SYSTEM DESIGN - 1

17/8/2022

del.icio.us → bookmarking

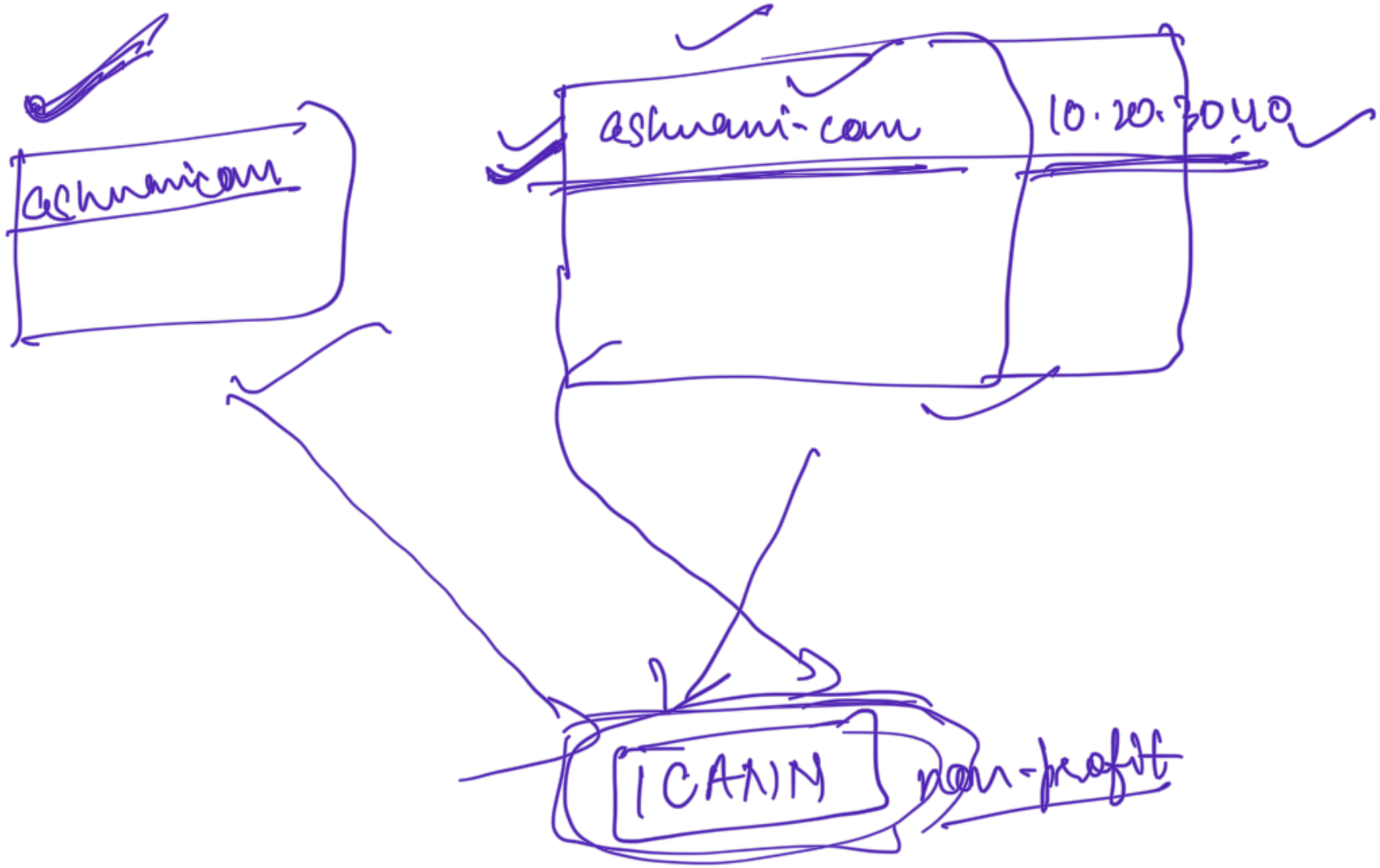
1. addbookmark(user_id, site-url)
2. getbookmarks(user_id)

Internet
↓
IP address

Human
↓
names
(domain name)

Godaddy, namecheap etc

ashwani.com


ashwani.com
ashwani.com
10.20.30.40
ICANIN
non-profit

Browser
DNS
IP
domain
domain -> IP

name server

who maintains DNS?

