EECS 591
Distributed Systems

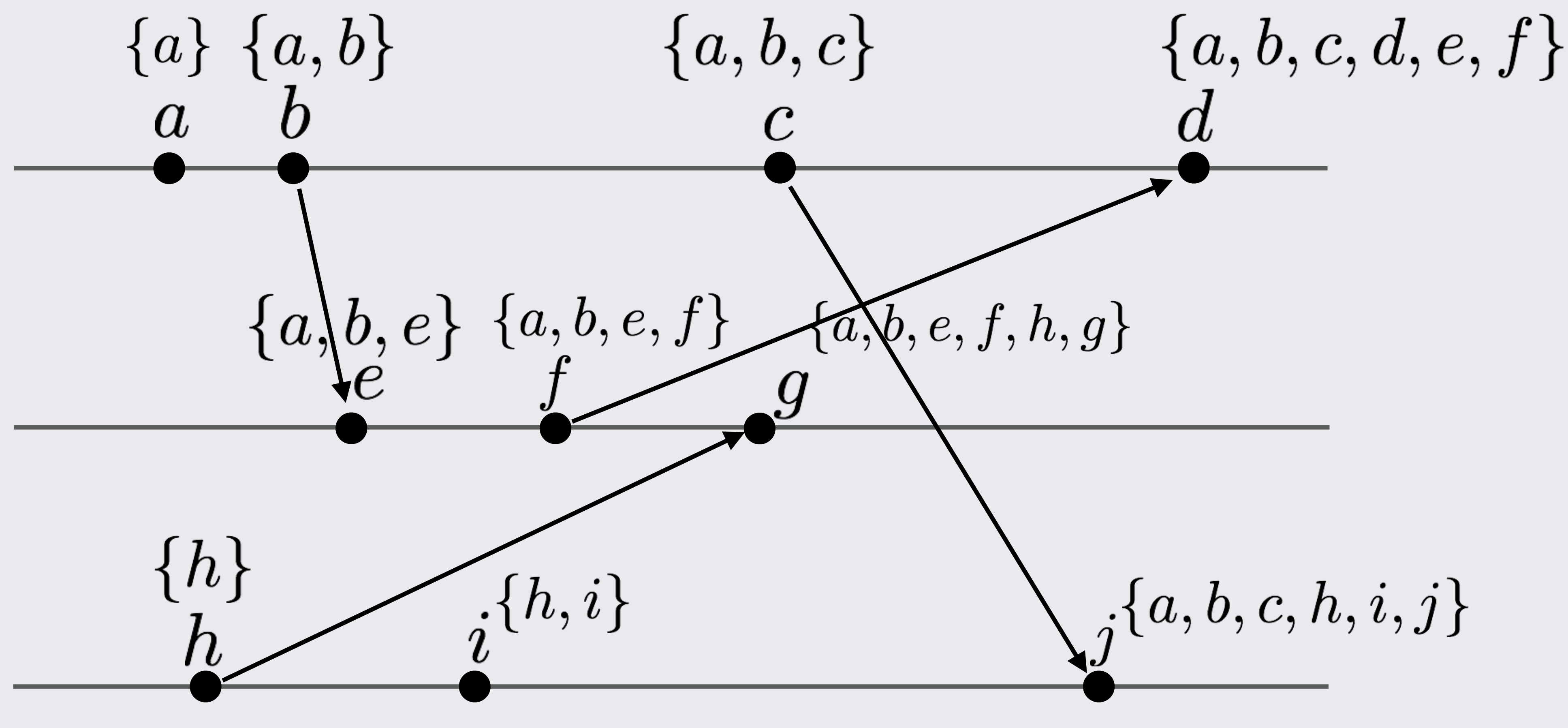
Manos Kapritsos

Fall 2021


PREVIOUSLY ON
DISTRIBUTED SYSTEMS

Implementing strong clocks

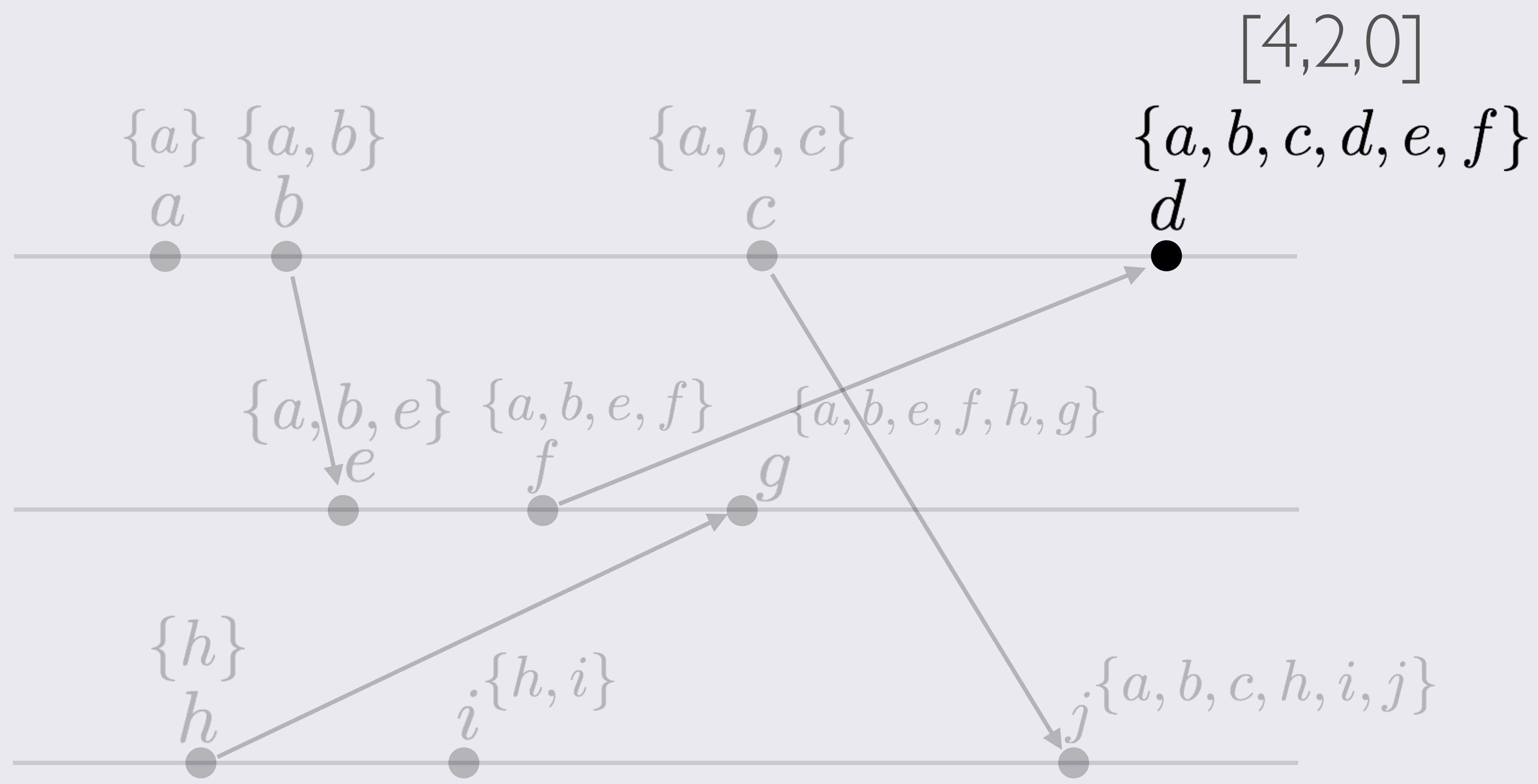
(the hard way)



