Parallelism

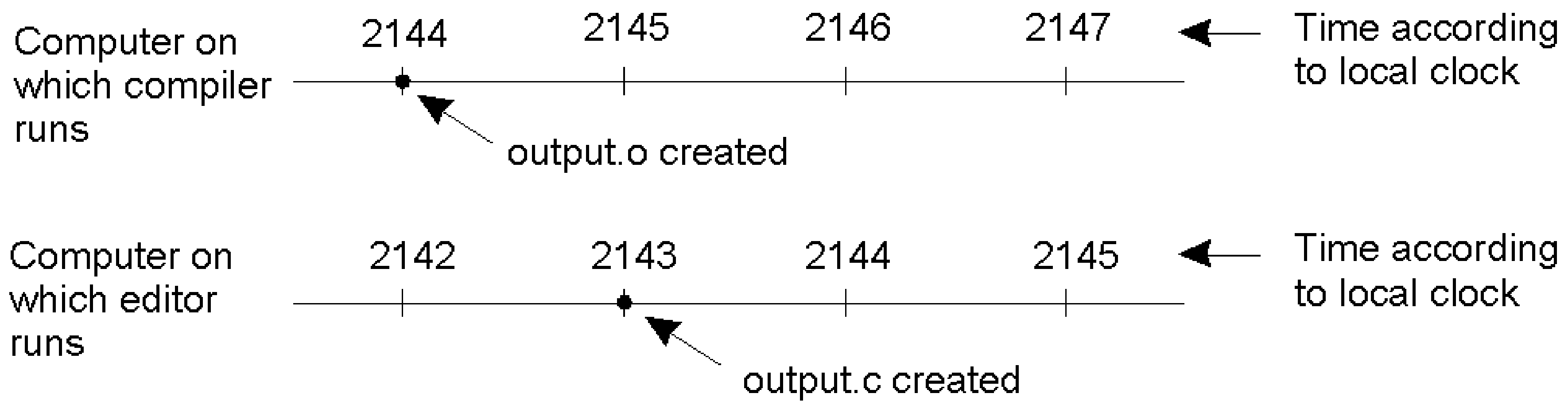
Master 1 International

Andrea G. B. Tettamanzi
Université de Nice Sophia Antipolis
Département Informatique
andrea.tettamanzi@unice.fr

Lecture 3 – Part b

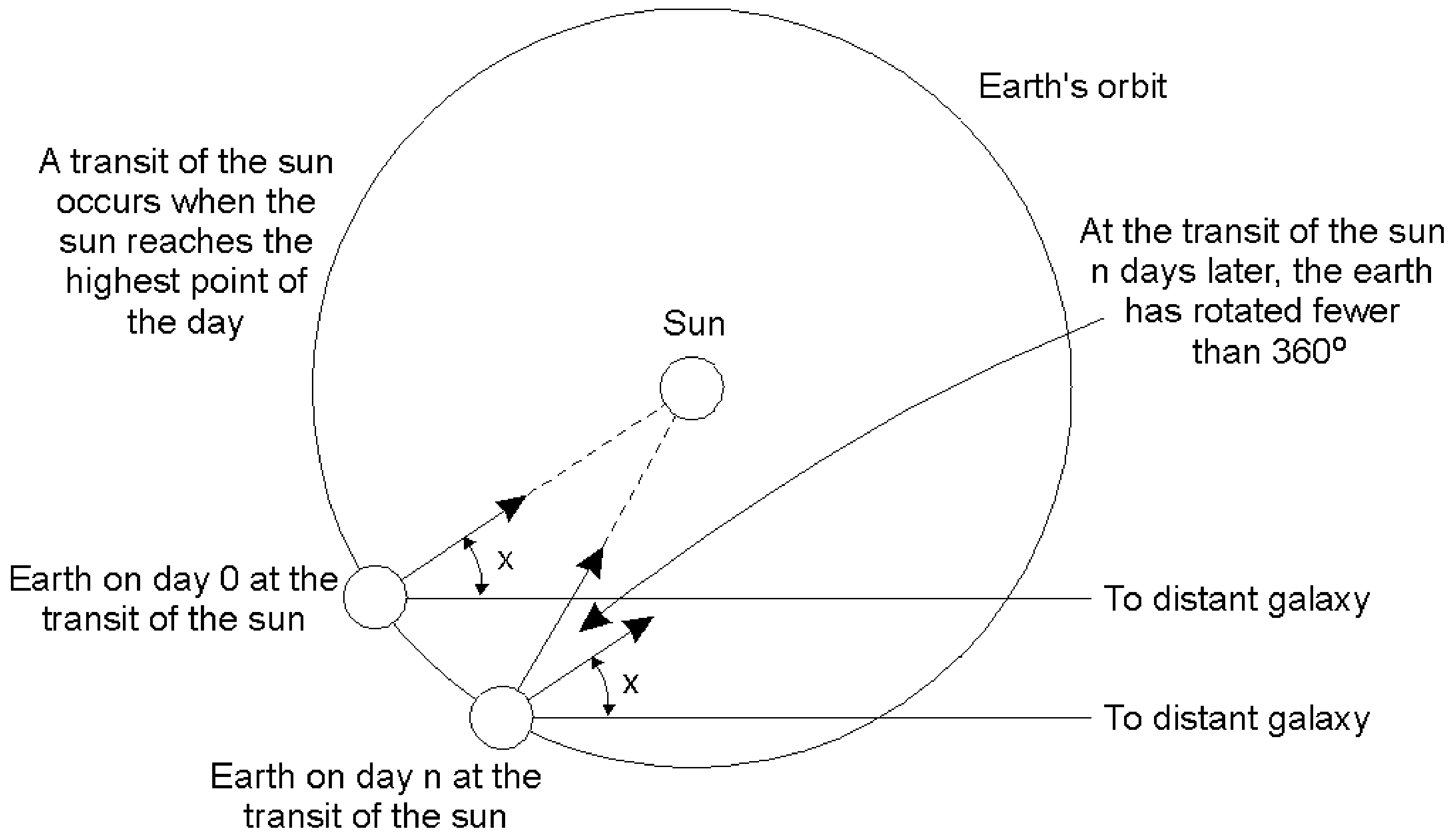
Synchronization

Clock Synchronization


Computer on which compiler runs
2144
2145
2146
2147
Time according to local clock
output.o created
Computer on which editor runs
2142
2143
2144
2145
Time according to local clock
output.c created

When each machine has its own clock, an event that occurred after another event may nevertheless be assigned an earlier time.

