Systèmes et algorithmes répartis

Causalité et datation

Philippe Quéinnec, Gérard Padiou

ENSEEIHT
Département Sciences du Numérique

4 octobre 2021

plan

Plan

Datation des événements

Objectif

Associer une date à chaque événement pour :

- ordonner les événements → compatible avec la causalité
- identifier les événements → dates uniques

Moyens

- Horloges matérielles
- Horloges logiques

Difficultés

- Pas d'horloge globale
- Tous les événements ne sont pas causalement liés
- Datation cohérente avec la relation causale :

$$\forall e, e' : e \prec e' \Rightarrow d_e < d_{e'}$$

Horloges matérielles

Idée

- Utiliser les horloges matérielles de chaque site
- Risque : Datation incohérente de l'événement de réception d'un message : la date de réception précède la date d'émission
- Possible si l'horloge du site de réception est en retard (suffisamment) sur celle du site de l'émetteur

Difficultés

- Cohérence avec la causalité
- Datation définissant un ordre total
- Pas d'unicité des dates

[Précis 3.3.1 pp.43–44]

Horloges matérielles

Solutions

- Synchronisation des horloges locales :
  $$\text{invariant} \quad \max_{i=1..N}(h_i) - \min_{i=1..N}(h_i) < \epsilon$$
- Causalité respectée si ϵ est inférieur au temps de transmission d'un message
- Unicité en utilisant couple $(date, id\ du\ site)$

Protocole de synchronisation d'horloges possible mais dans des contextes réseaux assurant une certaine qualité de service ou par l'usage d'un signal externe (horloge atomique, GPS)

Temps logique

Principes de base

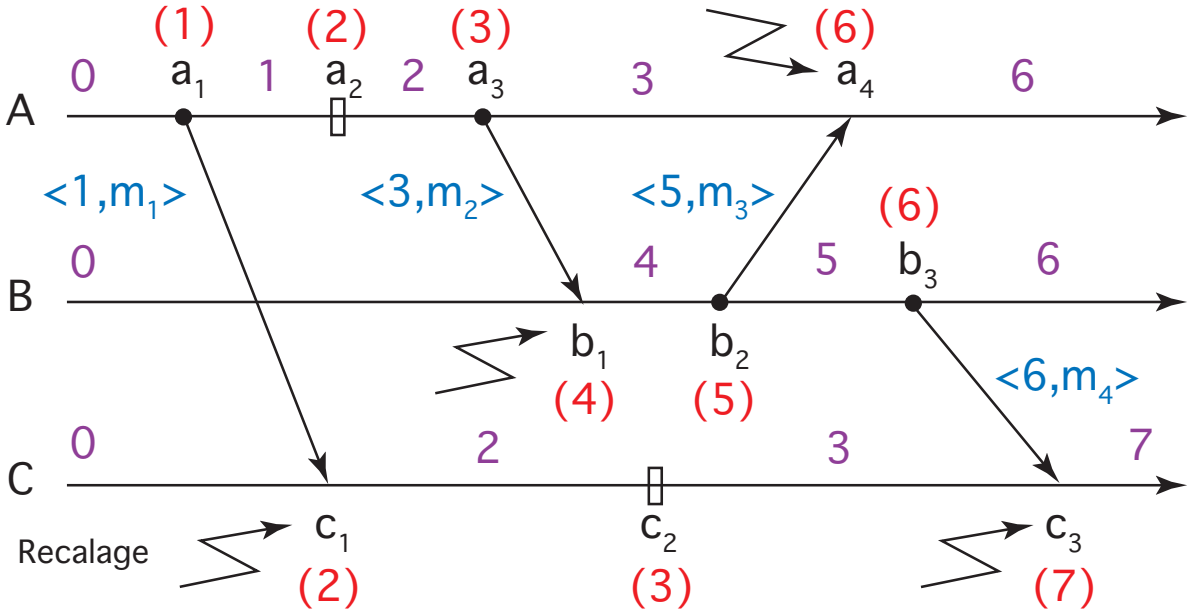
- Associer à chaque site une horloge logique locale
- L'horloge compte les événements au lieu du temps réel
- Surcharger les messages avec leur date d'émission
- Recaler si nécessaire l'horloge locale d'un site lors de chaque réception de message
- Avantage : vision plus abstraite d'un calcul réparti

[Précis 3.3.2 pp.44–47]

Une tentative...

