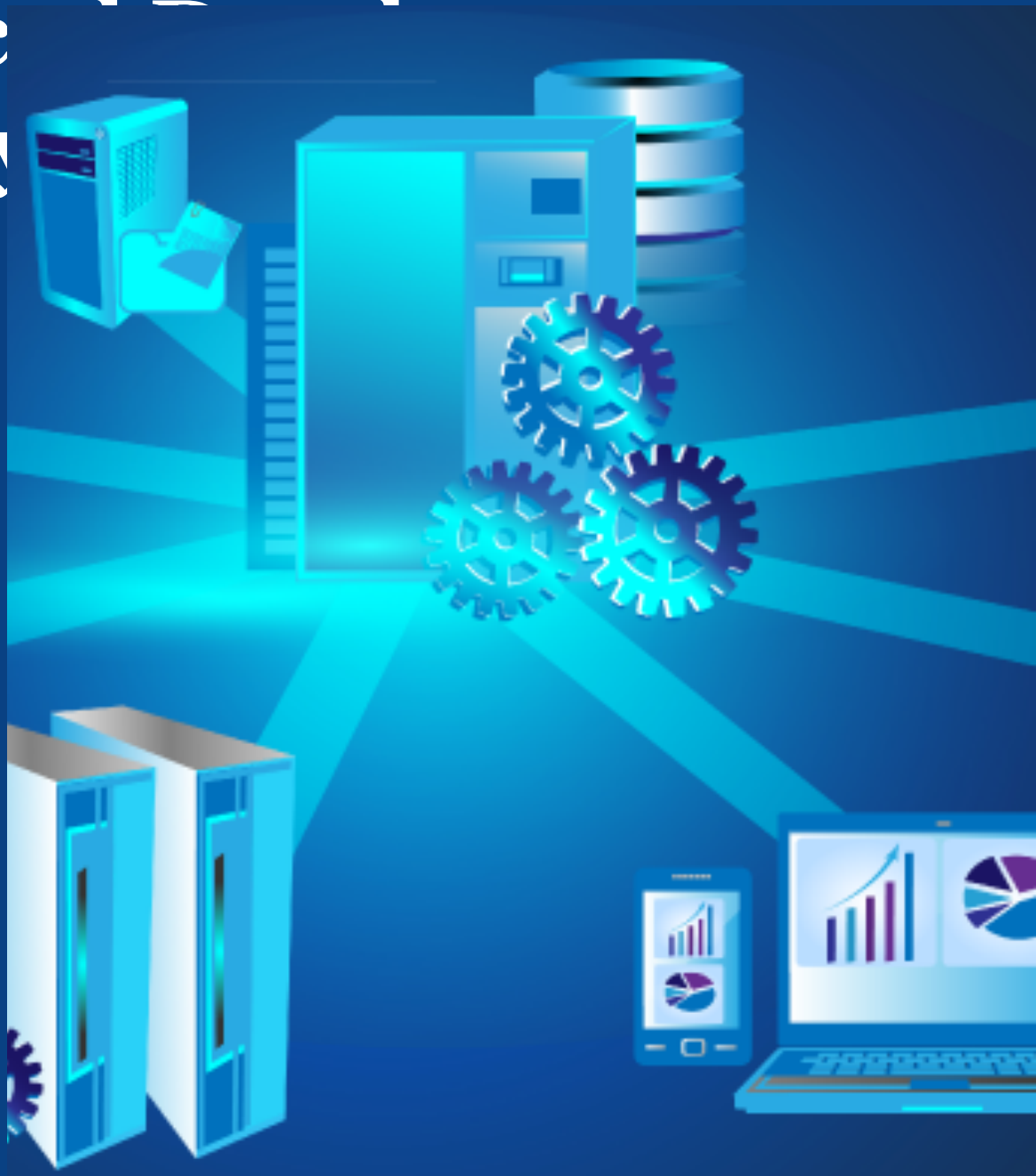
COMP90050: Advanc… Sy…

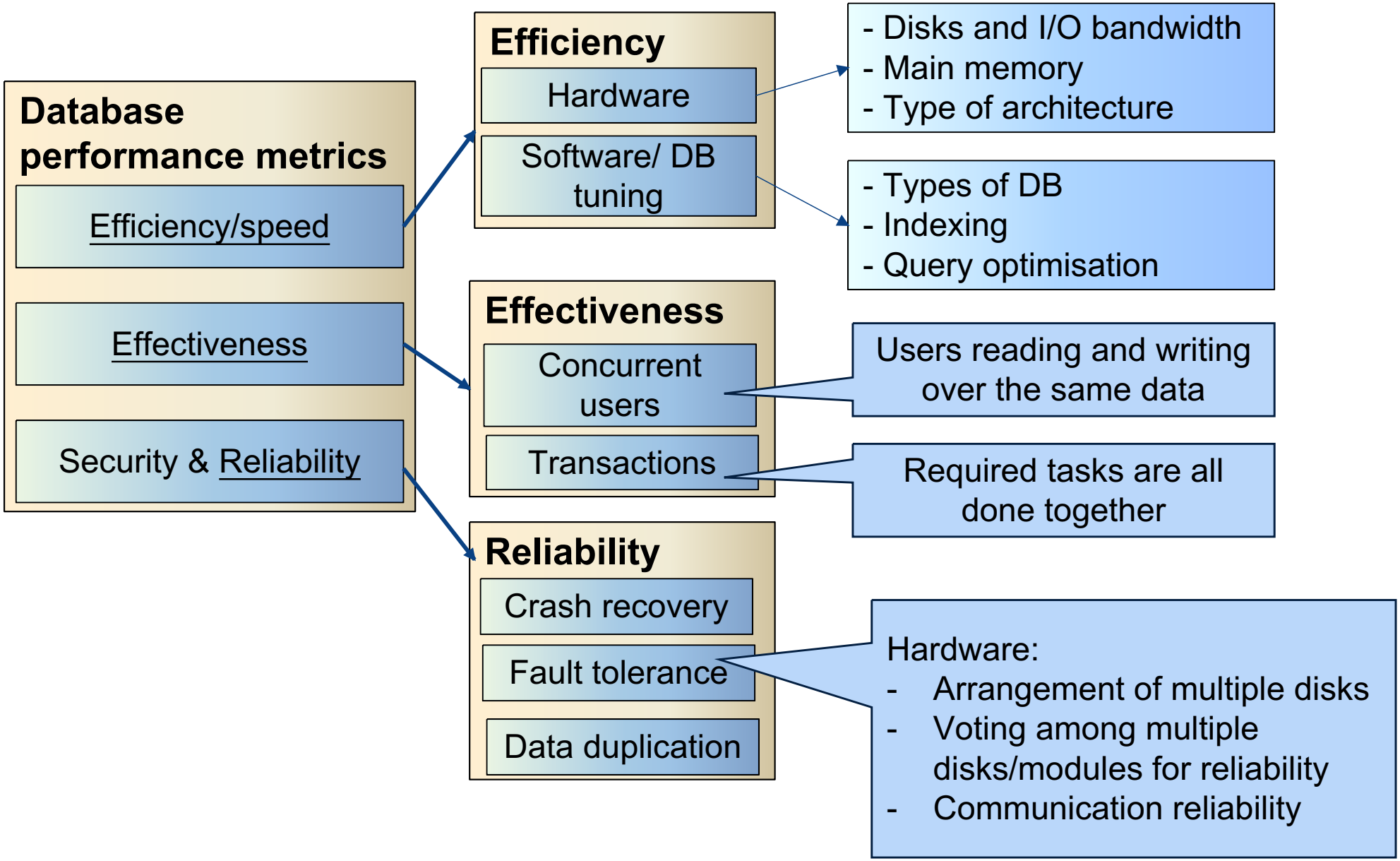


Lecturer: Farhana Choudhury (PhD)

Fault tolerance

Week 8

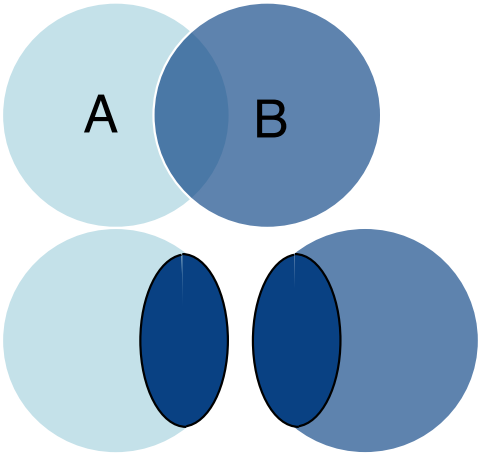
Core Concepts of Database management system



Fault Tolerance

The property that enables a system to continue operating properly in the event of the failure of some of its components.

Some statistics first!



P(A) = probability of an event A is happening in a **certain period.**

P(A and B) = probability both A and B happening in that period

= P(A)*P(B) assuming A and B are independent events.

P(A or B) = P(A) + P(B) - P(A and B)

= P(A) + P(B) - P(A)*P(B) [Assuming A and B are independent]

P(A) + P(B) [if P(A) and P(B) are very small]

Mean time to event A is, MT(A) = 1/P(A)
(The time elapsed before A occured)

If events A, B, have mean time MT(A), MT(B), then the mean time to the first event, MT(A or B) = 1/P(A or B)

The time that elapsed before one of these events occurs

If there are n events, each with the same probability p, then

Probability that one of the events occur = p + p + …. (n times)

= n*p [assuming p is small]

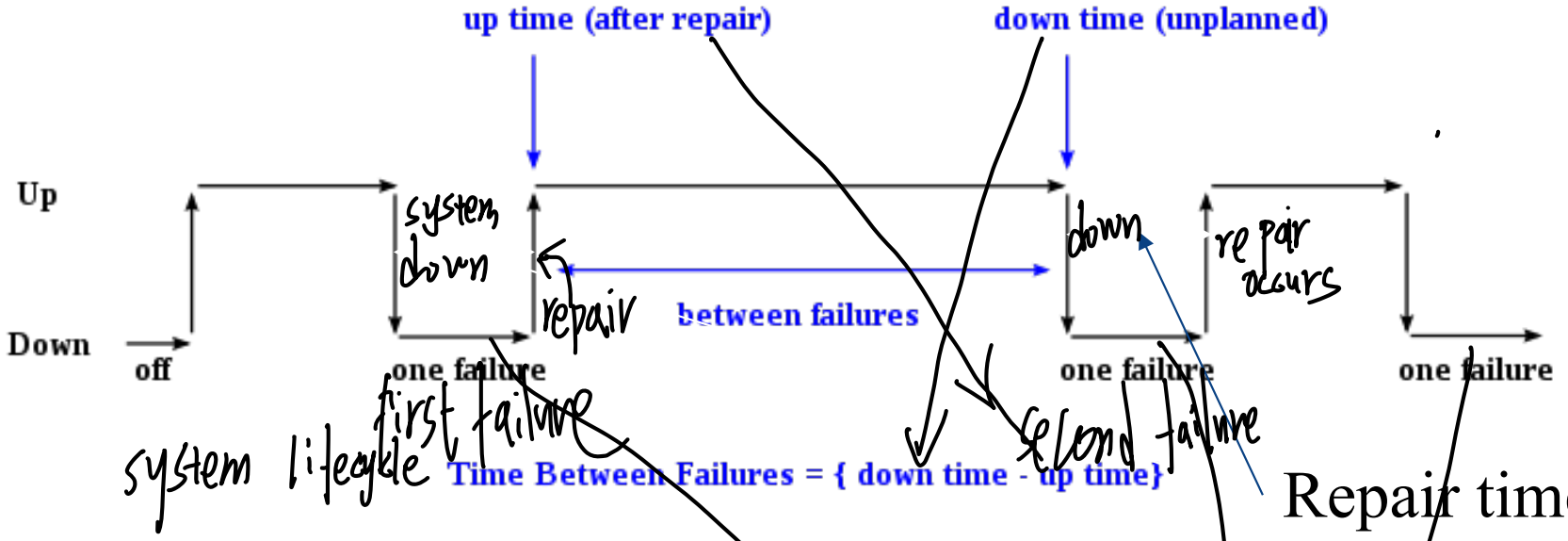
2. Mean time to one of the events (i.e., mean time to the first event)

