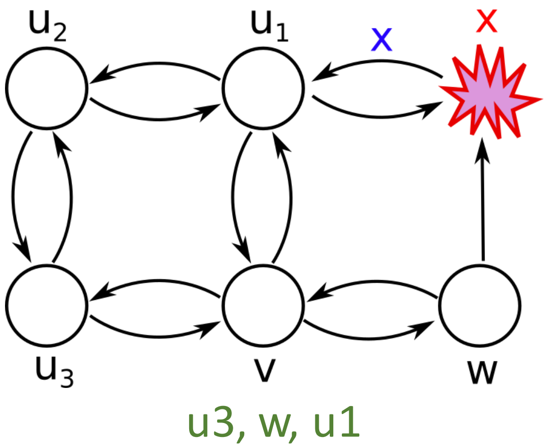
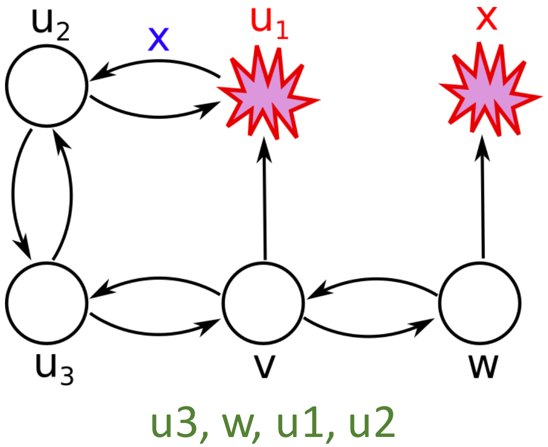
Computer Systems

Exercise 7

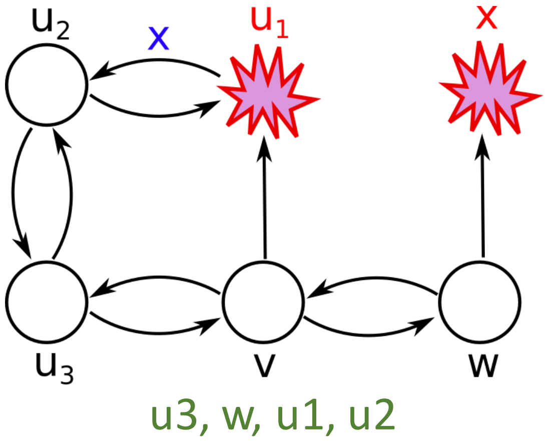
Low in sugar, fat and quizzes!



1. Node x crashes, u1 learns value, w does not



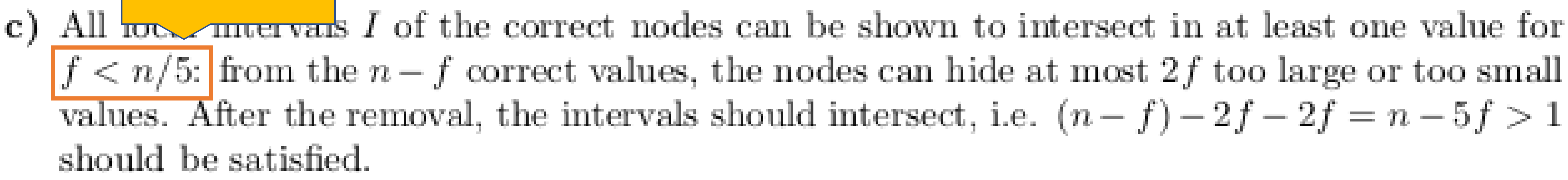
2. Node u1 crashes, u2 learns value, v does not



3. After 2 rounds, v will have learned the values of all nodes except x. In round 3 the value of x is at node u3, therefore v does not learn anything new and will terminate prematurely