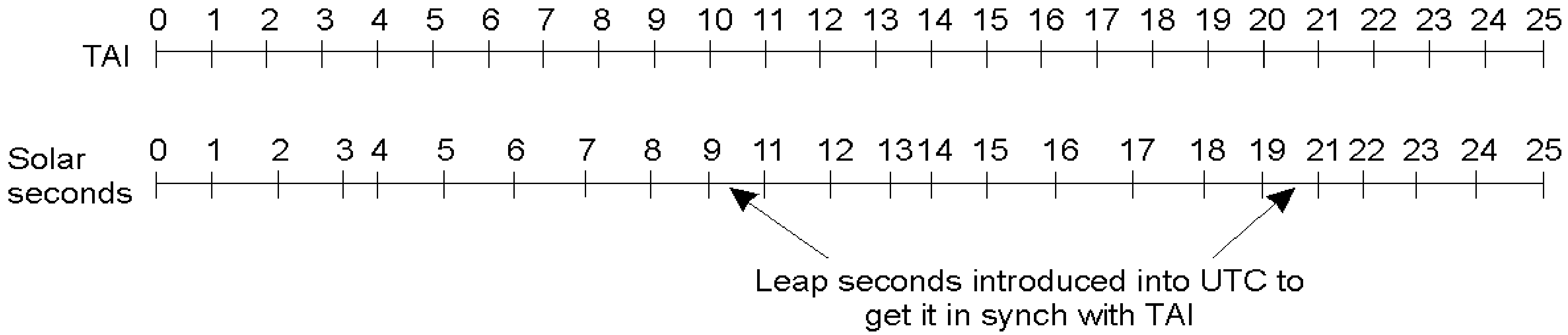
Physical Clocks (1)


Earth's orbit
A transit of the sun occurs when the sun reaches the highest point of the day
Sun
At the transit of the sun n days later, the earth has rotated fewer than 360º
x
Earth on day 0 at the transit of the sun
To distant galaxy
x
To distant galaxy
Earth on day n at the transit of the sun

Computation of the mean solar day.

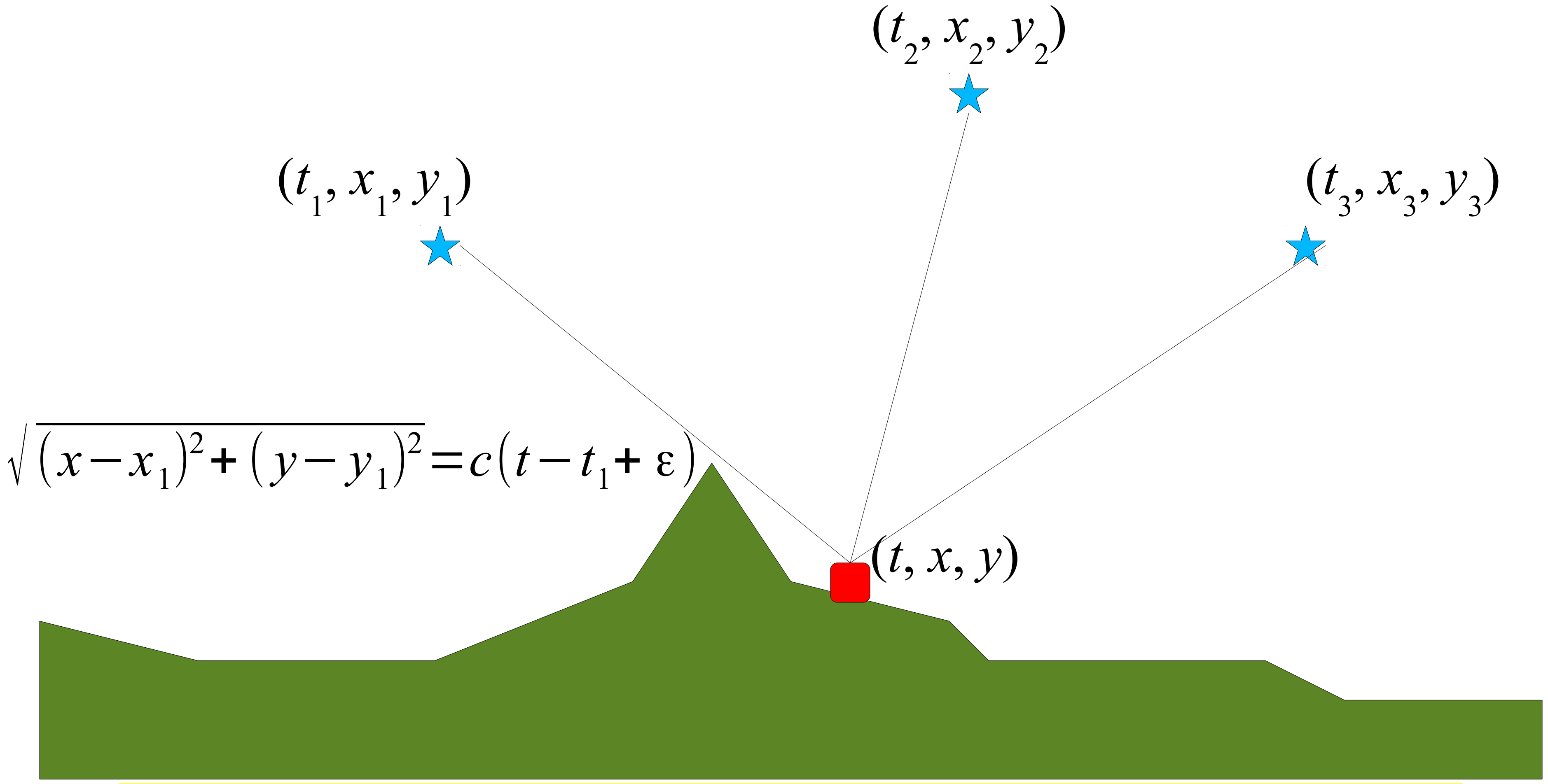
Physical Clocks (2)

