Denbora, kausaltasuna eta egoera globala

Sistema Banatuak

Mikel Larrea, KAT Saila, UPV/EHU

Denbora, kausaltasuna eta egoera globala

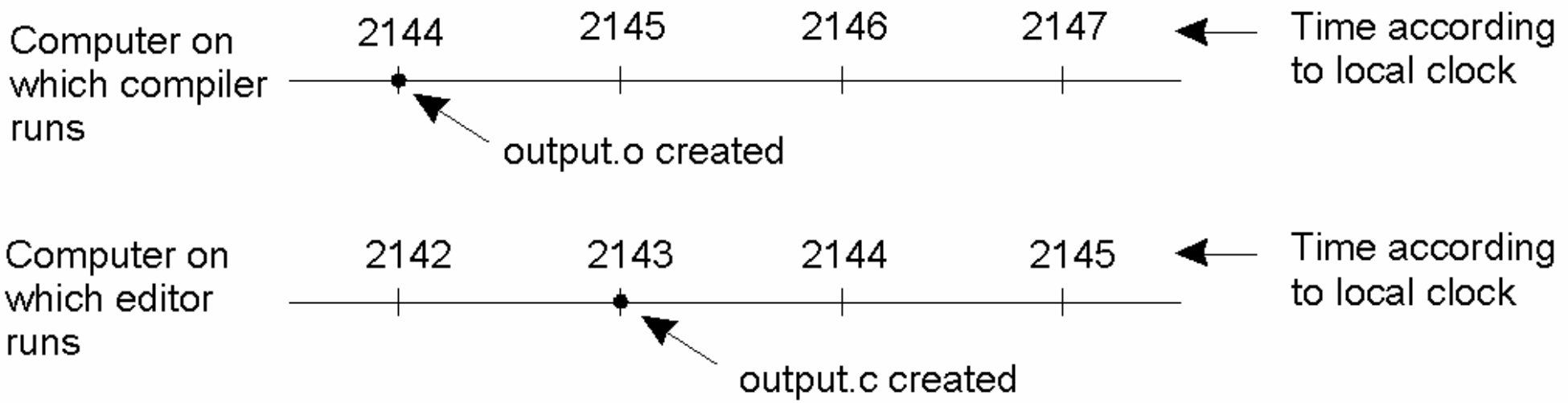
1 Sarrera

- Sistema banatuetan *egoera globala* nodo desberdinetan banatua dago
  - erloju estandarrak nodoetan
  - nodoen arteko komunikazioa: atzerapenak
  - nodoek egoera globalaren ikuspegi subjektiboa dute
- Adibideak:
  - mundu osoko banketxe guztiko kontuen saldoak une konkretu batean jakitea
  - lur planetako 6.000 milioi. biztanlea nor den jakitea (bataz beste, 5 jaiotza segundoko, sateliteren bidez seinalea transmititzeko segundo bat behar delarik)

1 Sarrera

- Soluzio proposamena: erloju zehatz bakarra + sare dedikatua seinalea atzerapenik gabe transmititzeko
  - ez da praktikoa (kostuagatik), edota eginezina (Internet)
- Soluzioa: denbora banatua
  - nodo bakoitzak bere erlojua du (denbora fisikoa lokala)
  - erloju hauen prezisioa mugatua denez, erreferentziazko denbora fisikoarekiko sinkronizatzen dira periodikoki
- Honetaz gain, egoera globalaren kudeaketarako gertaeren arteko ordena behar da (kausaltasuna):
  - denbora logikoa

Clock Synchronization

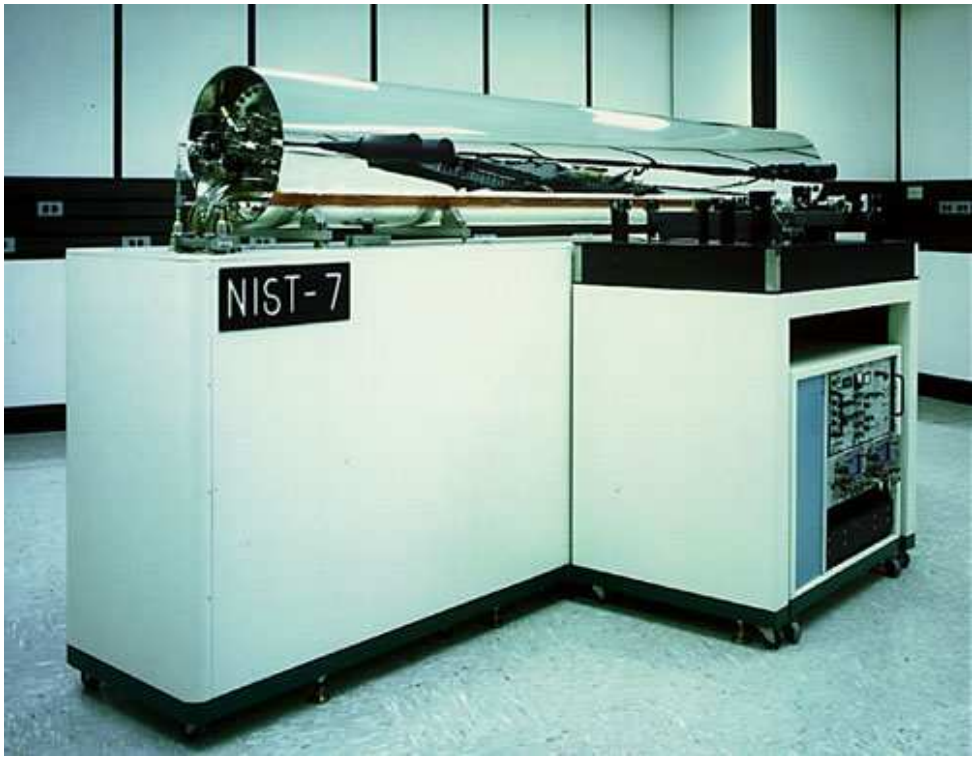


- When each machine has its own clock, an event that occurred after another event may nevertheless be assigned an earlier time.

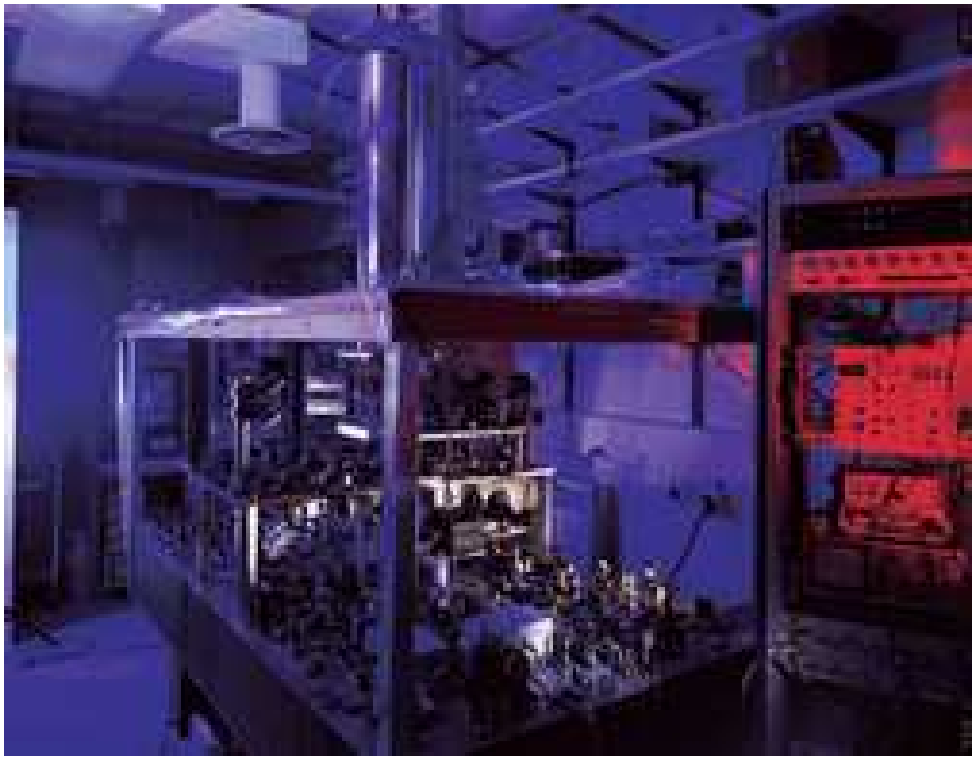
2 Denbora fisikoa

- Konputagailuetako erlojuak kuartzozkoak dira
  - oszilazio frekuentzia tenperaturarekin aldakorra da
  - deriba: ~10^{-6} (90ms-ko aldea egunero, 1s-koa 11,6 egunero)
- Erloju atomikoak: prezisio handia, oso garestiak
  - deriba: ~10^{-13} (9ns-ko aldea egunero, 1s-koa 300.000 urtero)
  - prezioa: $50.000 - $100.000 !!!
- Sinkronizazio mailak:
  - gertaera bat nodo baten noiz gertatu den jakitea: kanpoko sinkronizazioa
  - bi nodo desberdineko gertaeren arteko denbora tartea neurtzea, nodo bakoitzeko erloju lokala erabiliz: barneko sinkronizazioa
  - barnekoak ez du kanpokoa inplikatzen, baina alderantziz bai