= 1/(n*p)

= (1/p) * (1/n)

= m * (1/n) = m/n

A system's lifecycle



Module availability : measures the ratio of service accomplishment to elapsed time

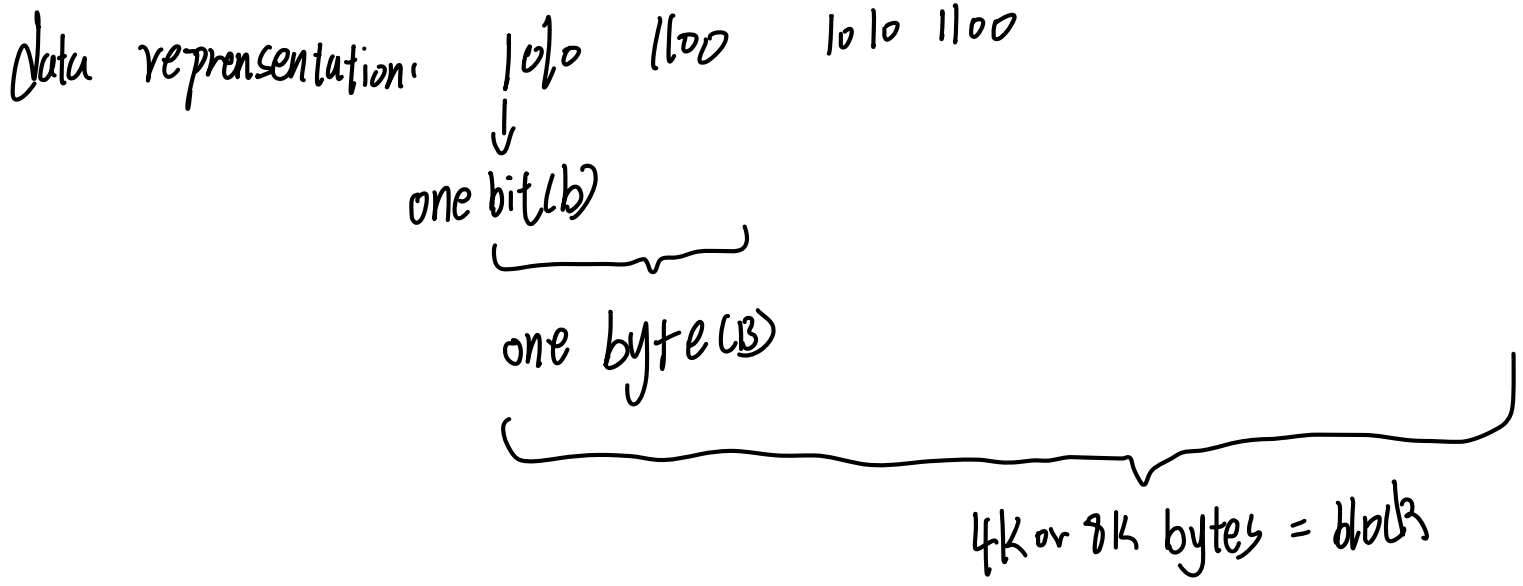
the **time** elapsing before a failure is experienced

$$= \frac{\text{Mean time to failure}}{\text{Mean time to failure} + \text{mean time to repair}}$$

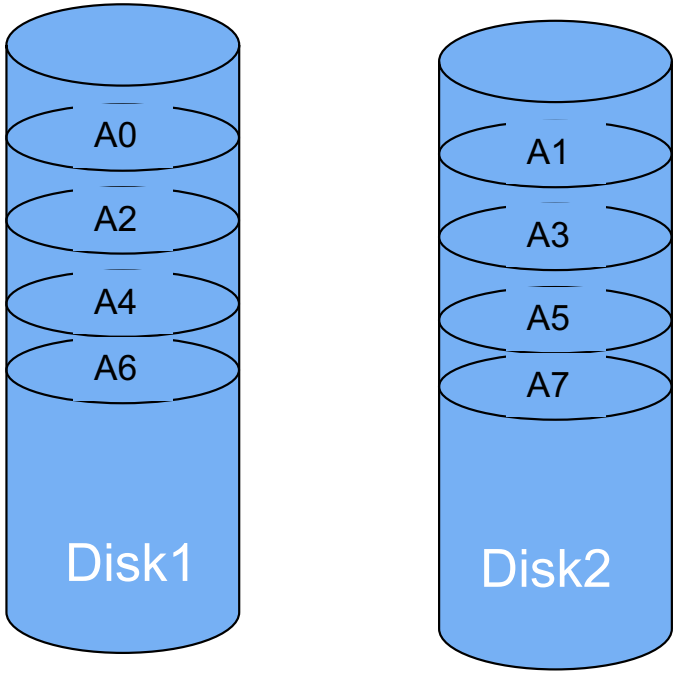
Fault tolerance by RAID

RAID: data will be stored in more than one disk
even one of the disks fail, the system will be still able to operate, and reading and writing operation can still be done there

Redundant Array of Independent Disks – different ways to combine multiple disks as a unit for fault tolerance or performance improvement, or both of a database system



RAID 0 (Block level Striping)



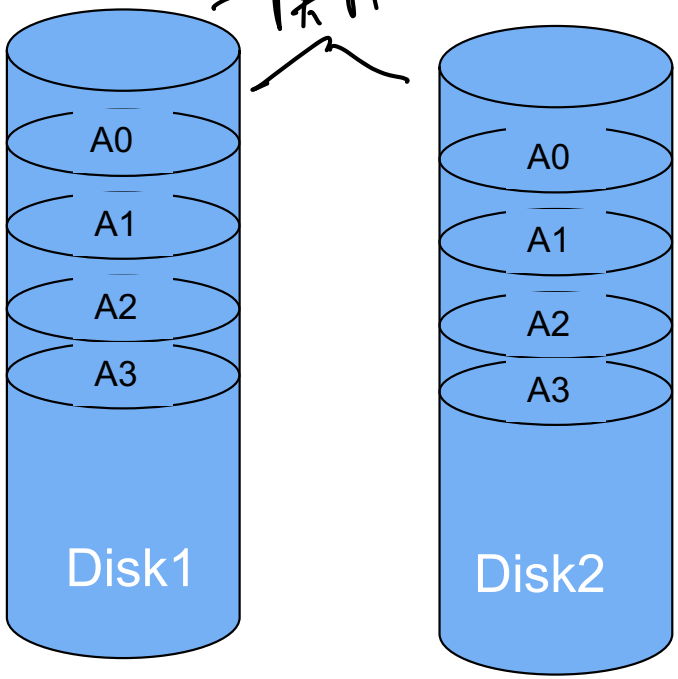
A0, A1, A2, ... are contiguous blocks of data of a file

- Provides balanced I/O of disk drives –throughput ~doubles
- Any disk failure will be catastrophic and MTTF reduces by a factor of 2

Higher throughput at the cost of increased vulnerability to failures

- A means Block (4K or 8K bytes of storage)
- MTTF = Mean Time To Failure

RAID 1 (mirroring)



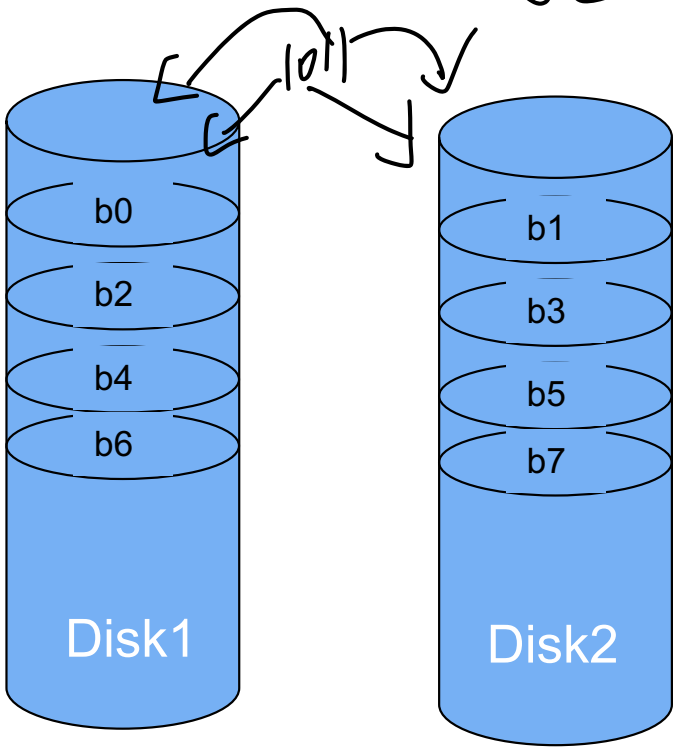
- Provides higher read throughput but lower write throughput (half of the total speed –i.e. single disk speed)
- Half storage utilization.
- MTTF increases substantially (quadratic improvement –i.e. MTTF^2!)

Continues to operate as long as 1 disk is functional

Calculation of MTTF values – in tutorials

- A means Block (4K or 8K bytes of storage)
- MTTF = Mean Time To Failure

RAID 2 (bit level striping)

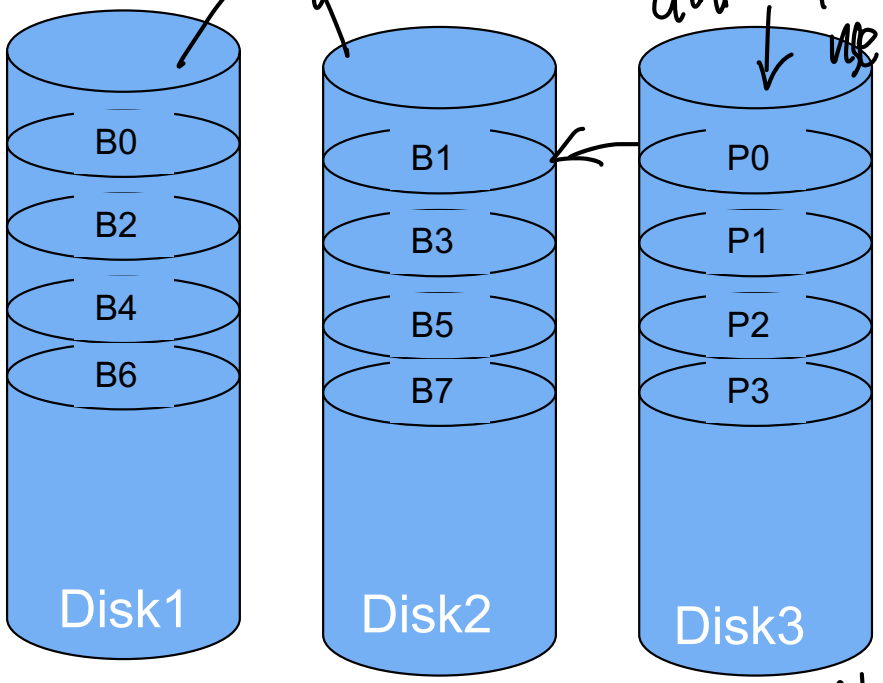


- Striping takes place at bit level
- Provides higher transfer rate (double the single disk)
- MTTF reduced by half as in RAID 0
- rarely used

striping at bit level is difficult to maintain and read.