Temps atomique international (TAI)



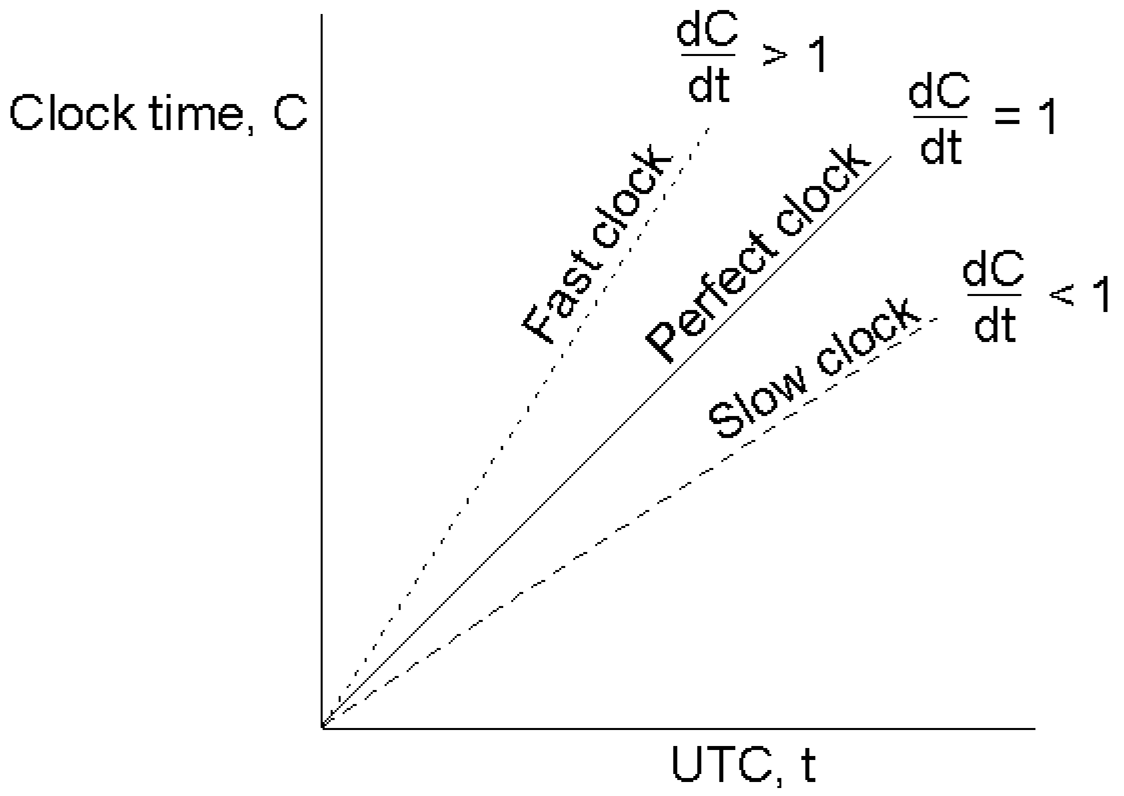
TAI seconds are of constant length, unlike solar seconds. Leap seconds are introduced when necessary to keep in phase with the sun.

Global Positioning System



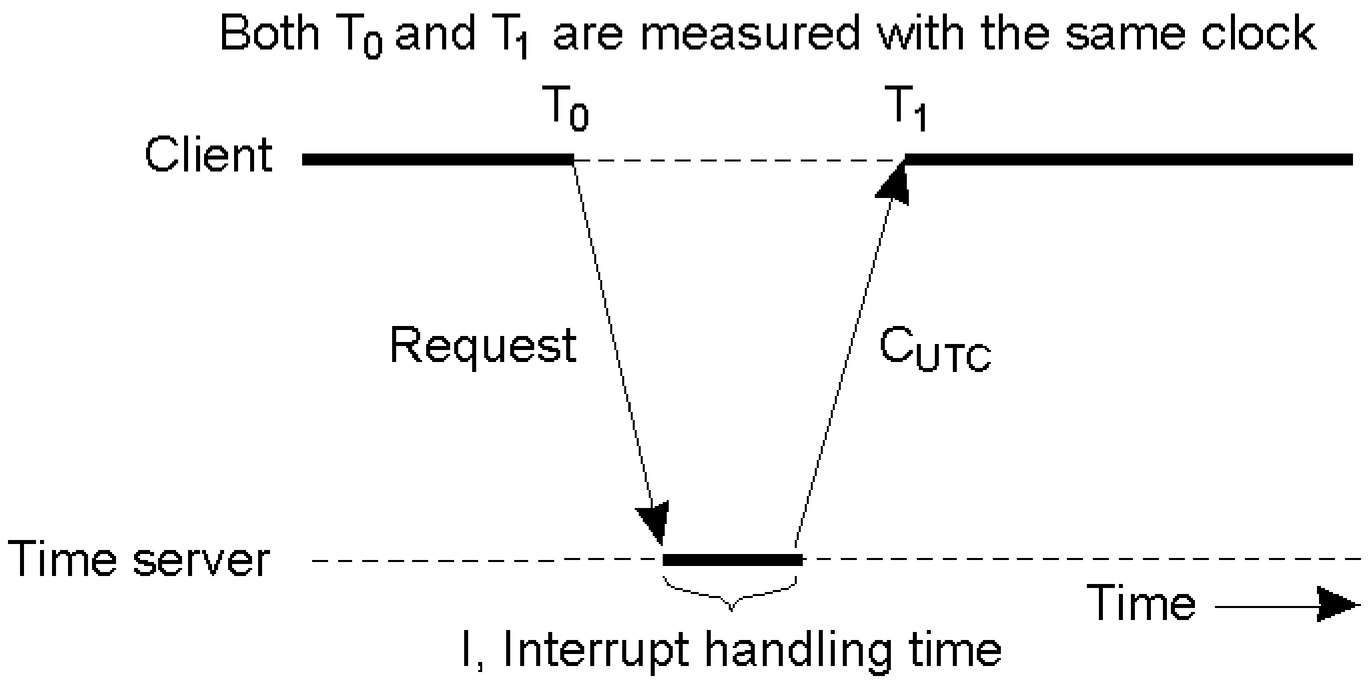
$$\sqrt{(x-x_1)^2+(y-y_1)^2}=c(t-t_1+\varepsilon)$$

