Database Management Systems II – Exercise 1

TECHNISCHE UNIVERSITÄT DARMSTADT

Robert Gottstein

gottstein@dvs.tu-darmstadt.de

Exercise 1.1
Amdahl‘s Law & Disk Performance

TECHNISCHE UNIVERSITÄT DARMSTADT

Amdahl‘s law for the effective performance speedup states:

$$S = \frac{1}{\frac{f}{k} + (1 - f)}$$

S = effective speedup (in times)

f = fraction of work in faster mode

k = speedup while in faster mode (in times)

Exercise 1.1
Amdahl‘s Law & Disk Performance

TECHNISCHE UNIVERSITÄT DARMSTADT

a) A DBMS Server spends 20% of its processing time in I/O operations. What speedup can be achieved if CPU speed is doubled (increased 10 times)?

20 % pt in I/O operations ➔ f = 80 % = 0.8
Double CPU speed ➔ k = 2

$$S = \frac{1}{\frac{f}{k} + (1 - f)}$$

$$S = \frac{1}{\frac{0.8}{2} + (1 - 0.8)}$$

$$S = \frac{1}{0.6} \approx 1.67$$

S = effective speedup (in times)
f = fraction of work in faster mode
k = speedup while in faster mode (in times)

Optimal: S = 2 ➔ 33% performance speedup lost!

Exercise 1.1
Amdahl‘s Law & Disk Performance

TECHNISCHE UNIVERSITÄT DARMSTADT

a) A DBMS Server spends 20% of its processing time in I/O operations. What speedup can be achieved if CPU speed is doubled (increased 10 times)?

20 % I/O operations ➔ f = 80 % = 0.8

CPU speed * 10 ➔ k = 10

$$S = \frac{1}{\frac{0.8}{10} + (1 - 0.8)} = \frac{1}{0.28} \approx 3.57$$

Optimal: S =10 ➔ 71.4 % performance speedup lost!

Exercise 1.1
Amdahl‘s Law & Disk Performance

TECHNISCHE UNIVERSITÄT DARMSTADT

Expected Speedup (k)	2	10
Benefit (k-1)	1	9
Speedup (Amdahl's Law)	1,67	3,57
Lost Expected Speedup (k-speedup)	0,33	6,43
Speedup Lost (decimal)	0,33	0,714

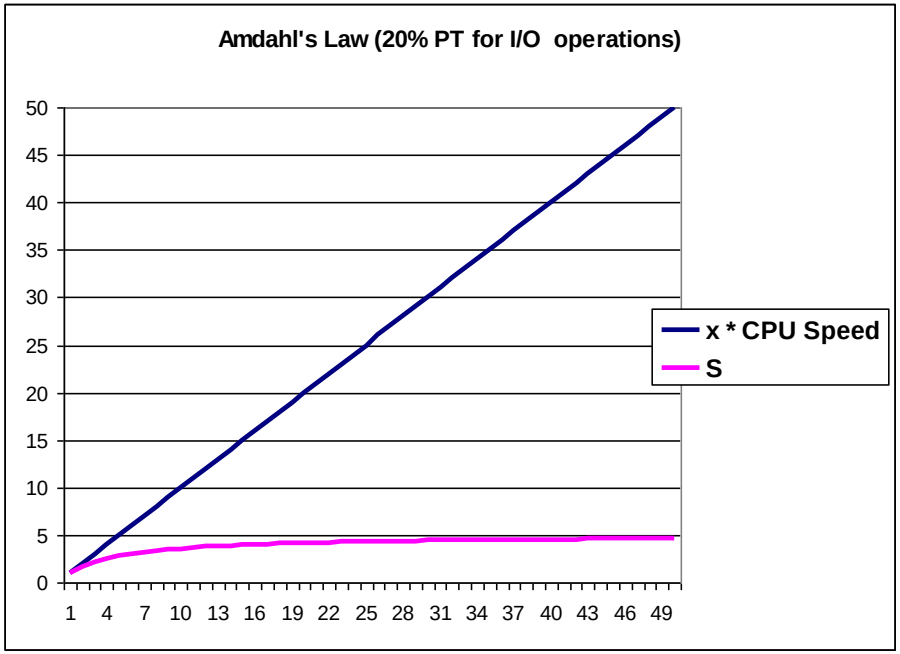
Speedup Lost → (k-amdahl)/benefit → How much of the additional speed is lost.

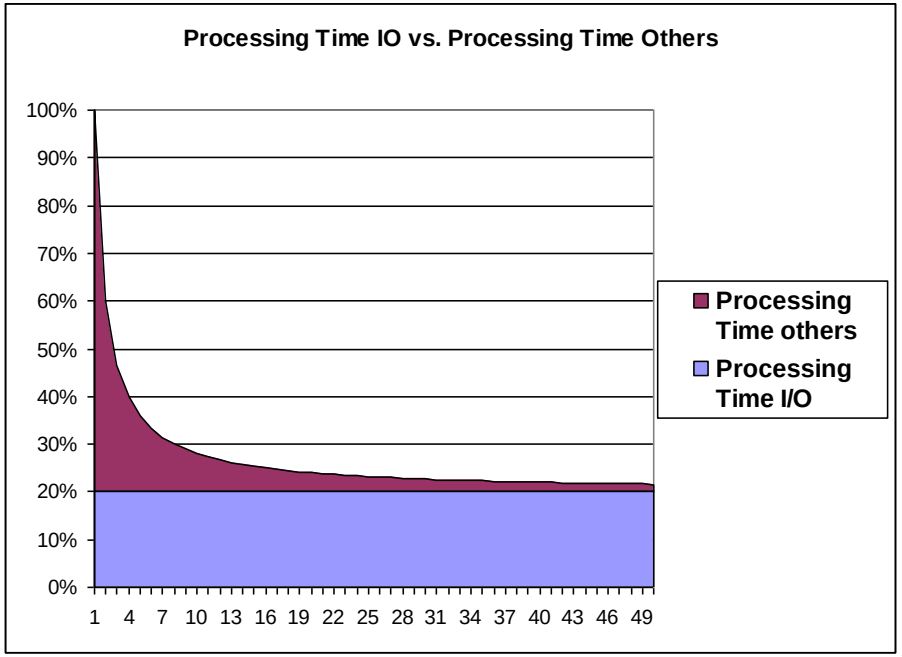
Divide the lost speedup (k-speedup) by the benefit (k-1)