♪♪♪

- Un compteur « horloge » sur chaque site
- Surcharge des messages et recalage de compteur



☺ $e \prec e' \Rightarrow d(e) < d(e')$

☺☹ Mêmes dates $\Rightarrow$ pas causalement liés

☹ $d(e) < d(e') \not\Rightarrow e \prec e'$ (ex : $d(c_2) < d(b_3)$ mais $c_2 \not\prec b_3$)

Horloge de Lamport

Propriétés

- Introduit un ordre total entre événements
  $\rightarrow$ Dates distinctes pour tout couple d'événements
- Date = (compteur local, numéro de site)
  ordre total sur les sites $\Rightarrow$ ordre lexicographique total

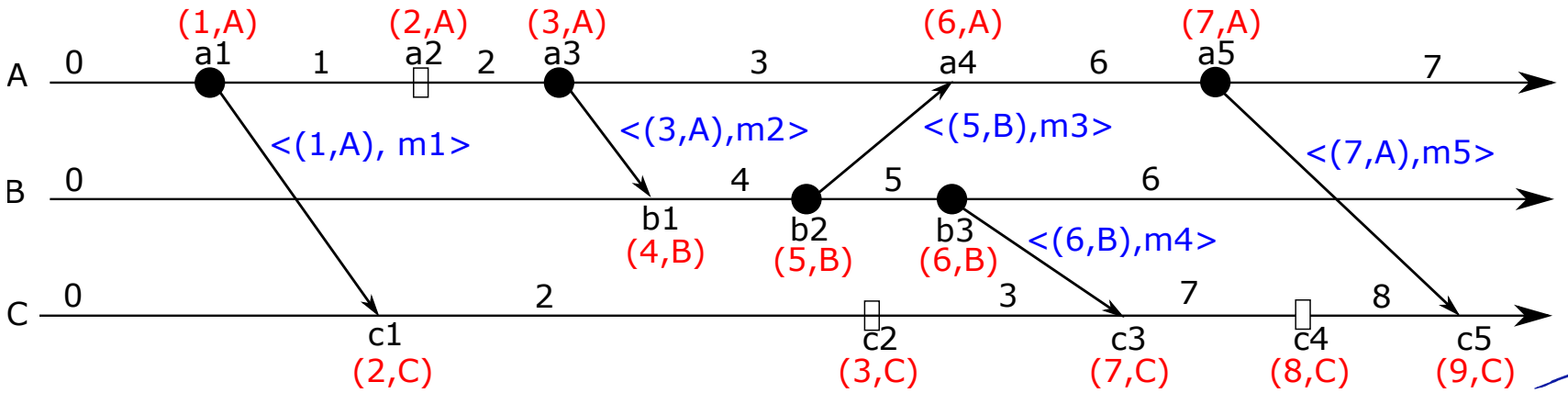
```
struct Date { int cpt; // compteur d'événements
              int s;   // numéro de site
}
bool prec (Date d1, Date d2) {
       return (d1.cpt < d2.cpt)
              || ((d1.cpt == d2.cpt) && (d1.s < d2.s));
}
```

1. *Time, Clocks and the Ordering of Events in a Distributed System*, Leslie Lamport. Communications of the ACM, July 1978.

Actions associées aux événements

Chaque site s possède une horloge entière H_s. Chaque événement est daté avec le couple (H_s après l'action, id du site).

Type d'événement sur un site s	Action sur le site s
Événement interne sur s	$H_s \leftarrow H_s + 1$;
Émission sur s de m	$H_s \leftarrow H_s + 1$; envoi de $\langle\langle H_s, s\rangle, m\rangle$;
Réception sur s de $\langle\langle dm, s'\rangle, m\rangle$	$H_s \leftarrow \max(H_s, dm) + 1$;



Horloge vectorielle de Fidge-Mattern (1988) ♪♪♪

Objectif

Représenter exactement la relation de causalité

Propriétés

- Utilisation de vecteurs de dimension égale au nombre de sites
- Pour un événement e, $HV(e)[i]$ = nombre d'événements du passé de e sur p_i (y compris e)
- Coût plus élevé : surcharge des messages par un vecteur

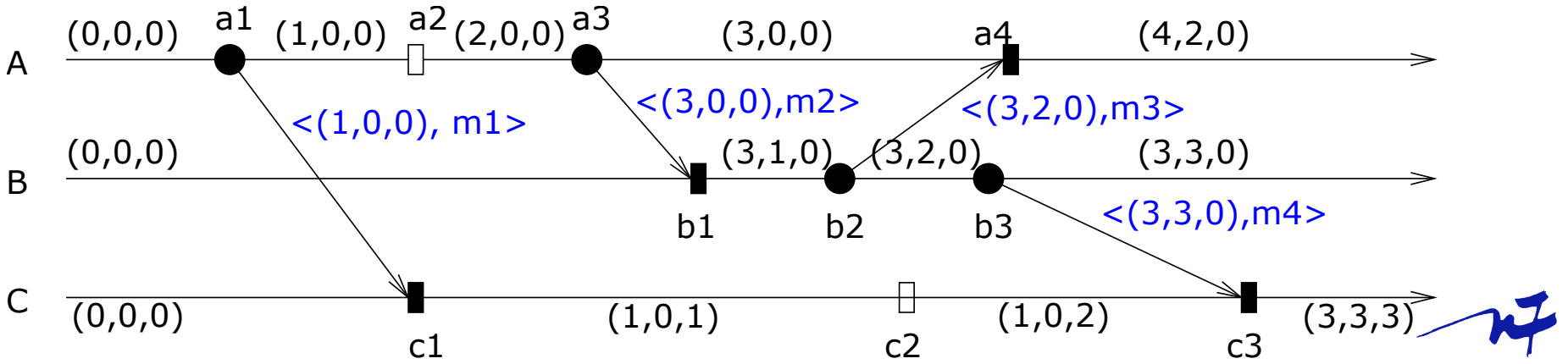
---

1. *Timestamps in Message-Passing Systems That Preserve the Partial Ordering*, Colin J. Fidge. 11th Australian Computer Science Conference, 1988.
2. *Virtual Time and Global State in Distributed Systems*, Friedemann Mattern. Int'l Workshop on Parallel and Distributed Algorithms, 1989.