Clock Synchronization Algorithms



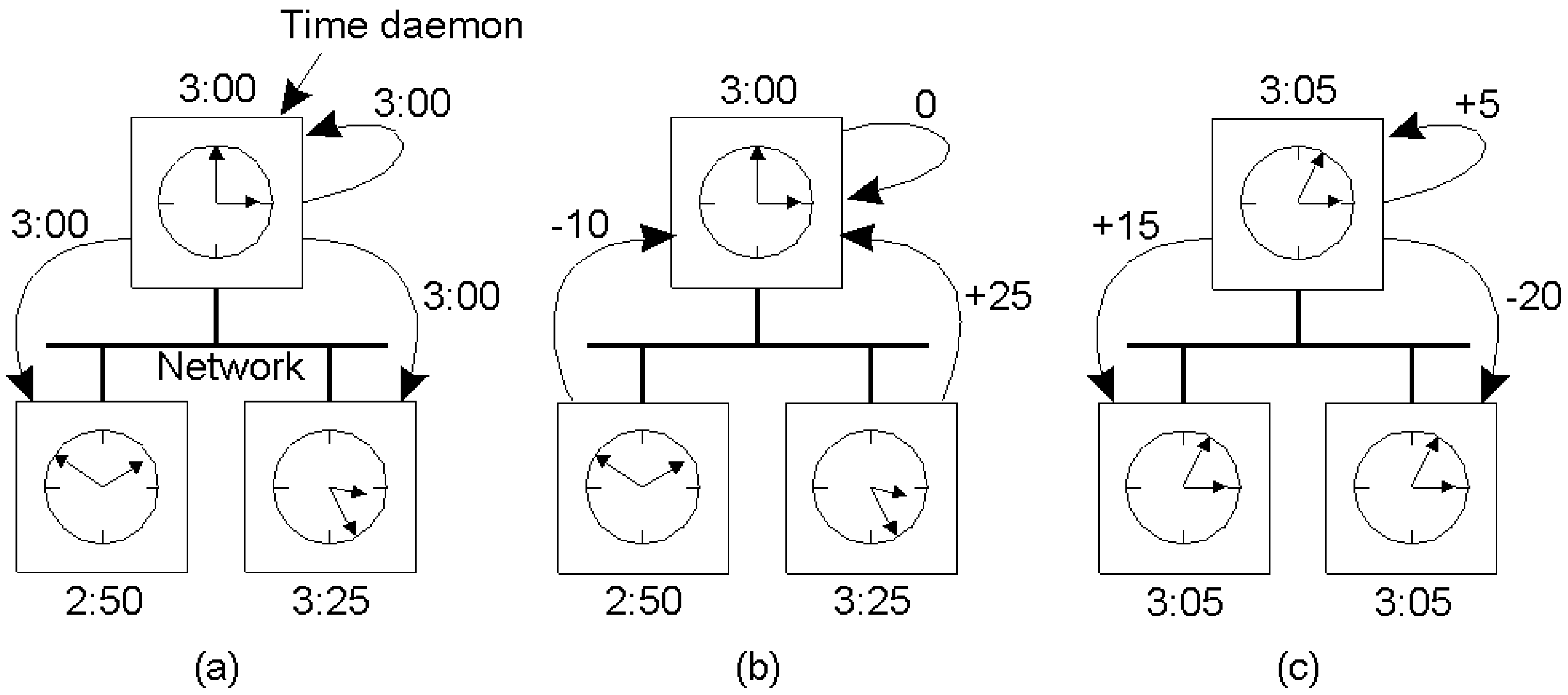
The relation between clock time and UTC when clocks tick at different rates.

Cristian's Algorithm


Both T_0 and T_1 are measured with the same clock
T_0
T_1
Client
Request
C_{UTC}
Time server
Time
I, Interrupt handling time

Getting the current time from a time server.