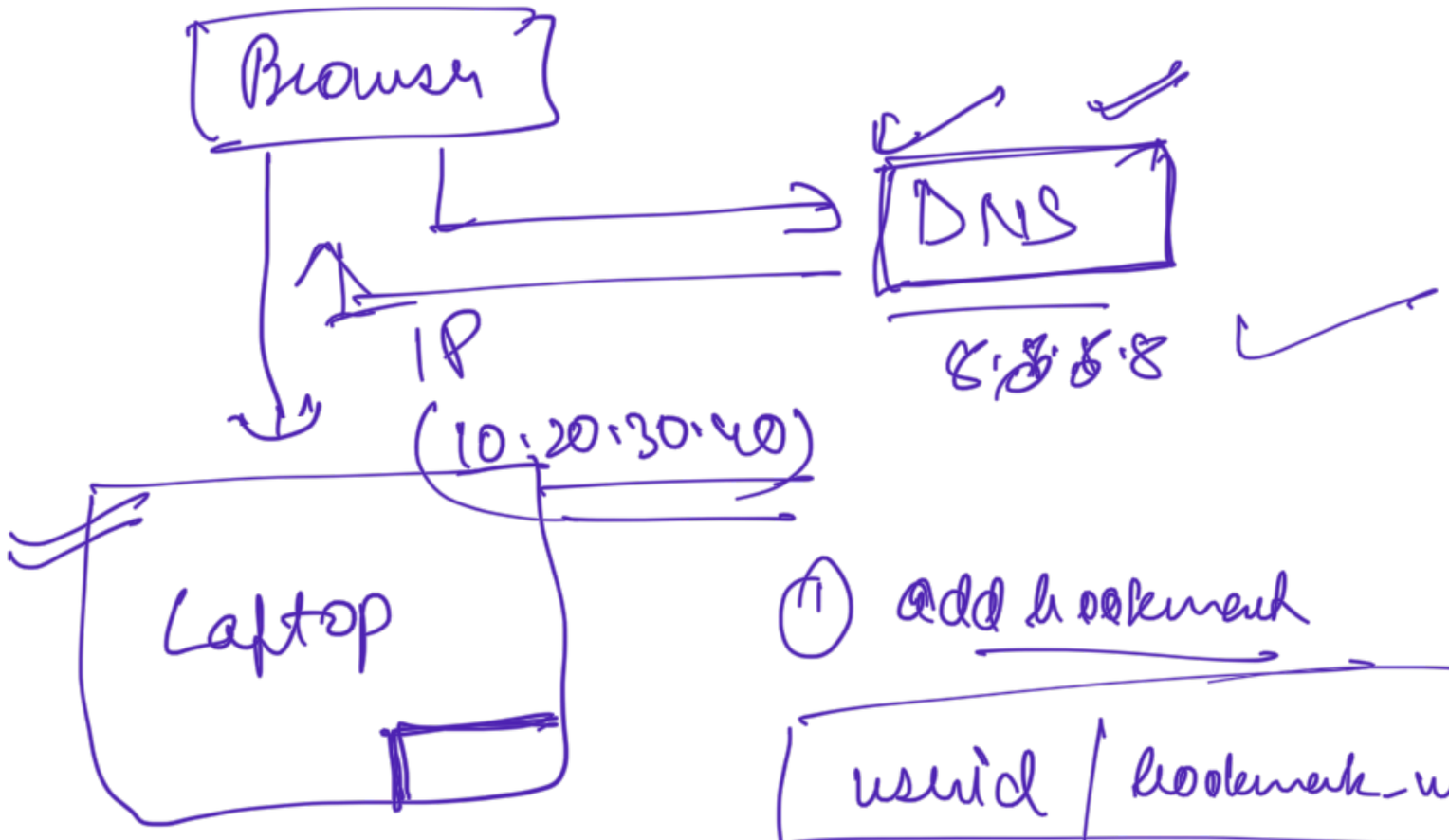
1. ISP
2. google, facebook (big companies)