Erloju atomikoak



NIST-7 (1993)
Deriba: 5×10^{-15}



NIST-F1 (2005)
Deriba: 5×10^{-16}

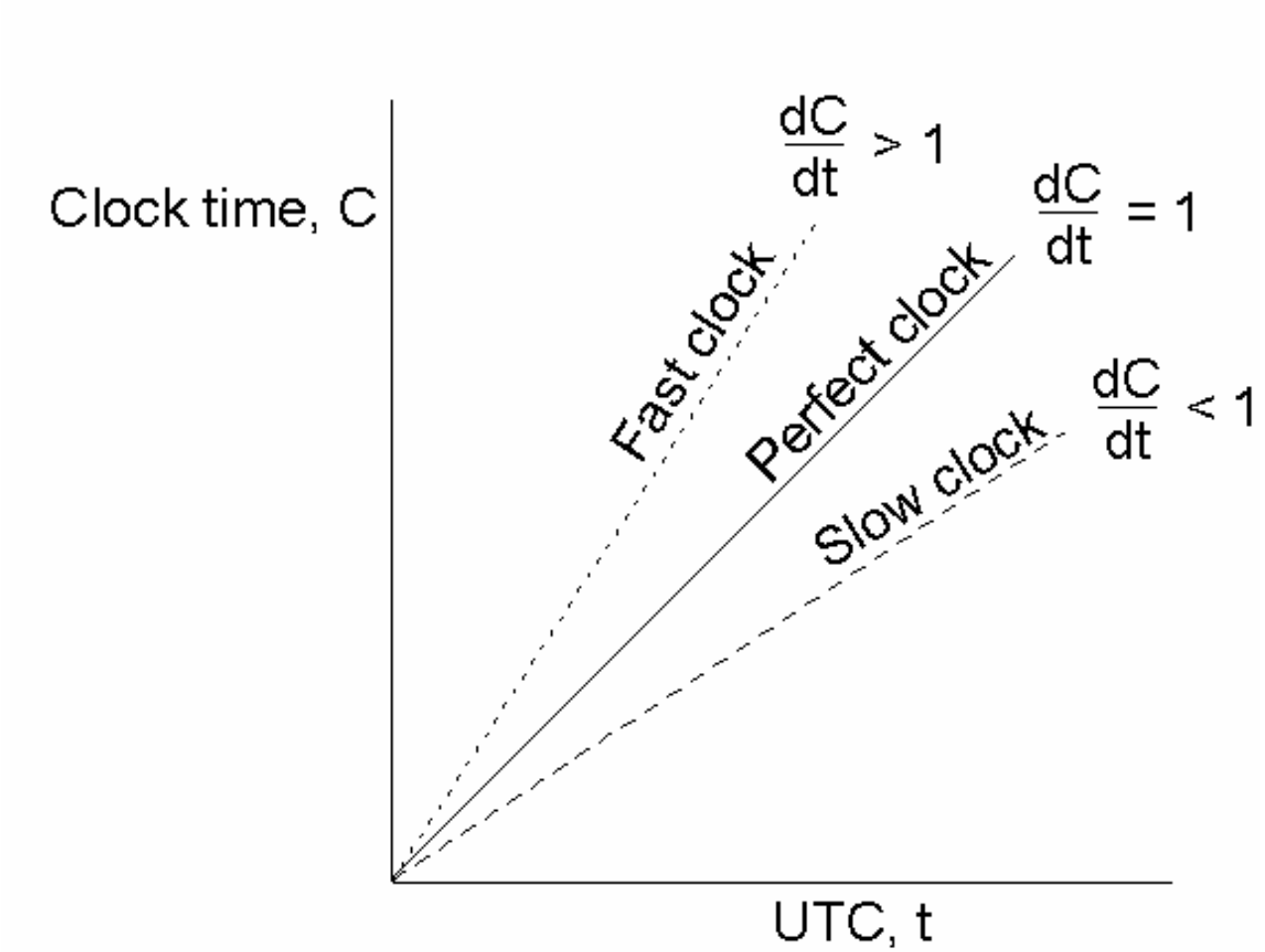
2 Denbora fisikoa

- Definizioak:
  - Eguzki-segundoa edo segundo astronomikoa: lurraren errotazio periodoaren 1/86.400 da (*mean solar second*)
    - gure eguneroko bizitzarako guztiz baliozkoa izan arren, lurraren abiadura ez da konstantea (pixkanaka geldotzen ari da)
  - Segundo atomikoa (*IAT*, 1967): Zesio-133 (Cs^{133}) atomoaren 9.192.631.770 egoera aldaketen denbora da. Erloju atomikoek denbora hau neurtzen dute
    - $3*10^{-8}$ko aldea dago (~3 ms egunean, ~1 s urtean)
  - Denbora unibertsala koordinatua (*UTC*): denbora atomikoan oinarritua, eta denbora astronomikoarekin sinkronizatua (aldea > 0,9 s $\Rightarrow$ segundo bat gehitu)

2 Denbora fisikoa

- Definizioak:
  - Erreferentziazko denbora fisikoa: normalean *UTC*
  - Bereizmena (*resolution*): erloju lokalaren erregistroaren bi eguneratze arteko periodoa
    - Nodoko bi *gertaera* kontsekutiboen arteko tarte minimoa baino txikiagoa izan behar da
  - Desbideratzea (*offset*, *skew*, θ): denbora lokalaren eta erreferentziazko denbora fisikoaren arteko aldea
  - Deriba (*drift*, δ): denbora unitateko desbideratzea (erlojuak aurreratu edota atzeratu egiten duena)
  - Prezisioa (*accuracy*): erloju lokala doitzerakoan bermatu daiteken desbideratze maximoa

Clock Synchronization Algorithms



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

2.1 Kanpoko sinkronizazioa

- Sinkronizazioa:
  - erlojuaren balioa erreferentziazko denbora fisikoarekiko doitze prozedura, aurredefinitutako prezisioa betez
- *UTC* denbora periodikoki irrati uhinez hedatzen da
  - seinalearen transmisio abiadura: ~$3*10^8$ m/s (1.000 km: ~3 ms)
  - transmisio abiadura aldakorra da, prezisioa galtzen delarik
- Hargailu komertzialen prezisioa iturriaren arabera:
  - Lurreko estazioak: 0,1 - 10 ms
  - *GOES* sateliteak (35.786 km): 0,5 ms
  - *GPS* sateliteak (20.200 km): 1 μs
- Aplikazioak: ordu zerbitzua, kontabilitatea

2.2 Barneko sinkronizazioa

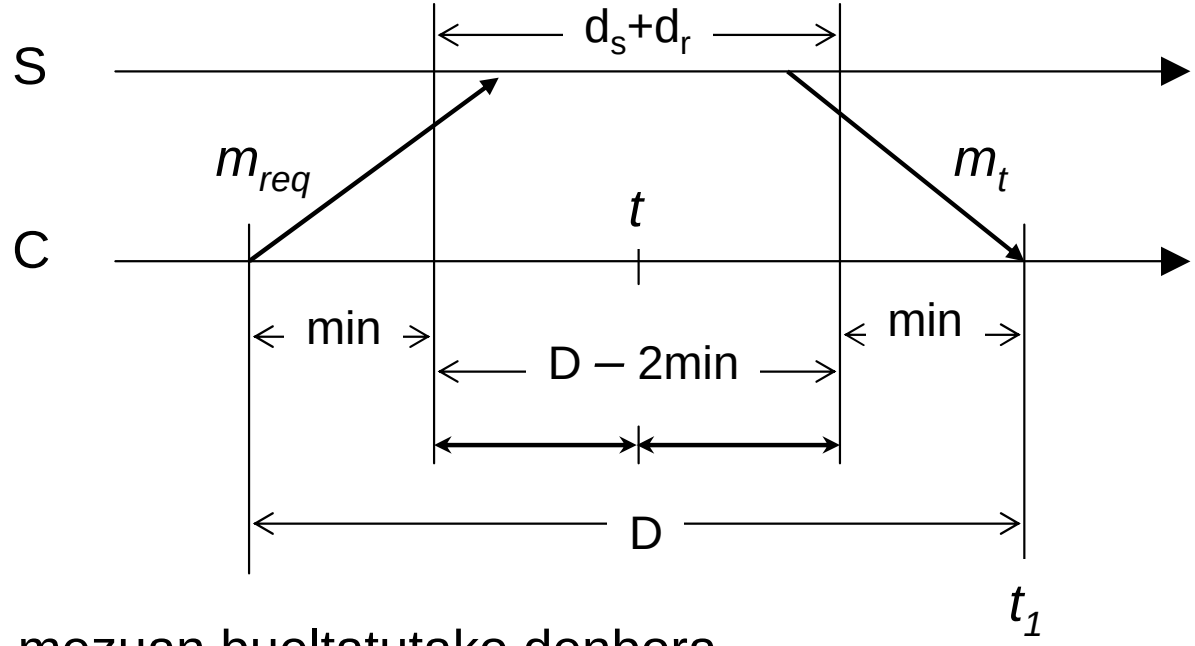
- Aplikazio banatu gehienetan nodoen erloju arteko sinkronizazioa kanpoko sinkronizazioa baino garrantzitsuagoa izaten da
  - gertaerei denbora-markak ezartzeko erabiltzen dira, gertaeren ordenazioa posible eginez
- Soluzio proposamena: nodo bakoitzak *UTC*rekiko sinkronizatzea
  - ez da praktikoa (kostuagatik)
- Soluzioa: barneko sinkronizaziorako algoritmoak
  - zentralizatuak: denbora zerbitzarietan oinarrituak
  - banatuak: estatistikoak

2.2.1 Algoritmo zentralizatuak

- Printzipioa:
  - denbora zerbitzari fidagarri bat erabiltzen du (adibidez, periodikoki *UTC*rekiko sinkronizatzen dena)
  - bezeroak zerbitzariari denbora eskatzen dio (m_{req})
  - zerbitzariak bere denbora erreferentzia bueltatzen dio bezeroari (m_t)
  - mezuen transmisioaren atzerapena: *min* + *d*
    - *min* transmisio denbora minimoa da. Konstantea da, eta sarearen ezaugarrien arabera kalkulatu daiteke
    - *d* aldakorra da, sistemaren kargaren araberakoa
- Adibidea: *Cristian*-en algoritmoa (1989)

2.2.1 Algoritmo zentralizatuak

Cristian-en algoritmoa



t(m_t): zerbitzariak m_t mezuan bueltatutako denbora

D: m_{req} bidali eta m_t jaso arteko denbora

min: mezuen transmisio denbora minimoa da

Suposaketa: zerbitzariak *t(m_t) D-2*min* tartearen erdian asignatzen du (bezeroan t denborari dagokio). Horrela prezisiorik onena lortzen da.