Exercise 1.1
Amdahl's Law & Disk Performance

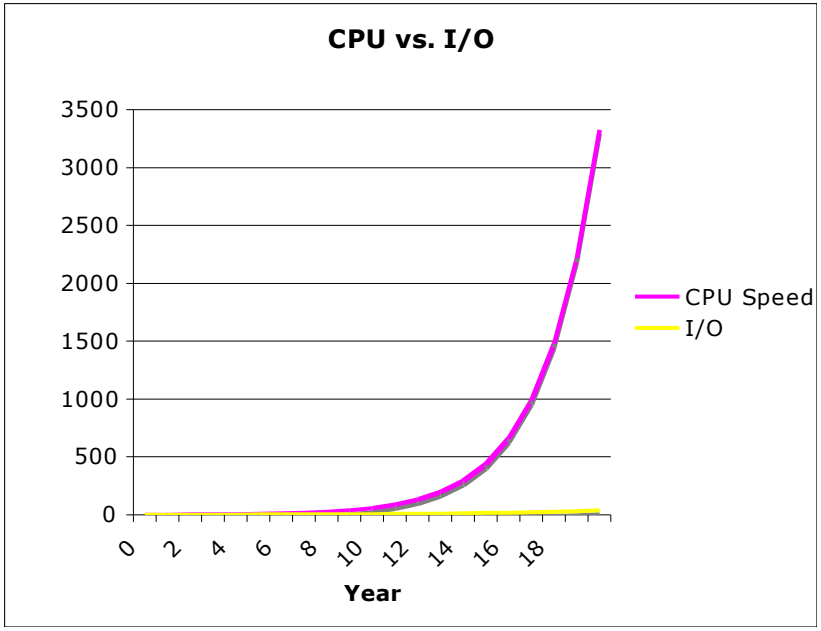
TECHNISCHE UNIVERSITÄT DARMSTADT

$$\lim_{k\to\infty} S = \frac{1}{\frac{0.8}{k} + 0.2} = 5$$





Exercise 1.1
Amdahl‘s Law & Disk Performance



Trends:

- CPU performance improves **50%** p.a.
- I/O performance only **10-20%** p.a.

➔**Disk I/O becomes a bottleneck!**

OLTP Systems spend approx. 60% in I/O
[Patterson: Computer Architecture
a quantitative approach]

Exercise 1.1
Amdahl‘s Law & Disk Performance

TECHNISCHE UNIVERSITÄT DARMSTADT

b) Given is a computer with 3000 MIPS and a hard drive with the following technical characteristics:

- *avg. seek time* = 8.5 ms
- *min. seek time* = 2 ms
- *transfer rate* = 444 MB/sec
- *rot. speed* = 7200 rpm
- *block size* = 512 bytes

How many instructions on *average* can the CPU execute during a block access?
What if the block is on *adjacent* track (min. seek time needed)?

Exercise 1.1
Amdahl's Law & Disk Performance

TECHNISCHE UNIVERSITÄT DARMSTADT

1. How many time is needed for one block access?

Disk access time = *seek time* + *rotational latency* + *transfer time*

$$\Rightarrow t_{access} = t_{seek} + \frac{1}{2} t_{rotation} + t_{transfer}$$

avg. seek time = 8.5ms *rot. speed* = 7200 rpm
transfer rate = 444 MB/sec *block size* = 512 bytes

$$\Rightarrow t_{access} = 8.5ms + \frac{1}{2} \cdot \frac{60 \cdot 1000}{7200} ms + \frac{512 \cdot 1000}{444 \cdot 1024^2} ms \approx 12.7ms$$

Note: Seek time (8.5 ms) and rotational delay (4.17 ms) dominate!
Transfer time is very small (0.001 ms)!

Exercise 1.1
Amdahl's Law & Disk Performance

TECHNISCHE UNIVERSITÄT DARMSTADT

2. How many instructions on average can the CPU execute during a block access?

3000 **M**illion **i**nstruction **p**er **s**econd ➔

$$\#\,Instructions = 3000\,million \cdot \left(\frac{12.7}{1000}\right) = 38.1\,million$$

Exercise 1.1
Amdahl's Law & Disk Performance

TECHNISCHE UNIVERSITÄT DARMSTADT

3. What if the block is on adjacent track (min. seek time needed)?

$$\Rightarrow t_{access} = 2.0ms + \frac{1}{2} \cdot \frac{60 \cdot 1000}{7200} ms + \frac{512 \cdot 1000}{444 \cdot 1024^2} ms \approx 6.2ms$$

$$\Rightarrow \#\,Instructions = 3000\,million \cdot \left(\frac{6.2}{1000} \right) = 18.6\,million$$

→ 20 Million Less than on a full seek

Exercise 1.1
Amdahl‘s Law & Disk Performance

TECHNISCHE UNIVERSITÄT DARMSTADT

- So for a IO System with the described properties the access should be:
  - as much random as possible
  - the smaller the block the better

➔WRONG!

- Slow random I/O kills the performance
- Try to do it sequential for those media
- See next slide…

Exercise 1.1
Amdahl‘s Law & Disk Performance

TECHNISCHE UNIVERSITÄT DARMSTADT

c) How to increase I/O bandwidth?

Hardware:

Decrease seek time

➔ improve mechanics

Decrease rotational latency

➔ increase rotations per minute (RPM)

Increase transfer rate

➔ increase disk density

➔Distribute data on multiple disks and parallelize access (disk striping)

Software:

Utilize sequential access to minimize positioning time

Exercise 1.1
Amdahl‘s Law & Disk Performance

TECHNISCHE UNIVERSITÄT DARMSTADT

- CPU performance has been improving 50% or more per year
- Disk positioning times only about 10% and disk transfer rates about 20% per year
- Performance gap between CPU and secondary storage systems
  ➔ **I/O Bottleneck**
- Disks rotate faster and transfer rates improve.
  ➔ **Seek times are the problem**

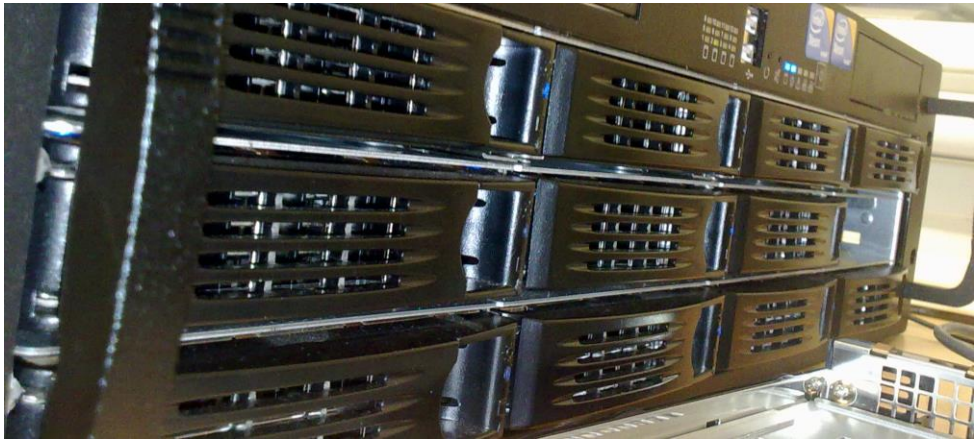
Use SSDs?