8.8.8.8
8.8.4.4

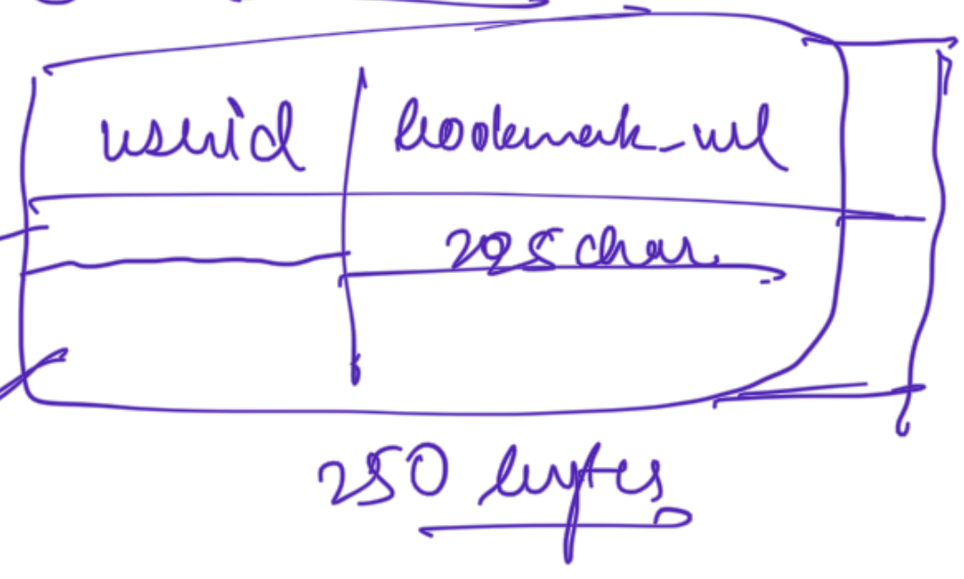
google.com

scaler.com

chat.scaler.com



① add bookmark



* 1 million new bookmarks/day

data saved / day = 250 bytes × 10^6

= 250 × 1000 × 1000 bytes

8 bits = 1 byte
~1000 byte = 1KB
~1000 KB = 1MB
~1000 MB = 1GB

250 KB

250 MB

= 250MB

4 days → 1 GB

40 GB space

No of days I can survive = $\frac{40\,GB}{1\,GB} = 4 \times 40$ days

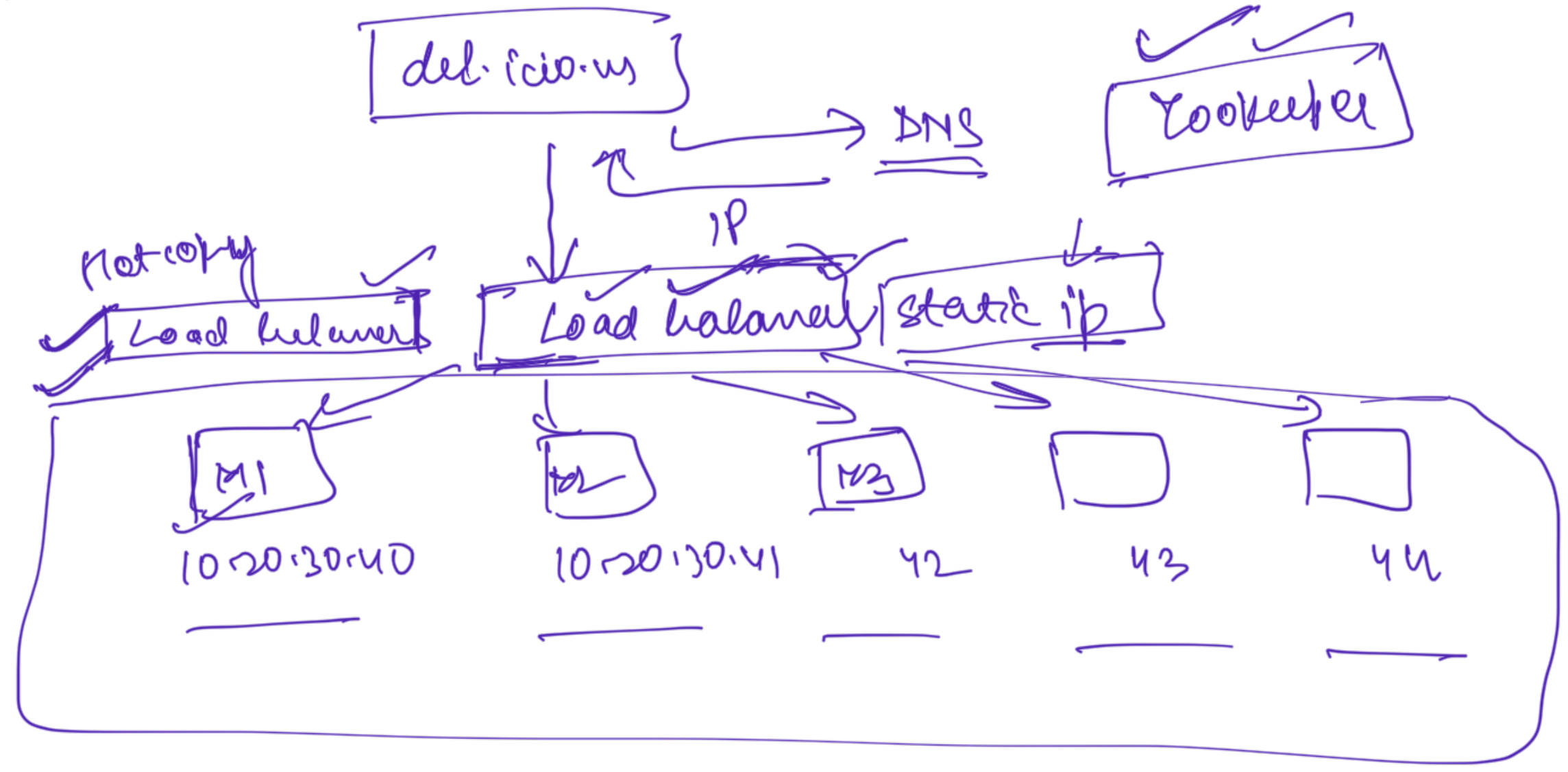
= 160 days

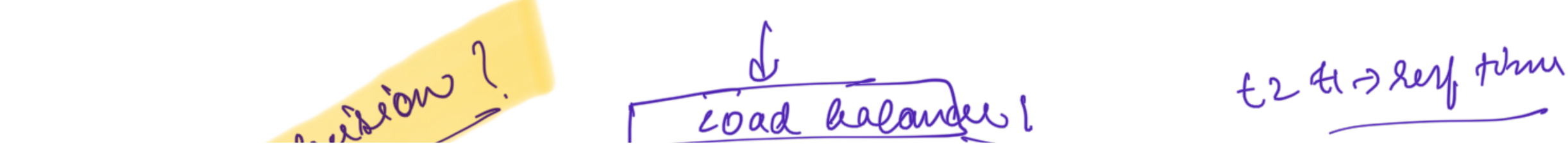
Solutions

① bigger laptop → vertical scaling.