The Berkeley Algorithm



a) The time daemon asks all the other machines for their clock values

b) The machines answer

c) The time daemon tells everyone how to adjust their clock

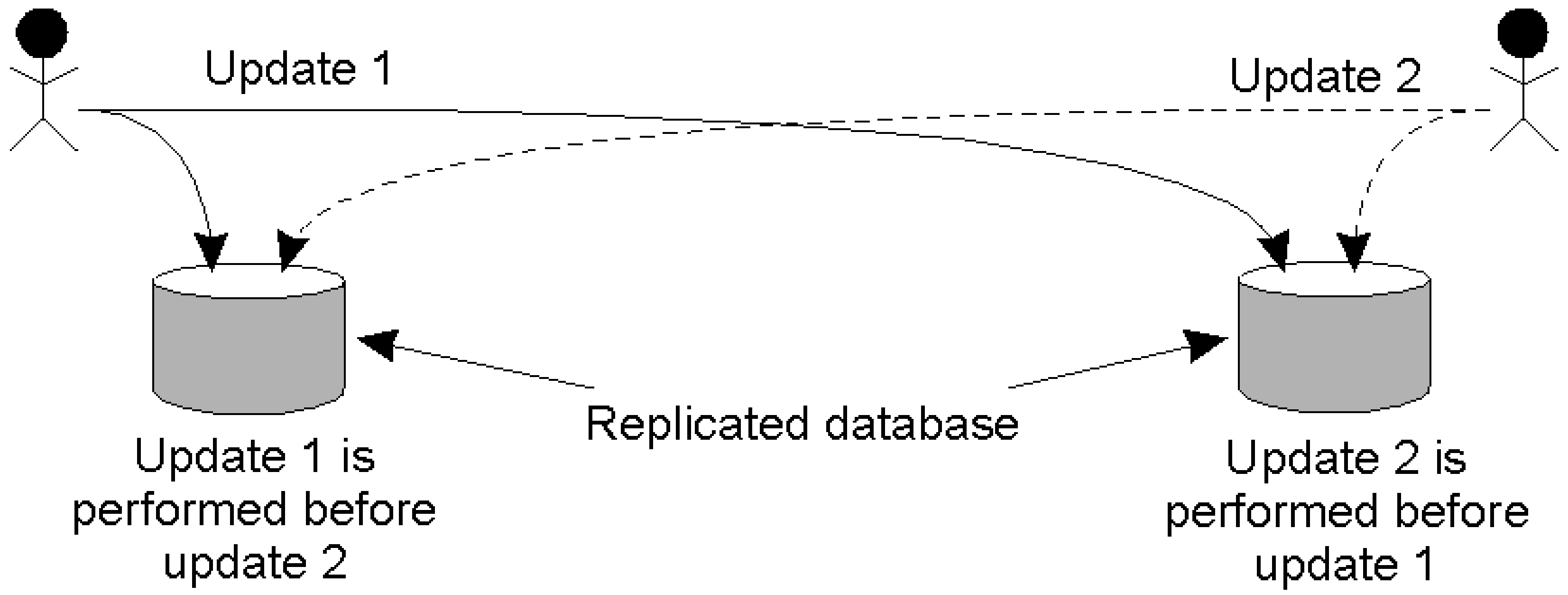
NTP

- Network Time Protocol (RFC 958 → RFC 5905)
- High-accuracy time synchronization for computers across the net
- In Unix : ntpd – NTP daemon, adjusts its own computer time
- Each daemon can be client, server, or peer for other daemons:
  - As client it queries reference time from one or more servers
  - As server it makes its own time available as reference time
  - As peer it compares its system time to other peers until all the peers finally agree about the "true" time to synchronize to
- Hierarchical time synchronization structure (strata)
- A daemon's stratum is the stratum of its time source + 1
- Radio clocks have a stratum number of 0

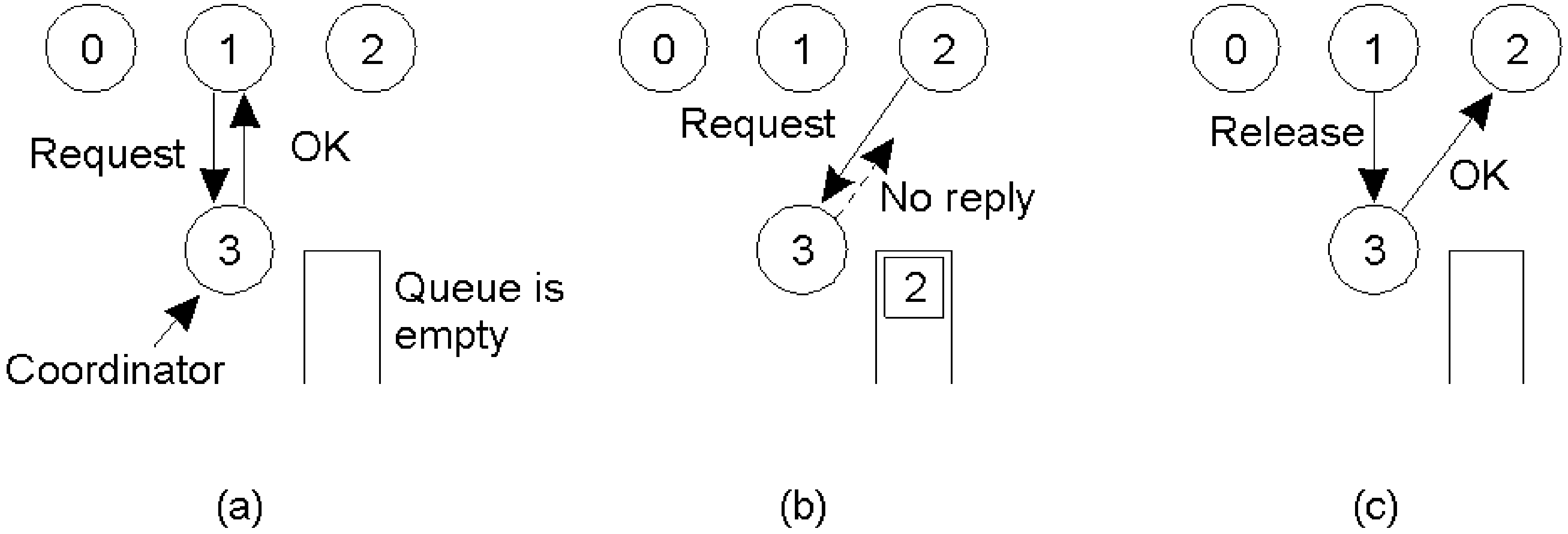
Lamport's Logical Clocks

- Relation $\rightarrow$
  - If a and b are events in the same thread and a comes before b, then $a \rightarrow b$
  - If a is the sending of a message by a thread and b is the receipt of the same message by a different thread, then $a \rightarrow b$
- Clock Condition: for any events, a and b,
  - If $a \rightarrow b$ then $C(a) < C(b)$
- Implementation
  - Each thread increments its clock between any two successive events
  - A massage contains C(a) as its timestamp; upon receiving it, the receiving thread sets its clock to max{clock, C(a) + 1}

Lamport Timestamps


Update 1
Update 2
Replicated database
Update 1 is performed before update 2
Update 2 is performed before update 1

Mutual Exclusion: A Centralized Algorithm



a) Process 1 asks the coordinator for permission to enter a critical region. Permission is granted

b) Process 2 then asks permission to enter the same critical region. The coordinator does not reply.