Strong clock condition: $p \to q \Leftrightarrow \theta(p) \subset \theta(q)$

IMPLEMENTING STRONG CLOCKS

(the hard way)



Strong clock condition: $p \to q \Leftrightarrow \theta(p) \subset \theta(q)$

VECTOR CLOCKS

Each process keeps a vector of natural numbers **VC**, one for each process

Update rules

If e_i is a local or send event at process i:

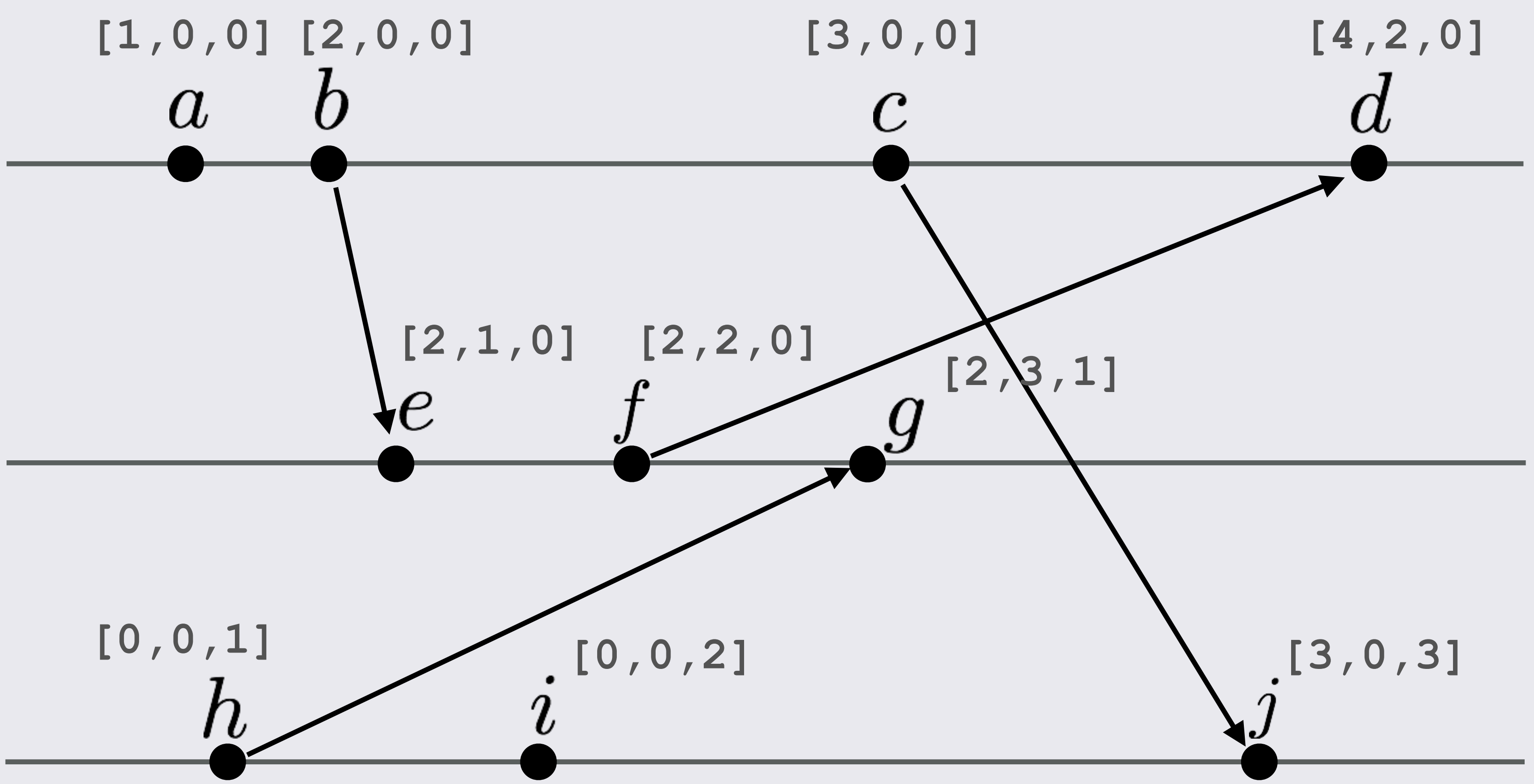
$VC(e_i)[i] := VC[i] + 1$ (Update the "local" counter)

If e_i is a receive event of message m:

$VC(e_i) := max\{VC, VC(m)\}$ (First "max" with the incoming VC…)

$VC(e_i)[i] := VC[i] + 1$ (…then update the "local" counter)

VECTOR CLOCKS



$VC(e_i)[j]$ = number of events executed by process j that causally precede e_i

Comparing vector clocks

Equality

$$V = V' \equiv \forall k : 1 \leq k \leq n : V[k] = V'[k]$$

(i.e. all elements are the same)