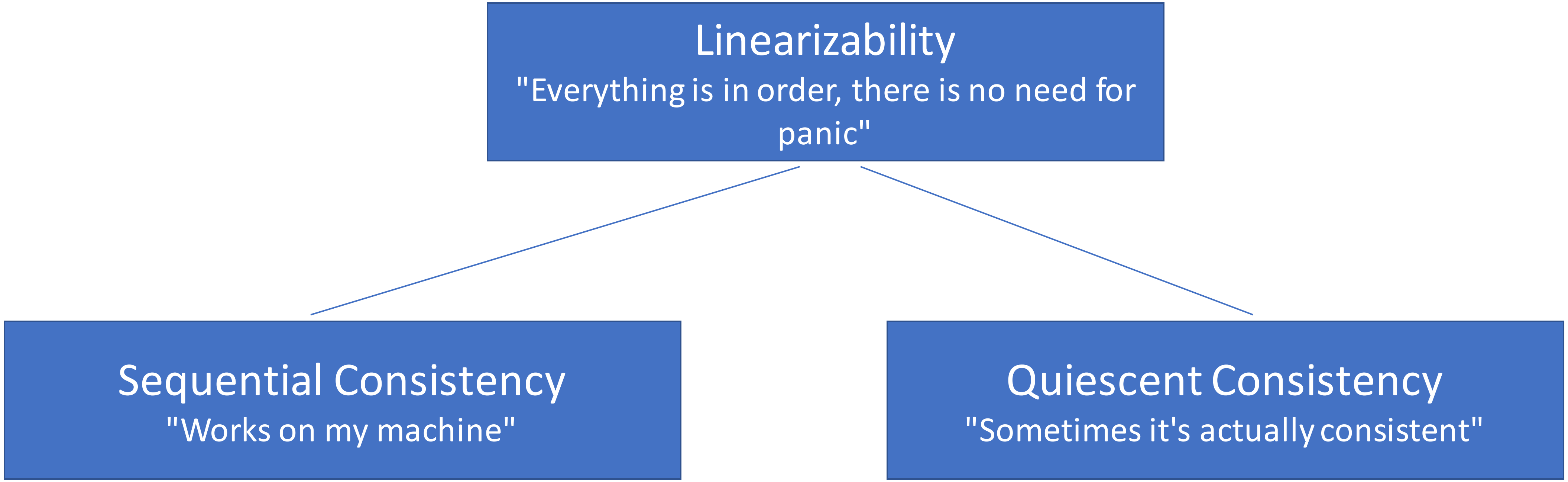
Last Exercise – 2.3

Not n/4!



c) All local intervals I of the correct nodes can be shown to intersect in at least one value for $f < n/5$: from the $n - f$ correct values, the nodes can hide at most $2f$ too large or too small values. After the removal, the intervals should intersect, i.e. $(n - f) - 2f - 2f = n - 5f > 1$ should be satisfied.

Consistency Models



Linearizability
"Everything is in order, there is no need for panic"

Sequential Consistency
"Works on my machine"

Quiescent Consistency
"Sometimes it's actually consistent"

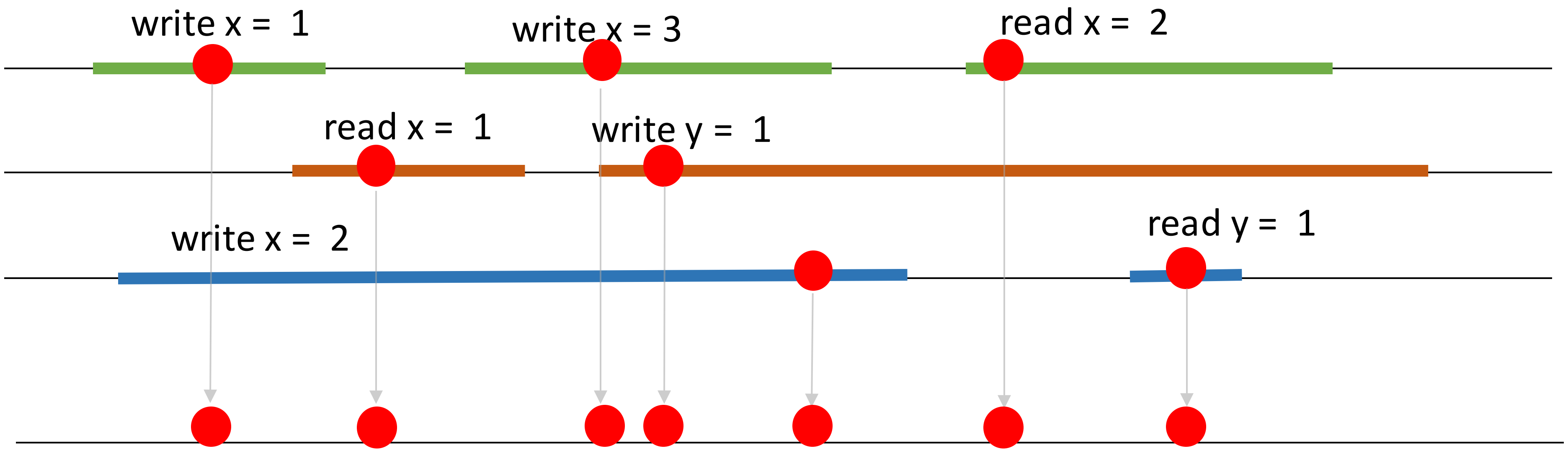
- A more in-depth explanation is available here:
  - http://coldattic.info/post/88/

Linearizability

Linearizability

"Everything is in order, there is no need for panic"

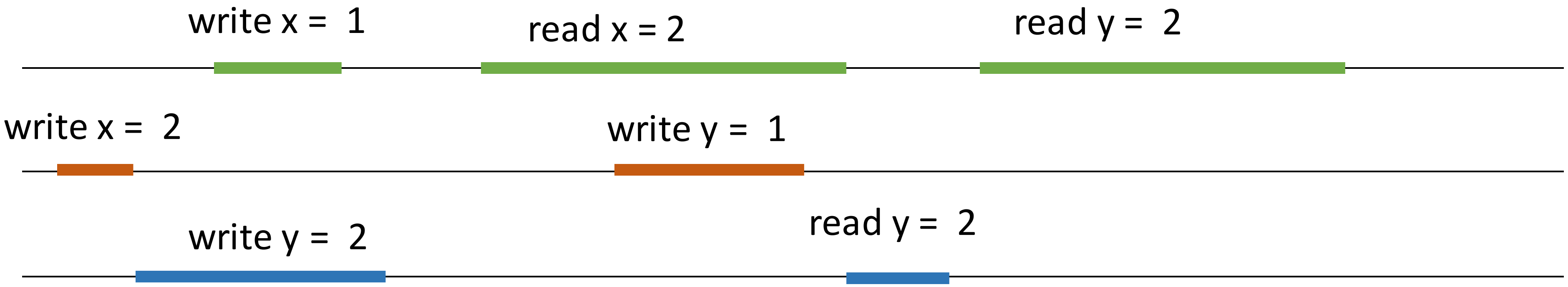
- “one global order”
- Linearizability -> put points on a “line”
- Strongest assumption, implies other two



write x = 1 < read x = 1 < write x= 3 < write y = 1 < write x = 2 < read x = 2 < read = 1