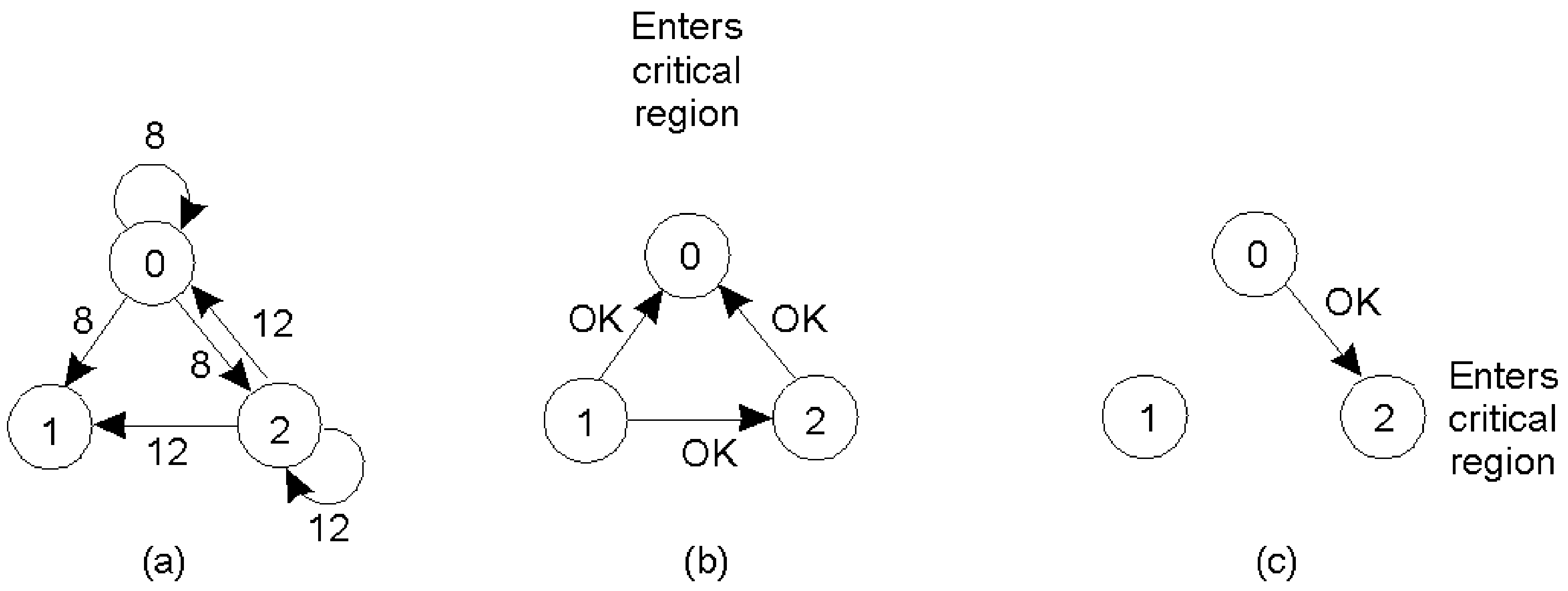
c) When process 1 exits the critical region, it tells the coordinator, when then replies to 2

A Decentralized Algorithm

- For each resource, *n* coordinators
- Access granted with *m* > *n*/2 authorizations
- Let *p* = prob that a coordinator resets in Δt,
- P[k] = *k* coordinators reset

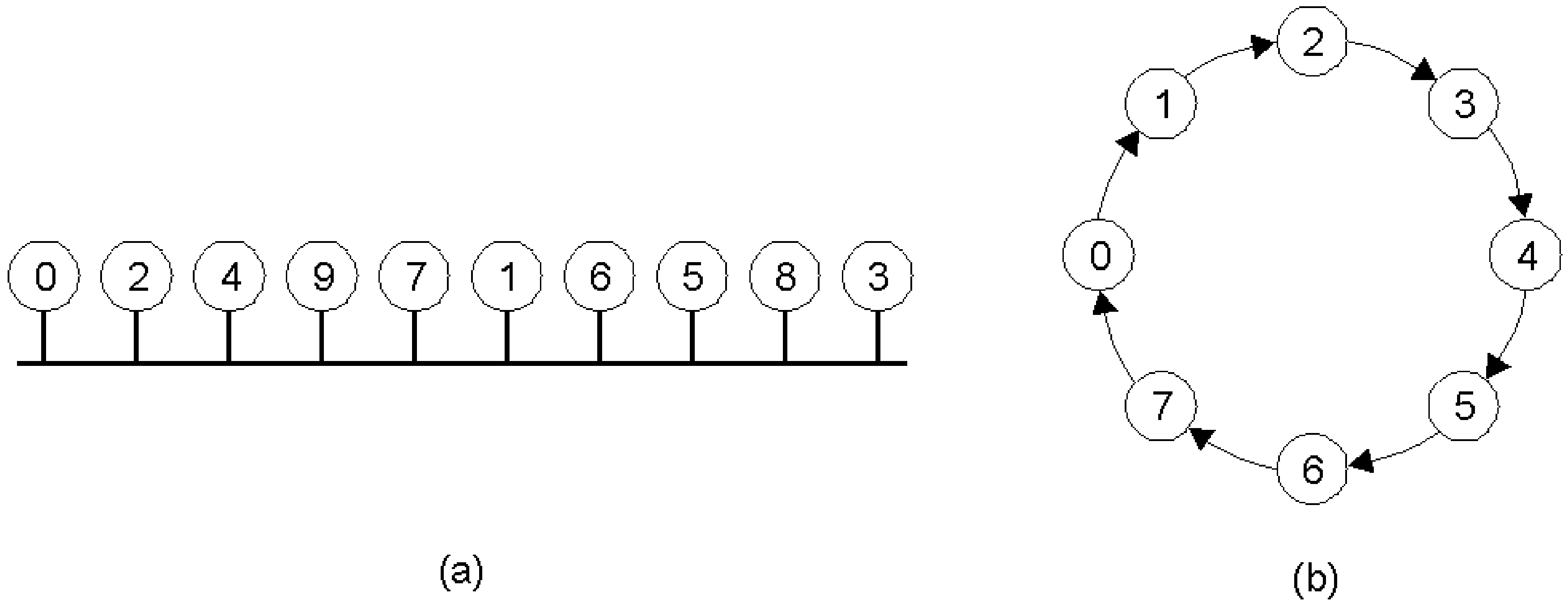
$$P[k] = \binom{m}{k} p^k (1-p)^{m-k}$$

A Distributed Algorithm



a) Two processes want to enter the same critical region at the same moment.
b) Process 0 has the lowest timestamp, so it wins.
c) When process 0 is done, it sends an OK also, so 2 can now enter the critical region.