↳ 100 GB

② multiple laptops → horizontal scaling


deliveroo
DNS
Zookeeper
IP
Not copy
Load balancer
Load balancer
static ip
M1
M2
M3
10.20.30.40
10.20.30.41
42
43
44


Load balancer
t2 t1 → ref time

How LB takes decision

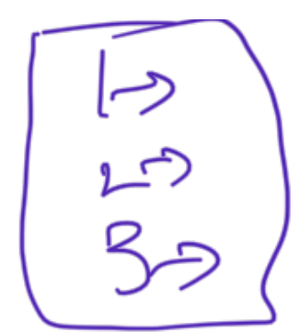
1. Round robin ✓
2. weighted RR

M1	16 GB	8 core	1 : 2
M2	32 GB	16 core	

3. Response time
4. Least load factor ✓

How LB detects machine health

1. Health check ⇒ LB is pulling data
2. Heartbeat ⇒ machines push the data



① getbookmarks

* divide using userid => shard key

Parameters

① LB should know which machine should we send the data

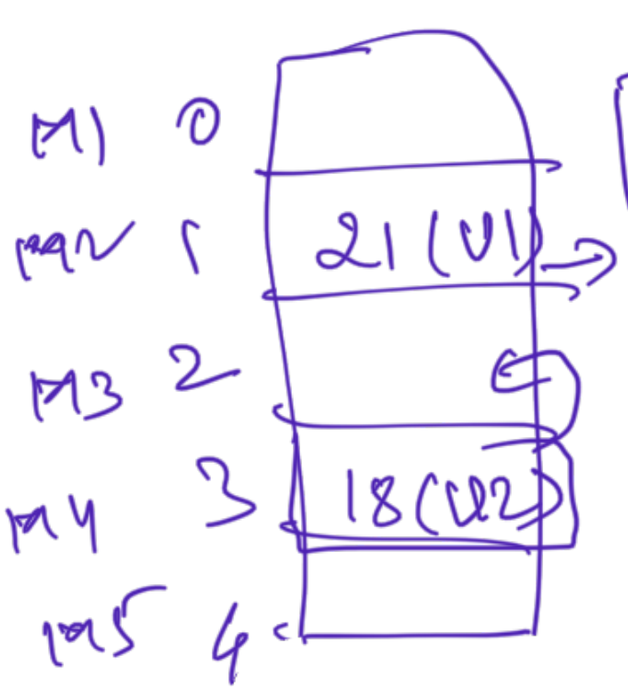
② req should go to 1 machine for a user.

③ adding / removing machine is easy.

✓ Solutions U1 → M1, U2 → M3 X

① map (table) — millions of users → table size is a mgjr issue.

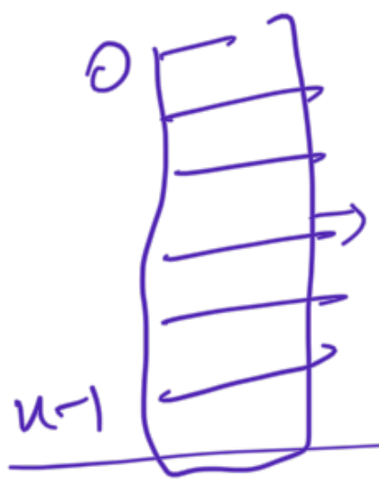
② hash & modulo —



hash fn → uid % no of machines (5)

U1 → user id → 21 % 5

U2 → 18 % 5 ⇒ 3



hf(uid)