Exercise 1.1
Amdahl‘s Law & Disk Performance

TECHNISCHE UNIVERSITÄT DARMSTADT

- HW RAID0 8x HDD, 15K RPM
  - BlockSize: 4KB
  - Random Read [IOPS]: ~**2000**
  - Random Write [IOPS]: ~**2500** (controller cache)

- Our System SW RAID0 8x SSD
  - BlockSize: 4KB
  - Random Read [IOPS]: ~**280 000**
  - Random Write [IOPS]: ~**42 000**



Exercise 1.1
Amdahl‘s Law & Disk Performance

TECHNISCHE UNIVERSITÄT DARMSTADT

Solid State Disks

- No moving parts → no seek → no rotational delay
- 10x Faster than HDD on Random reads (4k)
- Fundamentally different Characteristics
  - Read/ Write/ Erase Operations
  - Additional Layer to Translate
  - High Parallelism (Query Depth)

- Whats the bottleneck now?
  - SATA Interface?
  - Controllers?
  - Bridges?

Bsc./ Msc./ Diploma… Thesis available

Exercise 1.2

Motivation: The 80-20 Rule & Need for Disk Striping

„80% of all the accesses on 20% of the data“
→ Close to the truth!

Exercise 1.2
The 80-20 Rule & Need for Disk Striping

TECHNISCHE UNIVERSITÄT DARMSTADT

The load on a ***transaction processing system*** is typically made of read-modify-write operations which are mapped to corresponding **I/O** requests, normally having the size of a ***disk block***. The data is stored in files, distributed over the available physical disks, *where each file is completely stored on one physical disk*.
The so-called ***80-20 rule states that 80% of the accesses concern only 20% of the data files***.
It is observed in practice that the potential hardware parallelism (the joint bandwidth) of the *I/O system is not utilized efficiently* in order to provide the expected throughput (number of requests processed per unit time).

a) To what extent can this observation be explained using the 80-20 Rule?
b) How can the efficiency of the secondary storage system be improved in order to achieve better overall bandwidth and I/O throughput?

Exercise 1.2
The 80-20 Rule & Need for Disk Striping

TECHNISCHE UNIVERSITÄT DARMSTADT

a) To what extent can this observation be explained using the 80-20 Rule?

Just having multiple disks doesn't bring significant performance benefits unless we **make sure that these disks are kept busy**, so that their joint bandwidth is exploited.

We can achieve a joint bandwidth (I/O rate) close to the sum of the bandwidths (I/O rates) of individual disks only if **all disks are involved** in the processing of I/O requests (so that their bandwidths are utilized).

Problem (from the 80-20 Rule):

Most accesses concern **the same** data files:
➔concern the **same devices**
➔queues are formed for access to the respective devices, while other devices are idle (not utilized).

Exercise 1.2
The 80-20 Rule & Need for Disk Striping

TECHNISCHE UNIVERSITÄT DARMSTADT

b) How can the efficiency of the secondary storage system be improved in order to achieve better overall bandwidth and I/O throughput?

To achieve maximum performance and device utilization, data needs to be split and distributed among available disks. This is called Disk Striping and is the fundamental technique used in RAID systems to boost performance.

Striping brings the following benefits:

- hot spots and load is distributed
- more disks are utilized and higher overall disk bandwidth is achieved

Note:
I/O caching helps to reduce hot spots, but only for "small working sets“

Exercise 1.3

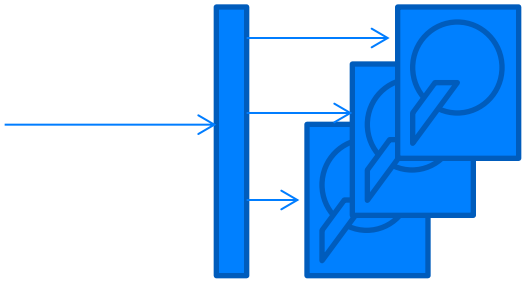
Choosing the Right Striping Unit

Exercise 1.3
Introduction

TECHNISCHE UNIVERSITÄT DARMSTADT

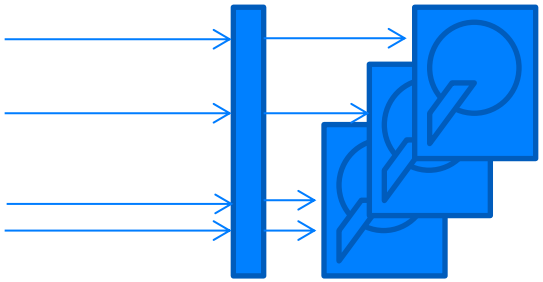
Parallelism:

when multiple disks in the array are involved in the processing of an individual request and they work in parallel.



Concurrency:

when multiple requests are processed simultaneously (concurrently) by different disks in the array.



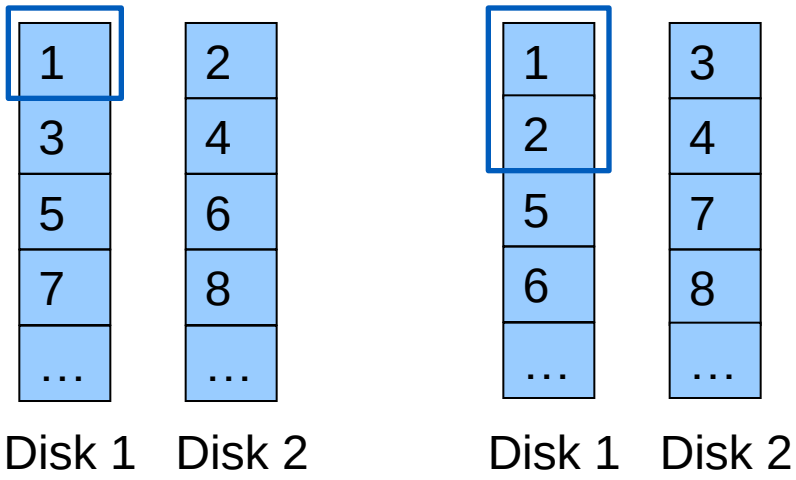
Concurrency of the processing ⇔ Concurrency of the workload.

Both parallelism and concurrency can lead to an increase in throughput.

Exercise 1.3
Introduction

TECHNISCHE UNIVERSITÄT DARMSTADT

Def. Striping Unit:
The maximum amount of logically contiguous data that is stored on a single disk.



Exercise 1.3
Choosing the right Striping Unit

TECHNISCHE UNIVERSITÄT DARMSTADT

The problem of choosing a right size for the striping unit is a problem of finding a **balance between parallelism and concurrency**.

