# Operating Systems

## Redundant Array of Inexpensive Disks (RAID)

Thomas Ropars

thomas.ropars@univ-grenoble-alpes.fr

2017

# References

The content of this lecture is inspired by:

- *Operating Systems: Three Easy Pieces* by R. Arpaci-Dusseau and A. Arpaci-Dusseau

Other references:

- *Modern Operating Systems* by A. Tanenbaum
- *Operating System Concepts* by A. Silberschatz et al.

# Agenda

# Agenda

Introduction

RAID levels

Summary

# Motivation

In a previous lecture, we have seen how disks basically work.

# Motivation

In a previous lecture, we have seen how disks basically work.

Sometimes we would like more:

- Storage space
  - What if I have too many data for a single disk?
- Performance
  - What if I do a lot of reads and writes (I/O bound)?
- Reliability
  - What if my disk fails?

# RAID

## History

- A Case for Redundant Arrays of Inexpensive Disks (RAID) by D. Patterson, G. Gibson and R. Katz (1988)
  - Argue that RAID can perform better than expensive disks
  - Argue that this is true despite decreased MTTF (mean time to failure)
  - Defines 5 levels of RAID (still valid)

- Some manufacturers talk about "Redundant Array of Independent Disks"

# RAID

## Interface

- From the point of view of the OS, a RAID system is just a large, reliable, efficient disk.
- Logical I/O: The OS issues logical I/Os to the RAID system

## Internals

- A standard connection (eg., SATA, SCSI)
- A set of disks
- Volatile memory for buffering
- Microcontroller(s) that operate the RAID logic
- Physical I/O: The RAID issues the physical I/Os to the disks

# Metrics

- Capacity: Given N disks, how much client data can be stored?
- Reliability: How many disks failures may a design tolerate?
  - In the following, we assume a **fail-stop** failure model: a disk fails by crashing.
- Performance:
  - Latency: single-request latency
  - Steady-state throughput: total throughput of many concurrent requests (sequential or random)

# Agenda

Introduction

RAID levels

Summary

# RAID 0: Striping

- RAID level 0 is no RAID!
  - no redundancy
- Spread the blocks across the disks in round robin fashion.
  - With N disks, N blocks read/write in parallel
  - Often implemented at the granularity of blocks (other: bit-level, byte-level) – can be multiple blocks

| Disk 0 | Disk 1 | Disk 2 | Disk 3 |
|---|---|---|---|
| 0 | 2 | 4 | 6 |
| 1 | 3 | 5 | 7 |
| 8 | 10 | 12 | 14 |
| 9 | 11 | 13 | 15 |

Figure: Striping 16 blocks on 4 disks (2-block granularity)

# RAID 0: Striping

## Metrics

- Reliability: does not tolerate failures
- Capacity: $C \times N$[1]
- Performance: optimal performance

| Throughput[2] | | Latency[3] | |
|---|---|---|---|
| Sequential Read: | $N \times S$ | Read: | $T$ |
| Sequential Write: | $N \times S$ | Write: | $T$ |
| Random Read: | $N \times R$ | | |
| Random Write: | $N \times R$ | | |

[1] C: capacity of one disk; N: number of disks

[2] S/R: throughput of one disk with sequential/random accesses

[3] T: latency of read/write with a single disk

# RAID 1: Mirroring

- RAID level 1 targets reliability
- It keeps several copies of each block
- Copies are stored on different disks

| Disk 0 | Disk 1 | Disk 2 | Disk 3 |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 1 | 1 | 1 | 1 |
| 2 | 2 | 2 | 2 |
| 3 | 3 | 3 | 3 |

Figure: Mirroring 4 blocks on 4 disks

# RAID 1: Mirroring

## Metrics

- Reliability: tolerates $N - 1$ disk failures
- Capacity: $C$
- Performance:
  - With random read, multiple reads can be issued in parallel on different disks
  - Write latency increases because we have to wait for slowest disk to finish

### Throughput

| | |
|---|---|
| Sequential Read: | $S$ |
| Sequential Write: | $S$ |
| Random Read: | $N \times R$ |
| Random Write: | $R$ |

### Latency

| | |
|---|---|
| Read: | $T$ |
| Write: | $\geq T$ |

# RAID 1+0: Mirroring + striping

- Apply striping across pairs of mirrored disks
- RAID 0+1 is also possible (mirror a striping array of disks)
  - Considered less reliable: one disk failure makes a full array unusable.