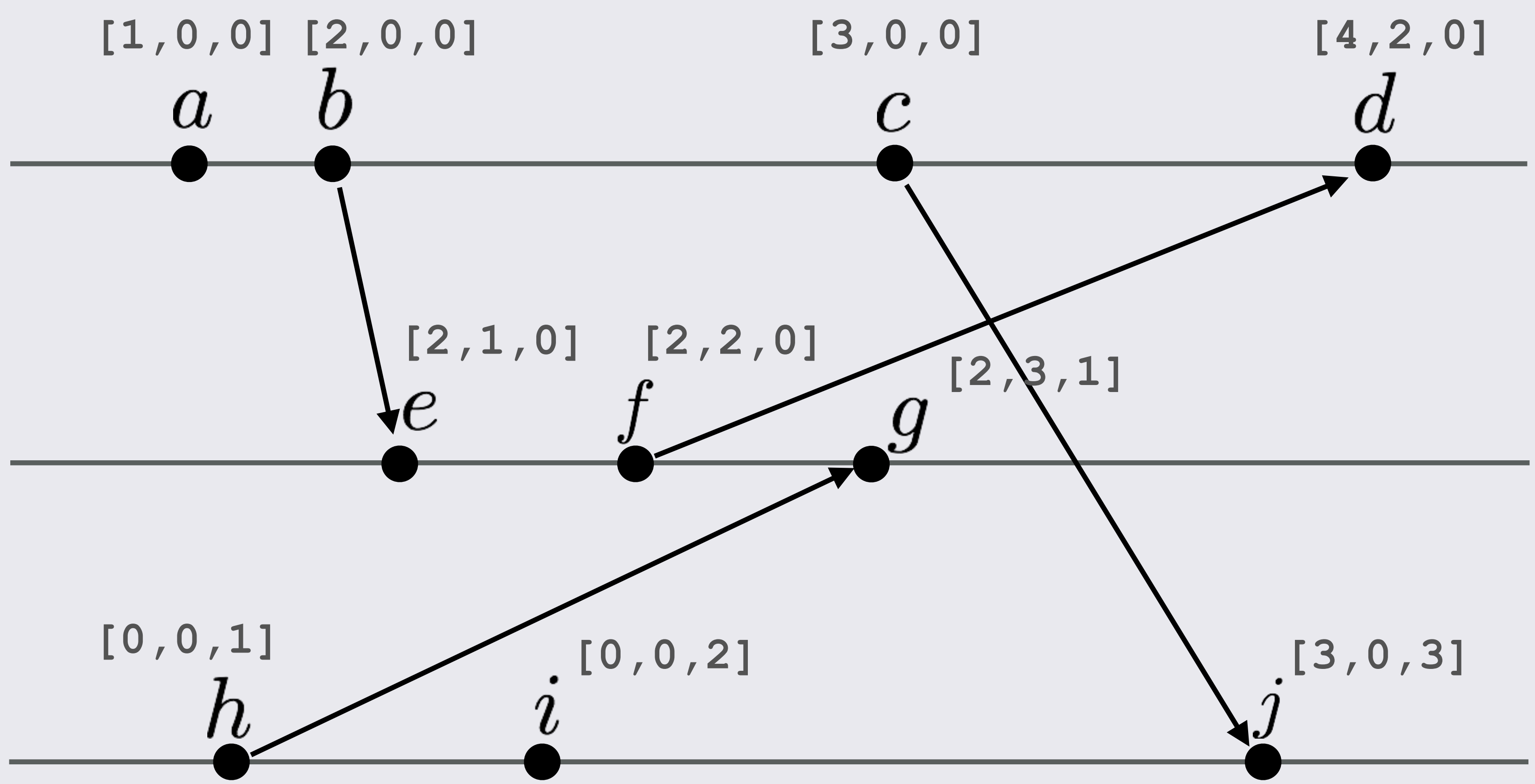
Inequality

$$V < V' \equiv (V \neq V') \wedge (\forall k : 1 \leq k \leq n : V[k] \leq V'[k])$$

Examples: `[2,0,0] < [2,0,1] < [3,0,1] < [4,1,1]`

Strong clock condition: $p \rightarrow q \Leftrightarrow VC(p) < VC(q)$

COMPARING VECTOR CLOCKS



Strong clock condition: $p \rightarrow q \Leftrightarrow VC(p) < VC(q)$

CAUSAL DELIVERY

A "monitor" process wants to record all messages
(e.g. deadlock detection, system snapshot, etc)

- Processes send copies of their messages to the monitor
- Only increment the local component of **VC** for send events

Causal delivery rules

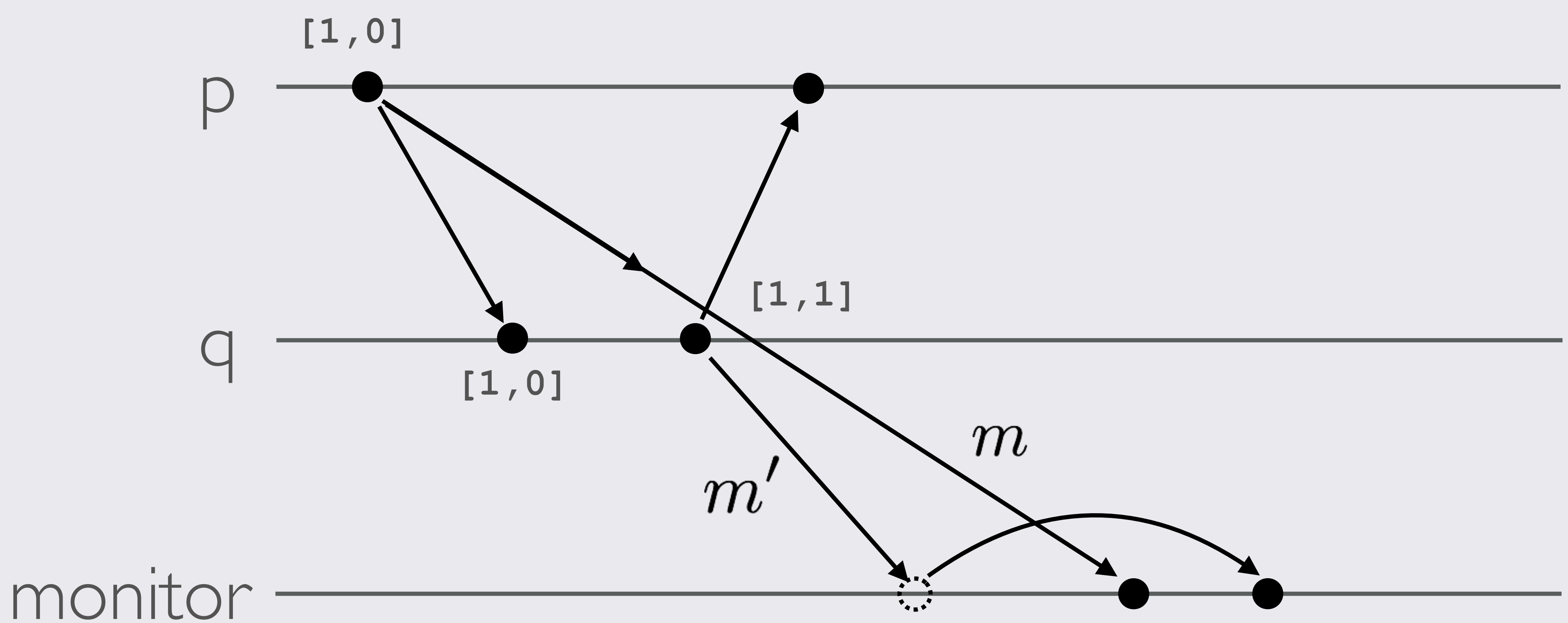
Monitor keeps an array D, where $D[i]$ is the number of messages delivered from process i