A critical factor that should be considered is the characterization of the **workload** (*request size distribution, level of workload concurrency*).
E.g.: HomePC with FullHD .mp4 videos vs. Dokument Server with .txt files

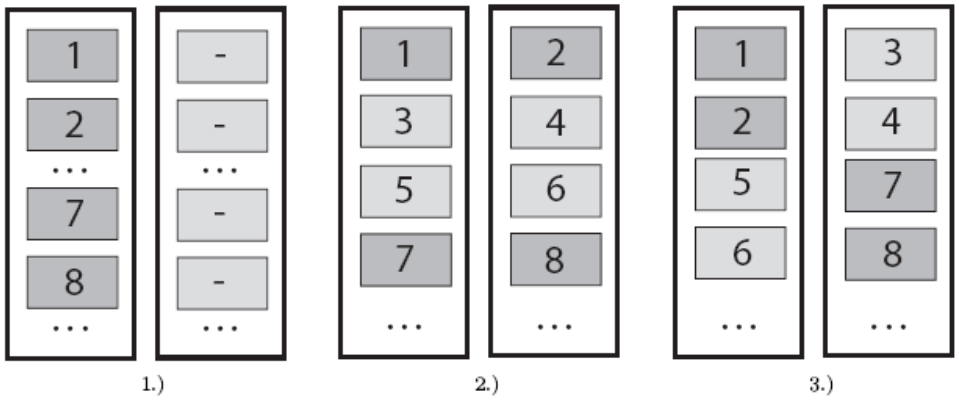
It usually doesn't make sense to use a striping unit smaller than 1 block. Accessing a block would in this case require positioning more than one heads.

Exercise 1.3
Choosing the right Striping Unit

We will now examine the effect of the **size of the striping unit** (the maximum amount of logically contiguous data that is stored on a single disk) on the processing of disk requests. Data is stored and addressed in units of a disk block. Two disks are available.

Following three configurations will be considered:

1. No striping: store on disk 1 until filled and then continue on disk 2
2. Striping with a striping unit of 1 block
3. Striping with a striping unit of 2 blocks



Exercise 1.3
Choosing the right Striping Unit

TECHNISCHE UNIVERSITÄT DARMSTADT

The access time for reading a block is determined by:

$$T_{acc} = kT_{pos} + T_{trans}$$

Mean positioning time for random accesses $k = 10$,
for adjacent blocks $k = 1$.

Exercise 1.3
Choosing the right Striping Unit

TECHNISCHE UNIVERSITÄT DARMSTADT

Examine the following 2 workloads (where R*i* stands for request *i*):

Workload a)

2 read requests sent *sequentially*:

R1:readBlocks(1,2)
R2:readBlocks(7,8)

Workload b)

2 read requests sent *concurrent*:

R1:readBlocks(1,2) *R2:readBlocks(7,8)*

Calculate the mean request response time (queuing time + service time) and the mean throughput (requests processed per unit time) for configurations 1.,2., and 3.

Exercise 1.3
Choosing the right Striping Unit

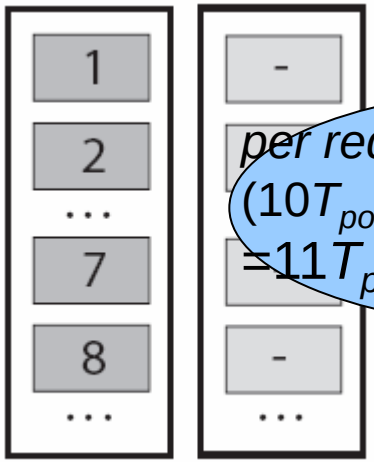
Configuration 1 (no striping):

a) 2 read requests sent ***sequentially***:

R1:readBlocks(1,2)

R2:readBlocks(7,8)

Disk 1	***Disk 2***
R1: position[1]	
R1: read[1]	
R1: position[2]	
R1: read[2]	
R2: position[7]	
R2: read[8]	



per request:
$(10T_{pos}+T_{trans})+(1T_{pos}+T_{trans})$
$=11T_{pos}+2T_{trans}$

➔This leads us to the following results:

$$Mean\ RT = \frac{2(11T_{pos}+2T_{trans})}{2} = 11T_{pos}+2T_{trans}$$

# of requests / time

$$ThrPut = \frac{2}{2(11T_{pos}+2T_{trans})} = \frac{1}{11T_{pos}+2T_{trans}}$$

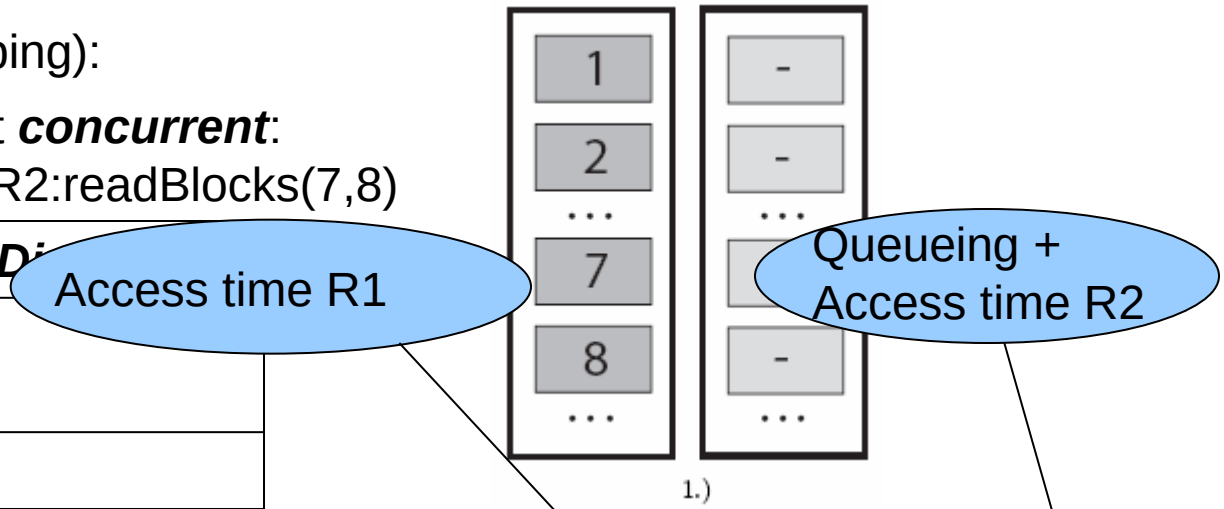
Exercise 1.3
Choosing the right Striping Unit

TECHNISCHE UNIVERSITÄT DARMSTADT

Configuration 1 (no striping):

b) 2 read requests sent ***concurrent***:
R1:readBlocks(1,2),R2:readBlocks(7,8)

Disk 1	***Di***
R1: position[1] R2: queue up	
R1: read[1]	
R1: position[2]	
R1: read[2]	
R2: position[7]	
R2: read[7]	
R2: position[8]	
R2: read[8]	



$$Mean\ RT = \frac{(11T_{pos} + 2T_{trans}) + 2(11T_{pos} + 2T_{trans})}{2}$$

$$Mean\ RT = 16{,}5T_{pos} + 3T_{trans}$$

$$ThrPut = \frac{2}{2(11T_{pos} + 2T_{trans})} = \frac{1}{11T_{pos} + 2T_{trans}}$$