n

U1 ⇒ 21 % 4 ⇒ 1

U2 ⇒ 18 % 4 ⇒ 2

---

(3) Consistent hashing

uuid

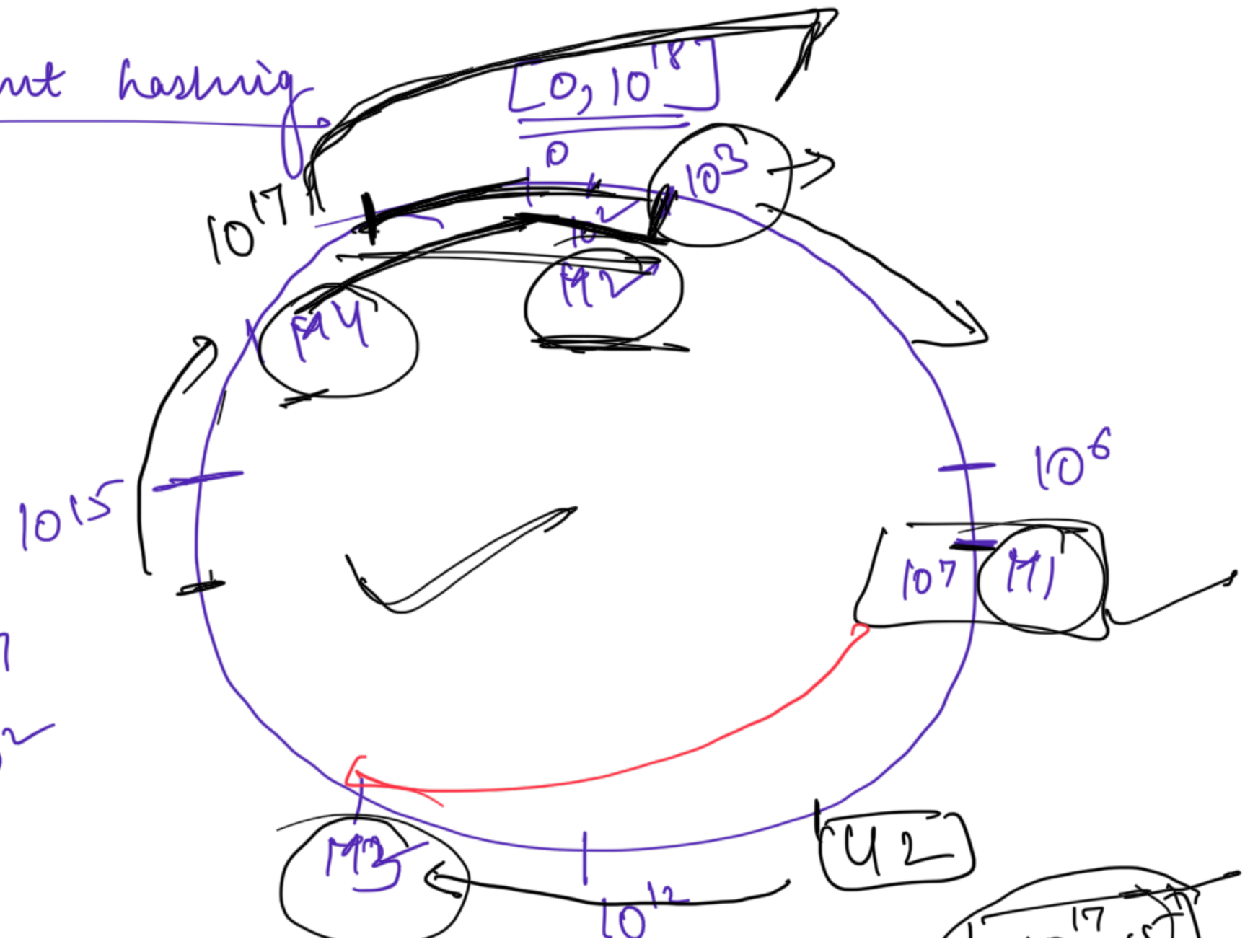
H1 → 10^3

0 > 10^18

M1 H2 → 10^7

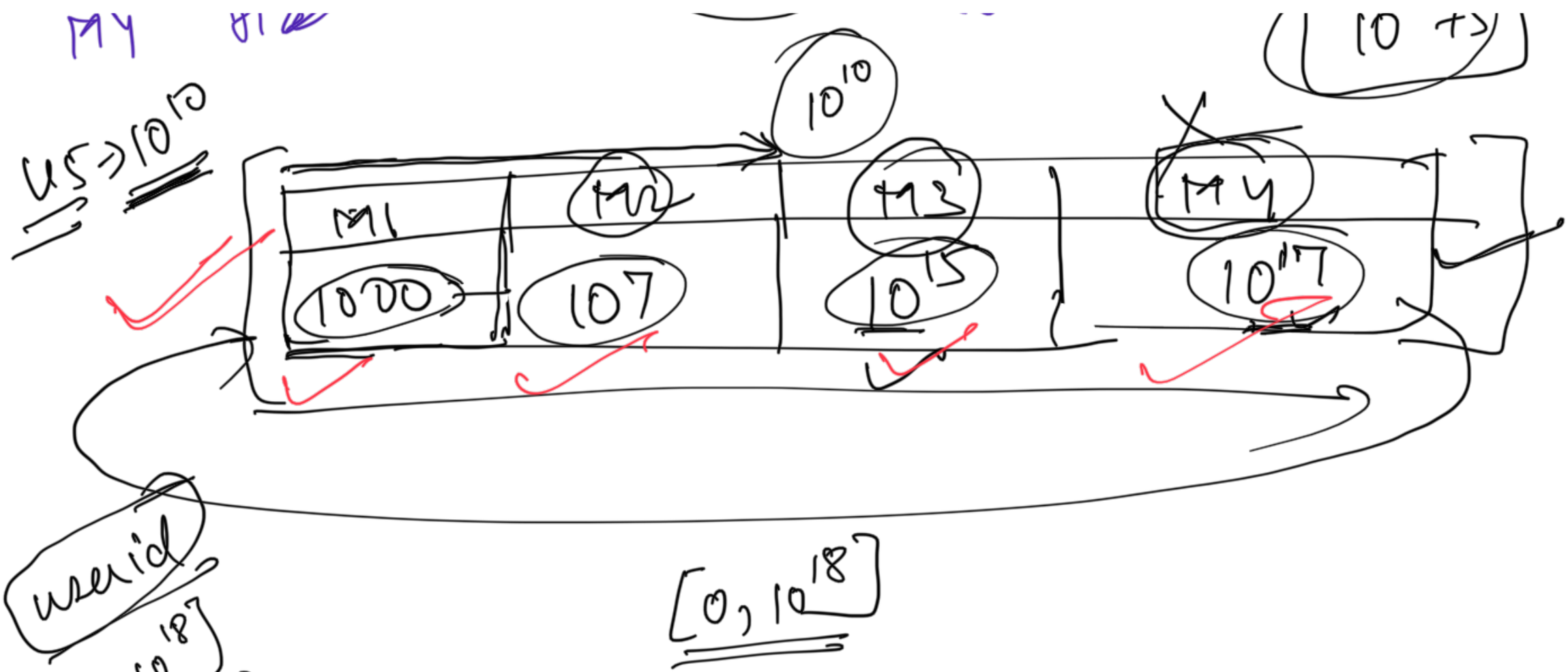