Monitor delivers message m from process j when:

$$D[j] = VC(m)[j] - 1$$

$$D[k] \geq VC(m)[k], \forall k \neq j$$

Causal delivery

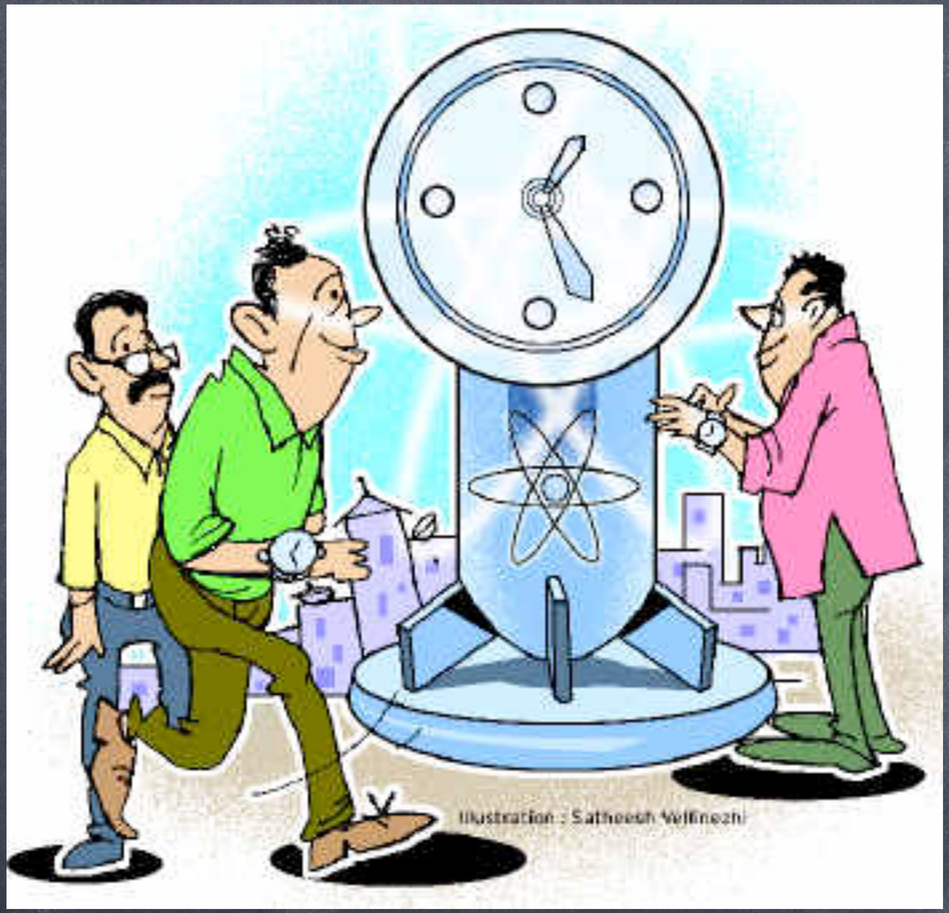


D	$(0,0)$	$(1,0)$	$(1,1)$
$D[j] = VC(m)[j] - 1$	✓	✓	✓
$D[k] \geq VC(m)[k], \forall k \neq j$	✗	✓	✓

Administrivia

- Remember to send me your picture if you haven't already

Clock synchronization



What time is it?

Roman generals v2.0

Attack at midnight!

12:00am

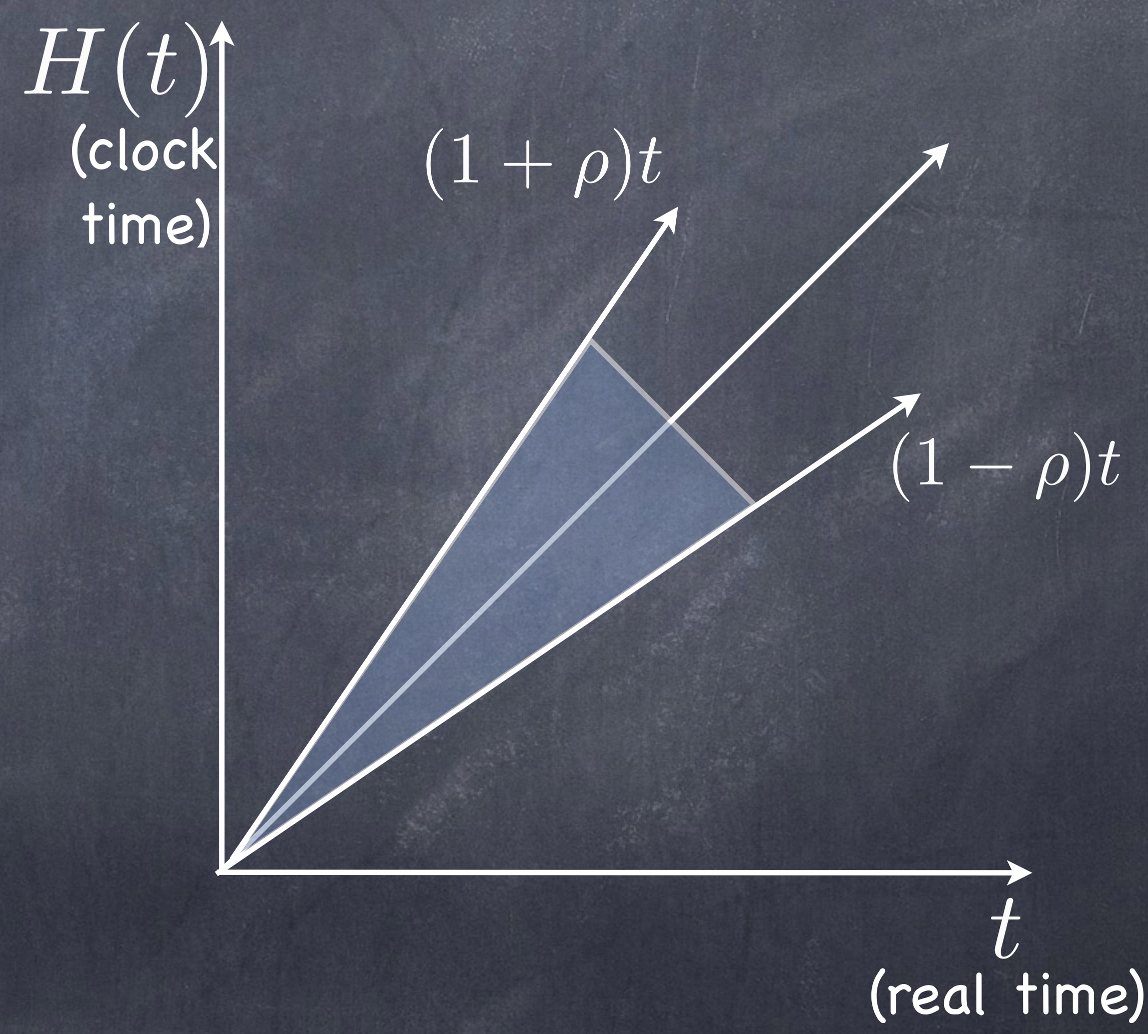
11:30pm

Clock drift

- Bound on drift: ρ

$$(1-\rho)(t-t') \leq H(t) - H(t') \leq (1+\rho)(t-t')$$

- ρ is typically small (10^{-6})
  - $\rho^2 \approx 0$
  - $\dfrac{1}{1-\rho} = 1+\rho$
  - $\dfrac{1}{1+\rho} = 1-\rho$



External vs internal synchronization

External Clock Synchronization:

keeps clock within some maximum deviation from an external time source.