Desbideratzea: $\theta = t - t(m_t) = t1 - D/2 - t(m_t)$

Prezisioa = *D/2 – min*

2.2.1 Algoritmo zentralizatuak

- *Cristian*-en algoritmoaren adibidea (3 eskaera):

Esk.	*D (ms)*	*t_1 (hh:mm:ss.ms)*	*$t(m_t)$ (hh:mm:ss.ms)*
(1)	*22*	*10:54:22.236*	*10:54:23.674*
(2)	*26*	*10:54:24.000*	*10:54:25.450*
(3)	*20*	*10:54:26.946*	*10:54:28.342*

  - Zein eskaera erabili beharko luke bezeroak? Zein litzateke zerbitzariarekiko prezisioa? Eta desbideratzea?

    Prezisioa = 10 *ms* Desbideratzea = 1406 *ms*

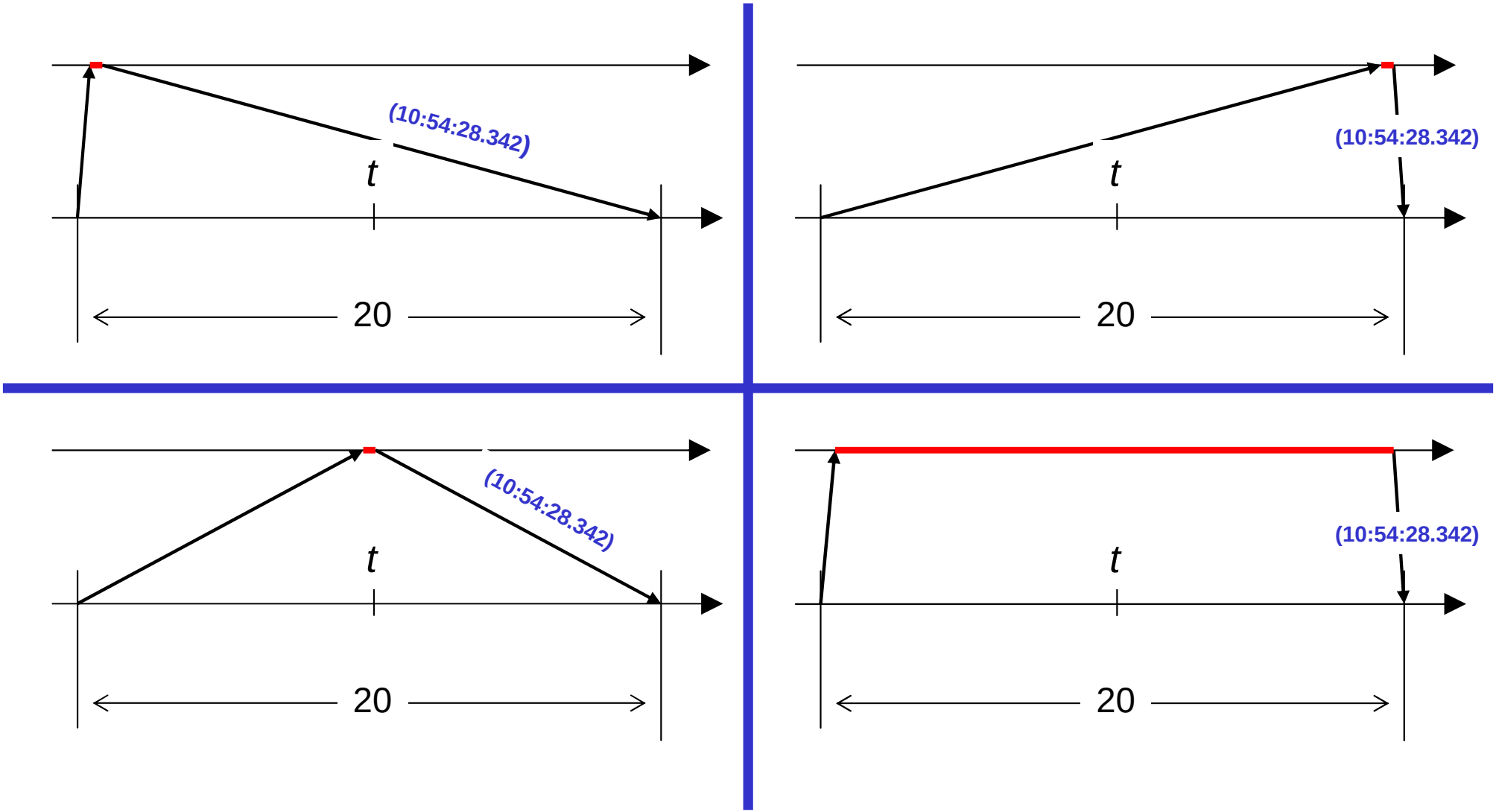
  - Transmisio denbora minimoa 7 ms dela jakinda, zerbait aldatzen al da aurreko erantzunetan?

    Prezisioa: 3 *ms*

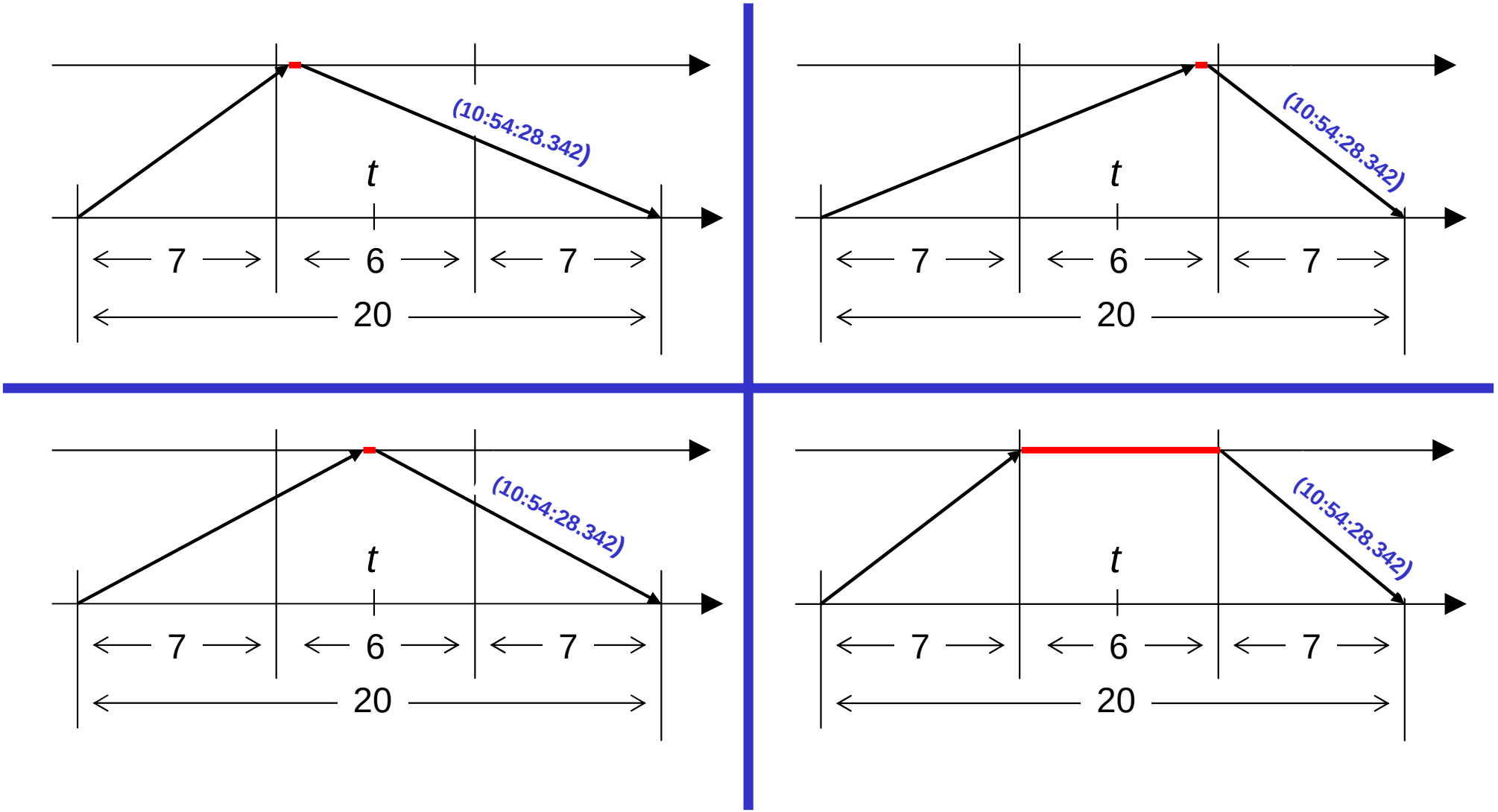
  - Zer beharko litzateke bezeroak 2 ms prezisioa lortzeko?

    min = 7 *ms* izanik, $D \leq$ 18 *ms*-ko eskaera

2.2.1 Algoritmo zentralizatuak



2.2.1 Algoritmo zentralizatuak



2.2.2 Algoritmo banatuak

- Algoritmo banatuen printzipio estatistikoa:
  - erlojuen desbideratzeak batezbesteko denborarekiko konpentsatzen dira. Denbora hau erreferentziatzat hartzen da barne sinkronizaziorako
- Adibidea: *Berkeley*-ko algoritmoa (1989)
  - koordinatzaileak beste nodoen denborak eskatzen ditu
  - koordinatzaileak batezbesteko denbora kalkulatzen du, neurri oso desbideratuak eta denboraz kanpo iritsitakoak baztertuz