- b means bit
- MTTF = Mean Time To Failure

RAID 3 (Byte level striping)



B0, B1, B2, B3, .. are bytes of data of file

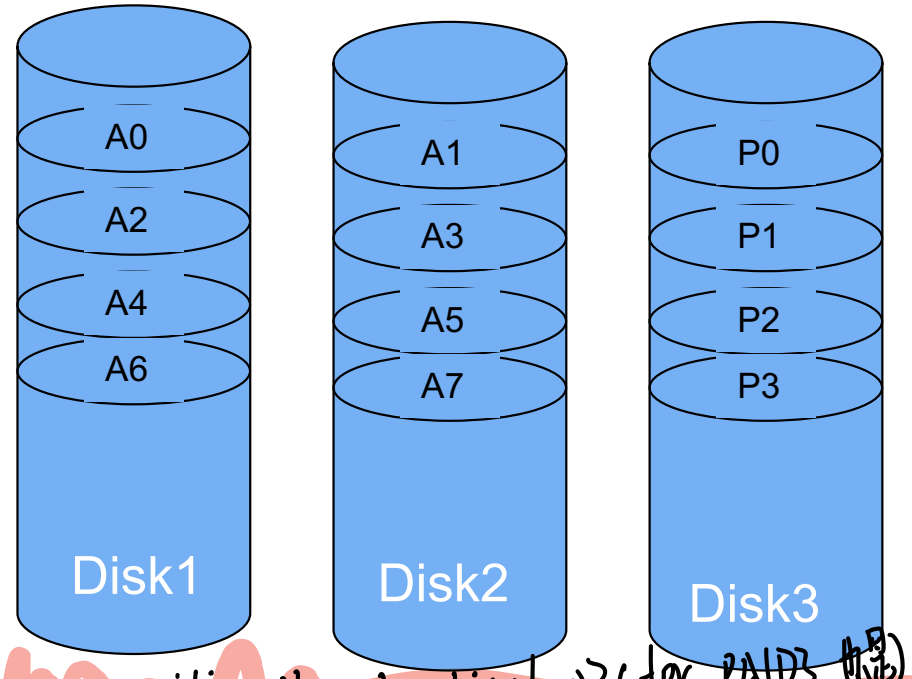
- Striping takes place at byte level
- Rarely used
- Provides higher transfer rate as in RAID 0
- P0 is parity for bytes B0 and B1
- $P_i = B_{2i} \oplus B_{2i+1}$, here $\oplus$ is exclusive-or operator

MTTF increases substantially (1/3 of RAID1 = $MTTF^2/3$), as 1 disk failure can be recovered from the data of the other 2 disks

- B means Byte
- P is parity
- MTTF = Mean Time To Failure
- Parity (or check bits) are used for error detection

RAID 4 (Block level level striping)

byte level ✗



- Striping takes place at block level
- Dedicated disk for parity blocks
- Provides higher throughput but very slow writes. Disk3 has more writes as Parity needs to be updated for every data write.
- MTTF increases substantially (same as RAID3)
- Pi = $A_{2i} \oplus A_{2i+1}$, here ⊕ is an exclusive-or operator

reading throughput is high (read from two disks in parallel), writing very slow (RAID3也一样)

Why writing throughput is slow? (for RAID3 也是)

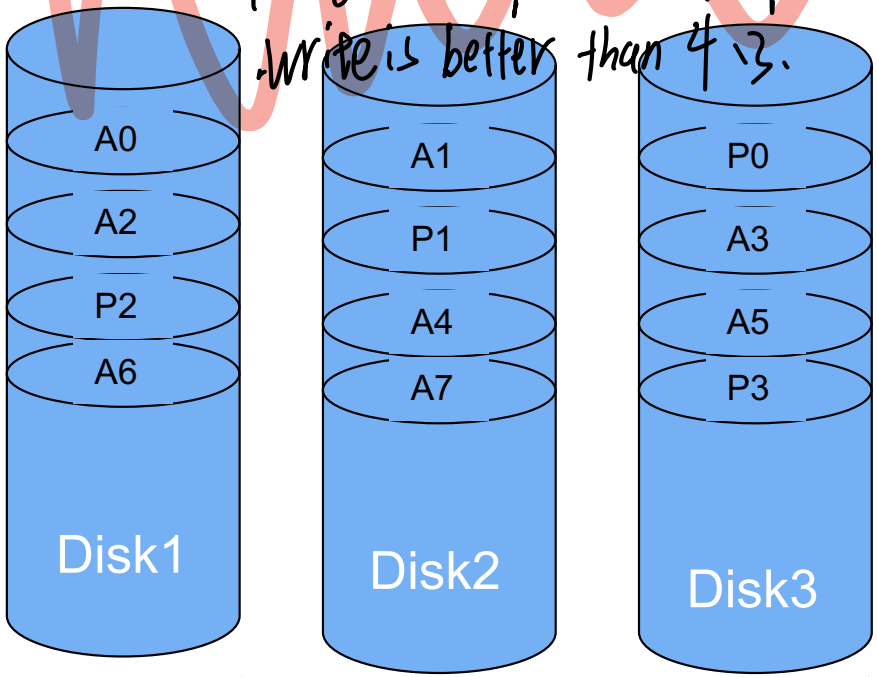
Calculate and update Parity, if anything gets updated in data, the corresponding parity has to be calculated as well. (one of the blocks)

So disk 3 gets more operations than any of those 2 disks.

So the whole write operation gets bottlenecked.

- A means Block (4K or 8K bytes of storage)
- P is parity
- MTTF = Mean Time To Failure

RAID 5 with 3 disks (Block level striping)

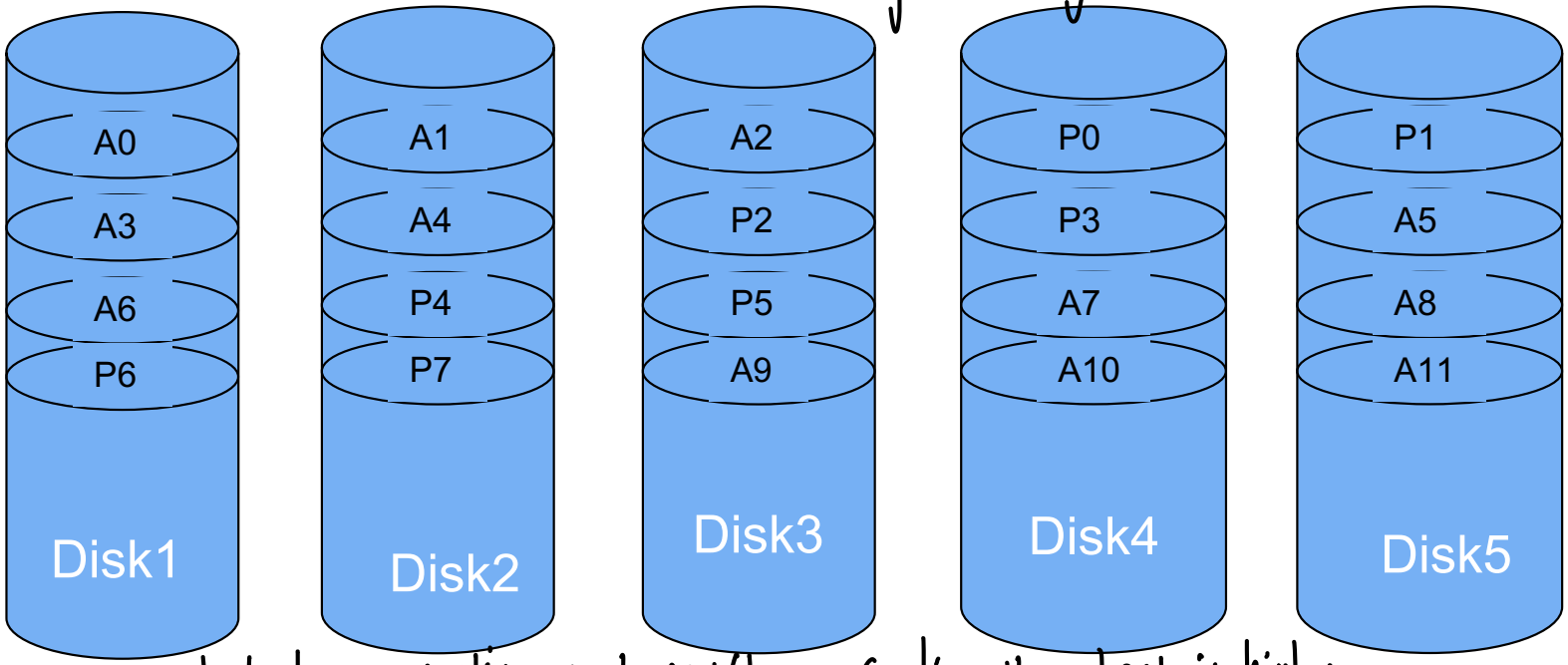


- A0,A1, A2, A3, .. are contiguous blocks of data of a file
- Striping takes place at block level
- Parity blocks are also striped
- Provides higher throughput but slower writes but better than RAID 4 as Parity bits are distributed among all disks and the number of write operations on average equal among all 3 disks.
- MTTF increases substantially (same as RAID3)
- Pi = $A_{2i} \oplus A_{2i+1}$, here $\oplus$ is an exclusive-or operator

- A means Block (4K or 8K bytes of storage)
- P is parity
- MTTF = Mean Time To Failure

RAID 6 (Block level level striping)

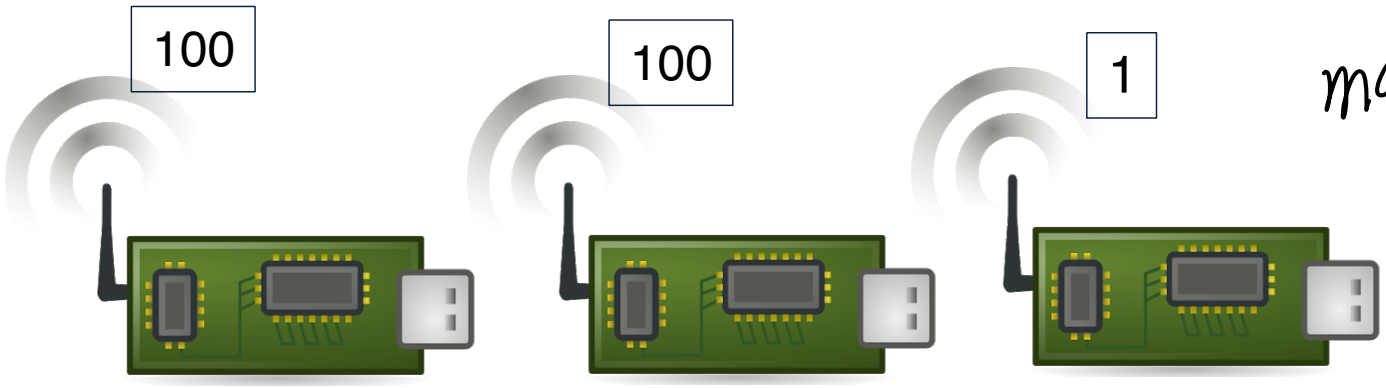
- A means Block (4K or 8K bytes of storage)
- P is parity



- Similar to RAID 5 except two parity bocks used.
- Reliability is of the order of MTTF^3/10
- P0 and P1 are parity blocks for blocks A0, A1 and A2. These are computed in such way that any two disk failures can be safe to recover the data.