Sequential Consistency

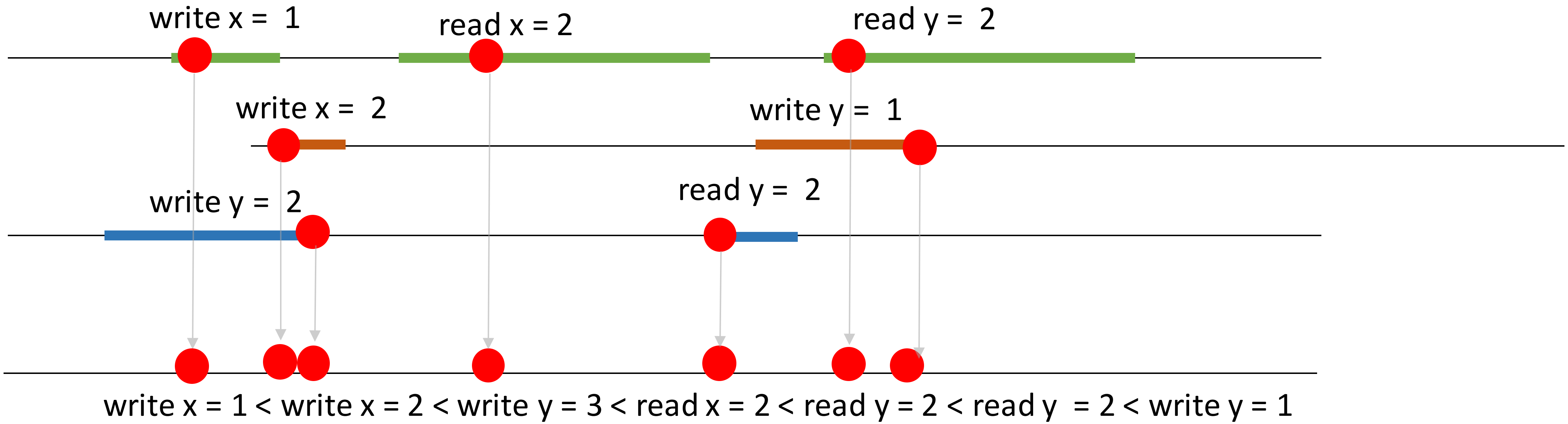
- similar as linearizability, but can ”shift” and “squeeze” threads compared to each other
- sequential consistency -> build “sequences”



Sequential Consistency

Sequential Consistency
"Works on my machine"

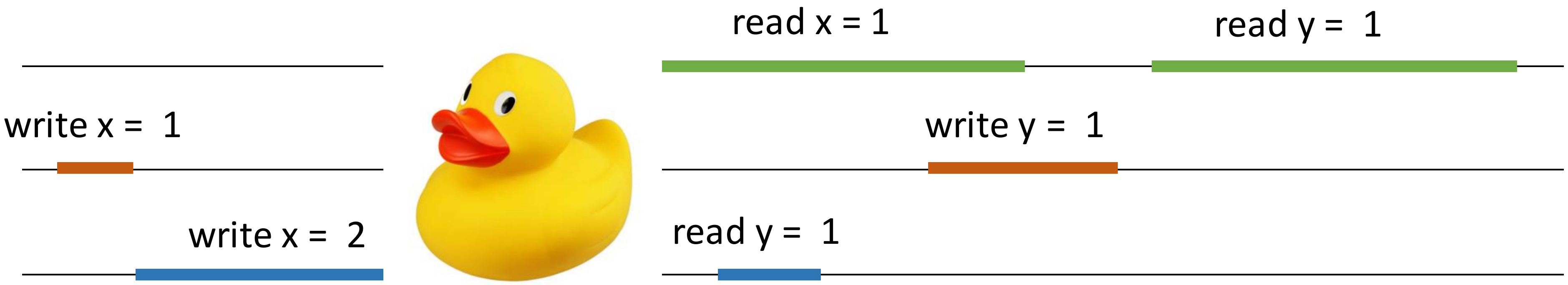
- similar as linearizability, but can ”shift” and “squeeze” threads compared to each other
- sequential consistency -> build “sequences”


write x = 1
read x = 2
read y = 2
write x = 2
write y = 1
write y = 2
read y = 2
write x = 1 < write x = 2 < write y = 3 < read x = 2 < read y = 2 < read y = 2 < write y = 1

Quiescent Consistency

Quiescent Consistency
"Sometimes it's actually consistent"