M2 H2 → 10^2

M3 H2 → 3



M4 M1

US => 10^{15}

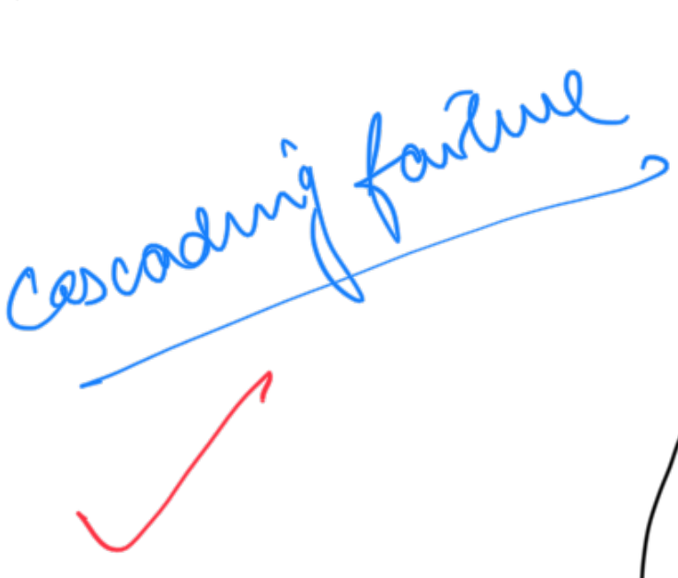


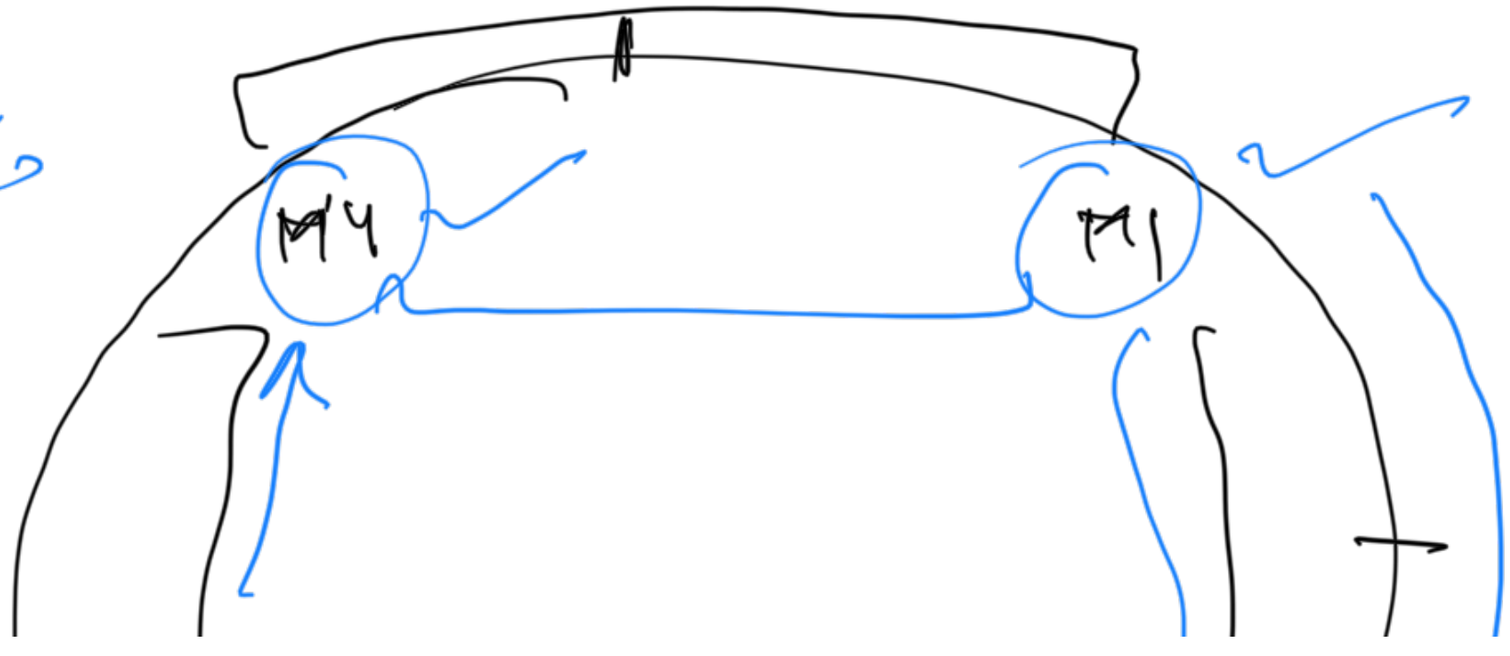
userid

$[0, 10^{18}]$

$[0, 10$

$[0, 10^{18}]$





H1(user_id)