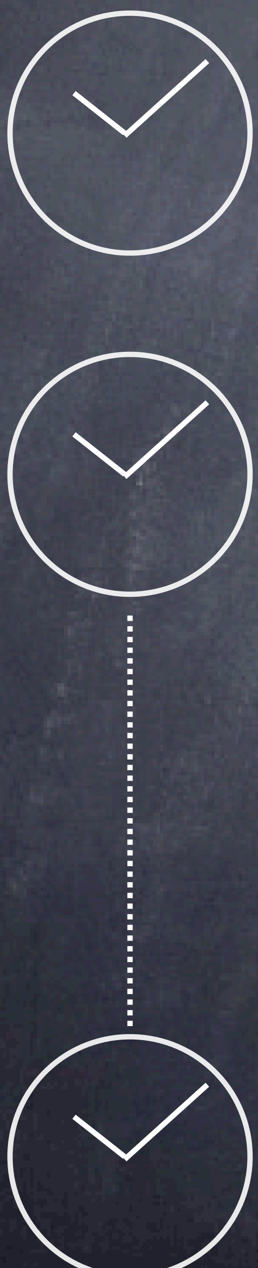
- exchange of info about timing events of different systems
- can take actions at **real-time** deadlines

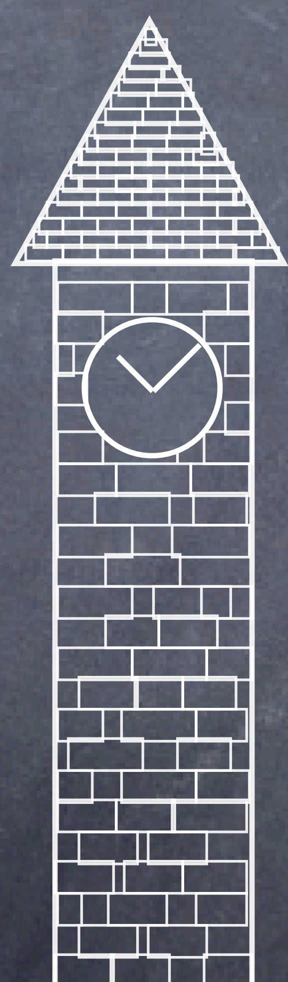
Internal Clock Synchronization:

keeps clocks within some maximum deviation from each other.

- can measure **duration** of distributed activities that start on one process and terminate on another
- can totally order events that occur in a distributed system

Probabilistic Clock Synchronization (Cristian)



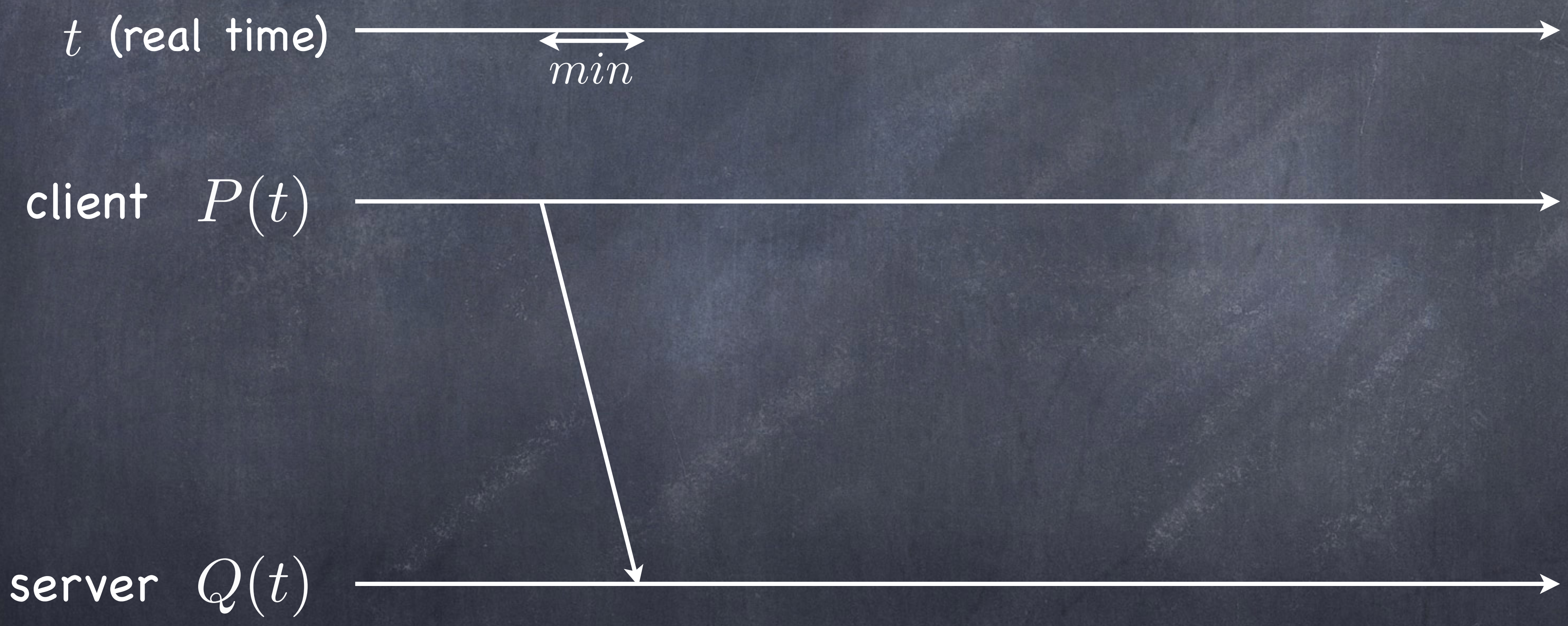


- Client-server architecture
- Server can be connected to external time source
- Clients read server's clock and adjust their own

How accurately can a client read the server's clock?