- synchronizes all threads whenever there is a time when there is no possible execution
- quiescent -> “Quietschente”



read x = 1

read y = 1

write x = 1

write y = 1

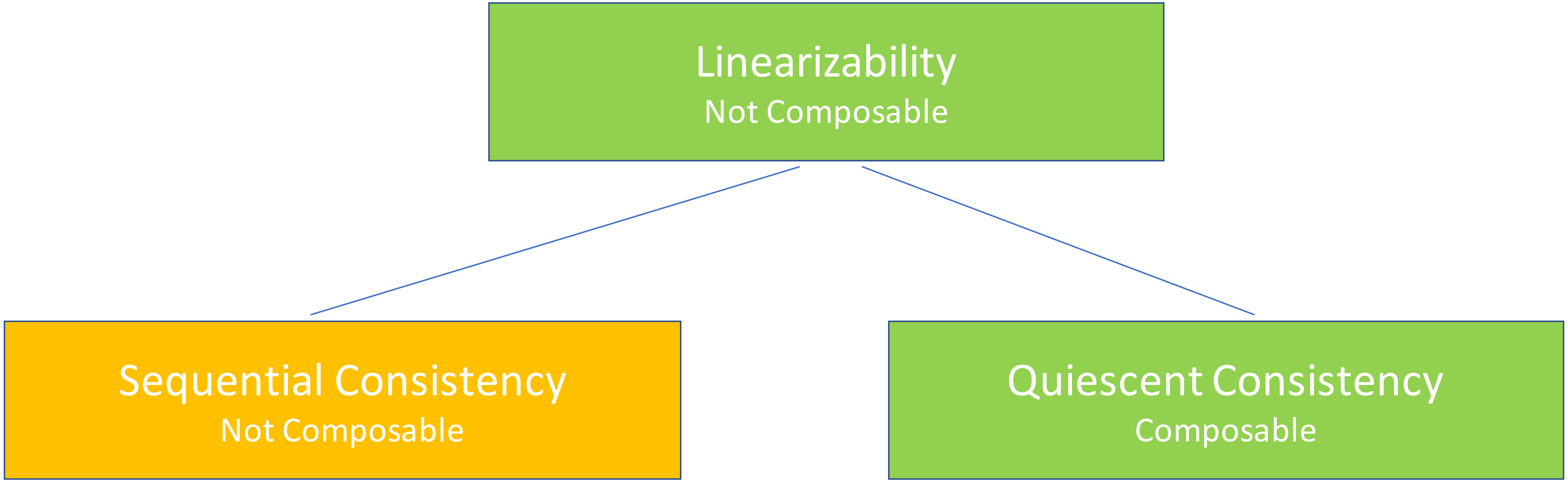
write x = 2

read y = 1

write x = 2 < write x = 1 < write y = 1 < ready y = 1 < read x = 1 < read y = 1

Composability

- Definition: If a consistency criterion applies to all objects individually, then also the whole execution is consistent