TECHNISCHE UNIVERSITÄT DARMSTADT

Summary configuration 1

- No parallelism
- No concurrency

Exercise 1.3
Choosing the right Striping Unit

TECHNISCHE UNIVERSITÄT DARMSTADT

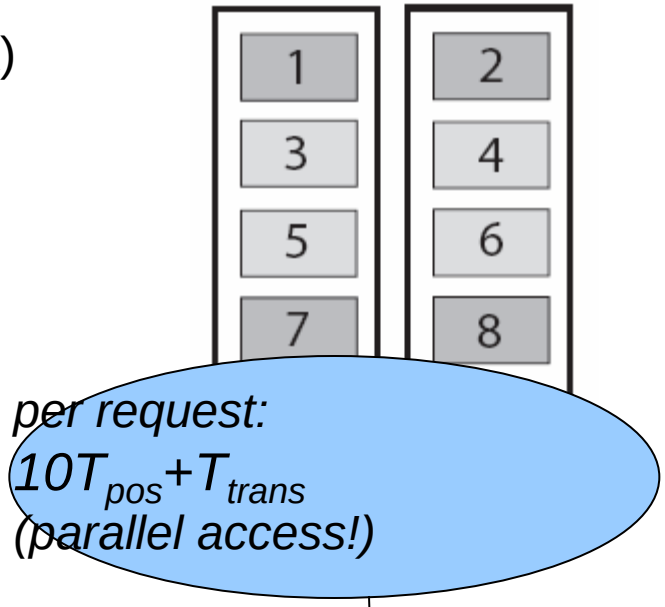
Configuration 2 (striping unit of 1 block)

a) 2 read requests sent ***sequentially***:

R1:readBlocks(1,2)

R2:readBlocks(7,8)

Disk 1	***Disk 2***
R1: position[1]	R1: position[2]
R1: read[1]	R1: read[2]
R2: position[7]	R2: position[8]
R2: read[7]	R2: read[8]



$$Mean\ RT = \frac{2(10T_{pos}+T_{trans})}{2} = 10T_{pos}+T_{trans}$$

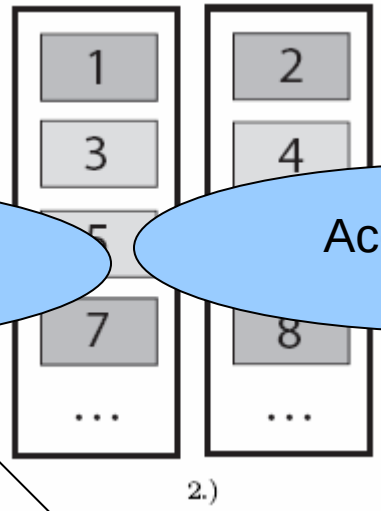
$$ThrPut = \frac{2}{2(10T_{pos}+T_{trans})} = \frac{1}{10T_{pos}+T_{trans}}$$

Exercise 1.3
Choosing the right Striping Unit

Configuration 2 (striping unit of 1 block)

b) 2 read requests sent **at the same time**:
R1:readBlocks(1,2),R2:readBlocks(7,8)

Disk 1	***Disk 2***
R1: position[1] R2: queue up	R1: position[2] R2: queue up
R1: read[1]	R1: read[2]
R2: position[7]	R2: position[8]
R2: read[7]	R2: read[8]



Queuing time + Access time R1

Access time R2

$$Mean\ RT = \frac{2(10T_{pos} + T_{trans}) + (10T_{pos} + T_{trans})}{2}$$

$$Mean\ RT = 15T_{pos} + 1.5T_{trans}$$

$$ThrPut = \frac{2}{2(10T_{pos} + T_{trans})} = \frac{1}{10T_{pos} + T_{trans}}$$

TECHNISCHE UNIVERSITÄT DARMSTADT

Summary for configuration 2
(smaller striping unit)

- **Parallelism:**
  - ➔ faster RT
  - ➔ higher TH
- **No concurrency**
  - ➔ concurrent requests blocked
  - ➔ worse Response Time for concurrent requests

- Good for low-concurrency workloads and large requests

Exercise 1.3
Choosing the right Striping Unit

TECHNISCHE UNIVERSITÄT DARMSTADT

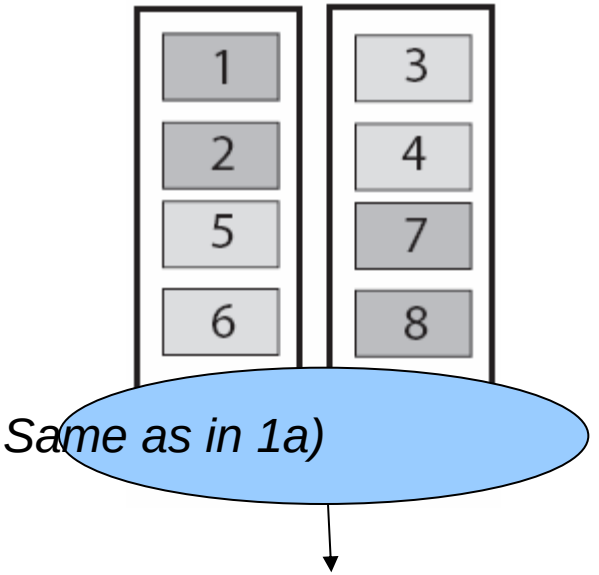
Configuration 3 (striping unit of 2 blocks)

a) 2 read requests sent *sequentially*:

R1:readBlocks(1,2)

R2:readBlocks(7,8)

Disk 1	***Disk 2***
R1: position[1]	
R1: read[1]	
R1: position[2]	
R1: read[2]	
	R2: position[7]
	R2: read[7]
	R2: position[8]
	R2: read[8]



$$Mean\ RT = \frac{2(11T_{pos} + 2T_{trans})}{2} = 11T_{pos} + 2T_{trans}$$

$$ThrPut = \frac{1}{11T_{pos} + 2T_{trans}}$$

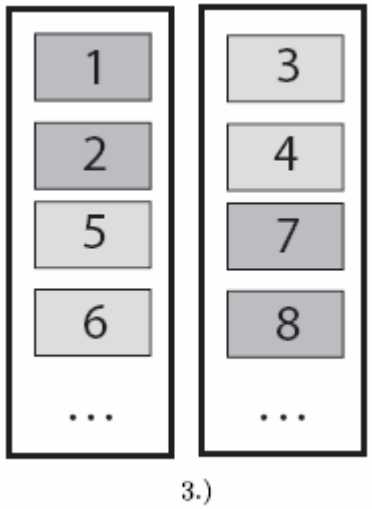
Exercise 1.3
Choosing the right Striping Unit