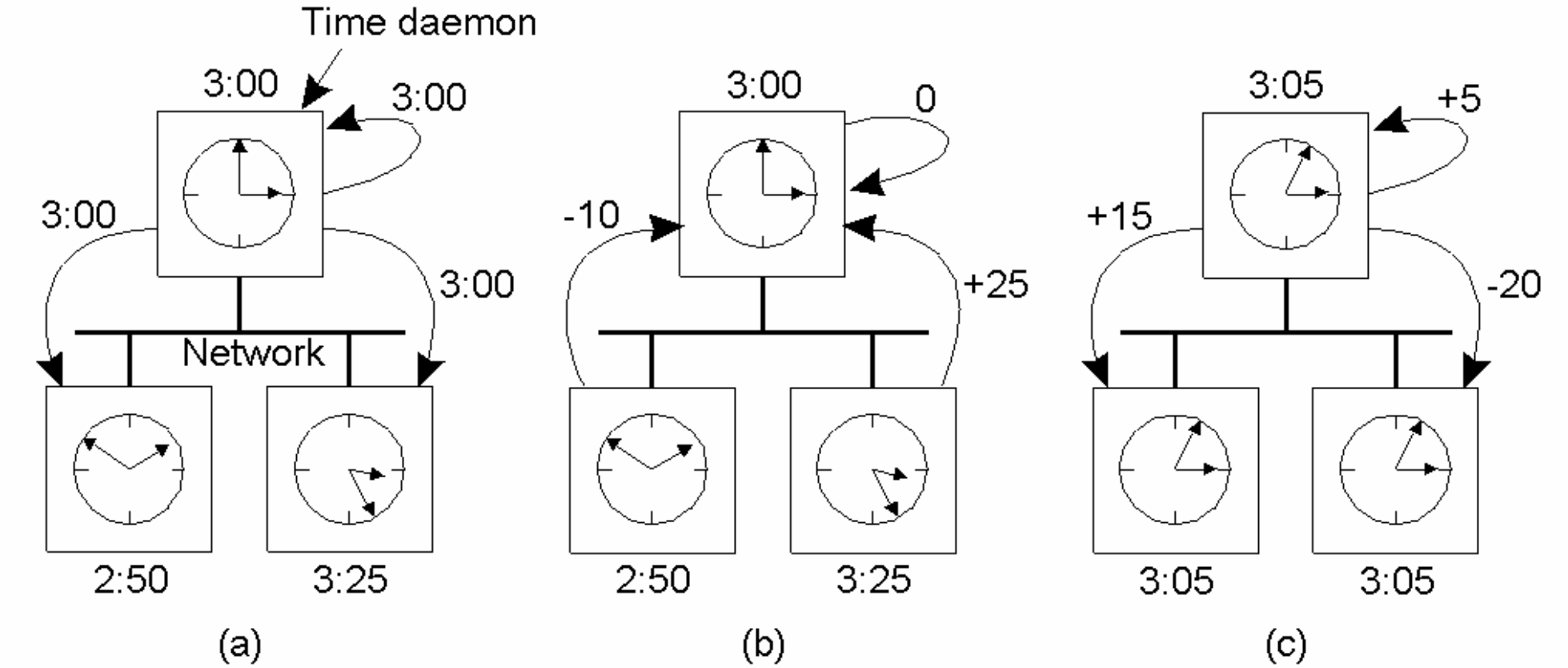
$$\mathbf{T = \Sigma(t(m_i) - D_i/2) / N}$$

  - azkenik, koordinatzaileak nodo bakoitzari duen desbideratzea (θ) bidaltzen dio, bere denbora sinkronizatu dezan

$$\mathbf{\theta_i = t(m_i) - D_i/2 - T}$$

  - koordinatzaileak huts egiten badu, berri bat hautatzen da

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

2.2.2 Algoritmo banatuak

- *Berkeley*-ko algoritmoaren adibidea:

	Nodoa	*D (ms)*	*t (hh:mm:ss.ms)*	***Desbideratzeak***
	N1 (koord.)	*0*	*10:54:23.118*	***-948 ms***
	N2	*22*	*10:54:22.236*	***-1842 ms***
	N3	*26*	*10:54:24.000*	***-79 ms***
Baztertua!	*N4*	*190*	*10:41:46.179*	***-757983 ms***
	N5	*20*	*10:54:26.946*	***+2870 ms***

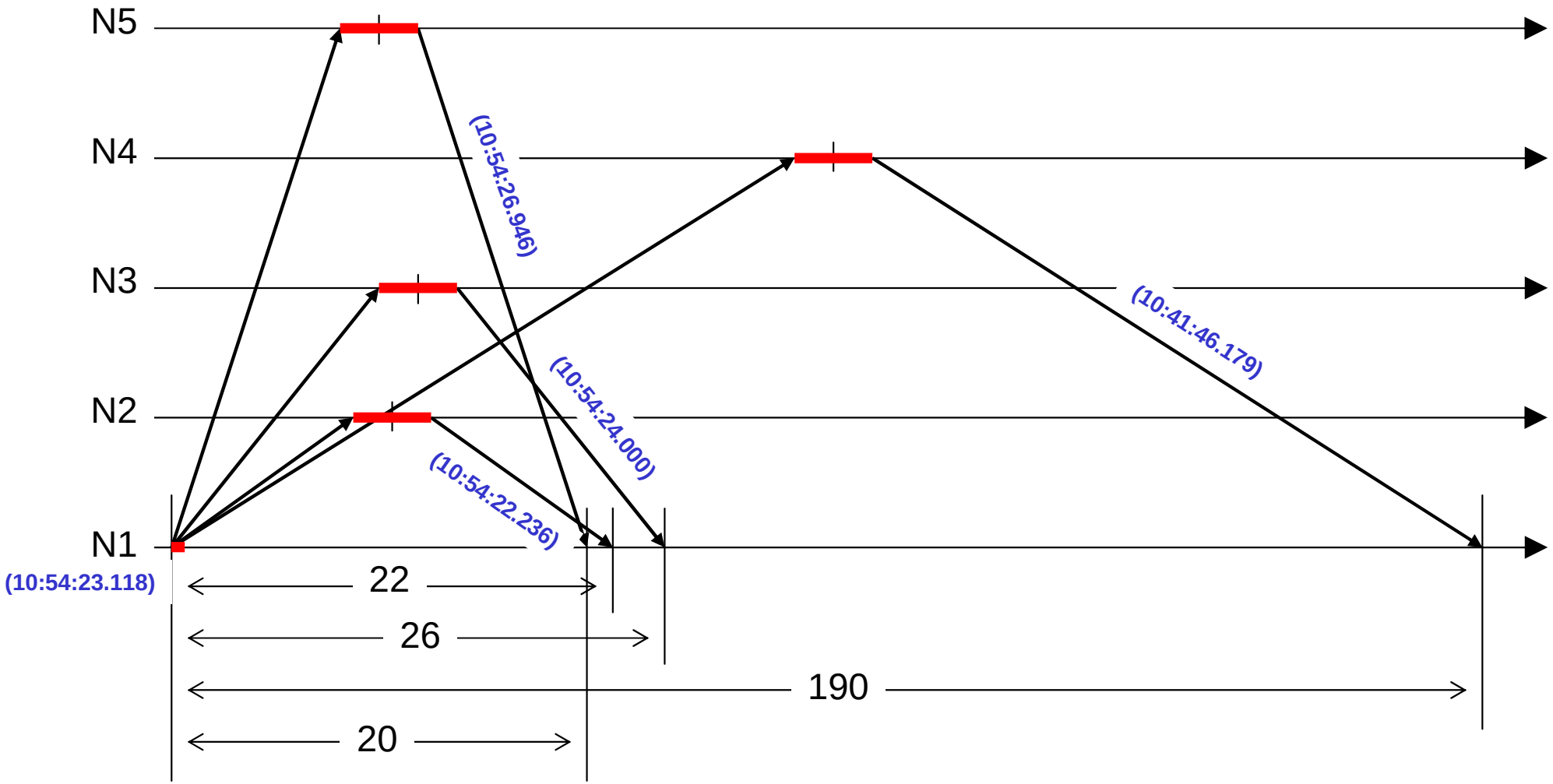
  - Kalkulatu sinkronizaziorako batezbesteko denbora eta nodo bakoitzari bidali beharreko desbideratzea

    Batezbesteko denbora = 10:54:24.067

  - *Cristian*-en metodoa erabiliz, kalkulatu nodo bakoitzaren prezisioa denbora sinkronizatzerakoan (*min* ezezaguna da)

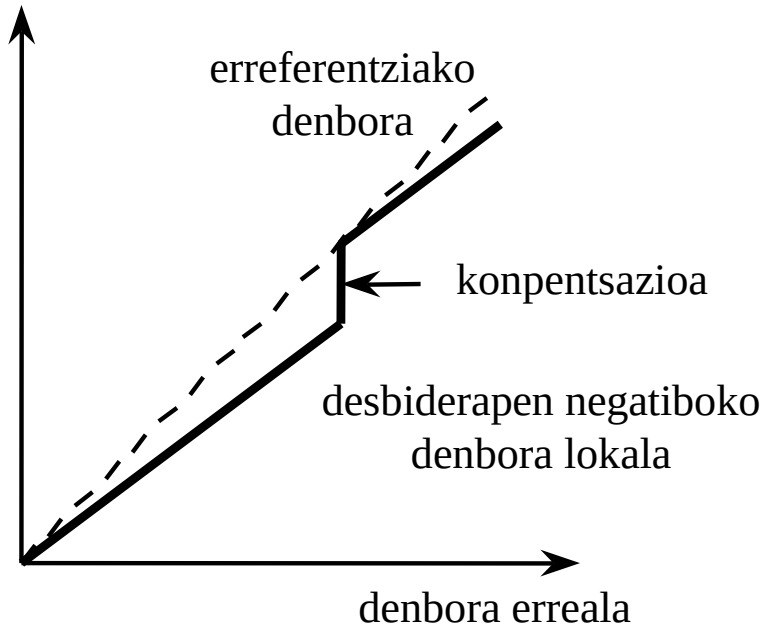
    D/2

2.2.2 Algoritmo banatuak



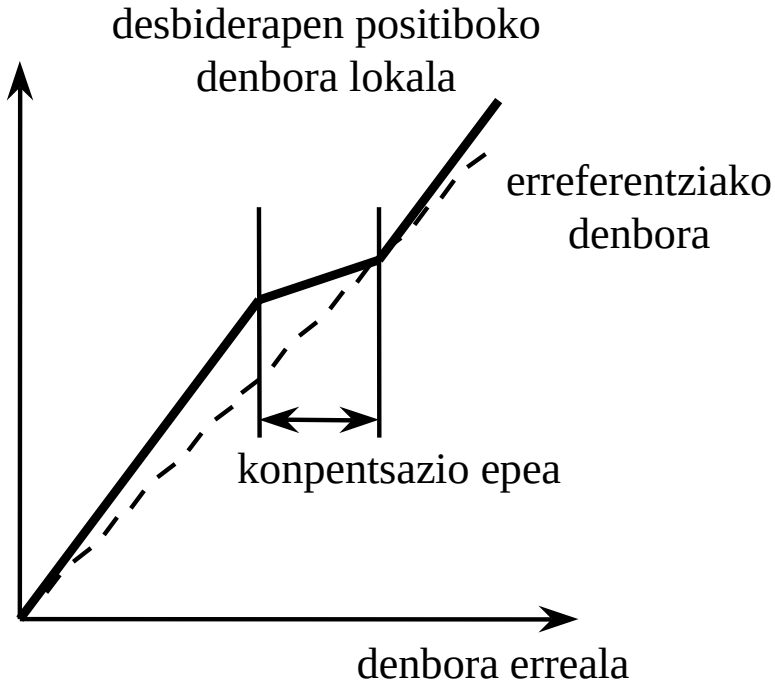
2.3 Desbideratzeen konpentsazioa

- *Denbora monotonoki handituz joan behar da!*



(a)