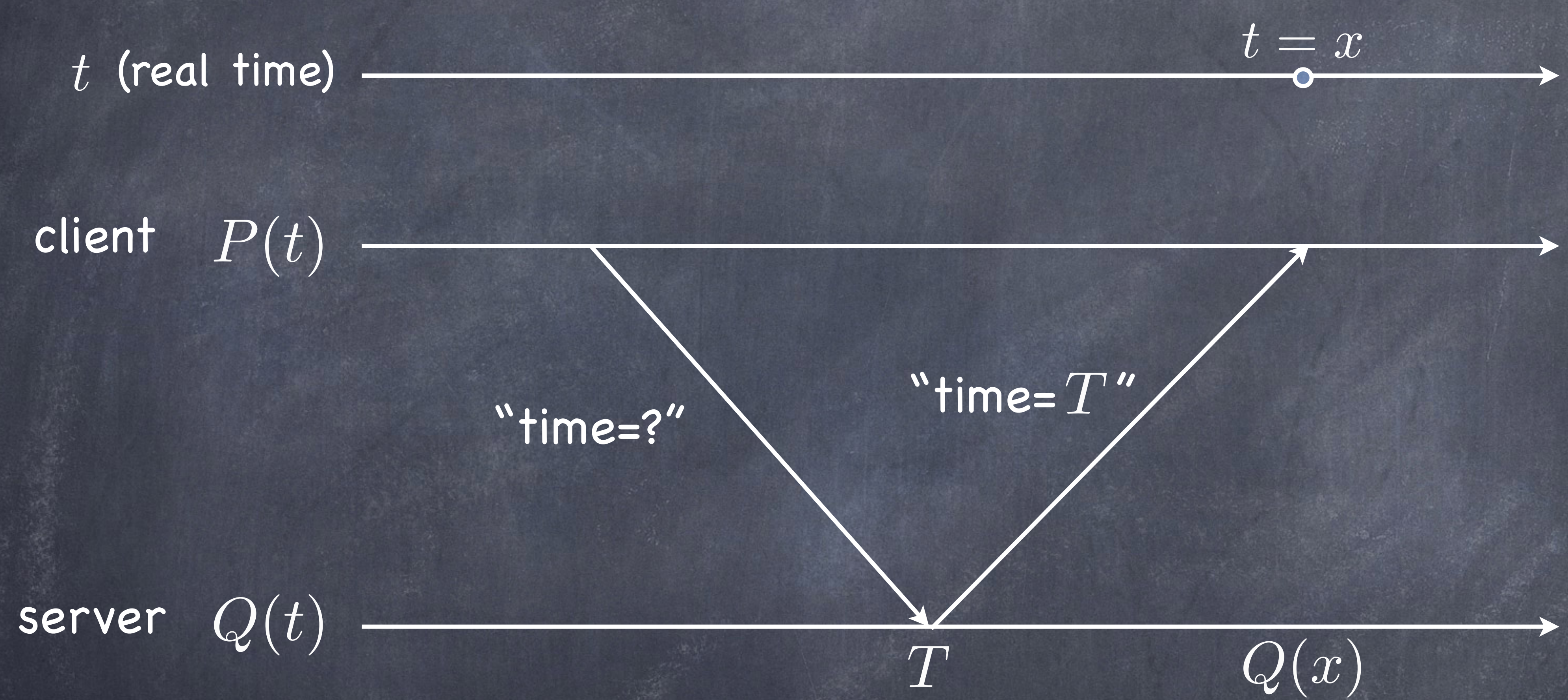
Setup and assumptions

Goal: Synchronize the client's clock with the server



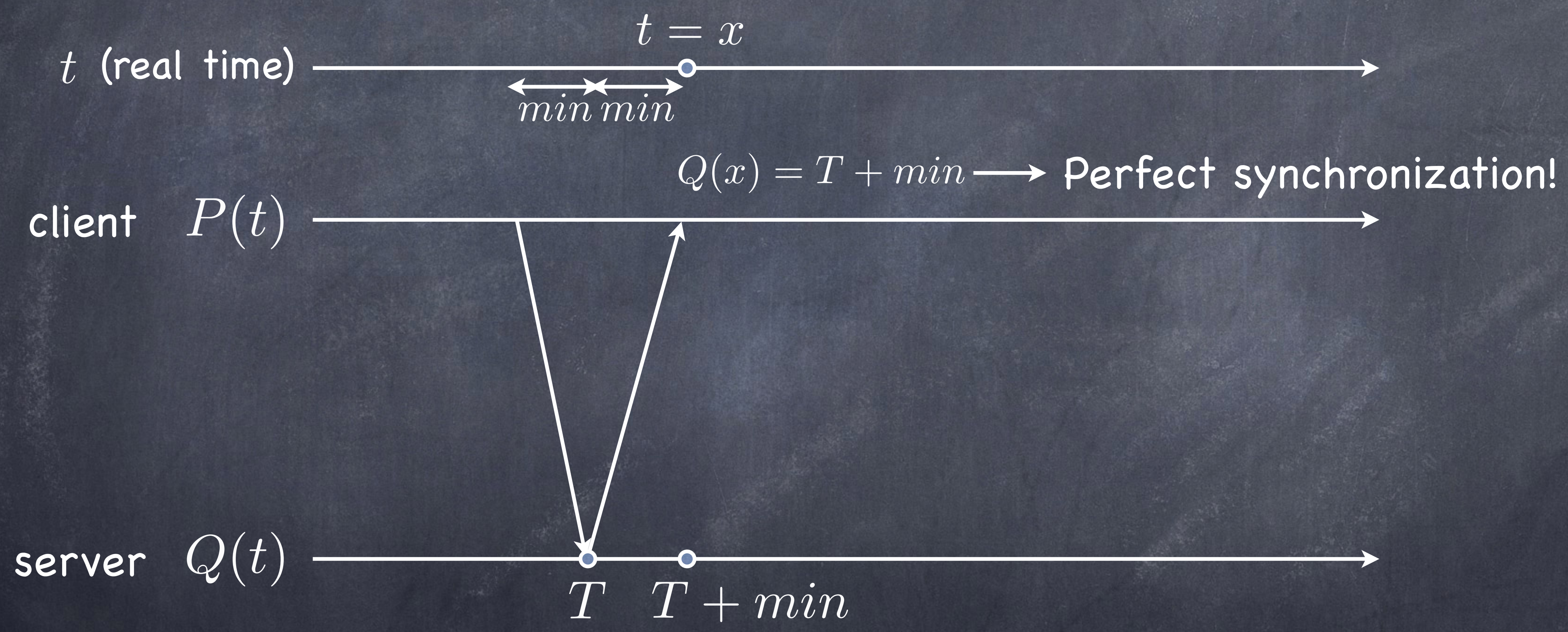
Assume that minimum delay is known
Assume that clock drifts are known (ρ for both)