TECHNISCHE UNIVERSITÄT DARMSTADT

Configuration 3 (striping unit of 2 blocks)

b) 2 read requests sent at the same time:
R1:readBlocks(1,2),R2:readBlocks(7,8)

Disk 1	***Disk 2***
R1: position[1]	R2: position[7]
R1: read[1]	R2: read[7]
R1: position[2]	R2: position[8]
R1: read[2]	R2: read[8]



3.)

$$Mean\ RT = \frac{(11T_{pos} + 2T_{trans}) + (11T_{pos} + 2T_{trans})}{2}$$

$$Mean\ RT = 11T_{pos} + 2T_{trans}$$

$$ThrPut = \frac{2}{11T_{pos} + 2T_{trans}}$$

Exercise 1.3
Choosing the right Striping Unit

TECHNISCHE UNIVERSITÄT DARMSTADT

Summary for configuration 3 (larger striping unit)

- **No Parallelism**
- **Concurrency**
  ➔ High TH
- Good for
  - *high-concurrency workloads*
  - *small requests*

Exercise 1.3
Choosing the right Striping Unit

TECHNISCHE UNIVERSITÄT DARMSTADT

Summary I

A). Sequential Requests (low-concurrency workloads):

	RT	ThrPut
1	$11T_{pos} + 2T_{trans}$	$\frac{1}{11T_{pos} + 2T_{trans}}$
2	$10T_{pos} + T_{trans}$	$\frac{1}{10T_{pos} + T_{trans}}$
3	$11T_{pos} + 2T_{trans}$	$\frac{1}{11T_{pos} + 2T_{trans}}$

Exercise 1.3
Choosing the right Striping Unit

TECHNISCHE UNIVERSITÄT DARMSTADT

Summary II

B).Concurrent Requests(high-concurrency workloads):

	RT	ThrPut
1	$16.5T_{pos} + 3T_{trans}$	$\frac{1}{11T_{pos} + 2T_{trans}}$
2	$15T_{pos} + \frac{3}{2}T_{trans}$	$\frac{1}{10T_{pos} + T_{trans}}$
3	$11T_{pos} + 2T_{trans}$	$\frac{2}{11T_{pos} + 2T_{trans}}$

Summary III

Sequential executions:

- Configuration 2 provides min. Resp.Time and max. Throughput.
- Configuration 3 doesn't bring any benefits
  (in general for low concurrency workloads with dominating small requests)

Exercise 1.3
Choosing the right Striping Unit

TECHNISCHE UNIVERSITÄT DARMSTADT

Is configuration 2 or 3 better for concurrent requests?

Case: Response Time:

$$RT(2b) > RT(3b)$$

$$(15T_{pos} + \frac{3}{2}T_{trans}) > (11T_{pos} + 2T_{trans})$$

$$T_{pos} > \frac{1}{8}T_{trans}$$

$$(\text{Typically: } T_{pos} \approx 1ms; T_{trans} \approx 0.01ms)$$

Since T_{pos} dominates, we have:

PT(2b) > PT(3b) ➔ **3 performs better**

Exercise 1.3
Choosing the right Striping Unit

TECHNISCHE UNIVERSITÄT DARMSTADT

Is configuration 2 or 3 better for concurrent requests?

Throughput:

$$ThrPut(2b) < ThrPut(3b)$$

$$\frac{2}{(10T_{pos} + T_{trans})} < \frac{4}{(11T_{pos} + 2T_{trans})}$$

$$20T_{pos} > 11T_{pos}$$

➔3 performs better

➔For concurrent requests configuration 3 provides min. RT and highest TP.

Exercise 1.3
Choosing the right Striping Unit

TECHNISCHE UNIVERSITÄT DARMSTADT

Factors affecting the choice of striping unit:

- **Workload concurrency** (avg. number of requests in the system at any given time)
  - *high-concurrency workloads prefer larger*
  - *low-concurrency workloads prefer smaller*
- **Request size distribution**
  - *small requests prefer larger*
  - *large requests prefer smaller*
- The **average positioning time** and **data transfer rate** of the disks in the array

Exercise 1.3
Choosing the right Striping Unit

TECHNISCHE UNIVERSITÄT DARMSTADT

Smaller striping unit leads to:

- **higher degree of parallelism**
  ➔higher transfer rate for most I/O requests (with exception of very small requests where multiple seeks are not worth it)
- **lower degree of concurrency**
  more disks cooperate in serving each request
  ➔more concurrent transactions blocked)
- less data read per disk positioning
- better for DSS (Data-Storage Server)
  low concurrency, large sequential accesses

Exercise 1.3
Choosing the right Striping Unit

TECHNISCHE UNIVERSITÄT DARMSTADT

Larger striping unit leads to:

- **lower degree of parallelism**
  ➔lower transfer rate for smaller requests, larger requests still benefit
- **higher degree of concurrency**
  multiple small requests can be serviced simultaneously
- more data read per disk positioning
- better for OLTP systems
  high concurrency, small random accesses

Exercise 1.3
Choosing the right Striping Unit

TECHNISCHE UNIVERSITÄT DARMSTADT

Rules of thumb (Chen, Patterson):

The striping unit choice is primarily dependent on the concurrency of the applied **workload** and is relatively insensitive to the request size distribution.

- **High level of concurrency**
  ➔ *choose larger striping unit*
- **Low level of concurrency**
  ➔ *choose smaller striping unit*
- **Long positioning time**
  ➔ *choose larger striping unit*
  (so that more data is read for each positioning)
- **Low transfer rate (large transfer time)**
  ➔ choose smaller striping unit (to enable parallel transfers)

Exercise 1.4

TECHNISCHE UNIVERSITÄT DARMSTADT

Evaluation of the different RAID levels

Exercise 1.4
Introduction

TECHNISCHE UNIVERSITÄT DARMSTADT

Each RAID level achieves ***a different tradeoff between performance, reliability and cost***.

The main distinguishing characteristics are:

- the *granularity of data interleaving* (bit vs. block-interleaved)
- how *redundant info is computed* – redundancy scheme (Parity, Hamming, Reed-Solomon)
- how *redundant info is distributed* across disk array

Additional configuration parameters:

- Striping Unit
- Parity Group Size

Exercise 1.4
Overview of RAID levels

TECHNISCHE UNIVERSITÄT DARMSTADT

Existing RAID levels:

RAID Level	**Name**
RAID 0	Striping
RAID 1	**Mirroring**
RAID 2	Memory style ECC
RAID 3	Bit-Interleaved Parity
RAID 4	**Block-Interleaved Parity**
RAID 5	**Block-Interleaved Distributed Parity**
RAID 6	Block-Interleaved Double Distributed Parity

Multiple (nested) RAID levels like 0+1, 1+0, 3+0, 0+3, 1+5 …

Exercise 1.4
Overview of RAID levels

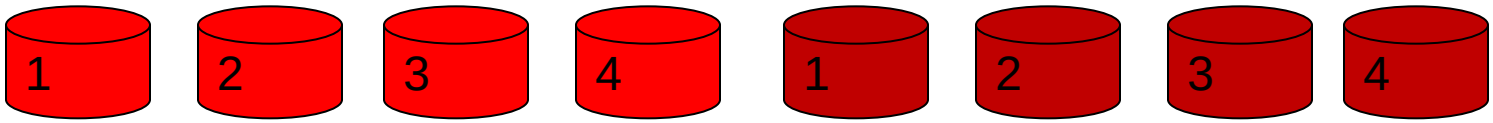
TECHNISCHE UNIVERSITÄT DARMSTADT

a) Describe the main differences between RAID Levels 1, 4, and 5.

b) Cost/Performance differences between RAID Levels 1, 4, and 5.

c) Summarize the pros and cons of RAID levels 1 and 5.