M1

H2(machines) = 1000

H3(machines) = 10^5

H4(rise) = 1000

= 10^{12}

H5

M3

M1

M2

M1

M2

M1

M2

M1

M3

M1

M2

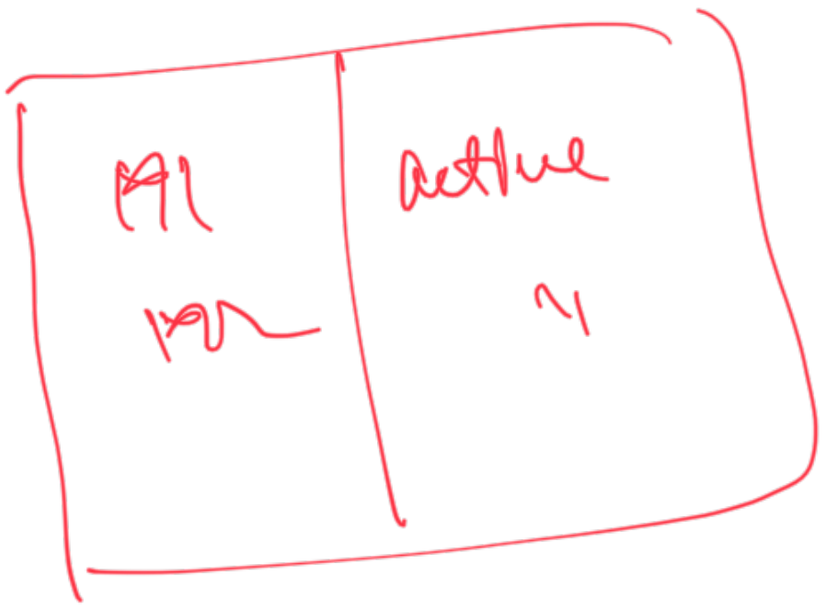
M1

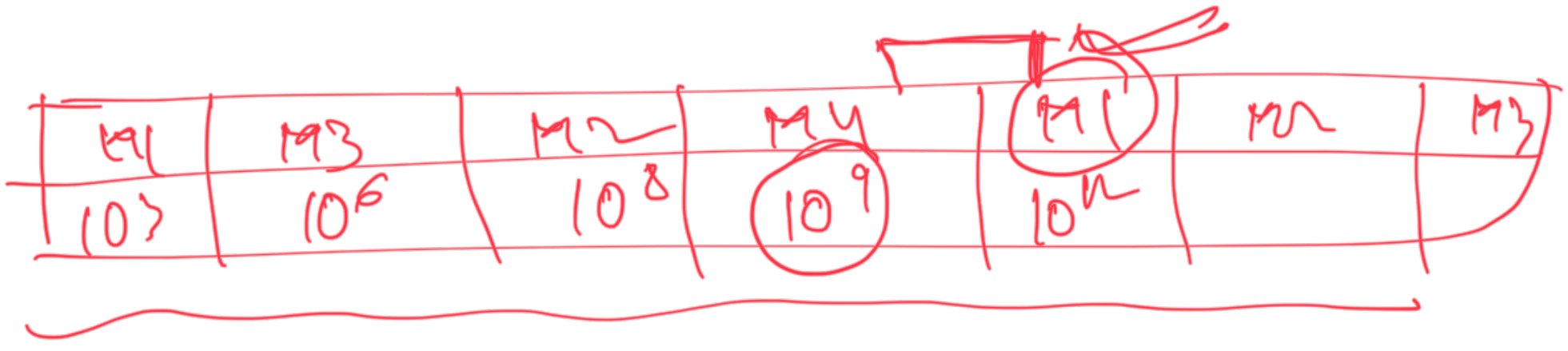