Fault Tolerance by voting

Use more than one module, voting for higher reliability



Failvote - Stops if there are no majority agreement

Failfast (voting)- Similar to failvote except the system senses which modules are available and uses the majority of the available modules.

- A 10 module Failfast system continues to operate until the failure of 9 modules where as Failvote stops when 5 modules fail.
- Failfast system has better availability than failvoting (since failvote stops when there is no majority agreement).

fail vote　5个都1个一致 √

4个都3个一致 √

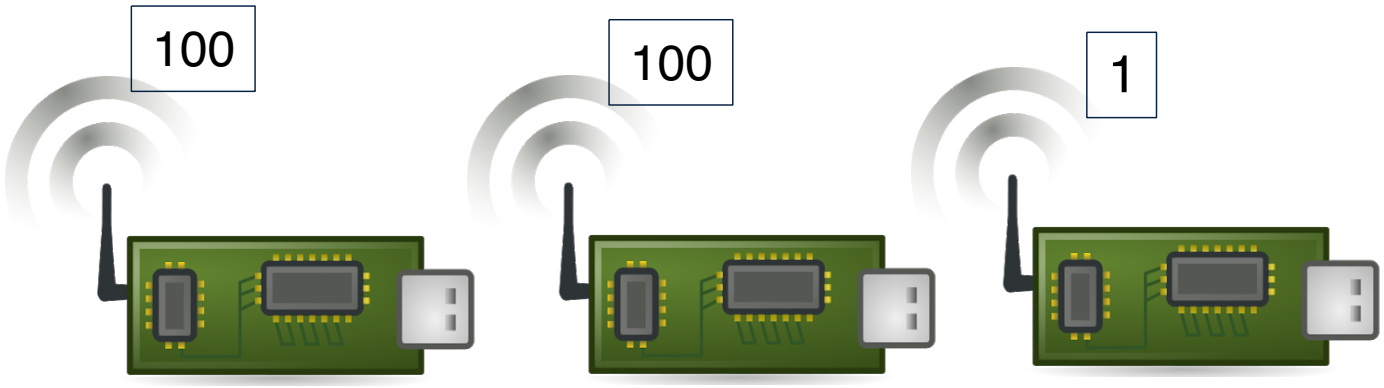
fail fast　系统能感知到哪个module正常运行，只对正常运行的计算进行majority voting

5个仅有两个正常运行 → 这2个进行majority voting　2个必须一致 √

available

Fault Tolerance ...

Failvote needs majority agreement to accept an action (eg, Read/write)

1. 

Accept

2. 100 100 ~~1~~ **Accept**

3. 100 ~~100~~ 1 **Do not accept**

If we start with 10 devices, the system works as long as 6 of them are working. Action is accepted when 6 or more agreeing on the decision.

The moment 5th one fails, system stops as there cannot be 6 devices agreeing

Fault Tolerance ...

In Failfast, we are only concerned of majority among the working ones. We are assuming that we can tell which ones are working. Hence we can continue to operate until 2 working ones and if both agree we can proceed with the action. But if they differ the system stops.

- 0 devices are faulty, we have 10 working and we need at least 6 to agree
- 1 device is faulty, we have 9 working and we need at least 5 to agree
- 2 devices are faulty, we have 8 working and we need at least 5 to agree
- 3 devices are faulty, we have 7 working and we need at least 4 to agree
- 4 devices are faulty, we have 6 working and we need at least 4 to agree
- 5 devices are faulty, we have 5 working and we need at least 3 to agree
- 6 devices are faulty, we have 4 working and we need at least 3 to agree
- 7 devices are faulty, we have 3 working and we need at least 2 to agree
- 8 devices are faulty, we have 2 working and we need both to agree
- 9 devices are faulty, we have 1working and we have to stop as nothing to compare!

Fault Tolerance for disks

Supermodule – Naturally, a system with multiple hard disk drives is expected to function with only one working disk (use voting when multiple disks are working/available, but still work even when only one is available)

ready from three disks if only one is available but I still want to use that disk to write or read.

Supermodule: still keep functioning with one working device or one working disk.

3 devices → failvote → 2 devices to agree (otherwise system fail) ⇒ 3个中有2个挂 系统不能用.

对failfast也一样 → 3个坏2个 → 剩一个不构成majority ⇒ the system will fail.

no majority.

voting

e.g. 2 devices → system fails for 1 failure. C if PC1 device fail) = P
pr(such a system to fail) = P+P MTTF = 1/2P (inverse) = 1/2 × 1/P = 1/2 m
→ 1 device的 MTTF m=10 years
1/2 × 10 = 5

Availability of failvote systems

If there are n events, mean time to the first event = m/n

Using voting system and 2 devices can reduce the MTTF, not a desirable thing.

Consider a system with modules each with MTTF of 10 years

Failvoting with 2 devices: → system down for 1 device failure

MTTF = 10/2 = 5 years (system fails with 1 device failure)

Failvoting with 3 devices: → system down for 2 devices failure.
3dev → 1st failure → 2nd failure

MTTF = 10/3 for the first failure + 10/2 for 2nd failure = 8.3 years.
MTTF for the 1st failure; MTTF of the 2nd failure

Lower availability for higher reliability (multiple modules agreeing on a value means that value is more likely to be accurate/reliable)

1st MTTF + 2nd MTTF = MTTF of full system.
one module ← m/n
= 10/3 + 10/2 = 8.3

But, but, but…. cannot we have both??

MTTF n: # of available devices. (3 for 1st, 2 for 2nd)

current

Fault tolerance with repair

MTTR 2hours << MTTF 修复时间相对长. 平均遇到故障时间多长.

With repair of modules: the faulty equipment is repaired with an average time of MTTR (mean time to repair) as soon as a fault is detected (Sometimes MTTR is just time needed to replace)

Typical Values for recent disks:

MTTR = Few hours (assuming we stock spare disks) to 1 Day

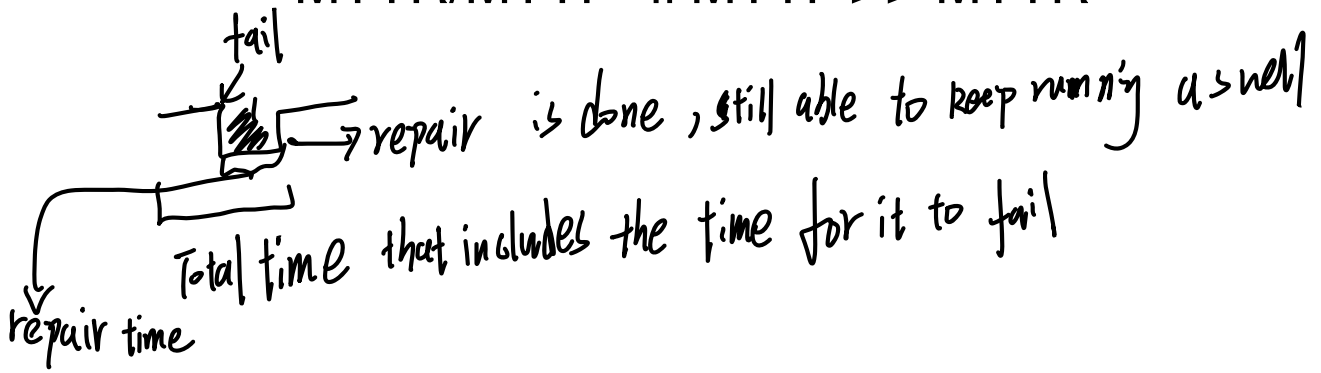
MTTF = 750000 hours (~ 86 years) [hard fault]

Probability of a particular module is not available

= MTTR/(MTTF+MTTR)

≅ MTTR/MTTF if MTTF >> MTTR

repair is done as soon as fault is detected



Fault tolerance of a supermodule with repair

Probability that (n-1) modules are unavailable, $= P_{n-1} = P \times P \times \cdots P = P^{n-1} = \left(\frac{MTTR}{MTTF}\right)^{n-1}$

Probability that a particular i^{th} module fails, $= P_f = \frac{1}{MTTF}$

Probability that the system fails with a particular i^{th} module failing last =

$P_{n-1} \times P_f = \frac{1}{MTTF} \times \left(\frac{MTTR}{MTTF}\right)^{n-1}$

Probability that a supermodule fails due to any one of the n modules failing last, when other (n-1) modules are unavailable

3 devices, 3 different ways that the last one fails

prob that supermodel system fails is $n \times \frac{1}{MTTF} \times \left(\frac{MTTR}{MTTF}\right)^{n-1}$

Communication reliability

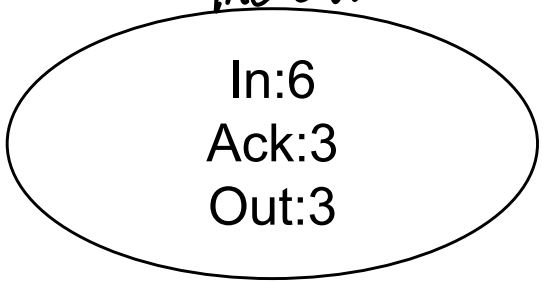
ensure the communication between them is reliable

Out = #messages sent

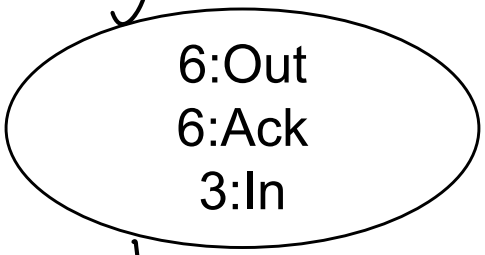
In= #messages received

Ack = #acknowledgements

the data that has been transmitted and received directly is ensured.