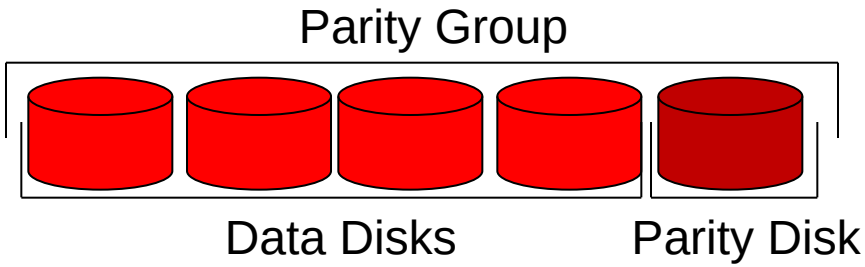
Exercise 1.4
RAID 1 - Mirroring



- Same data is written to primary disk and mirror
  ➔ **twice as many disks needed**
    ➔data is always fully redundant
    ➔bad storage efficiency
- Data can be read from either copy
  ➔ **Very efficient reads:**
  double the read bandwidth to a data item
- If availability and throughput are more important than storage efficiency

Exercise 1.4
RAID 4 - Block-Interleaved Parity

TECHNISCHE UNIVERSITÄT DARMSTADT

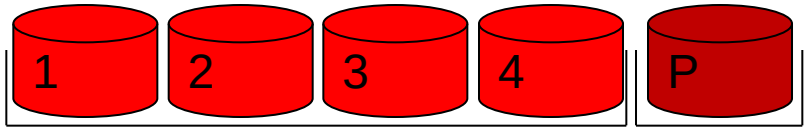


- Interleaving across disks in blocks of arbitrary size
- **Read:** 1 disk access *if* accessed unit < block size
- **Write:** update requested data blocks, compute parity, update parity disk ➔
- **Even small writes need 4 accesses:**
  1 r(old data), 1 w(new data), 1r(old parity), 1w(new parity)
- **Large writes use XOR over all touched disks**
- *Main problem:*
  Parity disk becomes bottleneck for concurrent writes

Exercise 1.4
RAID 4 - Block-Interleaved Parity

TECHNISCHE UNIVERSITÄT DARMSTADT

Example:



Data Disks Parity Disk

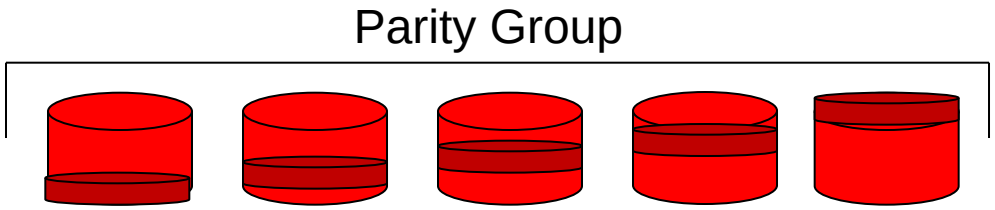
2 concurrent small writes:

1. Step:
   one writes data to Disk 1,
   one to Disk 2
   ➔No Problem
2. Step:

Both must write new parity

➔Parity Disk becomes bottleneck

Exercise 1.4
RAID 5 - Block-Interleaved Distributed Parity



- Parity distributed uniformly over all disks
  ➔**Eliminates hot-spot on parity disk**
- **Better read performance (e.g. than RAID 4):**
  data distributed over all disks, more disks utilized
- **Best read/write performance for most requests**
- Main problem remains:
  Inefficiency of small writes
  4 accesses needed as compared to 2 accesses in RAID 1

Exercise 1.4
Describe the main differences between RAID Levels.

TECHNISCHE UNIVERSITÄT DARMSTADT

Notes:

- RAID 1 is similar to RAID 5 with parity group size of 2.

- The problem of selecting a RAID level is related to the more general problem of choosing a right **parity group size** and **striping unit**.

Appendix

RAID 01 vs. 10

RAID 10 vs. RAID 01

TECHNISCHE UNIVERSITÄT DARMSTADT

E.g. for ten disks:

RAID 01 (Mirror of Stripes):

Divide the ten disks into **two sets of five**. Turn each set into a RAID 0 array containing five disks, then mirror the two arrays.

RAID 10 (Stripe of Mirrors):

Divide the ten disks into **five sets of two**. Turn each set into a RAID 1 array, then stripe across the five mirrored sets.

Note:

Be careful with the names. *Most* of the industry use them in the way that if RAID *X* is applied first and then RAID *Y*, that is RAID *XY* (*X*/*Y*, *X*+*Y*). But other companies reverse the order!!! They might call a RAID *XY* RAID *YX*!

RAID 10 vs. RAID 01

TECHNISCHE UNIVERSITÄT DARMSTADT

What are the differences???

No differences:

- Drive requirements
- Capacity
- Storage efficiency
- Performance (nearly no difference)