A Token Ring Algorithm



a) An unordered group of processes on a network.

b) A logical ring constructed in software.

Comparison

Algorithm	Messages per entry/exit	Delay before entry (in message times)	Problems
Centralized	3	2	Coordinator crash
Distributed	2 (n – 1)	2 (n – 1)	Crash of any process
Token ring	1 to ∞	0 to n – 1	Lost token, process crash

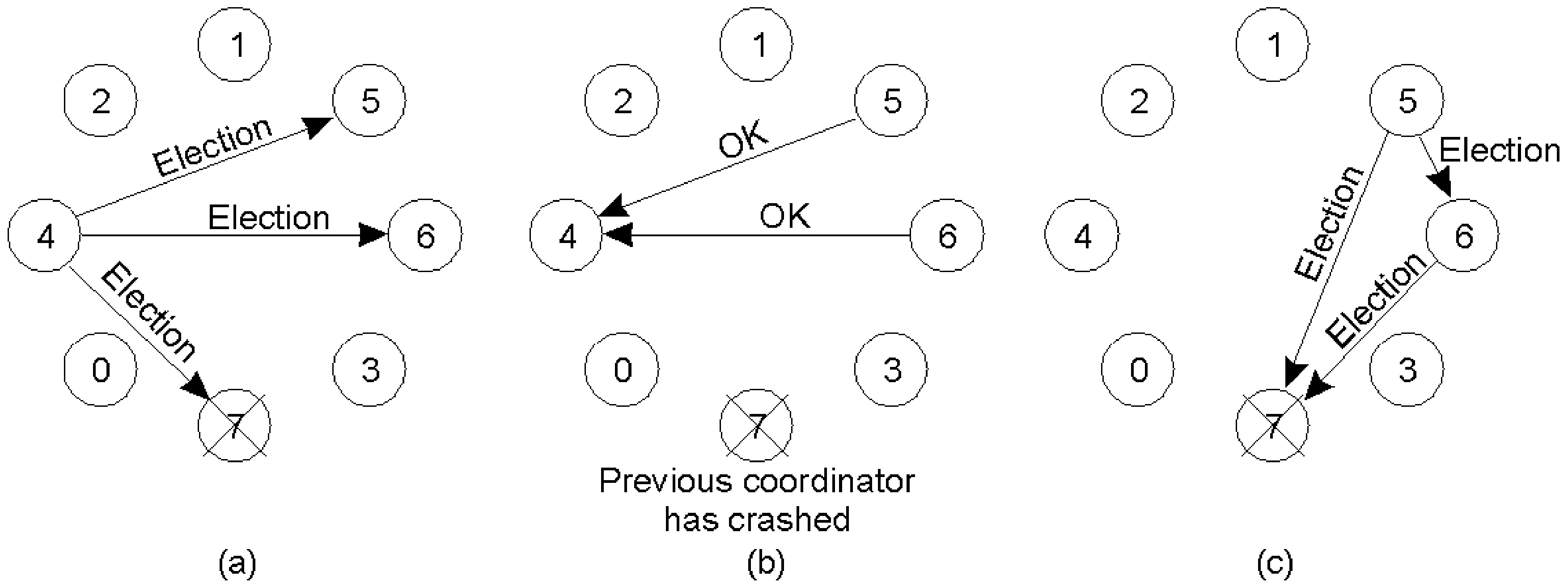
A comparison of three mutual exclusion algorithms.

Election Algorithms

- How is coordinator to be selected dynamically?
- N.B.: in some systems, chosen by hand (e.g., file server) → single point of failure

- Questions:
  - Centralized or decentralized?
  - Which one is more robust?

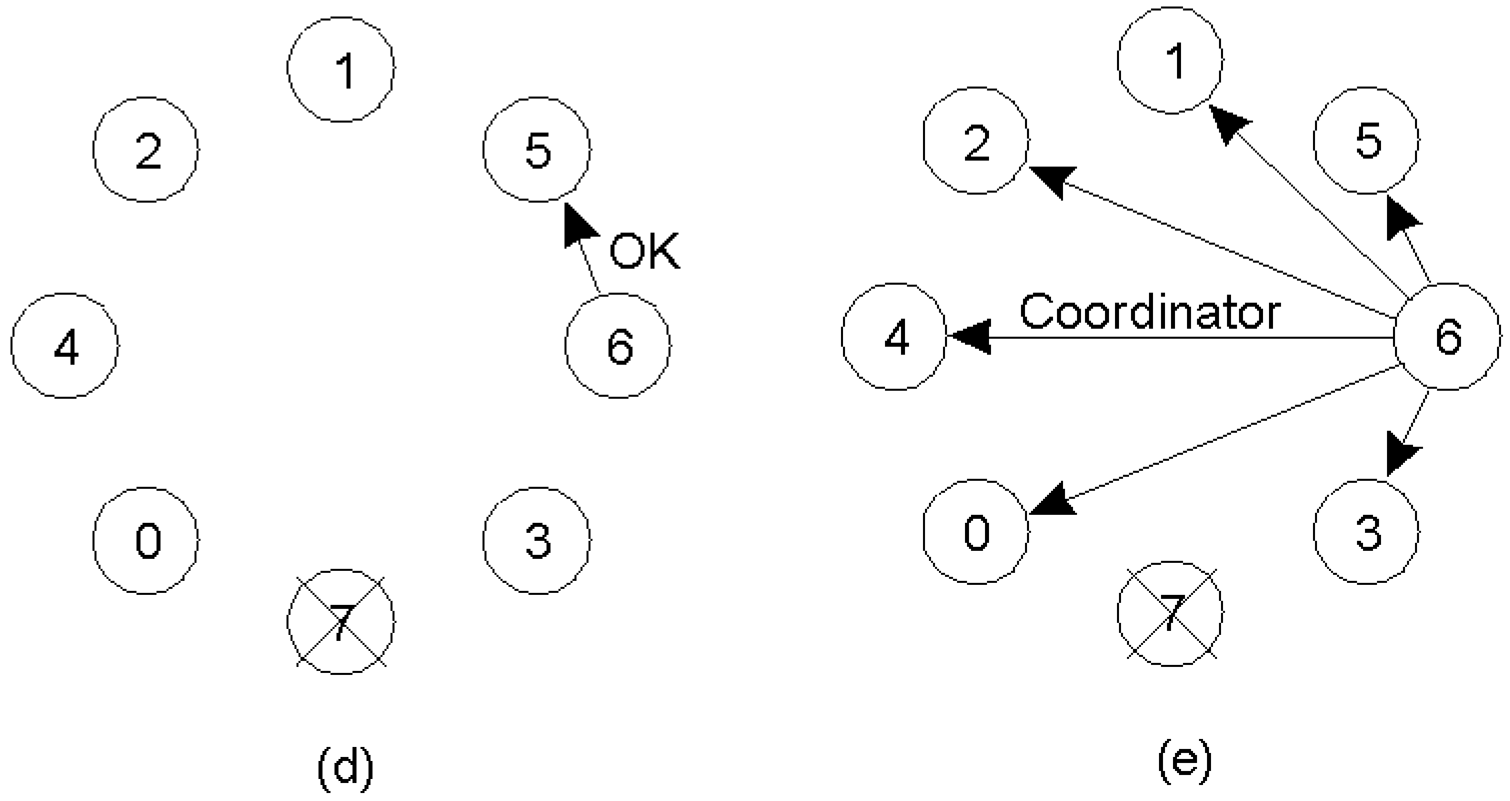
The Bully Algorithm (1)