Linearizability

Not Composable

Sequential Consistency

Not Composable

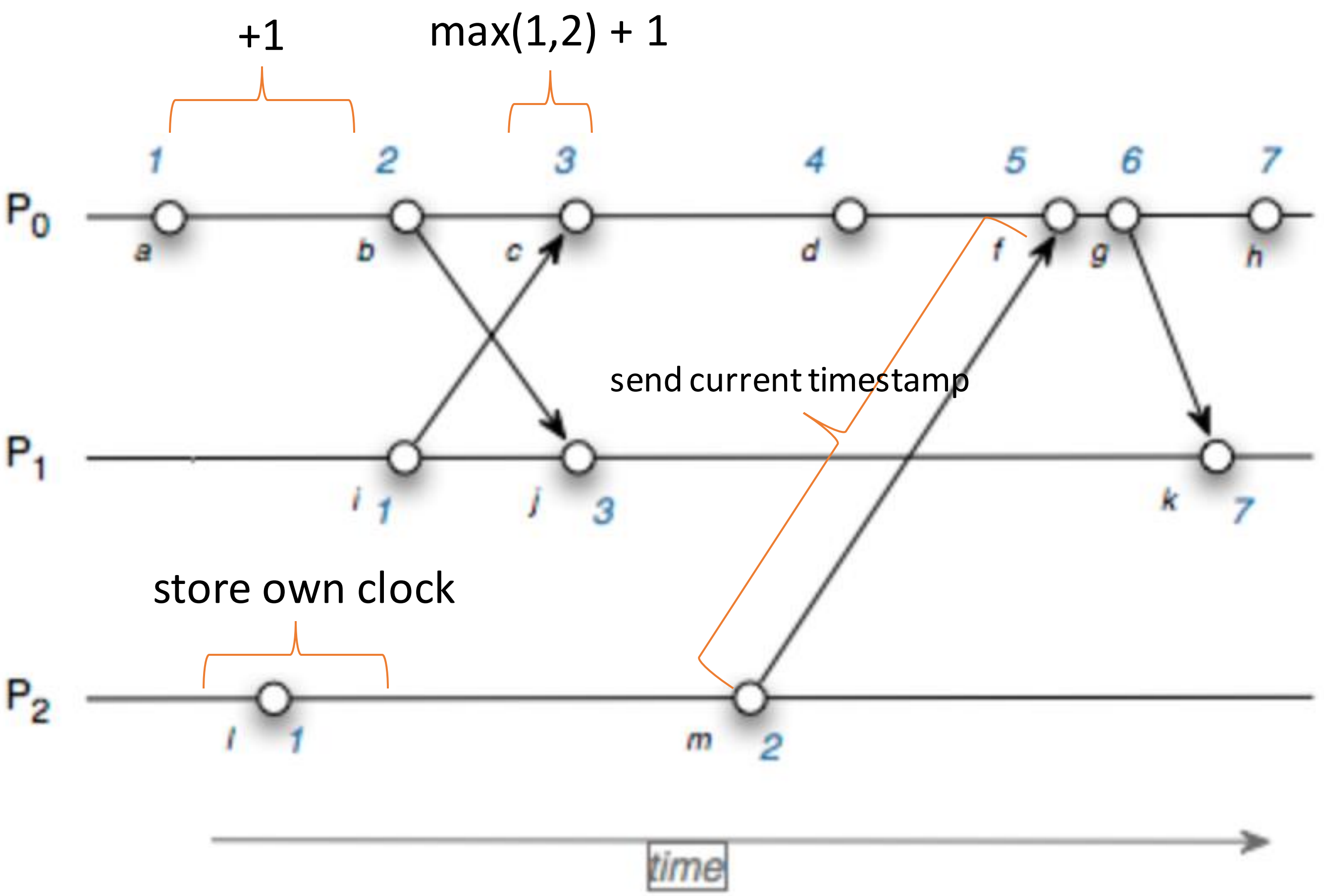
Quiescent Consistency

Composable

Logical Clocks

- happened before relation „->“ holds:
  - If f < g on the same node
  - Send happens before receive
  - If f -> g and g -> h then f -> h (Transitivity)
- c(a) means timestamp of event a
- **logical clock: if a -> b, then c(a) < c(b)**
- **strong logical clock: if c(a) < c(b), then a -> b** (in addition)

Lamport Clock



+1

max(1,2) + 1

send current timestamp

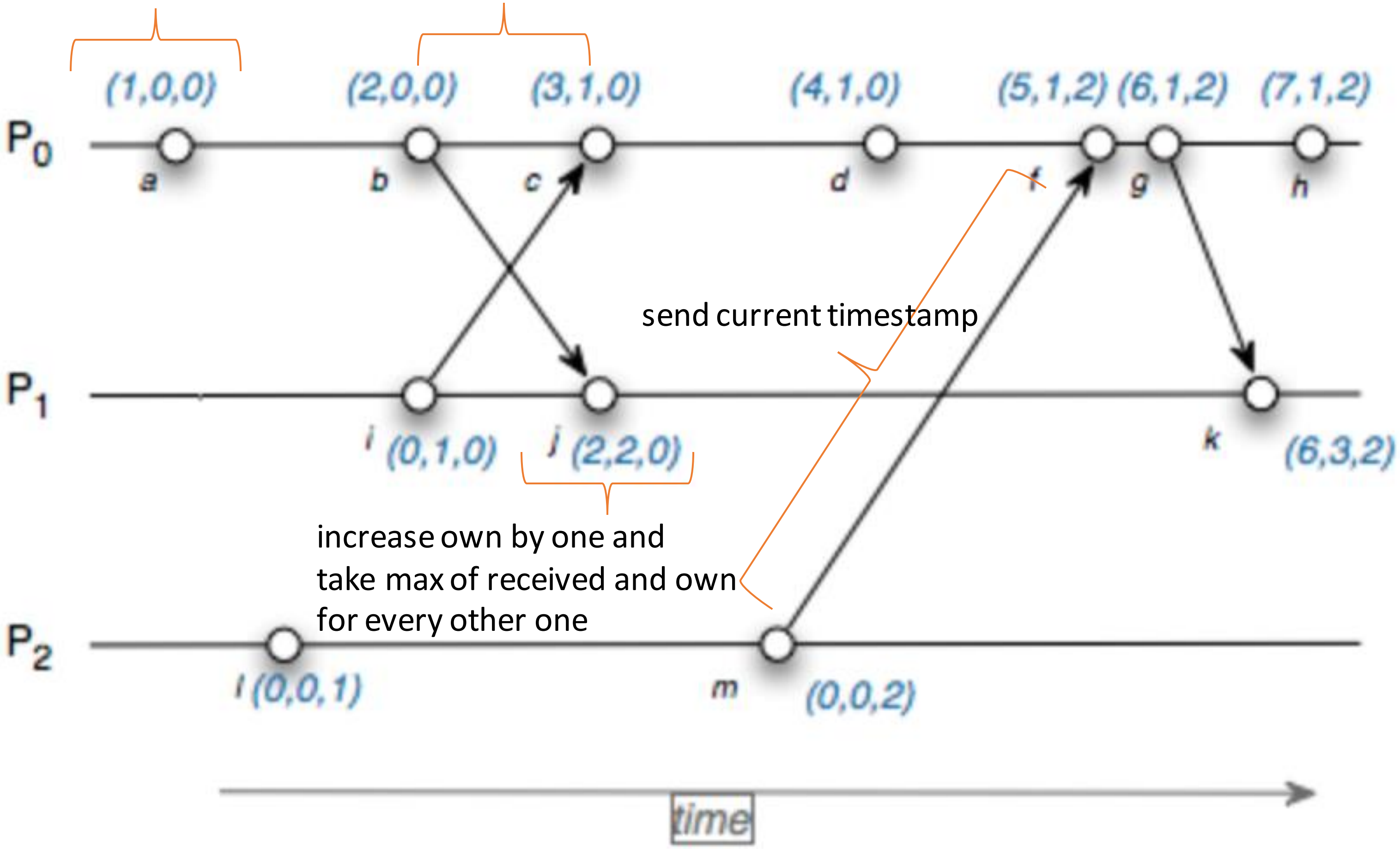
store own clock

https://www.cs.rutgers.edu/~pxk/rutgers/notes/clocks/index.html

Lamport Clock

- Is a logical clock (so if a -> b then c(a) < c(b))
- but the reverse does not hold, so not a strong logical clock