➔**The big difference: *Fault tolerance!***

RAID 10 vs. RAID 01

TECHNISCHE UNIVERSITÄT DARMSTADT

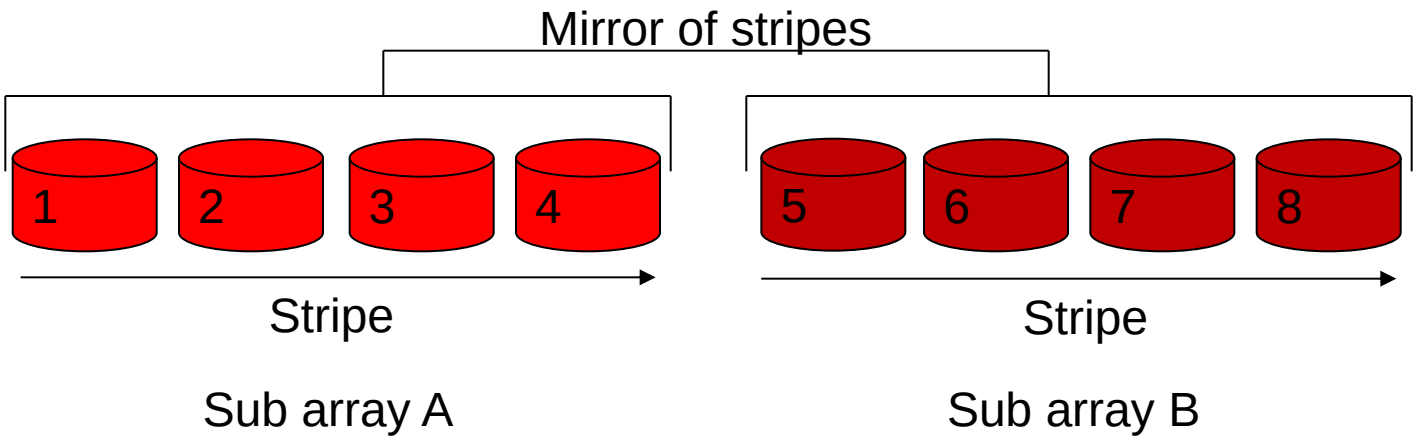
Implementation of multi level RAIDs:

- Most controller compose a *“super array”* on top of *“sub arrays”*
- Often the “sub arrays” which form the “super array” *(also called sets)* are considered to be “single units”
- **The controller only considers one of these “single units” to be “up” or “down” as a whole**
- Redundancy only exists **within** a “sub array”, not **between** “sub arrays” (even if they have the same data like a other sub array)

RAID 10 vs. RAID 01

TECHNISCHE UNIVERSITÄT DARMSTADT

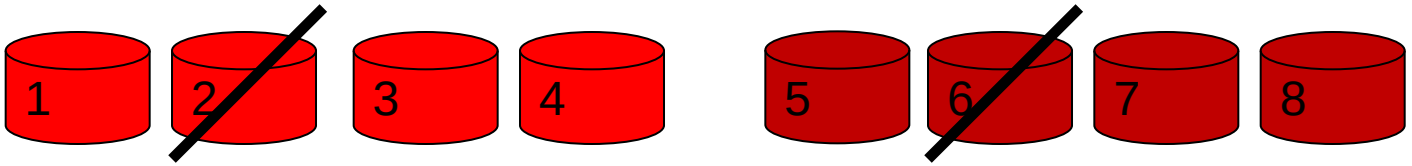
RAID 01 with eight disks



RAID 10 vs. RAID 01

TECHNISCHE UNIVERSITÄT DARMSTADT

Fault tolerance of RAID 01



Sub array A Sub array B

Disk 2 fails:

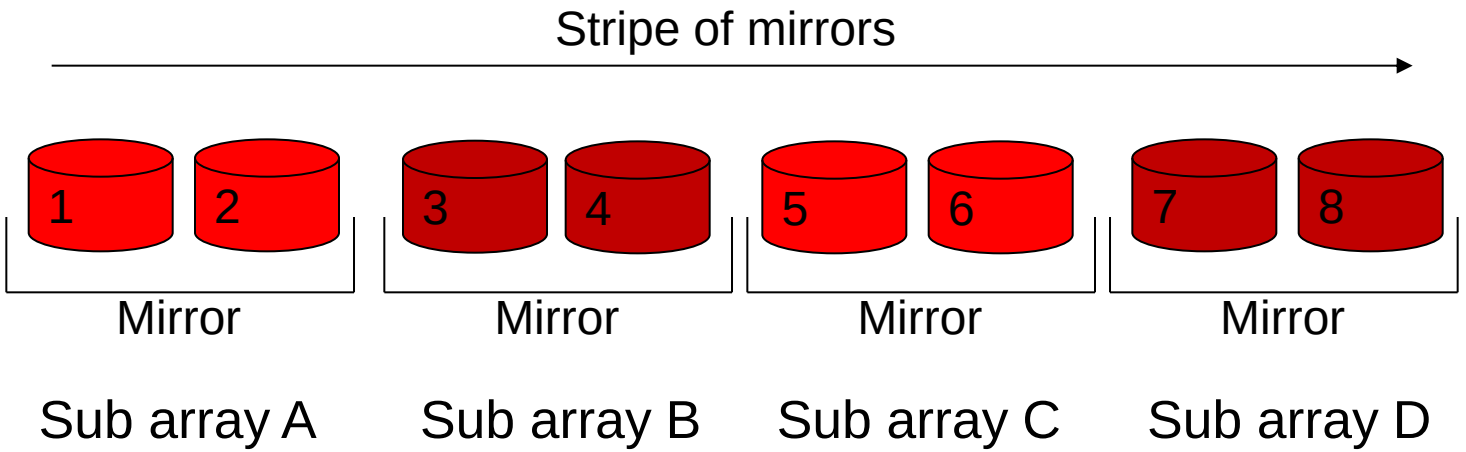
➔Sub array A is down!
➔Only sub array B is available

What happens if any disk of sub array B fails, e.g. disk 6?

➔Sub array B is down!
➔**The entire array is down!**

RAID 10 vs. RAID 01

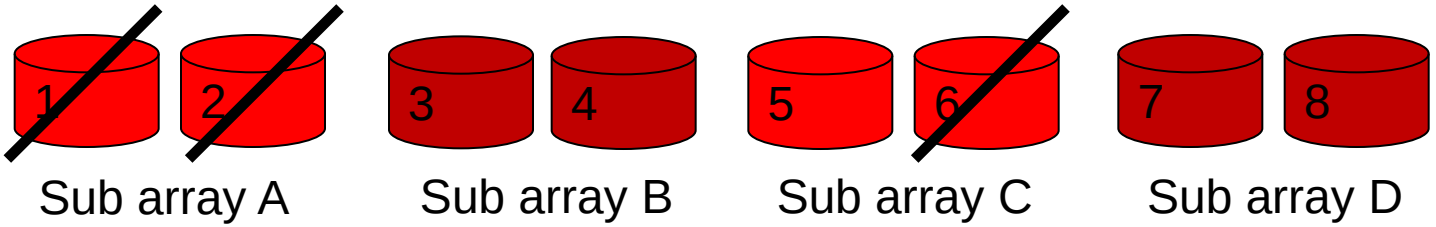
RAID 10 with eight disks



RAID 10 vs. RAID 01

TECHNISCHE UNIVERSITÄT DARMSTADT

Fault tolerance of RAID 10



Disk 2 fails:

➔Sub array A is still available

Any disk of another sub array fails, e.g. disk 6:

➔All sub arrays are still available

Only if *two disks of the same sub array fail*, the entire array is down (e.g., disk 1)!

Summary

- RAID 10 is more robust than RAID 01
- RAID 01 could be theoretically as fault tolerant as RAID 10

  If controllers running RAID 01 were smart they would use after a disk failure the corresponding disk of the other sub array.

  In general: **Controllers are not smart!!!!**