node A



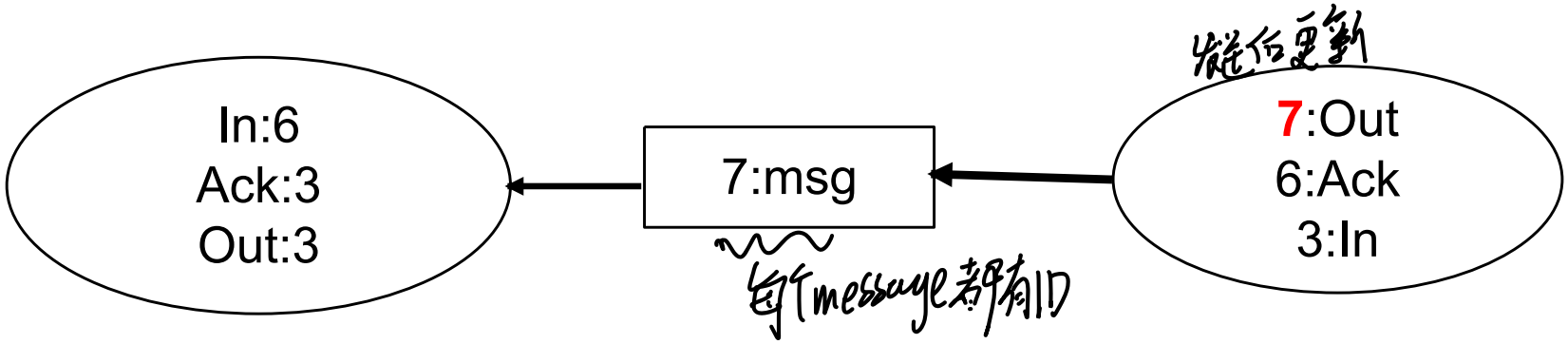
node B

Communication reliability

Out = #messages sent

In= #messages received

Ack = #acknowledgements



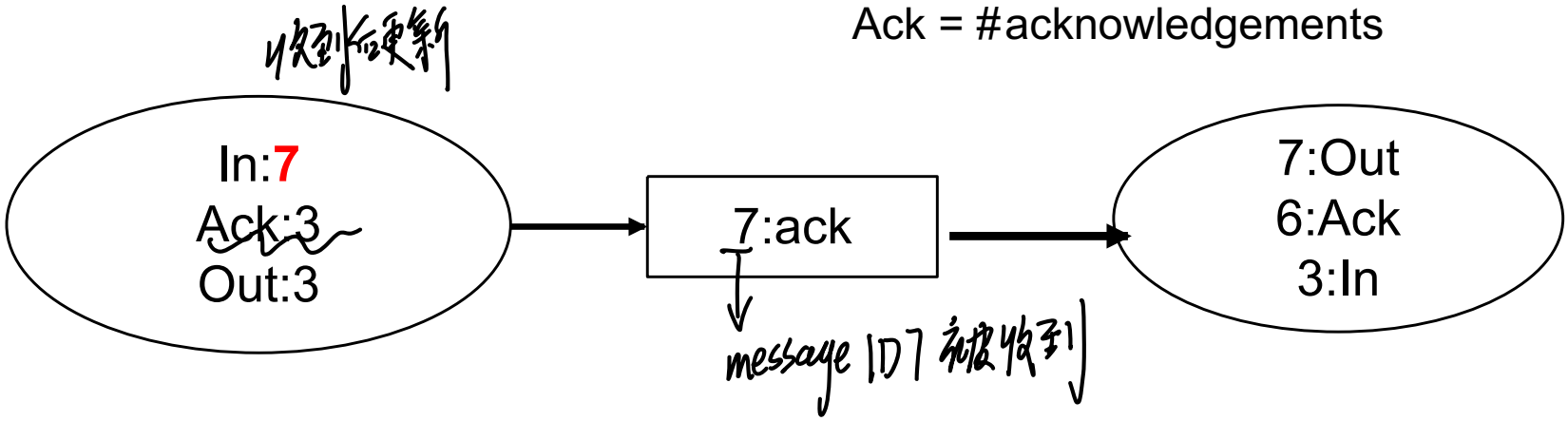
Communication reliability

Out = #messages sent

In= #messages received

Ack = #acknowledgements

Reliable message passing



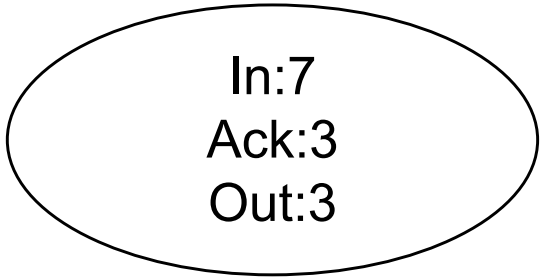
Communication reliability

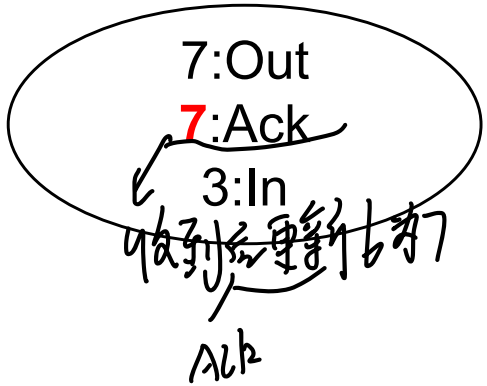
Reliable message passing

Out = #messages sent

In= #messages received

Ack = #acknowledgements

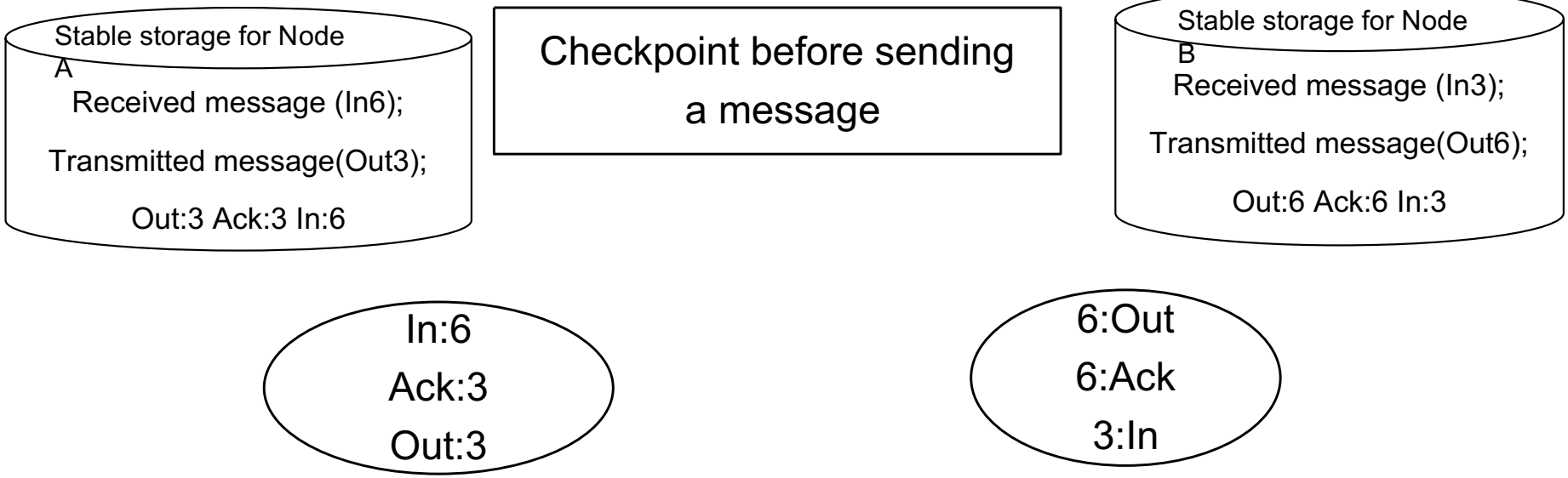




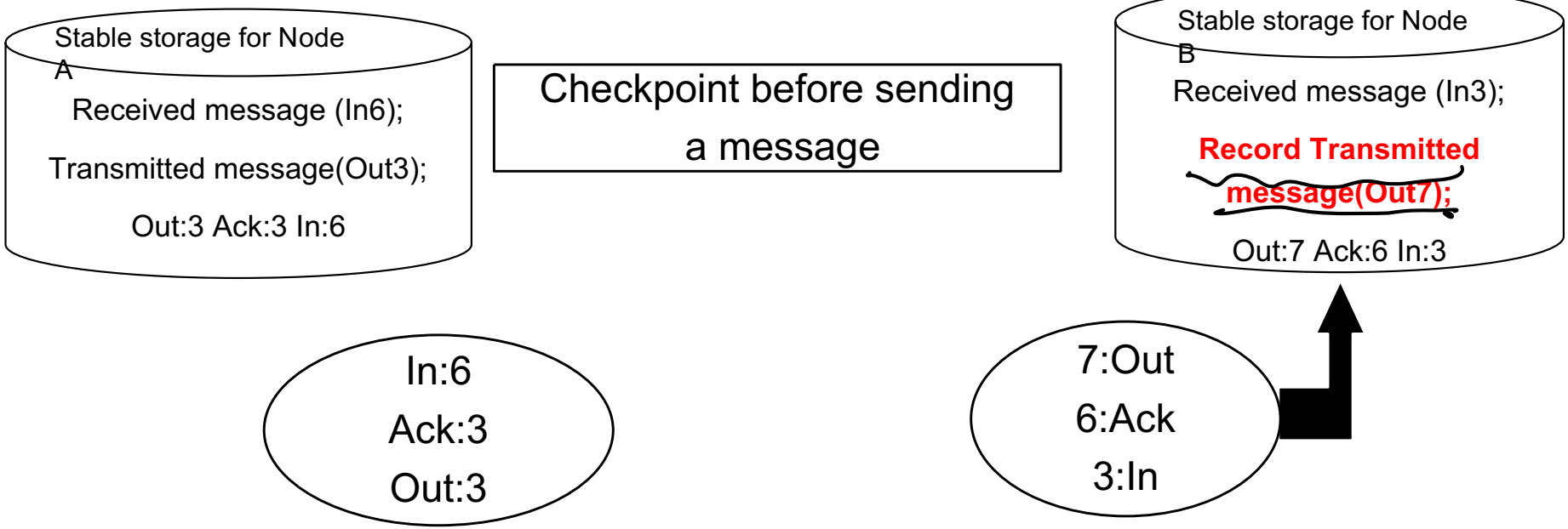
to make sure each message doesnot get lost → stable communication

Stable storage → fault tolerance (data not get lost easily)