The bully election algorithm

- Process 4 holds an election
- Process 5 and 6 respond, telling 4 to stop
- Now 5 and 6 each hold an election

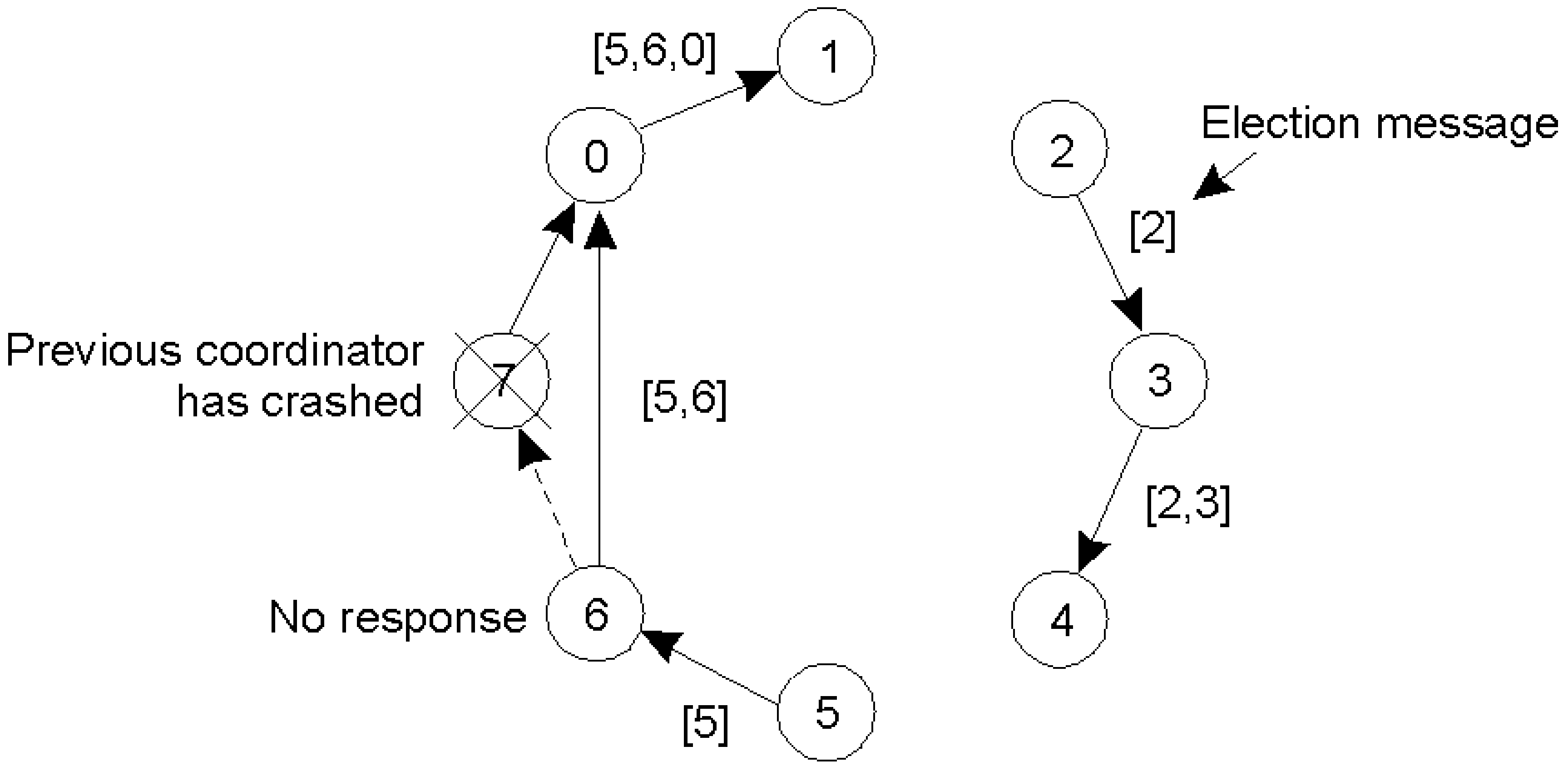
The Bully Algorithm (2)



d) Process 6 tells 5 to stop

e) Process 6 wins and tells everyone

A Ring Algorithm



Superpeer Election

- How can we select superpeers such that:
  - Normal nodes have low-latency access to them
  - Superpeers are evenly distributed
  - There is a predefined fraction of superpeers
  - Each superpeer does not serve more than a fixed number of normal nodes

Superpeer Election in DHTs

- Reserve a fixed part of ID space for superpeers
- S = desired number of superpeers
- m = bit-length of keys
- Reserve $k = \lceil \log_2 S \rceil$ bits for superpeers

- Routing to superpeers: send message for key p
- to node responsible for p & $\underbrace{11...11}_{k}00...00$

Thank you for your attention!