Desbideratze negatiboa

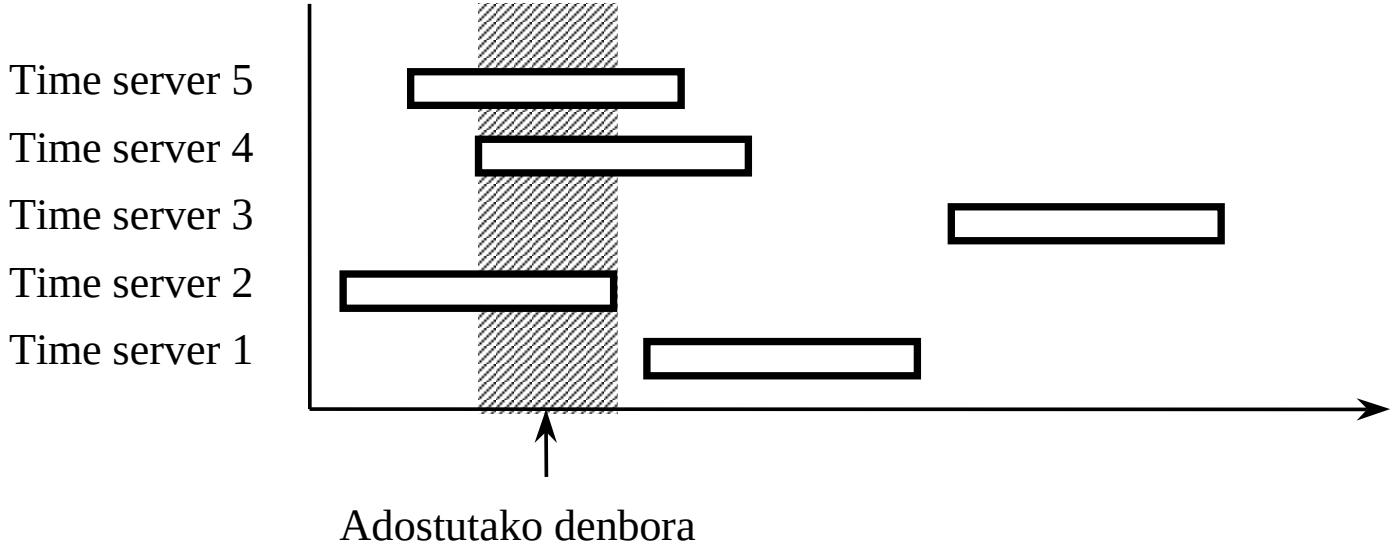


(b)

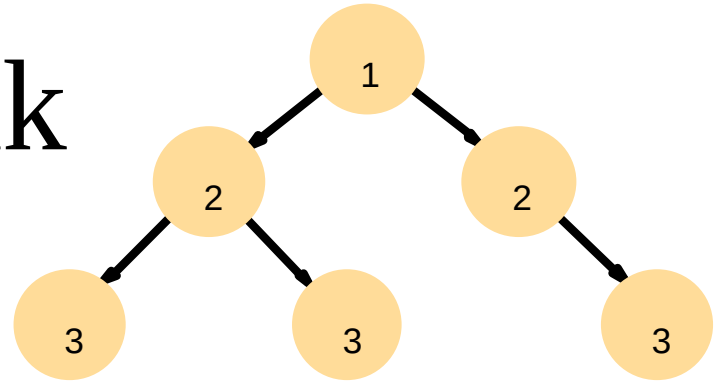
Desbideratze positiboa

2.4 Adibideak

- *Distributed Time Server (DTS)*
  - algoritmo banatua
  - denbora zerbitzariak (*time servers*): *UTC*
  - bezeroak (*time clerks*)



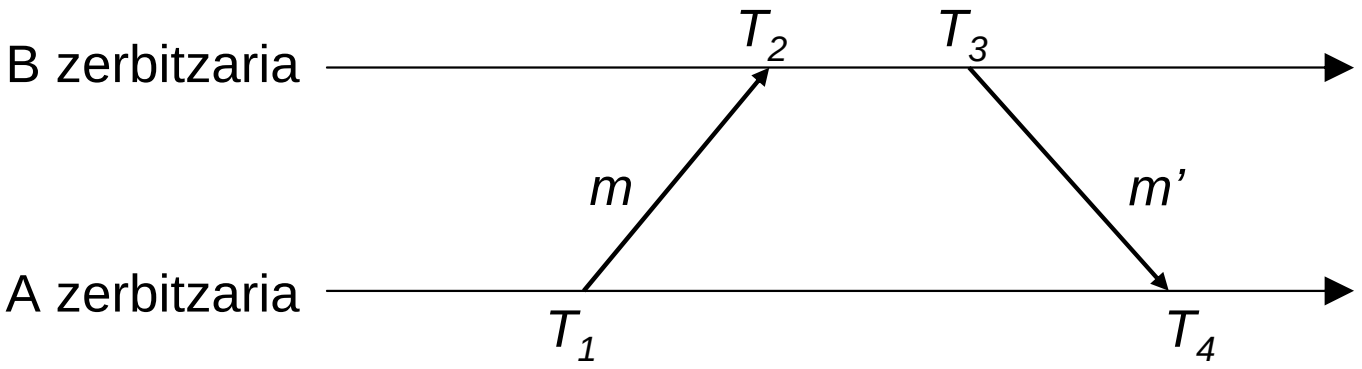
2.4 Adibideak



- *Network Time Protocol (NTP)*
  - Interneten estandarra bihurtu da
  - *UTC*rekiko sinkronizazio erredundantea eskaintzen du
  - denbora zerbitzariak mailatan banaturik daude (*strata*)
    - zerbitzari nagusiak *UTC* erreferentzia dute (strata 1)
    - 2. mailako zerbitzariak (strata 2): nagusiekiko sinkronizatuak
    - ...
  - Operazio moduak:
    - sare lokaletan, euskarri egokia dagoenean: *multicast* modua
    - prezisio handiagoa behar denean: prozedure deia modua
    - barne sinkronizazio hobeagoa behar denean: modu simetrikoa
  - Zerbitzarien sarea birkonfiguratu egin daiteke

2.4 Adibideak

- *Network Time Protocol (NTP):* modu simetrikoa



$T_2 = T_1 + t + \theta$
$T_4 = T_3 + t' - \theta$

Hipotesia: A atzeraturik B-rekiko (θ)

t, t' ≥ 0 m eta m' mezuen atzerapenak dira

$d_i = t + t' = T_2 - T_1 + T_4 - T_3 = (T_4 - T_1) - (T_3 - T_2)$

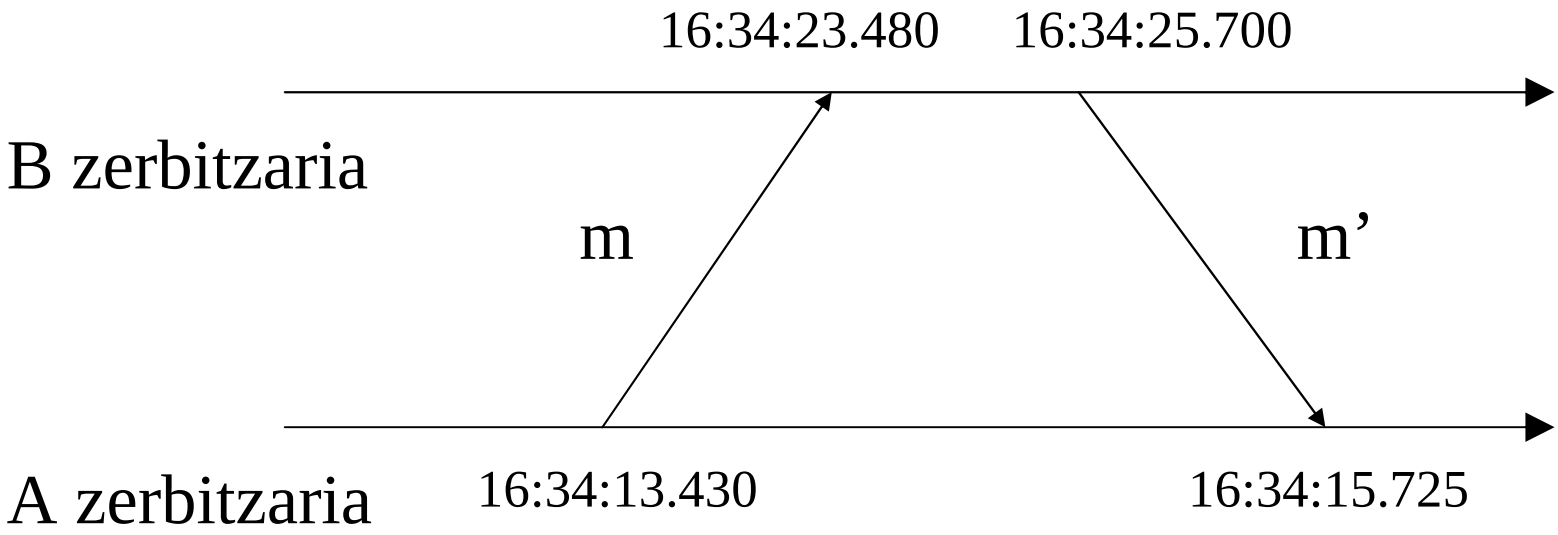
$\theta = \theta_i + (t' - t) / 2$, non $\theta_i = (T_2 - T_1 + T_3 - T_4) / 2$