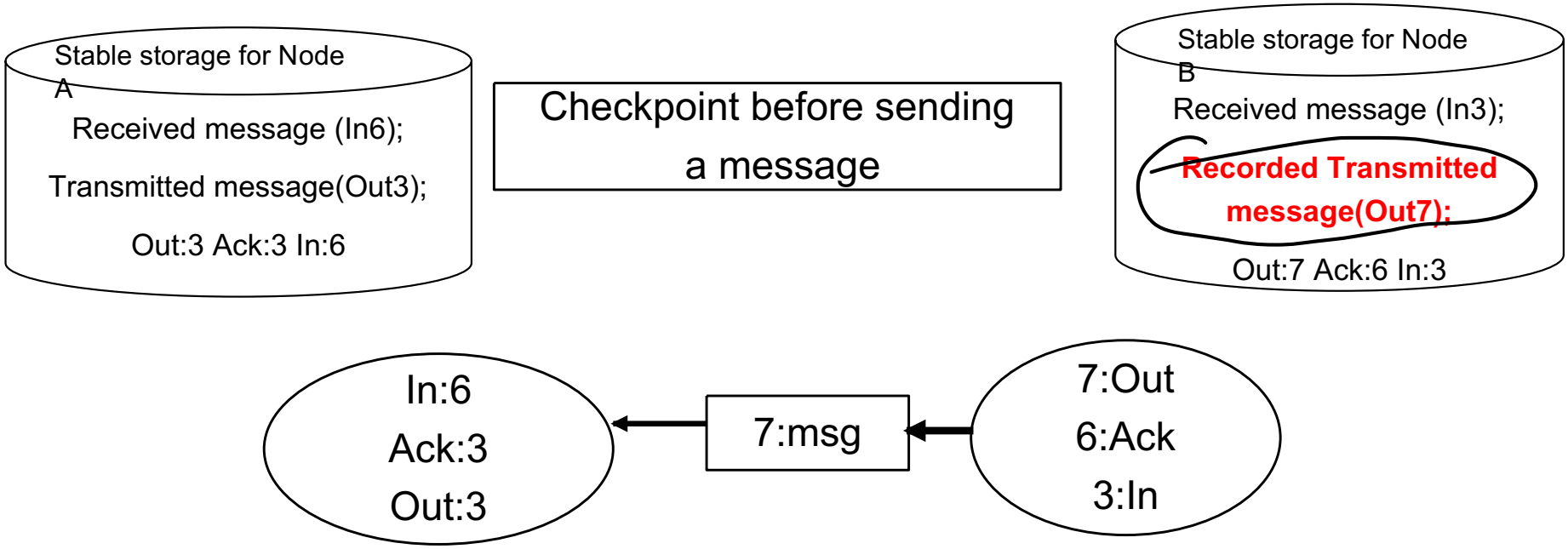
Communication reliability



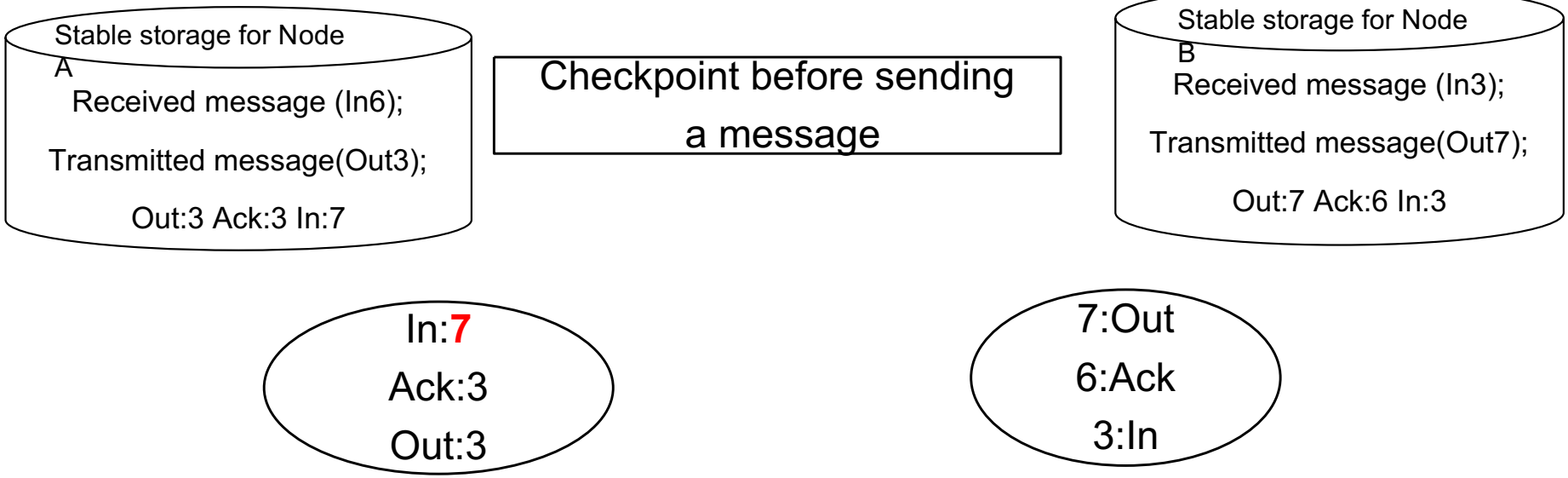
Communication reliability



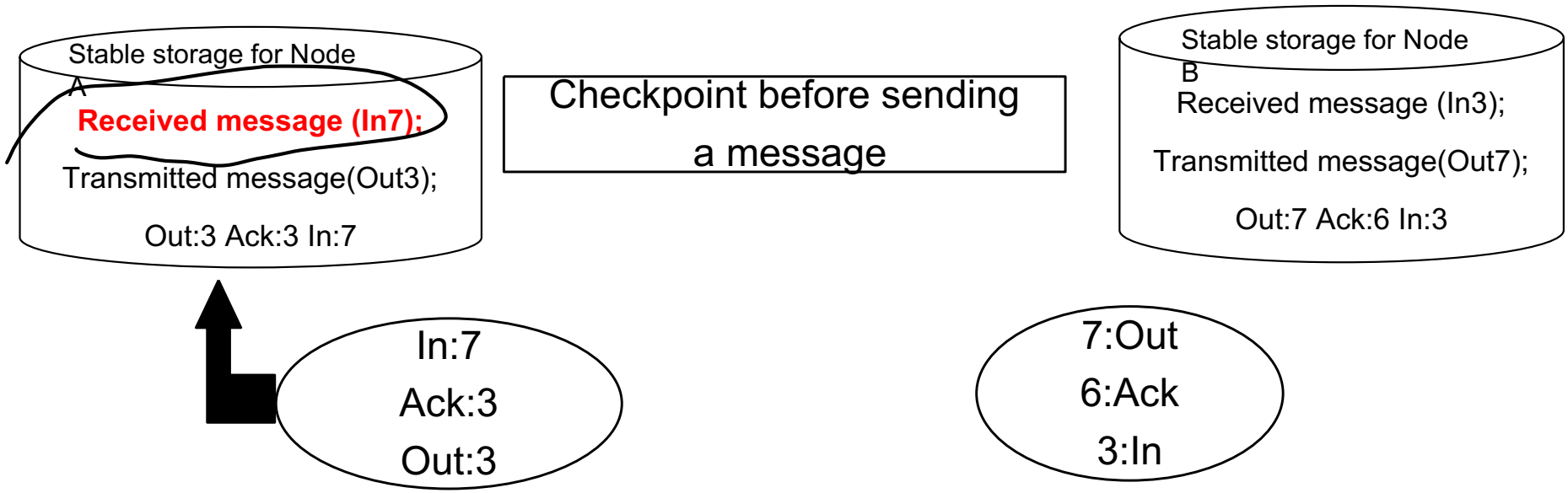
Communication reliability



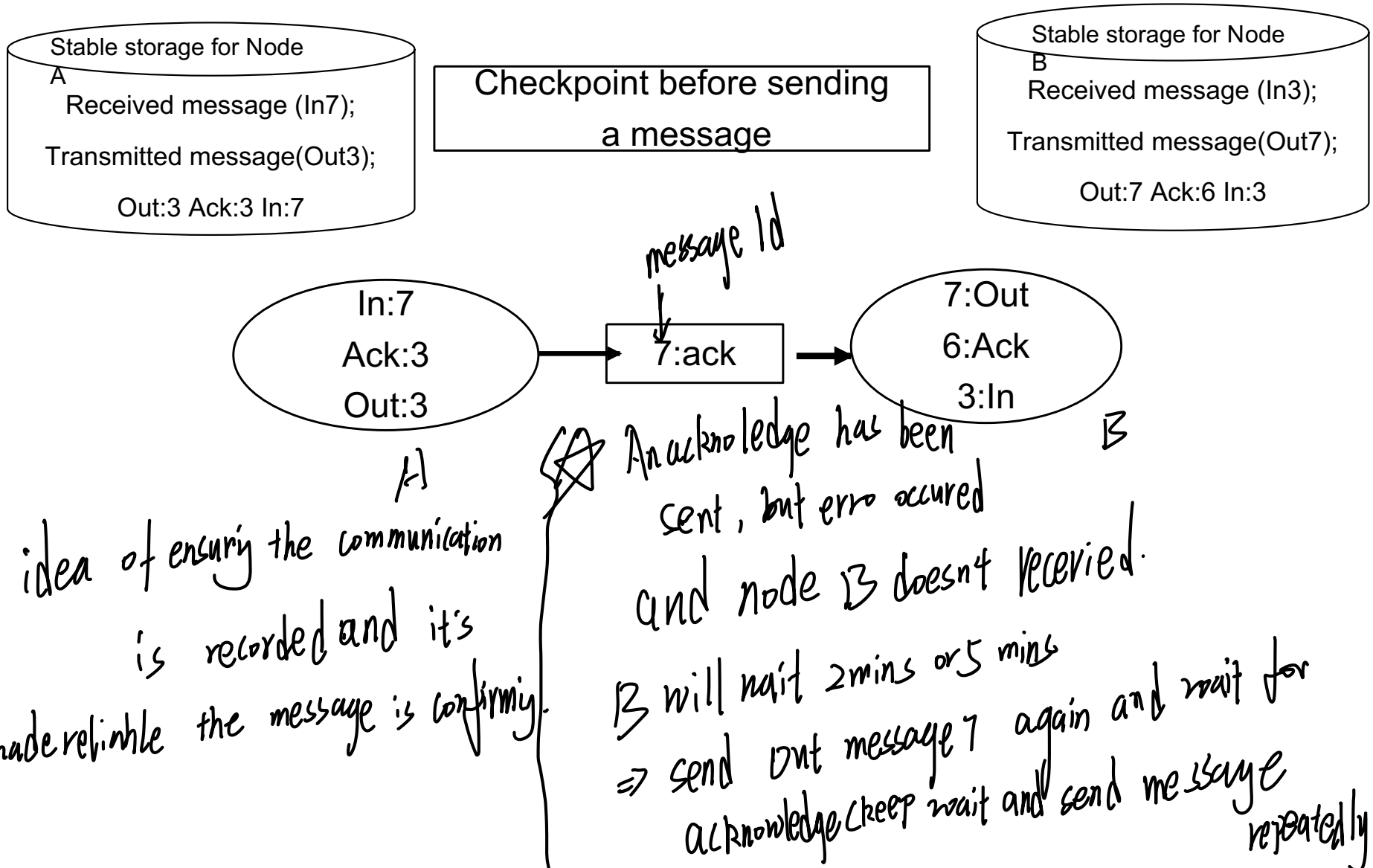
Communication reliability



Communication reliability

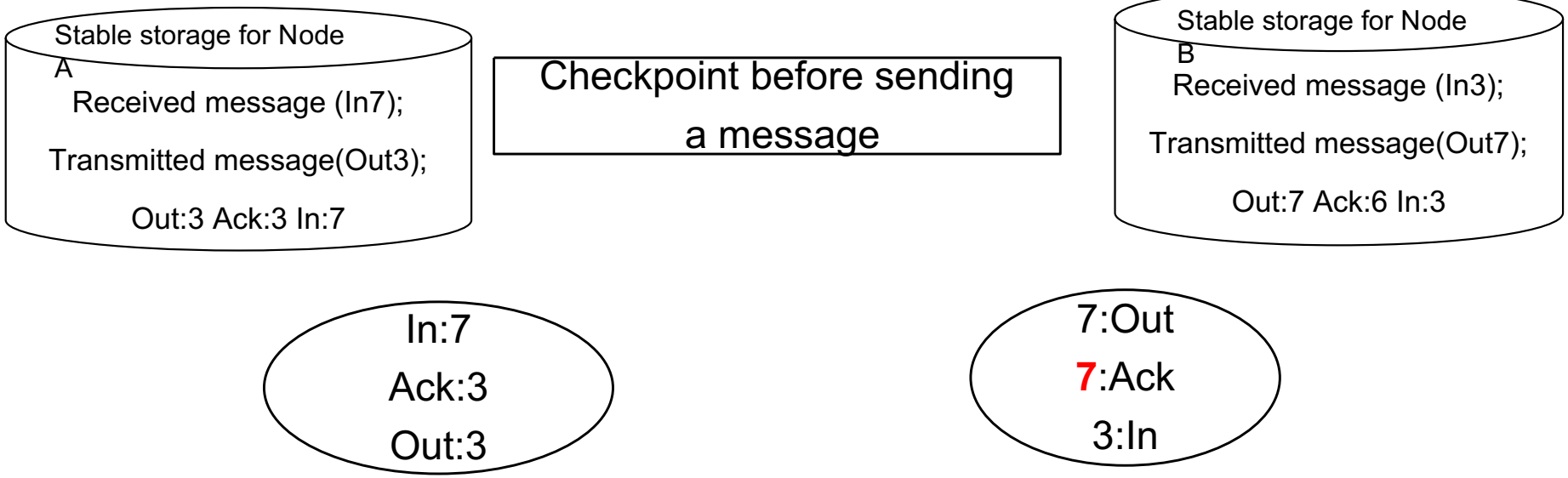


Communication reliability

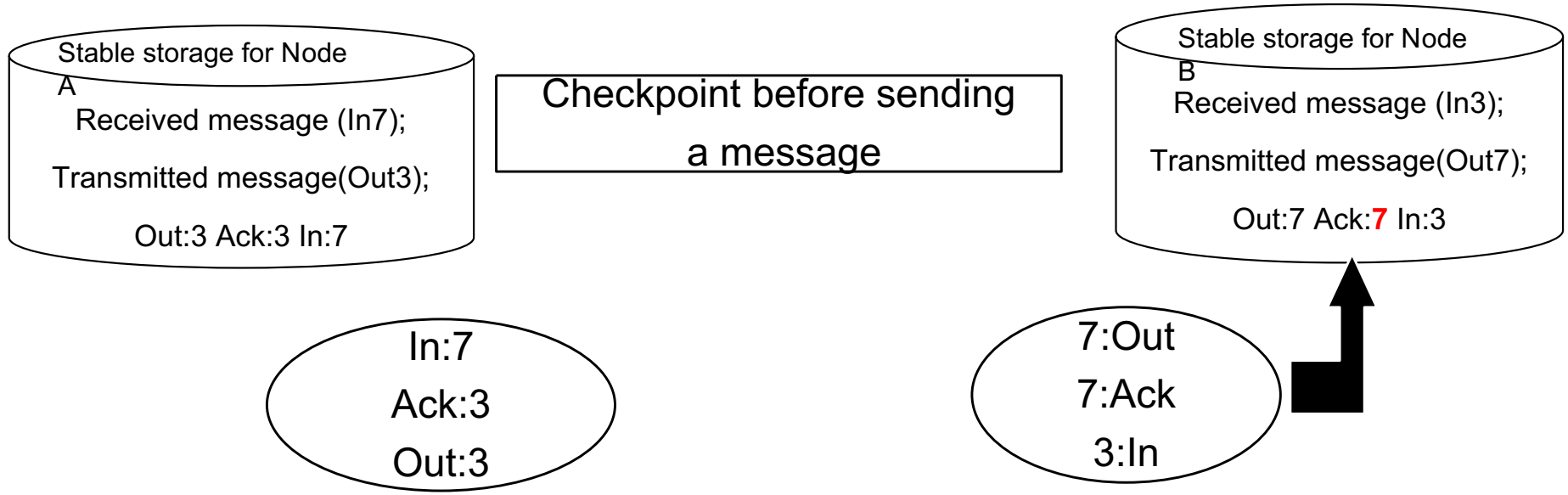


as long as it doesnot receive the acknoledge.
once receive acknowledgment => message is done completely that it has been received by the other node correctly.

Communication reliability

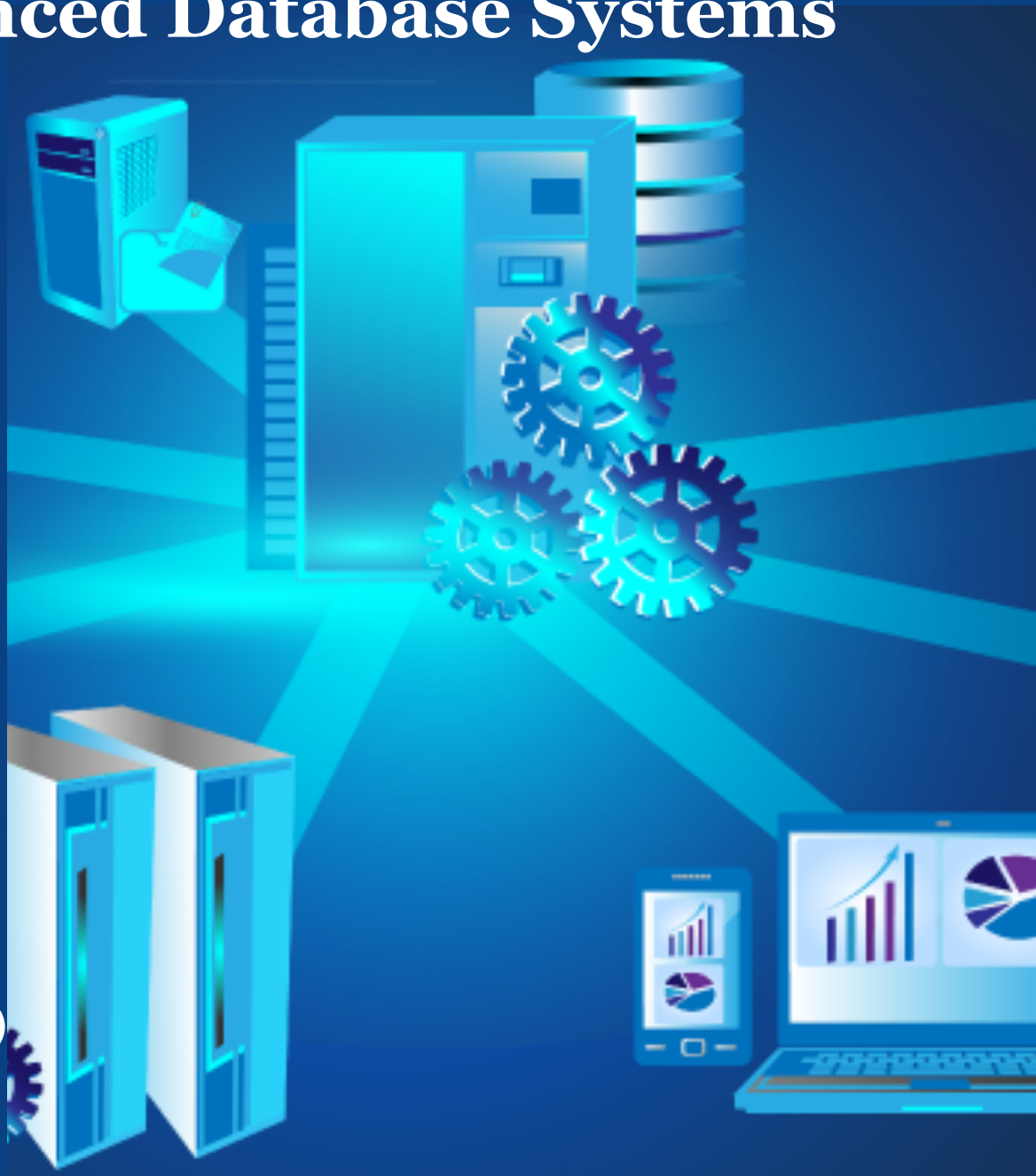


Communication reliability



THE UNIVERSITY OF MELBOURNE

COMP90050 Advanced Database Systems