Vector Clock



now vector of clocks

increase own clock for event

send current timestamp

increase own by one and
take max of received and own
for every other one

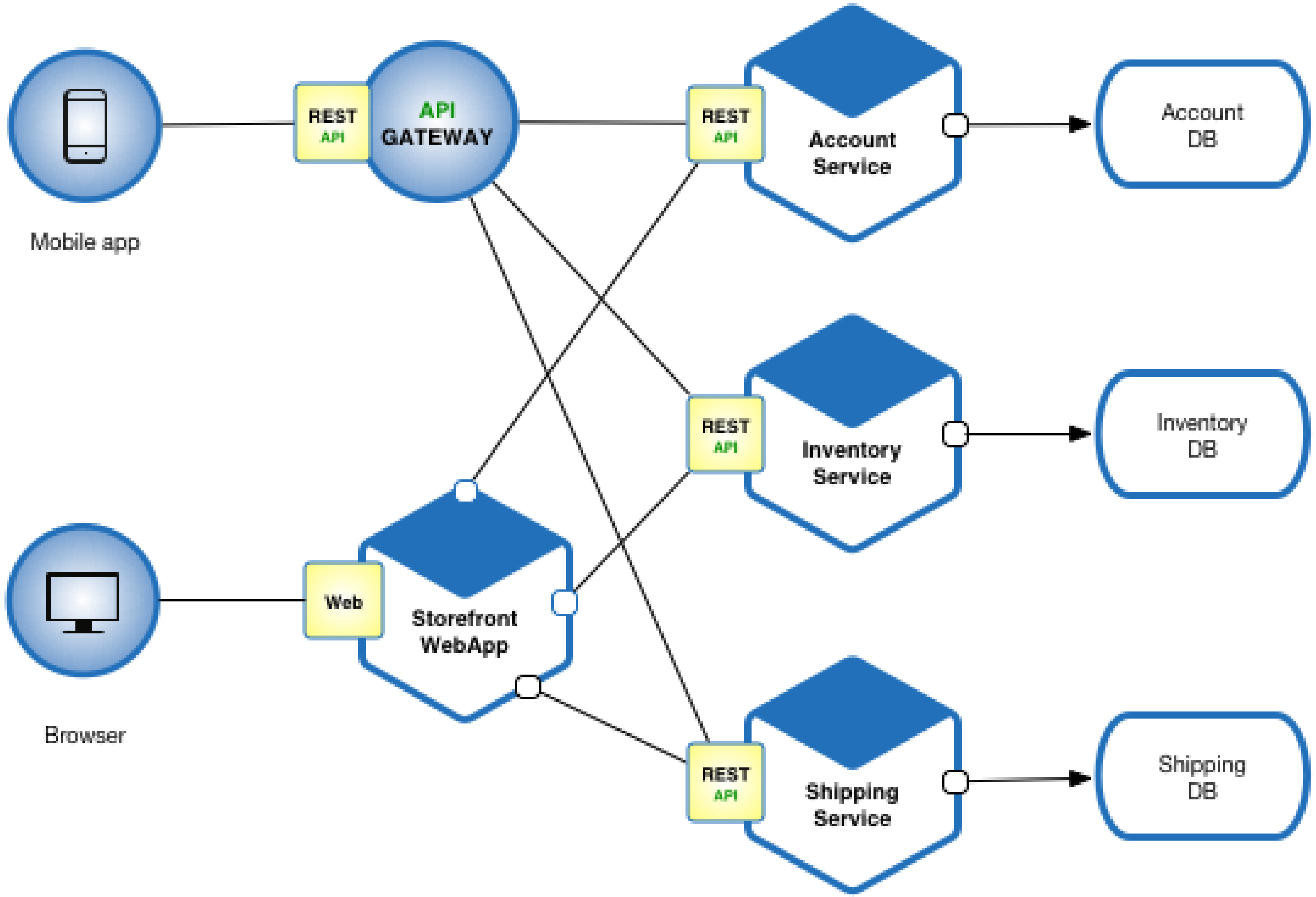
Vector Clock

- what does c(a) < c(b) mean now?
    - if all the entries are in a <= b and at least one entry where a < b
- is a logical clock (so if a -> b then c(a) < c(b))
- is also a strong logical clock (if c(a) < c(b) -> a -> b)
    - intuition: because in order to achieve c(a) < c(b), all entries have to be at least as big, so a message from a must have reached b (not necessarily directly) so that b has the right a value

Consistent Snapshot

- Cut
    - prefix of a distributed execution
- Consistent Snapshot
    - a cut for which holds that for every operation g in that cut, if f->g, then also f is there
    - -> if all “connected” preceding operations are included
- with number of consistent snapshots, one can make conclusions about degrees of concurrency in system

Side Note: Microservices



Microservices

+Easier to write and refactor

+Scalability

+Easy to debug

+Security

-Communication Overhead

-Data Serialization

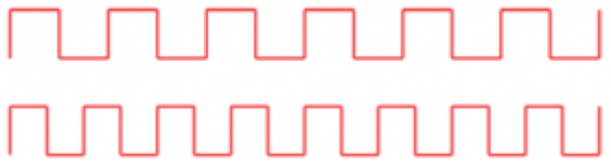
-Hard to debug

-Security

Clocks are awful

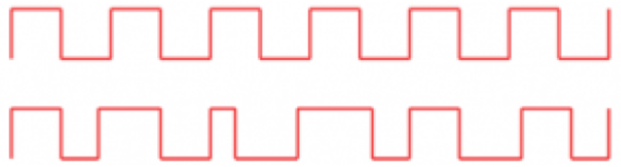
- Drift:

  Slow, persistent errors



- Jitter:

  Unpredictable, spurious errors



THINGS
YOU MIGHT
NOT KNOW

Timekeeping is even worse!