The protocol



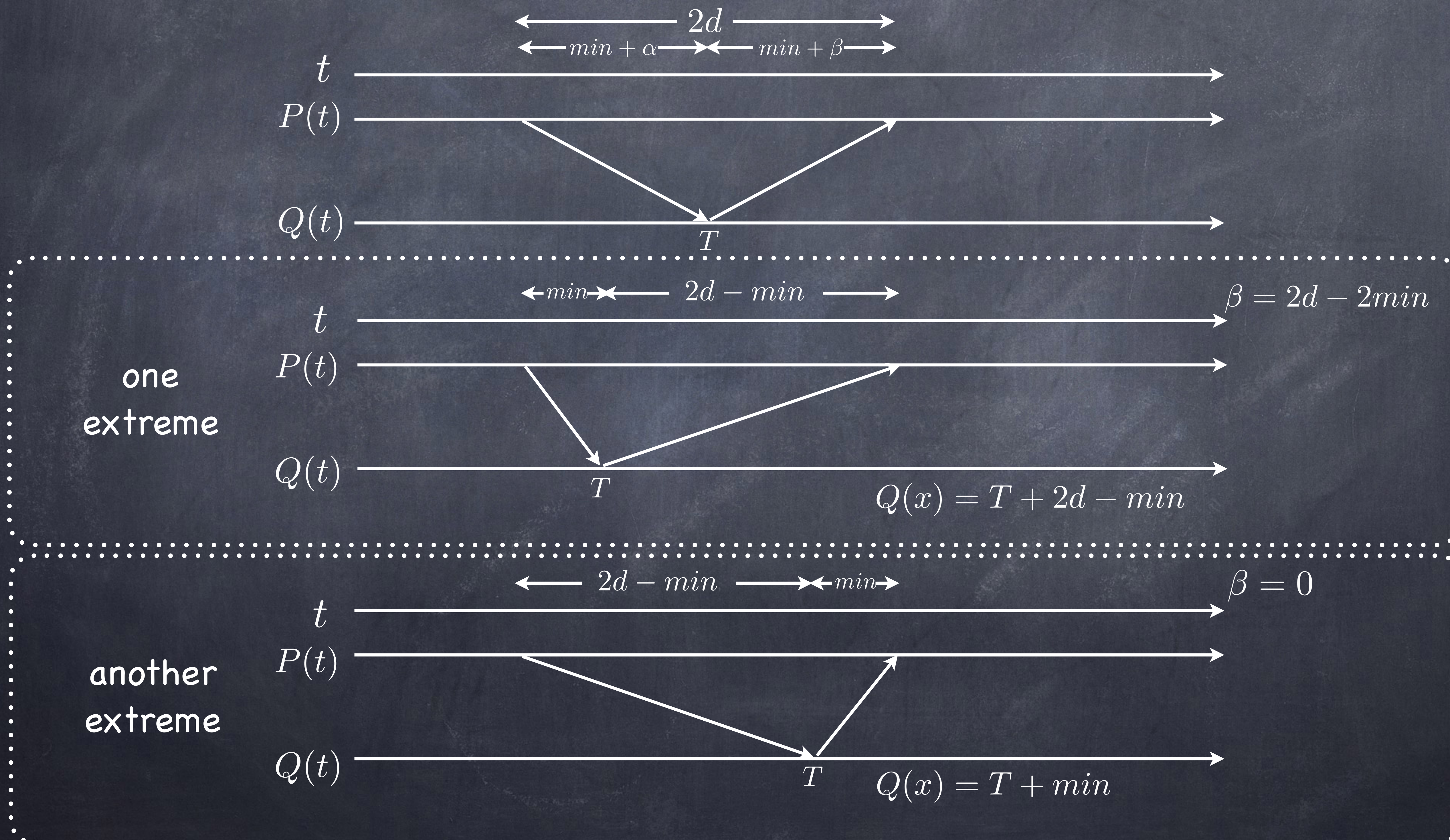
Question: what is $Q(x)$?