Semester 2, 2024

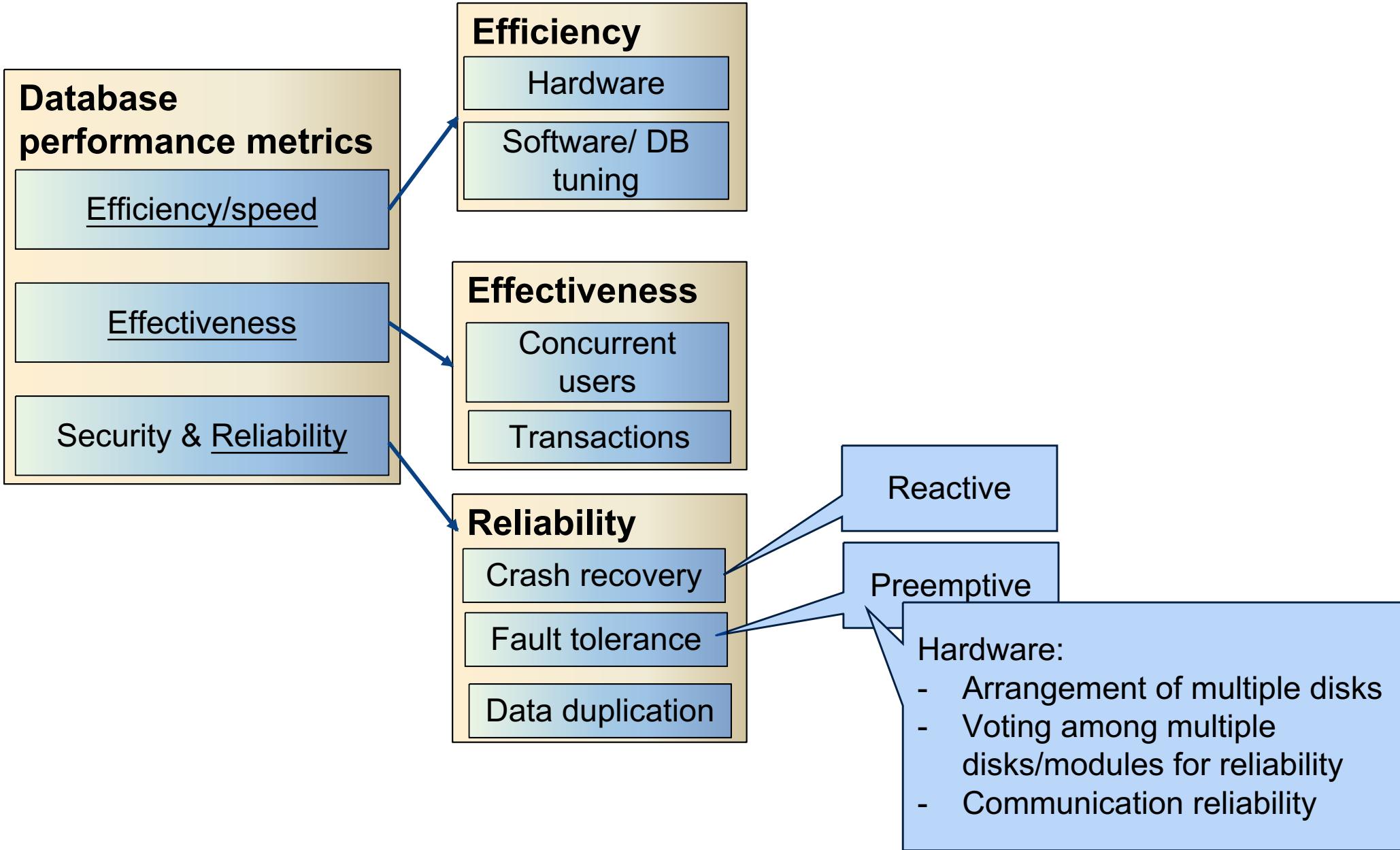
Lecturer: Farhana Choudhury (PhD)

Live lecture – Week 8

Presentation details

- Announcement
- Register asap
- Check FAQs

Core Concepts of Database management system



Fault Tolerance

The property that enables a system to continue operating properly in the event of the failure of some of its components.

We have covered

- Statistics crash course
- Lifecycle of a system
- Different fault tolerance techniques
- Communication reliability

P(A and B) = P(A)*P(B) assuming A and B are independent events.

P(A or B) = P(A) + P(B) - P(A and B)

= P(A) + P(B) - P(A)*P(B) [Assuming A and B are independent]

P(A) + P(B) [if P(A) and P(B) are very small]

This is not a math subject!