There are ¼h time zones!

Countries switch time zones!

Countries switch calendars!

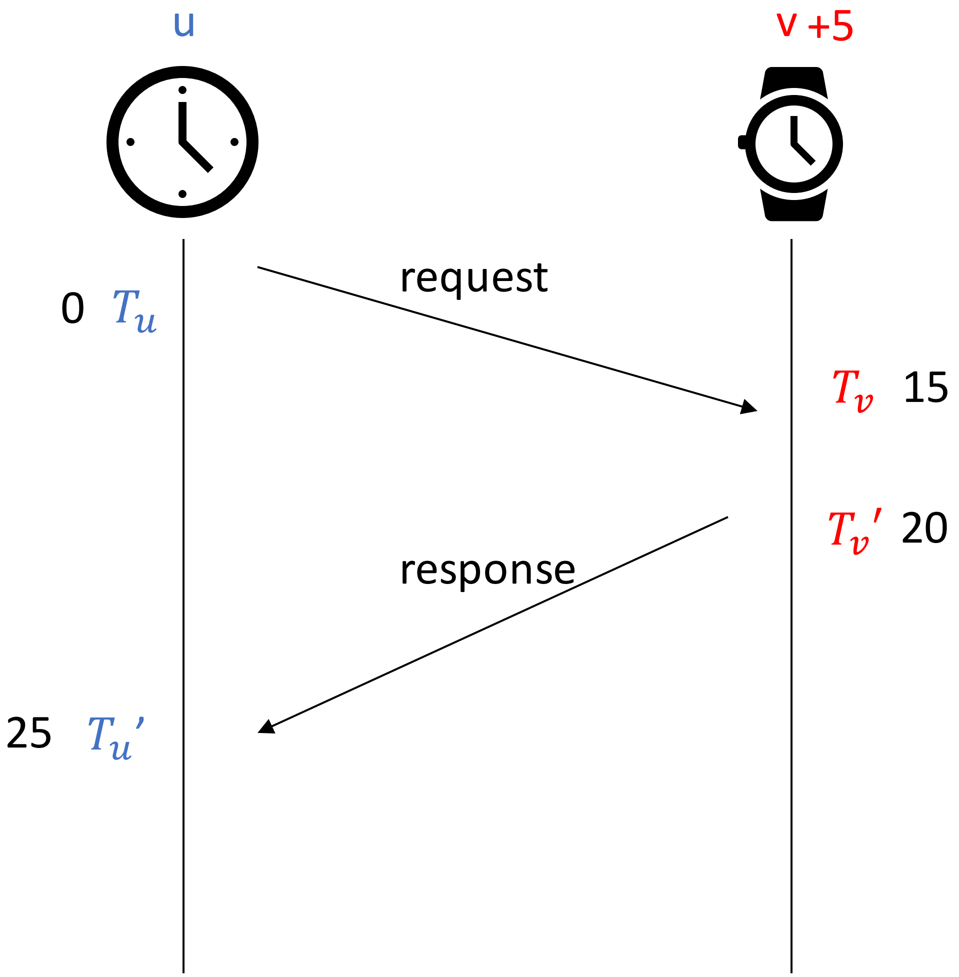
Some minutes have 61 seconds!

A day is not 24 hours long!

A year is not 365 days!

It's not even 365.25!