| Disk 0 | Disk 1 | Disk 2 | Disk 3 |
|---|---|---|---|
| 0 | 0 | 2 | 2 |
| 1 | 1 | 3 | 3 |
| 4 | 4 | 6 | 6 |
| 5 | 5 | 7 | 7 |

Figure: RAID 1+0 on 4 disks

# RAID 1+0: Mirroring + striping

## Metrics

- Reliability: tolerates 1 disk failure (worst case)
- Capacity: $N/2 \times C$
- Performance:
  - Throughput is improved thanks to striping
  - Mirrored disks can serve different requests on random reads
  - Write latency increases because we have to wait for slowest disk to finish (worst case seek and rotational delay)

### Throughput

| | |
|---|---|
| Sequential Read: | $N/2 \times S$ |
| Sequential Write: | $N/2 \times S$ |
| Random Read: | $N \times R$ |
| Random Write: | $N/2 \times R$ |

### Latency

| | |
|---|---|
| Read: | $T$ |
| Write: | $\geq T$ |

# RAID 4: Parity-based redundancy

- Fault tolerance with reduced capacity lost
- Computes a parity block for each strip of blocks and store it on a separate disk
  - A bit-wise XOR is used to compute parity data (a parity block has the same size as a normal block)
  - If one disk fails, its data can be recovered based on the parity data and the data in the other disks

| Disk 0 | Disk 1 | Disk 2 | Disk 3 | Disk 4 |
|---|---|---|---|---|
| 0 | 1 | 2 | 3 | $PP_0$ |
| 4 | 5 | 6 | 7 | $PP_1$ |
| 8 | 9 | 10 | 11 | $PP_2$ |
| 12 | 13 | 14 | 15 | $PP_3$ |

Figure: RAID level 4 on 5 disks

# RAID 4: Parity-based redundancy

## Metrics

- Reliability: tolerates 1 disk failure
- Capacity: $(N-1) \times C$
- Performance:
  - For sequential writes, parity blocks can be written in parallel with the data stride
  - Writing a single block requires reading the block and the parity block first to be able to update the parity block
  - For random writes, the parity disk becomes the bottleneck (problem of small writes)

**Throughput**

| | |
|---|---|
| Sequential Read: | $(N-1) \times S$ |
| Sequential Write: | $(N-1) \times S$ |
| Random Read: | $(N-1) \times R$ |
| Random Write: | $R/2$ |

**Latency**

| | |
|---|---|
| Read: | $T$ |
| Write: | $2T$ |

# RAID 5: Rotating Parity

- Same advantages as RAID 4 but without the small writes performance issue
- Rotates the parity blocks across disks

| Disk 0 | Disk 1 | Disk 2 | Disk 3 | Disk 4 |
|---|---|---|---|---|
| 0 | 1 | 2 | 3 | $PP_0$ |
| 4 | 5 | 6 | $PP_1$ | 7 |
| 8 | 9 | $PP_2$ | 10 | 11 |
| 12 | $PP_3$ | 13 | 14 | 15 |
| $PP_4$ | 16 | 17 | 18 | 19 |

Figure: RAID level 5 on 5 disks

# RAID 5: Rotating Parity

## Metrics

- Reliability: tolerates 1 disk failure
- Capacity: $(N-1) \times C$
- Performance:
  - Random reads can use all disks
  - Random writes allow using all disks in parallel but each write require 4 I/0 operations (read/write of data and parity block)

| Throughput | | Latency | |
|---|---|---|---|
| Sequential Read: | $(N-1) \times S$ | Read: | $T$ |
| Sequential Write: | $(N-1) \times S$ | Write: | $2T$ |
| Random Read: | $N \times R$ | | |
| Random Write: | $N/4 \times R$ | | |

# Agenda

Introduction

RAID levels

Summary

# Summary

- Only interested in performance: RAID 0
- Random I/O performance and reliability: RAID 1+0
- Reliability and capacity: RAID 4
- Reliability and capacity + Random I/O performance: RAID 5

# References for this lecture

- *Operating Systems: Three Easy Pieces* by R. Arpaci-Dusseau and A. Arpaci-Dusseau
  - Chapter 38: Redundant Disk Arrays (RAID)
- If you are interested in the topic:
  - D. Patterson, G. Gibson, and R. Katz. *A case for redundant arrays of inexpensive disks (RAID)*. ACM, 1988.