$\theta_i - (d_i / 2) \leq \theta \leq \theta_i + (d_i / 2)$

θ_i *desbideratzearen estimazioa;* $(d_i / 2)$ *prezisioa*

2.4 Adibideak

- *Network Time Protocol (NTP):* ariketa



  - Desbideratzearen estimazioa?
  - Prezisioa? **$t + t' = 0.075$, $\theta_i = 10.013$**

$\theta = 10.013 \pm 0.038$ (9.975 .. 10.051)

3 Denbora logikoa eta gertaeren ordena

- Erlojuak sinkronizatzerakoan lor dezakegun prezisioa mugatua denez, zenbait kasutan denbora fisikoak ez du balio sistema banatuko nodo desberdineko gertaerak ordenatzeko
  - nodo berdineko gertaerak ordenatzea posible da (baldin eta monotonizitatea errespetatzen bada)
  - aldiz, nodo desberdineko gertaerekin, kausaltasun erlazioak distortsionatu daitezke
- Bestalde, aplikazio askorentzat gertaeren arteko ordena du garrantzia (eta ez bere denbora fisikoa)

3.1 Gertaeren eredua

- Sistema eredua: mezu-trukearen bidez soilik komunikatzen diren prozesu multzoa

  bidali(p_i, mezua)

  jaso(p_j, mezua)

  p_i eta p_j mezuaren igorlea eta hartzailea direlarik

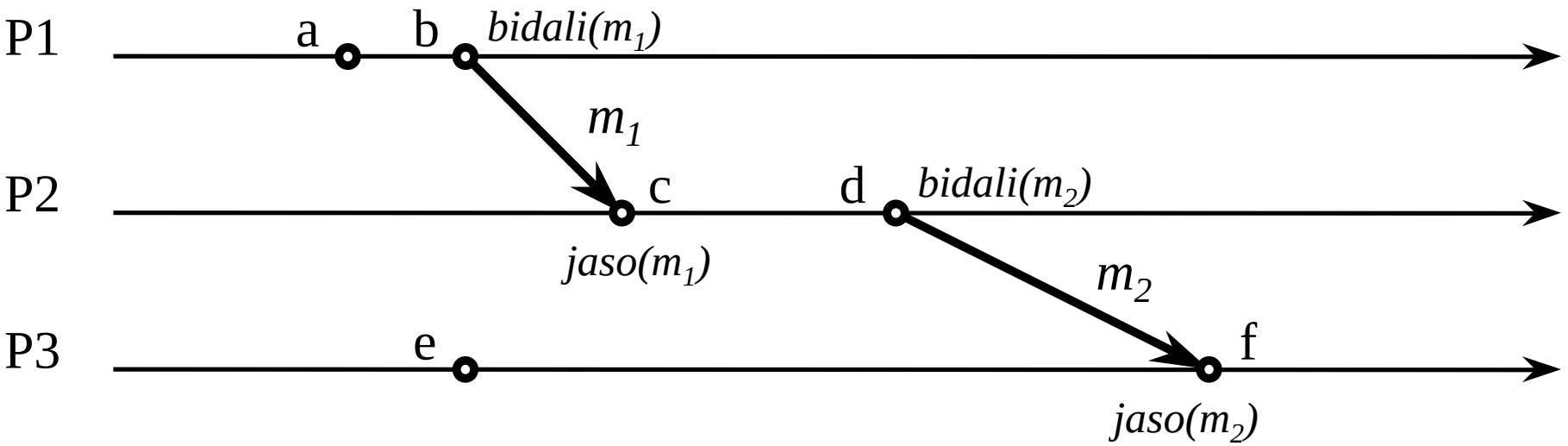
- Sinplifikazioa: nodoko prozesu bat suposatzen da
- Prozesu bakoitzak gertaera sekuentzia bat sortzen du. Hiru motako gertaerak daude:
  - Mezu bat bidaltzea (*bidali* exekutatzerakoan)
  - Mezu bat jasotzea (*jaso* eta *entregatu* bereiztuko ditugu)
  - Barneko gertaerak (beste guztiak, komunikaziorik gabekoak)

3.1 Gertaeren eredua

- Prozesu berdineko gertaerak ordenatzeko nahikoa da *x* gertaera bakoitzari dagokion denbora lokala ezartzea (bereizmena nahikoa bada): *T(x)*
- Sistemako bi gertaeren arteko kausaltasun erlazioa dagoela esaten da ($x \rightarrow y$, “*x y baino lehen gertatu da*”, “*happened before*”) baldin eta:
  1. *x* eta *y* prozesu berdineko gertaerak dira, eta $T(x)<T(y)$
  2. *x* eta *y m* mezuaren *bidali(m)* eta *jaso(m)* gertaerak direnean (prozesu desberdinetan edota prozesu berean)
  3. Beste gertaera *z* existitzen denean, non $x \rightarrow z$ eta $z \rightarrow y$ (itxiera trantsitiboa)

3.1 Gertaeren eredua

- Bi gertaeren artean ez bada kausaltasun erlaziorik konkurrenteak edo aldiberekoak dira: $x \,||\, y$



$$a \rightarrow b \quad b \rightarrow c \quad a \rightarrow c \quad a \rightarrow f \quad a \,||\, e \quad e \,||\, d$$

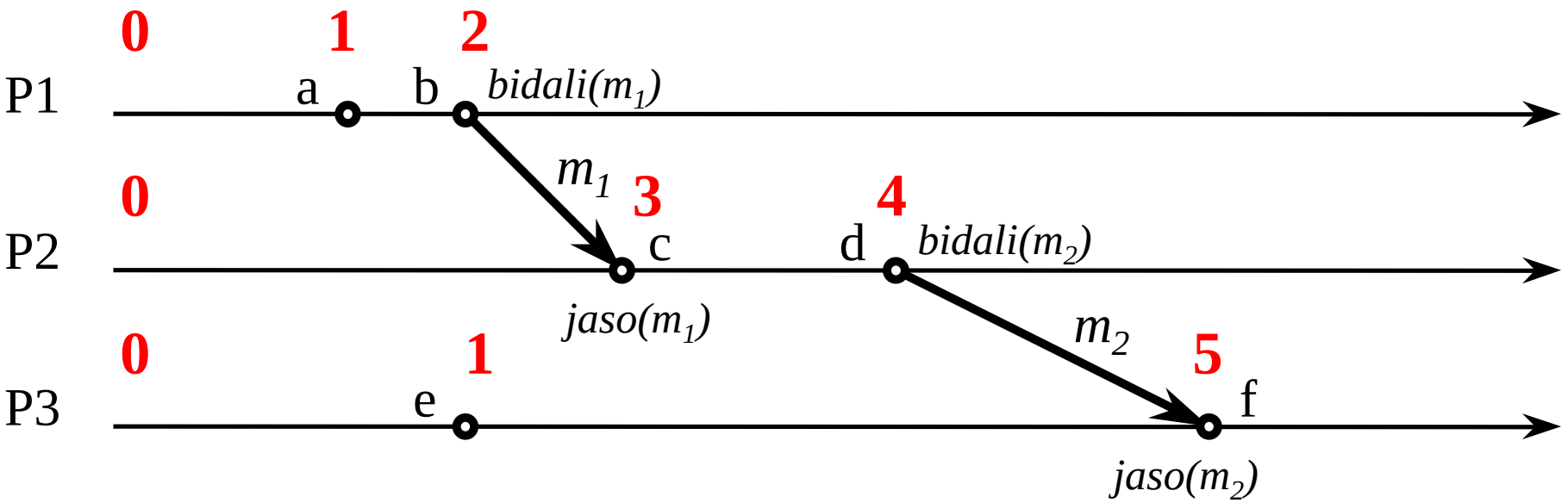
3.2 Lamporten erloju logikoak

- 1978. proposatua, kausaltasun erlazioak adierazteko. Prozesu bakoitzak P_i bere erloju logiko lokala du C_i, gertaerei denbora markak ezartzeko (kontadore asinkrono bat nahikoa da)
- Algoritmoa:
  - hasieran, $C_i = 0, \ \forall i$
  - p_i prozesuan gertaera bat gertatu baino lehen: $C_i = C_i + 1$
  - p_j-k p_i-ri mezu bat bidaltzerakoan, p_j-k bere erloju logikoaren balioa mezuarekin batera bidaltzen du, C_m. p_i-k mezua jasotzerakoan, bere erloju logikoa eguneratzen du:

$$(1)\ C_i = max(C_i, C_m), \quad (2)\ C_i = C_i + 1$$

  - Arazoa: $C_i(x) < C_j(y)$ ez du $x \rightarrow y$ inplikatzen

3.2 Lamporten erloju logikoak



$$a \rightarrow b \quad b \rightarrow c \quad a \rightarrow c \quad a \rightarrow f \quad a \,||\, e \quad e \,||\, d$$

3.3 Denborazko bektoreak

$$V_i(x) < V_j(y) \Leftrightarrow x \rightarrow y \text{ } \underline{\textit{bermatzen dute}}$$

- V: N sarrerako bektorea, N prozesu kopurua delarik
- $V_i[i]$: P_i prozesuaren erloju logikoa (lokala)
- $V_i[j]$: P_i prozesuak P_j-ren erlojuari buruz ezagutzen duen azken balioa