NTP



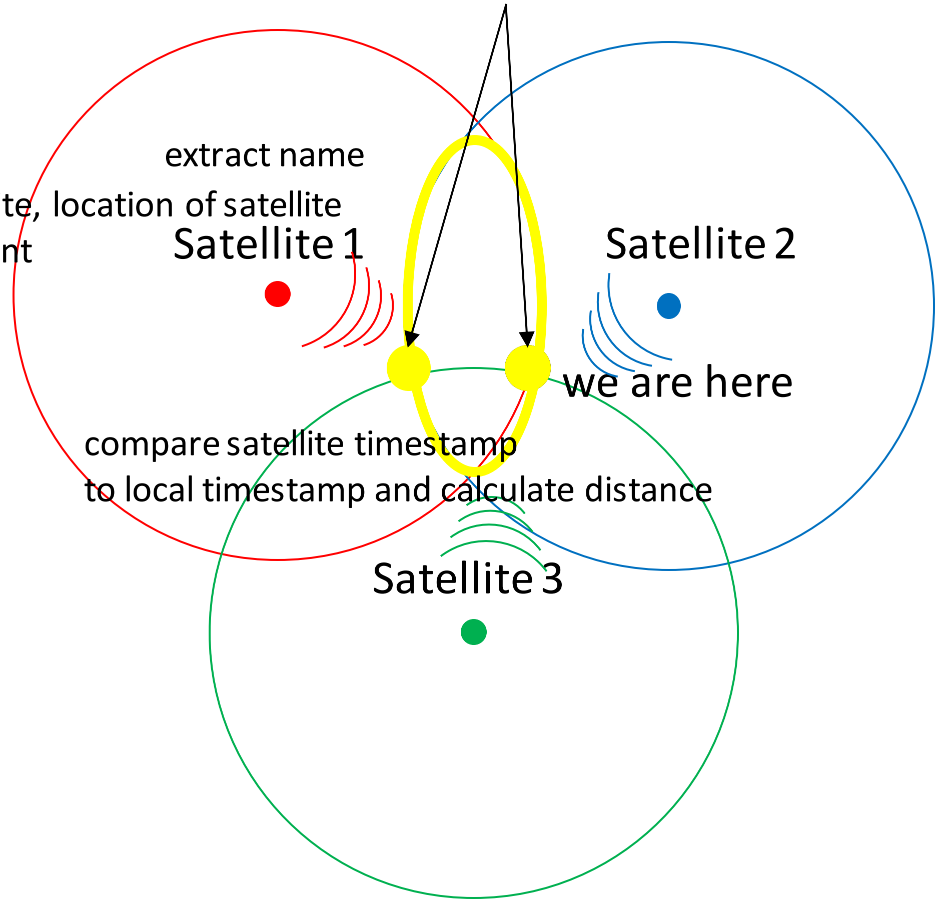
propagation delay = 10

processing time = 5

propagation delay = 10

skew = (15-0) +(20-25)/2 = 5

GPS – General idea



one of them close to earth distance, one far away

extract name

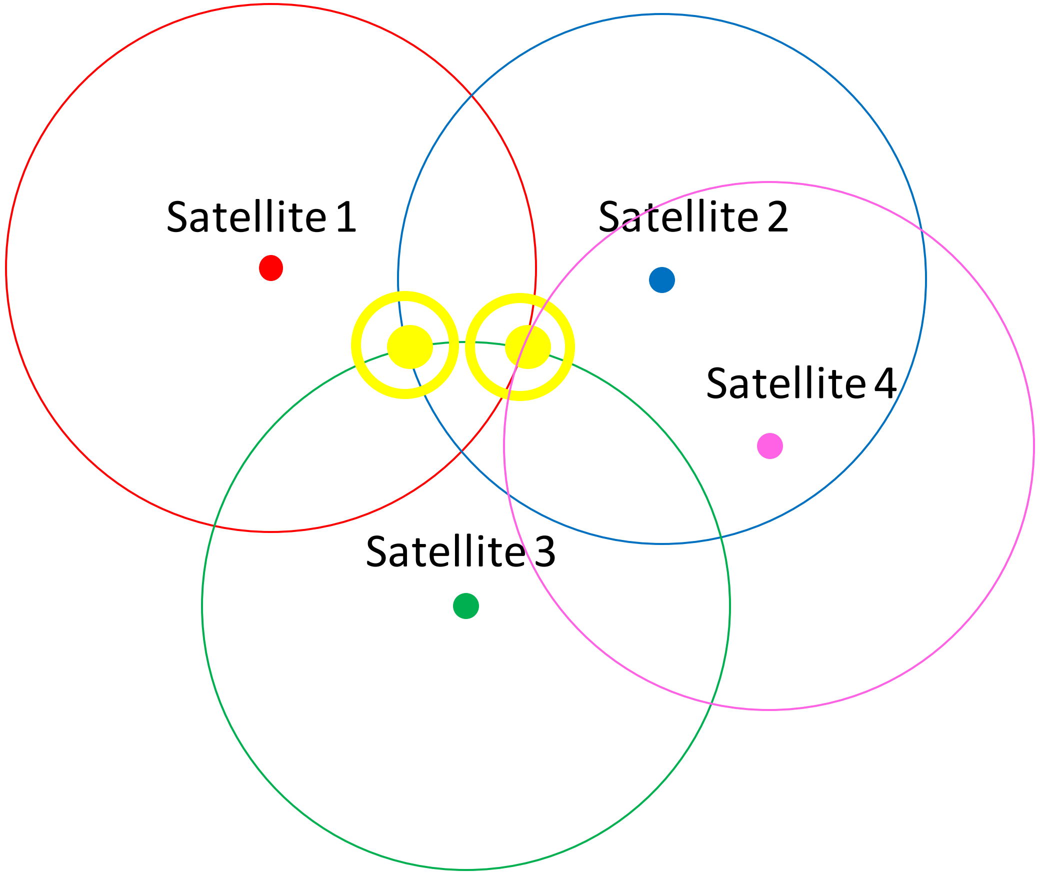
transmits name of satellite, location of satellite
and timestamp when sent

compare satellite timestamp
to local timestamp and calculate distance

GPS - Problem

- Problem: we do not have the same time as the satellite, so calculating the distance might not be accurate
- Solution: take measurement from fourth satellite!

GPS - Refined



GPS - Problem

- Problem: Bandwidth is super low (50 bit/s), so orbit information is sent every 30s
- Solution: go google it

Quiz

1.1 Clock Synchronization

a) Assume you run NTP to synchronize speakers in a soccer stadium. Each speaker has a radio downlink to receive digital audio data. However, there is no uplink! You decide to use an acoustic signal transmit by the speaker. To synchronize its clock, a speaker first plays back an acoustic signal. This signal is picked up by the NTP server which responds via radio. The speaker measures the exact time that passes between audio playback and radio downlink response. What is likely the largest source of error?

b) What are strategies to reduce the effect of this error source?

c) Prove or disprove the following statement: If the average local skew is smaller than x, then so is the average global skew.

d) Prove or disprove the following statement: If the average global skew is smaller than x, then so is the average local skew.