Understanding the concepts (e.g., the event that both disks fail at the same time is statistically 'how much' less likely than a single disk fail)

Fault tolerance by RAID

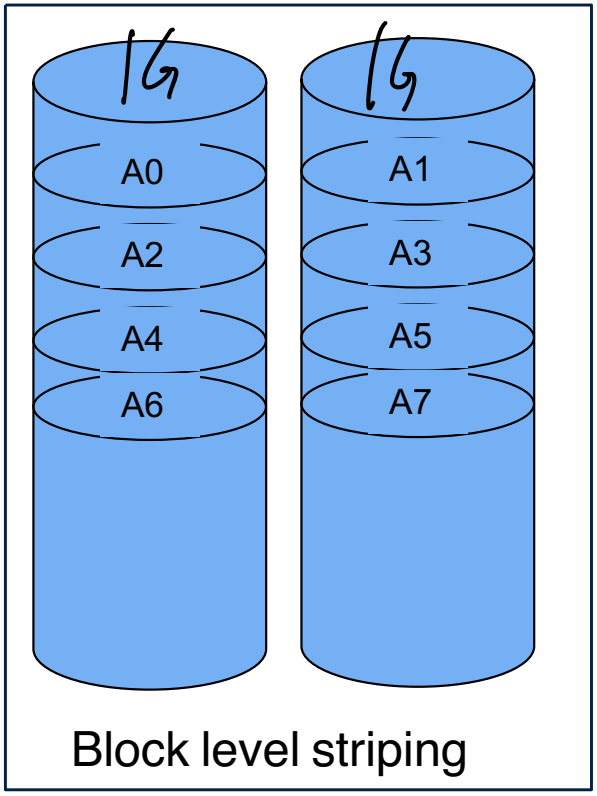
Redundant Array of Independent Disks – different ways to combine multiple disks as a unit for fault tolerance or performance improvement, or both of a database system

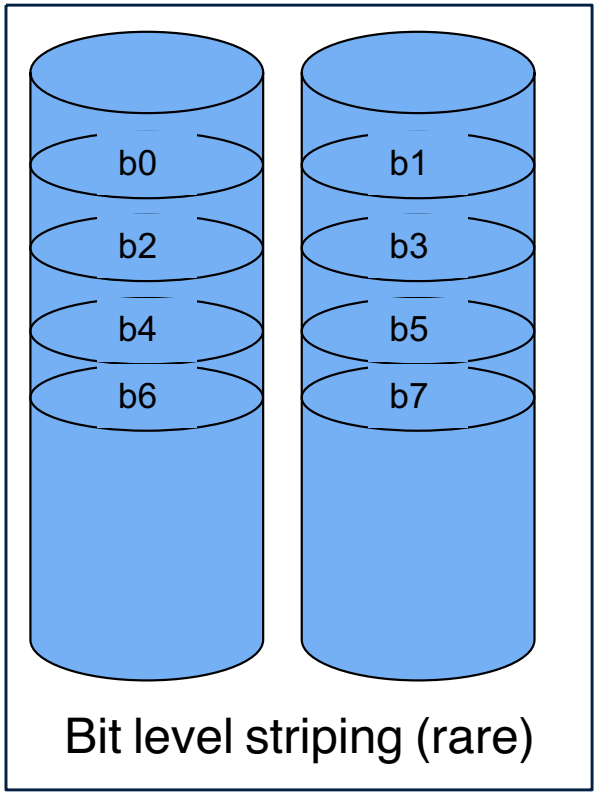
Choosing the suitable RAID level

The factors to consider:

- Reliability
- Performance
- Storage utilization
- Price/number of disks
- Additional overhead?

Example summary: RAID 0 and RAID 2





Storage utilization?

Number of disks needed?

Overhead difference?

Provides balanced I/O of disk drives

Provides higher throughput (~doubles)

Any disk failure will be catastrophic

MTTF reduces by a factor of 2

Higher throughput at the cost of increased vulnerability to failures

Calculation of MTTF values – in tutorials

Comparison:

RAID 0 与 RAID 2 一样

① Storage utilization: 对上面所说的，但有时候有 mirror 配置 (RAID 1). 2 blocks 只储存一半的 pure info. (100% storage utilization vs 50% storage utilization)

② # of disks needed: (RAID 0 VS RAID 2 一样)

③ overhead difference? Stripping in bit level cost more than strippy in block level. In 0 scenario: When we write more dividy or partitiony overhead involved. When we ready, we need to merge data from two disks more frequently to form meaningful data that can be displayed to users (much more overhead on RAID 0)

④ reliability performance: Any disk failure will be catastrophic (if one block failure, either has same then the whole system will not be able to provide any sort of data. (data will become meaningless.)

⑤ System throughput: we can read from two disks in parallel, higher throughput => almost double higher

⑥ balanced I/O of disk drives: the same # of write and read operations happen at both disks. (No bottleneck disk that we have to wait for disk write finish to move to the other one)

RAID – exercise

Which of the following RAID setting can safely recover data even for two disk failures?

RAID 5 with 3 disks — 0%

RAID 1 with 3 disks ✓ — 0%

Three replications ⇒ two failure. Still have third disk (can do read and write

None of the above — 0%

Time for a poll - Pollev.com/farhanachoud585

If somthy failed can I still recover data?

RAID is commonly used for fault tolerance system.

use RAID 1 to store research data (people's GPS data)

☆ data: As people move, their phone continously send GPS to the server. over someting that's big amount of data get collected. ☆

why?: data is important, if I lose it I cannot collect them again (S3 I need) mirror to duplicate dataset). Just guarantee I don't lose it.

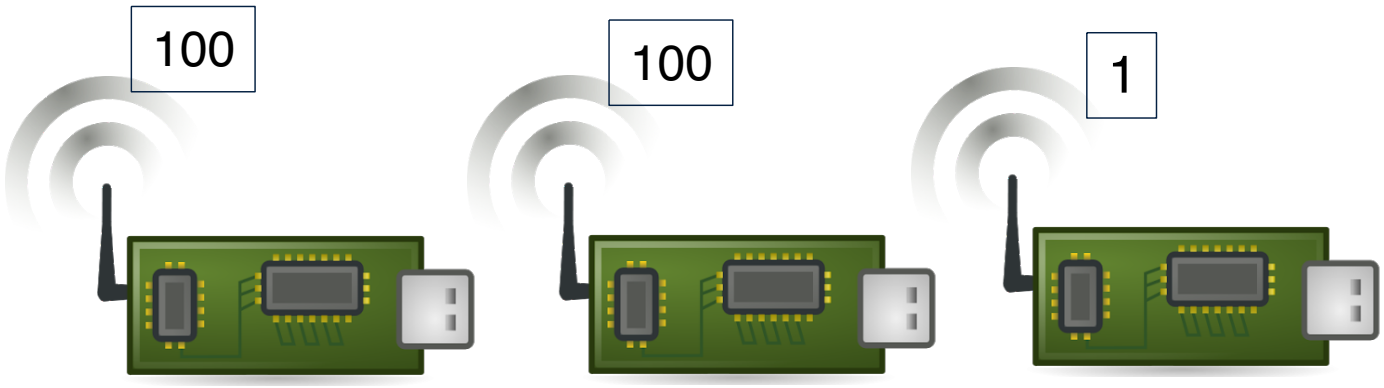
only for research purpose, only I don't care anythiy about read throughput. Not munyusers or somethy really needsto read faster.

how much reliability or better throughput ? ⇒ 不同RAID levels

Fault Tolerance by voting

Check if the data one got is correct

Use more than one module, voting for higher reliability



Failvote - Stops if there are no majority agreement

Failfast (voting)- Similar to failvote except the system senses which modules are available and uses the majority of the available modules.

Supermodule – A system with multiple modules that use voting when multiple modules are working/available, but still work even when only one is available

→ if only one device can run, the system can still keep operating.

Fault Tolerance by voting

In a Failfast system, which of the following cases we can accept an action?

failvote ⇒ check the total # of devices the do majority voting

for for failfast 只看 available (working) devices

Total number of devices	No. of working devices	No. of agreeing devices	Accept?
10	6	4	√
10	6	3 (3 is not a majority of 6)	X
10	5	3	√
5	5	3	√
5	4	2	X
5	2	2	√
5	1	-	X

10

Fault Tolerance by voting

In a Failfast system, which of the following cases we can accept an action?

Total number of devices	No. of working devices	No. of agreeing devices	Accept?
10	6	4	y
10	6	3	n
10	5	3	y
5	5	3	y
5	4	2	n
5	2	2	y
5	1	-	n

11

Communication reliability

- Why important for databases? for database ? why we learn that?

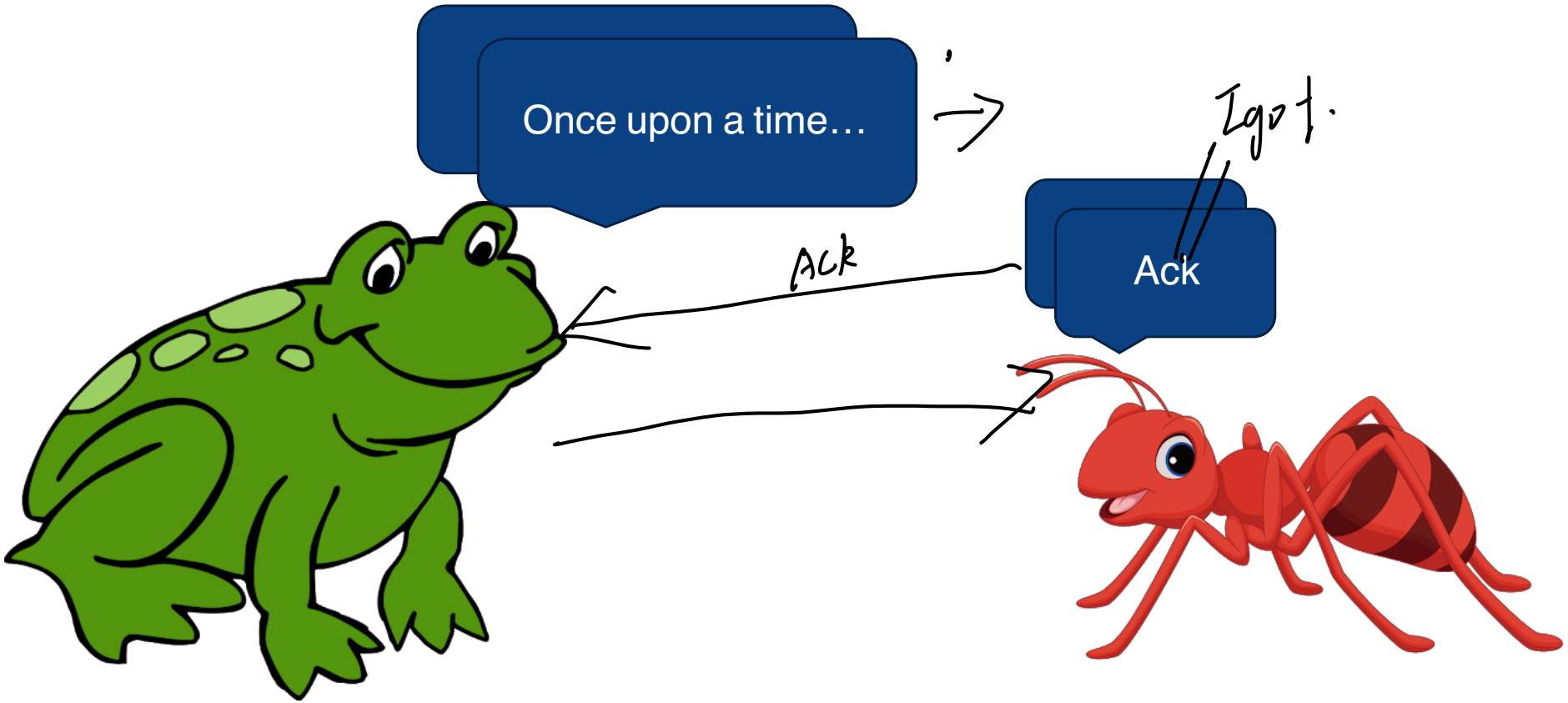
☆ the nodes of distributed database need to communicate with each other,

also : Input in a database does not only from users, but also come from automated or other systems. If input comes ⇒ some acknowledgement or message passing involved ⇒ the message or the input has been received correctly. check it

additionally: many functionalities of database need ensure the reliability of communication query

for consistency: insertion, updates. (communication and messages need to be passed)

Communication reliability



13

Communication reliability

If acknowledgement not received -

ensure the info sent between DB can be correctly received.

☆ What if the knowledge has not been received?: It will assume that the other party has not received the message ⇒ then it will keep trying the party

→ Counter here to count how many attempts it tried

14

Core Concepts of Database management system