- Algoritmoa:
  - Hasieran, $V_i[j] = 0, \ \forall i, j$
  - p_i prozesuan gertaera bat gertatu baino lehen: $V_i[i] = V_i[i] + 1$
  - p_j-k p_i-ri mezu bat bidaltzerakoan, p_j-k bere denborazko bektorea mezuarekin batera bidaltzen du, V_m. p_i-k mezua jasotzerakoan, bere denborazko bektorea eguneratzen du:

    (1) $\forall k: V_i[k] = max(V_i[k], V_m[k])$, (2) $V_i[i] = V_i[i] + 1$

3.3 Denborazko bektoreak

- Definizioa:

$$V1 < V2 \Leftrightarrow \forall i\ V1[i] \leq V2[i] \text{ and } \exists j\ V1[j] < V2[j]$$

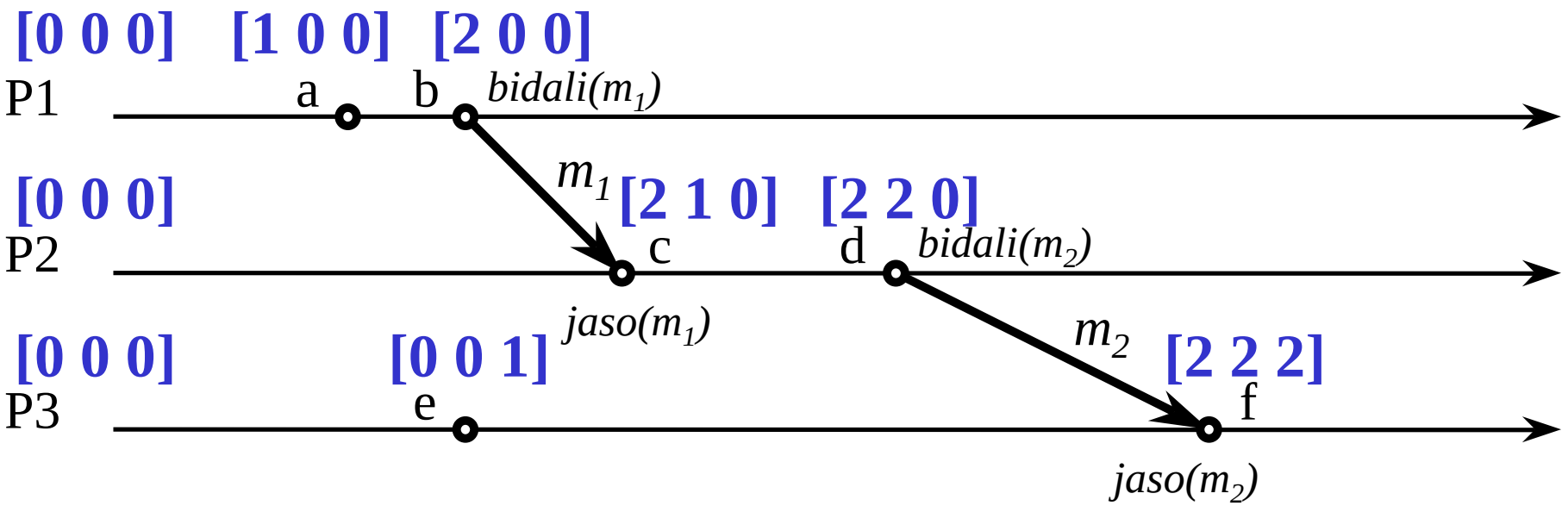
- Sistemako bi gertaeren *(x, y)* arteko kausaltasun erlazioa egongo da baldin eta

$$V(x) < V(y) \text{ edota } V(y) < V(x)$$

- Bestela, bi gertaerak konkurrenteak dira
- Denborazko bektoreak kausaltasuna prezisio guztiz adierazten dute:

$$V_i(x) < V_j(y) \Leftrightarrow x \rightarrow y$$

3.3 Denborazko bektoreak



$$a \rightarrow b \quad b \rightarrow c \quad a \rightarrow c \quad a \rightarrow f \quad a \,||\, e \quad e \,||\, d$$

4 Egoera globala eta sendotasuna

- Sistema eredua:
  - N prozesu P_i (i = 1, 2, ..., N)
  - historia(P_i) = h_i = < e_i^0, e_i^1, e_i^2, ... >
  - historiaren aurrizkia h_i^k = < e_i^0, e_i^1, ..., e_i^k >
  - s_i^k: P_i prozesuaren egoera, e_i^k gertaeraren ondoren
  - s_i^0: P_i prozesuaren hasierako egoera
- Sistemaren egoera globala: S = (s_1, s_2, ..., s_N)
  - Egoera batzuk sendotasunik gabekoak izan daitezke
- Ebaketa: prozesuen historia-aurrizkien batura
  - C = $h_1^{c1} \cup h_2^{c2} \cup ... \cup h_N^{cN}$ = (c_1, c_2, ..., c_N)

4 Egoera globala eta sendotasuna

- Ebaketaren muga: azken gertaeren batura

$$(c_1, c_2, ..., c_N) \Rightarrow (e_1^{c1}, e_2^{c2}, ..., e_N^{cN})$$

- Ebaketa bakoitzari egoera global bat dagokio

$$(c_1, c_2, ..., c_N) \Rightarrow (s_1^{c1}, s_2^{c2}, ..., s_N^{cN})$$

- C ebaketa sendoa da baldin eta edozein (e, e') gertaerentzat:

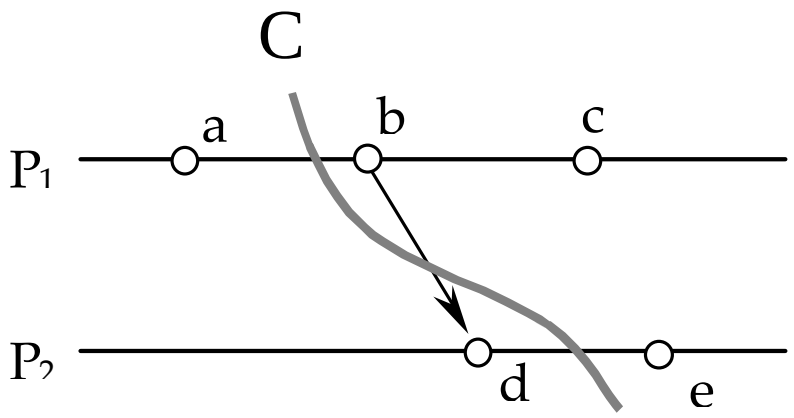
$$(e \in C) \wedge (e' \rightarrow e) \Rightarrow e' \in C$$

  - Bestela, C sendotasunik gabeko ebaketa da

- Ebaketa sendo bakoitzari egoera global sendo bat dagokio

4 Egoera globala eta sendotasuna

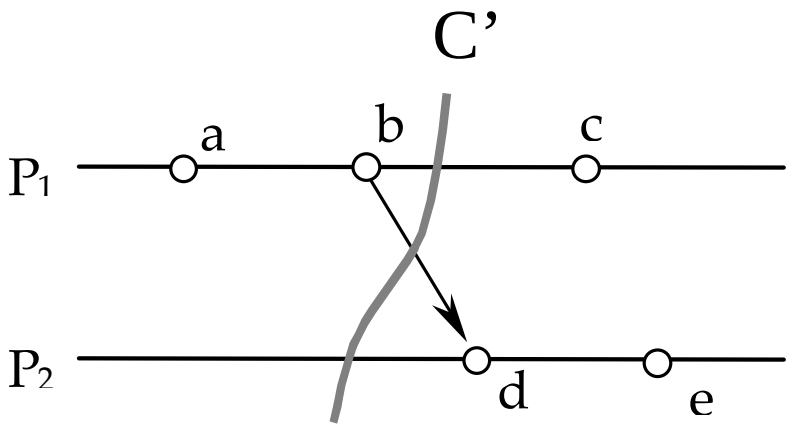
- Ebaketa sendoa: mezu baten jasotze gertaera ebaketa barruan badago, orduan mezuaren bidaltze gertaera ere ebaketaren barruan egon behar da



(a)

sendotasunik gabekoa

$d \in C, b \rightarrow d, b \notin C$



(b)

sendoa

4 Egoera globala eta sendotasuna

- Ebaketa bat sendoa den jakiteko, denborazko bektoreetaz balia gaitezke:

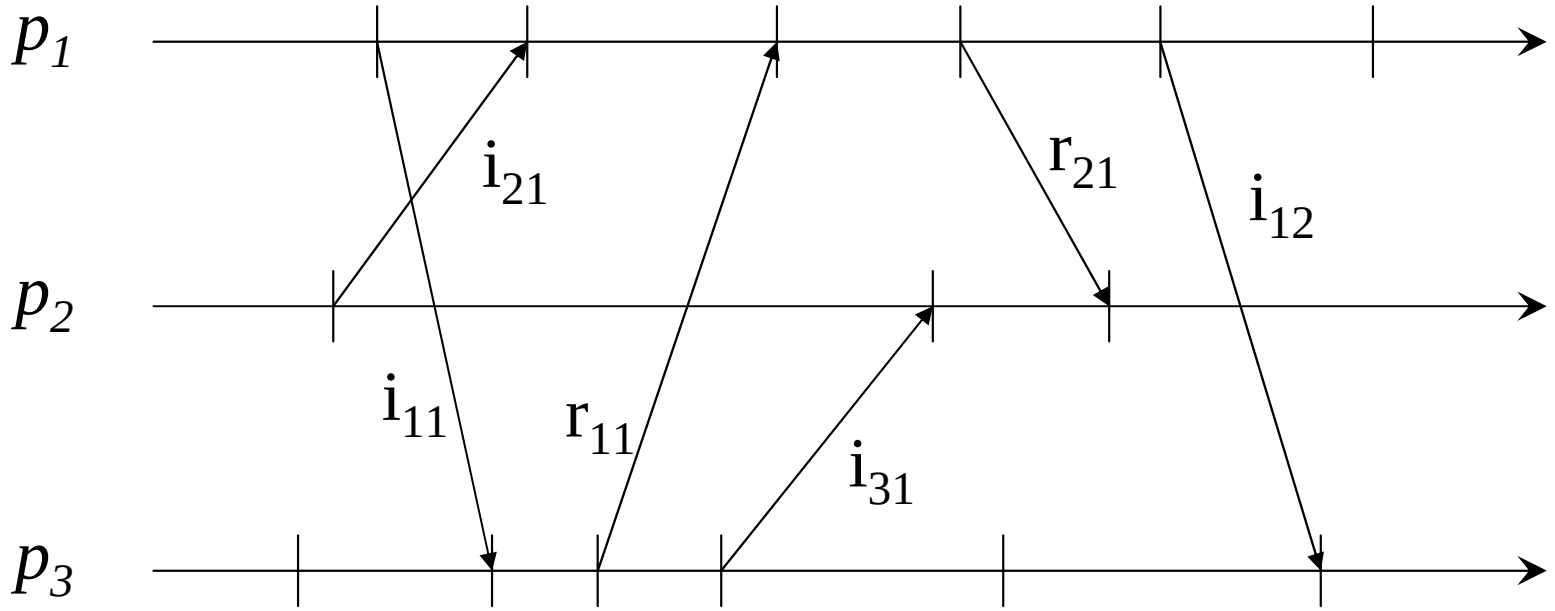
$$\forall i, j: V_i[i](e_i^{ci}) \geq V_j[i](e_j^{cj})$$