Ideal scenario

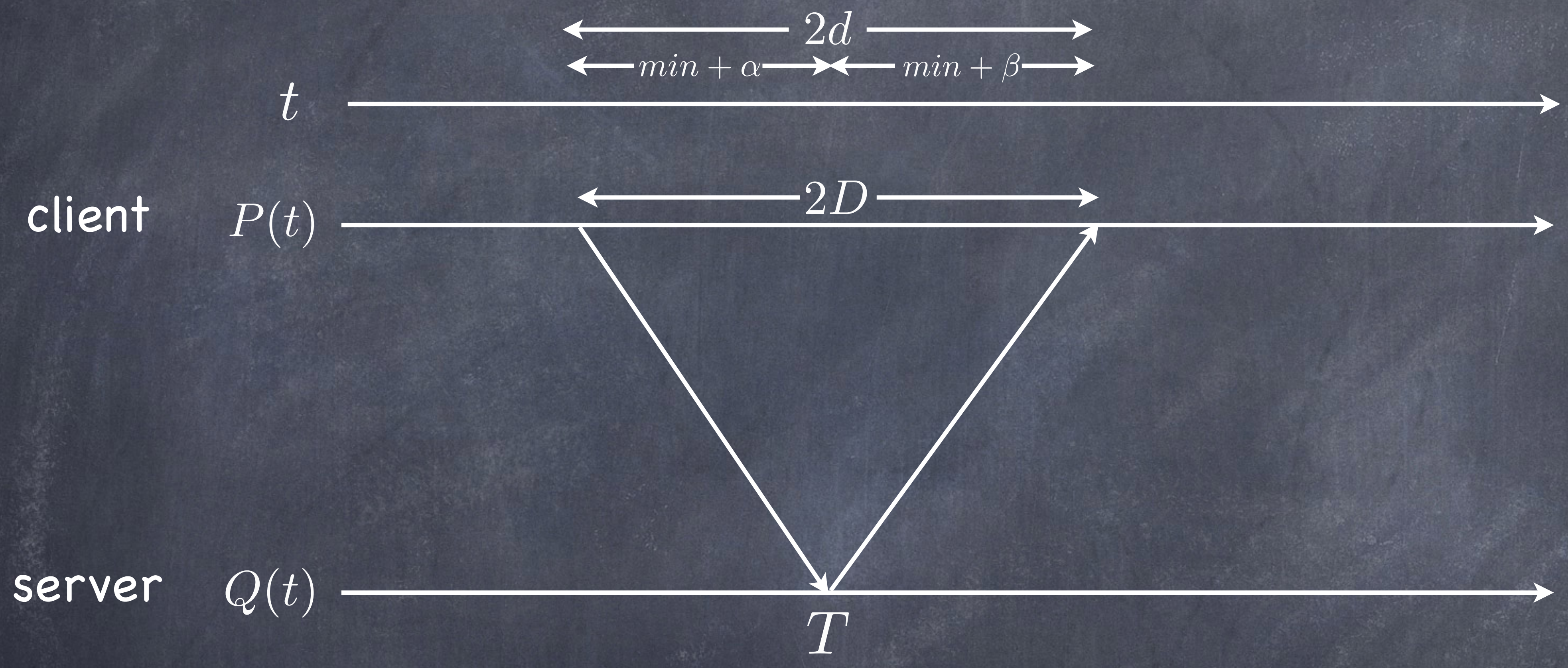


Assume no clock drift

Problem #1: message delay

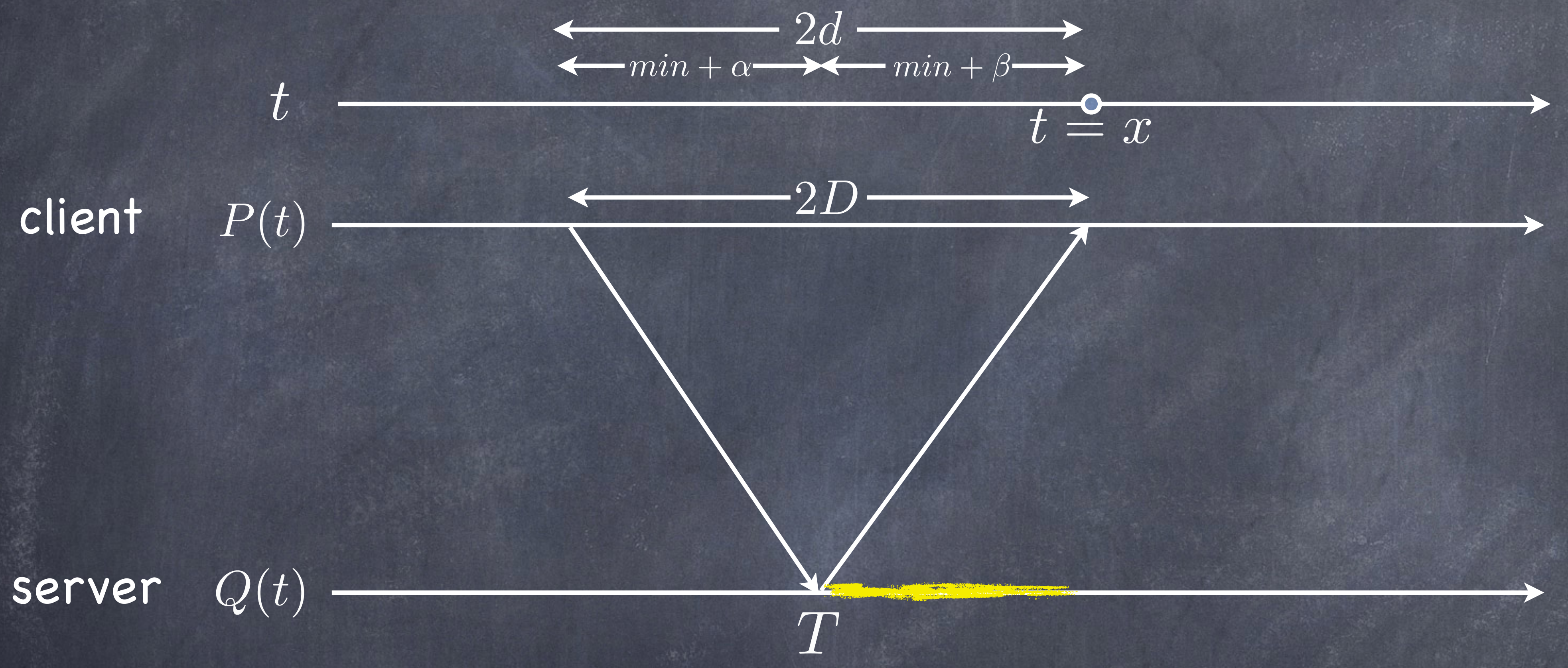


Problem #2: client drift



$$2d(1-\rho) \leq 2D \leq 2d(1+\rho)$$

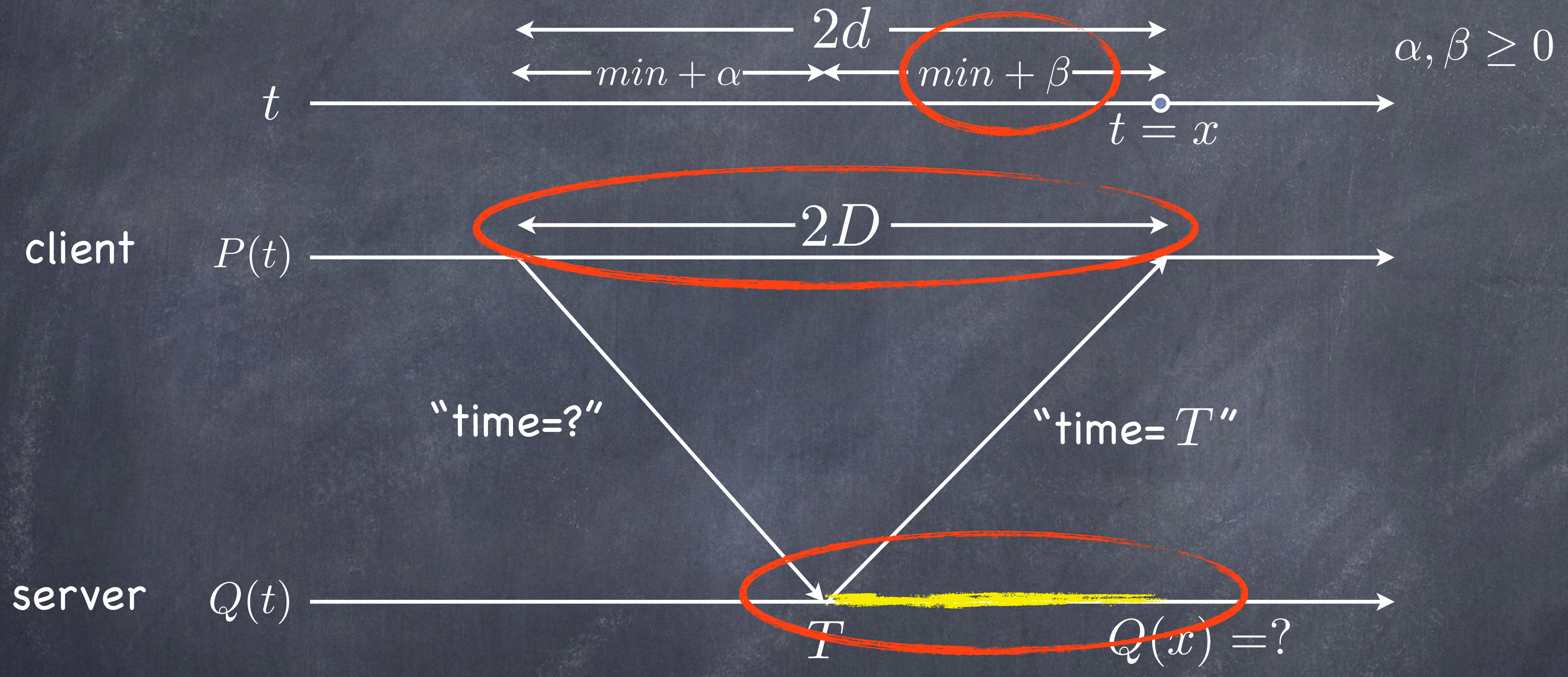
Problem #3: server drift



During ▬▬ the server's clock drifts

Even if you know β, there is still some uncertainty!

Cristian's algorithm



Cristian's algorithm

Naive estimation: $Q(x) = T + (min + \beta)$

(take server's drift into account)

$$Q(x) \in [T + (min + \beta)(1 - \rho), T + (min + \beta)(1 + \rho)]$$

$0 \leq \beta \leq 2d - 2min$ (take delay into account)

$$Q(x) \in [T + (min+0)(1 - \rho), T + (min+2d - 2min)(1 + \rho)]$$
$$= [T + (min)(1 - \rho), T + (2d - min)(1 + \rho)]$$

$2d \leq 2D(1 + \rho)$ (take client's drift into account)

$$Q(x) \in [T + (min)(1 - \rho), T + (2D(1 + \rho) - min)(1 + \rho)]$$
$$= [T + (min)(1 - \rho), T + 2D(1 + 2\rho) - min(1 + \rho)]$$

Client's estimation and precision

Client's best guess: $Q(x) = T + D(1+2\rho) - min \cdot \rho$

Maximum error: $e = D(1+2\rho) - min$

You can keep trying, until you achieve the required precision

(if that precision is reasonable)