M3

M1

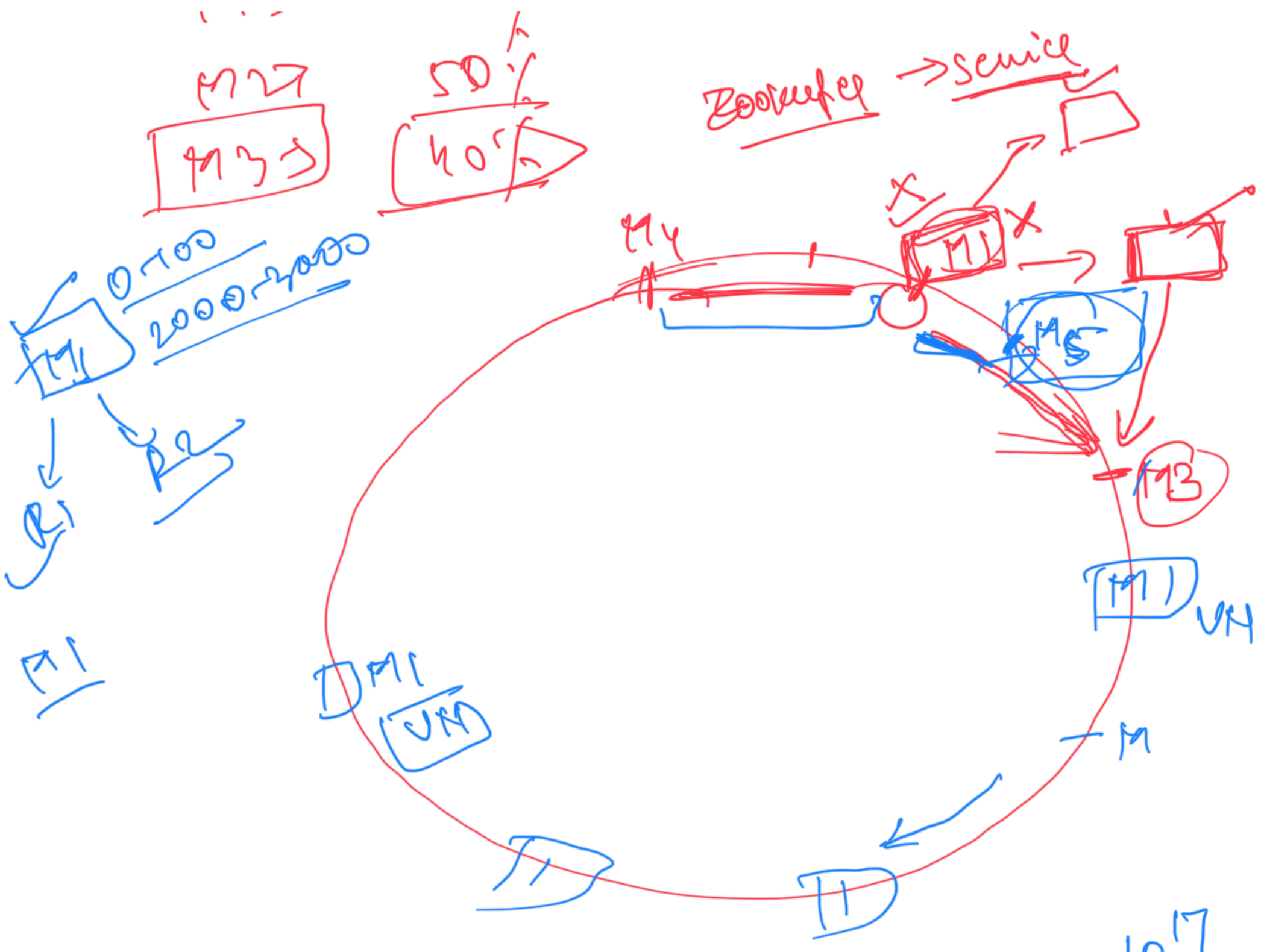
M2

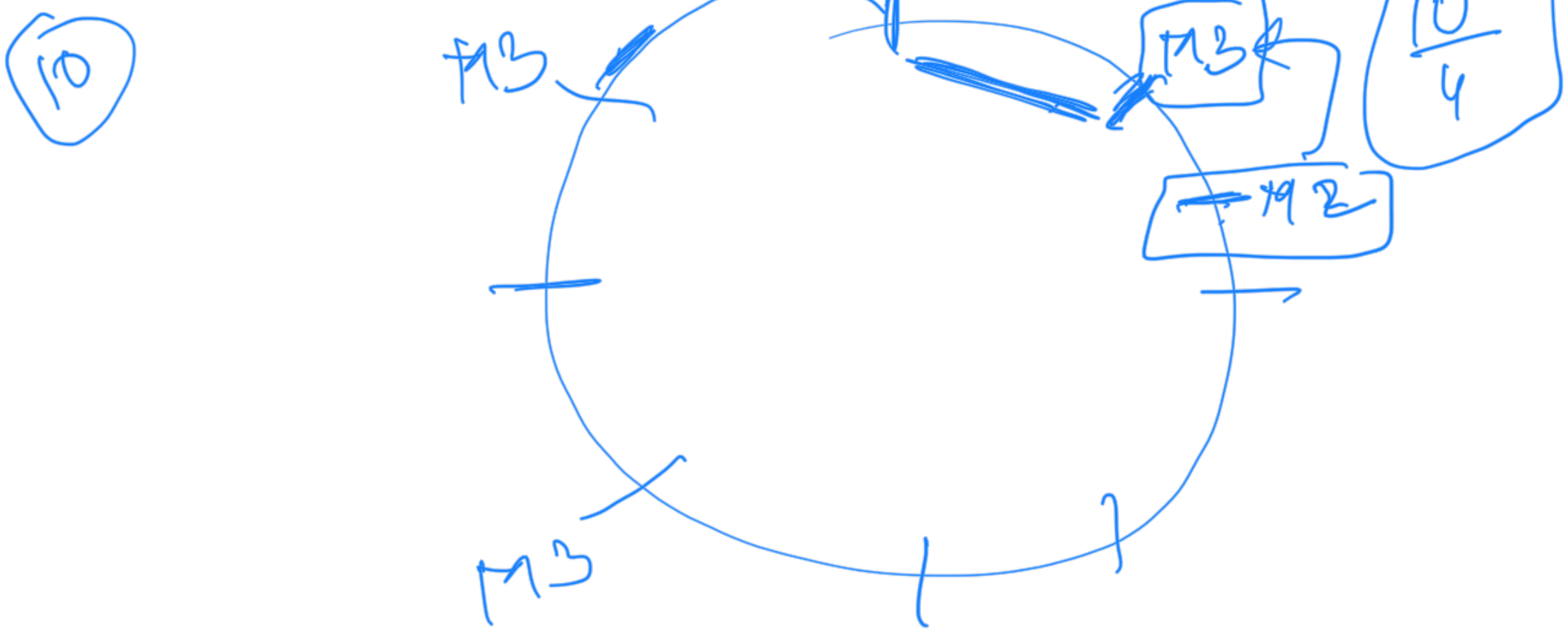
virtual nodes





M1 → CPU 60%


Zookeeper → service
50%
40%
200-2000
10^17


M1
M2
M3
M3
M2
M3
$\frac{10^{18}}{4}$