- Prozesuek informazio guztiaren zati bat bakarrik dutenez, ebaketa sendo bat eraikitzeko (eta bide batez, egoera global sendo bat kalkulatzeko) algoritmo banatu bat exekutatuko dute
  - Adibidea: Chandy eta Lanport-en algoritmoa (1985)
- Erabilpenak: elkar-blokeaketak, berreskuratze puntuak, prozesaketaren amaiera

4 Egoera globala eta sendotasuna

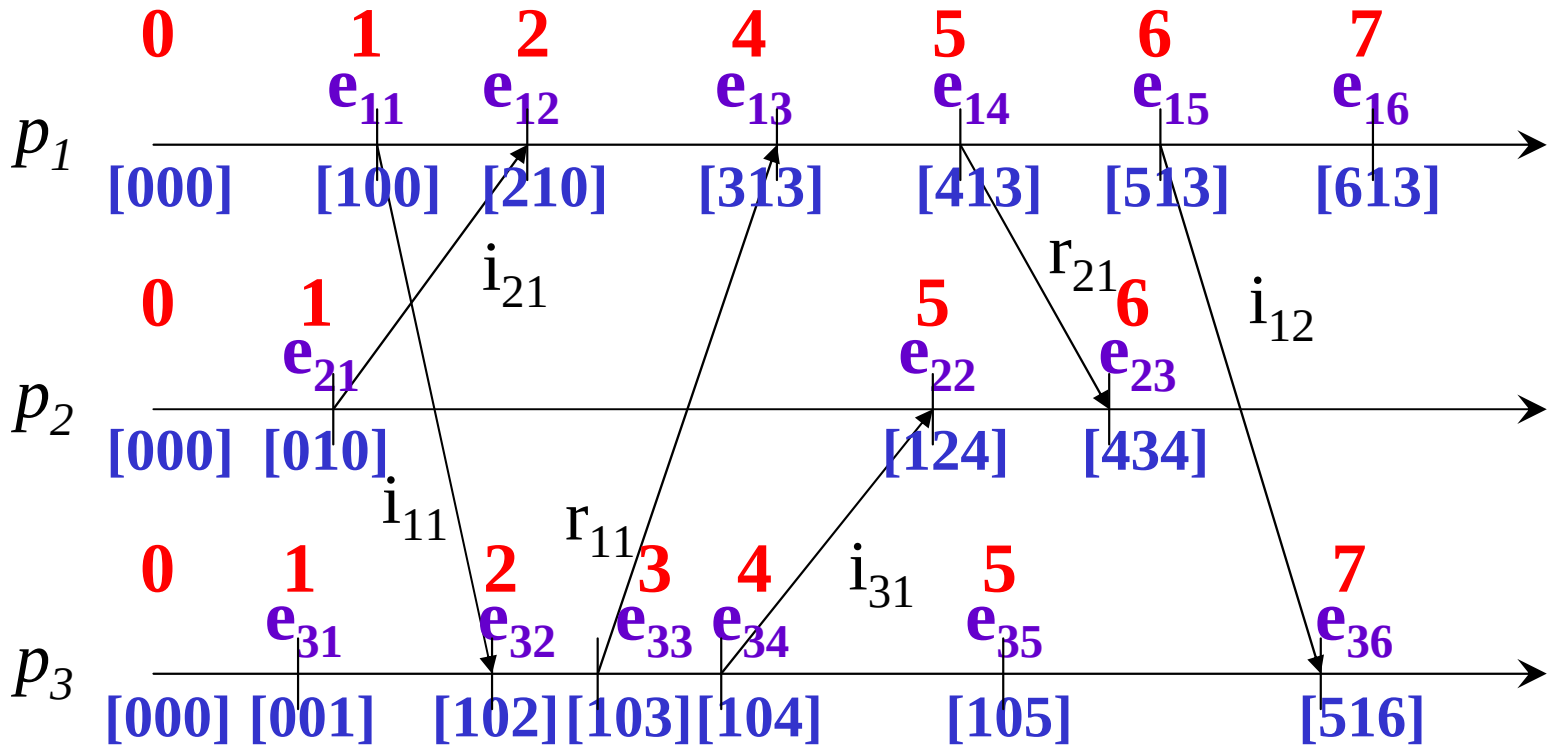
- Chandy eta Lanport-en *snapshot* algoritmoa:
  1. process p_1 (the initiator of the snapshot) saves its state s_1 and broadcasts the message *SNAPSHOT* to P (the set of processes).
  2. Let process p_i receive the *SNAPSHOT* message the first time from some process p_j (p_j can be different from p_1). At that time, p_i saves its state s_i and forwards the *SNAPSHOT* message to P. The state of the channel c_{ji} is set to empty, and p_i starts logging the messages received on the channels c_{ki} (for all $k \neq j$).
  3. When p_i receives *SNAPSHOT* from p_k, then the computation of the state of the channel c_{ki} is complete. As soon as p_i has received *SNAPSHOT* from all the processes in P, the computation of the snapshot is terminated.

4 Egoera globala eta sendotasuna



i: eskaria (*invocation*)
r: erantzuna (*reply*)

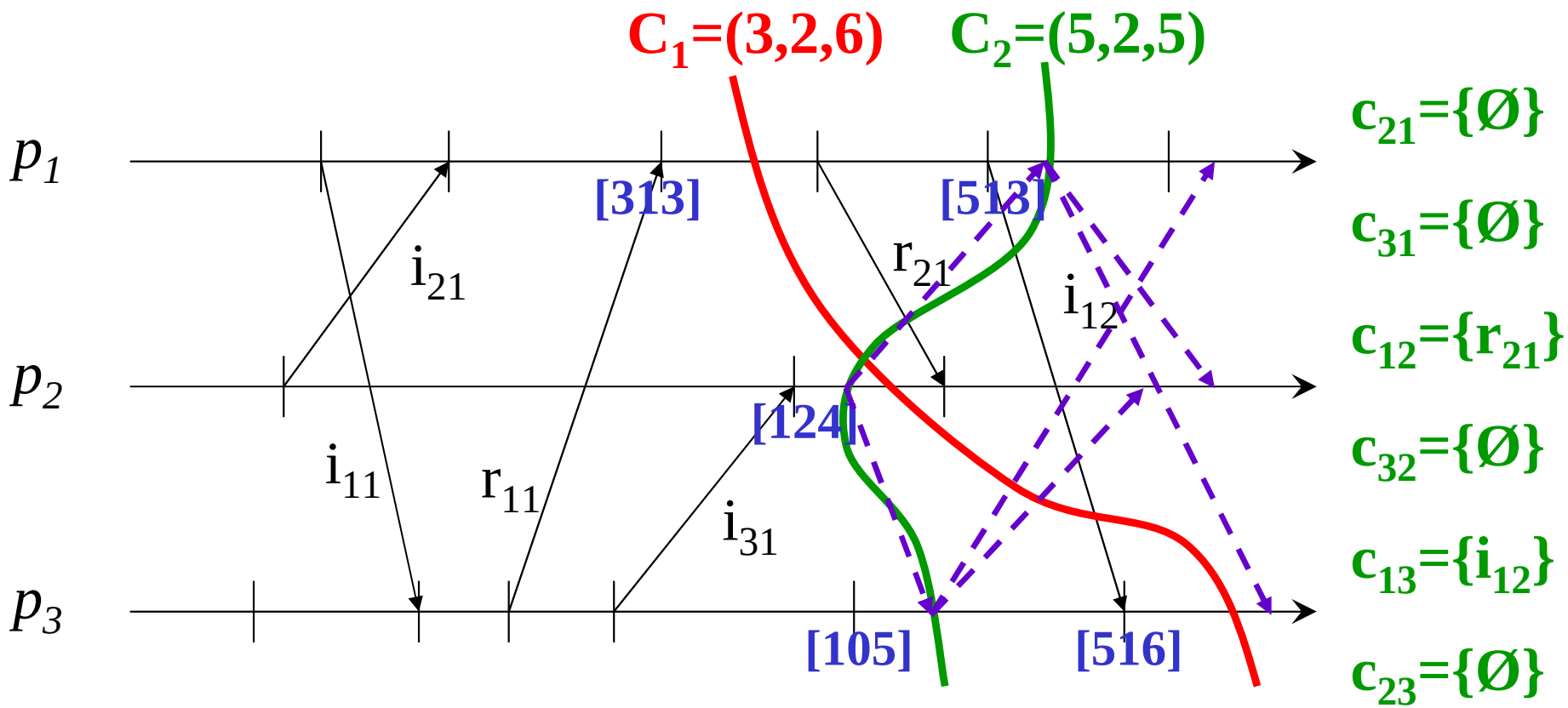
4 Egoera globala eta sendotasuna



i: eskaria (*invocation*)

r: erantzuna (*reply*)

4 Egoera globala eta sendotasuna



i: eskaria (*invocation*)

r: erantzuna (*reply*)