Actions associées aux événements

Chaque site s possède une horloge vectorielle H_s.
Chaque événement est daté avec le résultat de l'action.

Type d'événement sur un site s	Action sur le site s
Événement interne sur s	$H_s[s] \leftarrow H_s[s] + 1$
Émission sur s de m	$H_s[s] \leftarrow H_s[s] + 1$ envoi de $\langle H_S, m \rangle$
Réception sur s de $\langle dm, m \rangle$	$H_s[s] \leftarrow H_s[s] + 1$ $H_s[s'] \leftarrow \max(H_s[s'], dm[s']), \forall s' \neq s$



Horloge vectorielle de Fidge-Mattern

Expression des relations entre dates

$$
\begin{aligned}
D \leq D' &\triangleq \forall i : D[i] \leq D'[i] \\
D < D' &\triangleq D \leq D' \wedge \exists k : D[k] < D'[k] \\
D \parallel D' &\triangleq \neg(D < D') \wedge \neg(D' < D)
\end{aligned}
$$

Datation isomorphe à l'ordre causal

$$
\begin{aligned}
e \prec e' &\Leftrightarrow D(e) < D(e') \\
e \parallel e' &\Leftrightarrow D(e) \parallel D(e')
\end{aligned}
$$

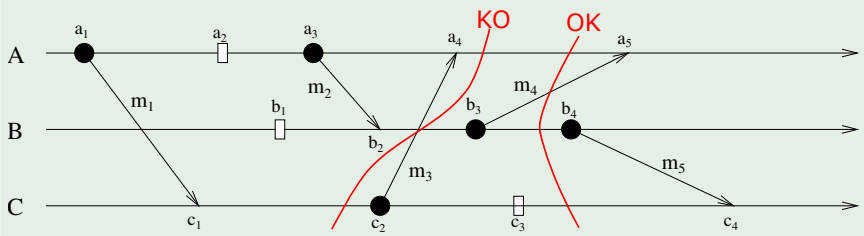
Datation des coupures

On identifie une coupure C par $(c_1, \ldots, c_n)$, ses événements maximaux sur chaque site (les événements internes à la coupure sont implicites)

Date d'une coupure

- Date d'une coupure $C = (c_1, \ldots, c_n)$:
  $HV(C) \triangleq sup(HV(c_1), \ldots, HV(c_n))$
- $C_2 \subsetneq C_1 \Leftrightarrow HV(C_2) < HV(C_1)$

Exemple (CH2, modèle standard)



$KO = (a_4, b_2, c_1)$, $HV(KO) = (4, 2, 2)$
$OK = (a_4, b_3, c_3)$, $HV(OK) = (4, 3, 3)$

$d(a_4) = (4, 0, 2)$,
$d(b_2) = (3, 2, 0)$, $d(b_3) = (3, 3, 0)$,
$d(c_1) = (1, 0, 1)$, $d(c_3) = (1, 0, 3)$

Datation et coupure cohérente

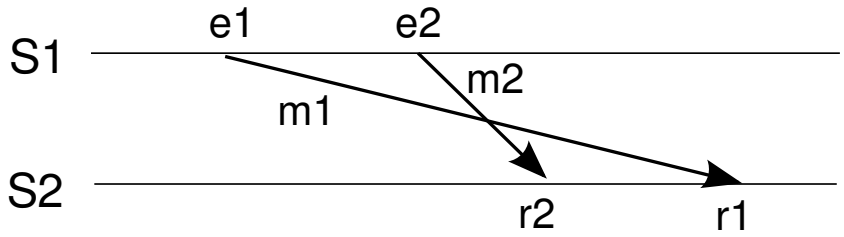
Coupure cohérente

C cohérente $\Leftrightarrow HV(C) = \langle HV(c_1)[1], \ldots, HV(c_n)[n] \rangle$

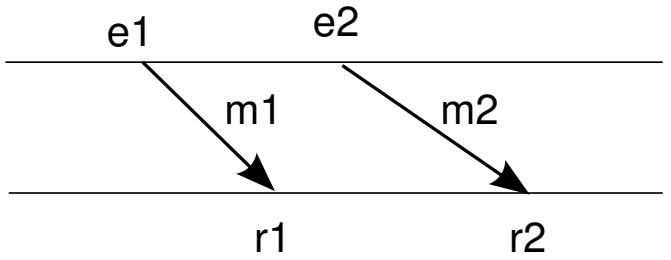
- Soit C cohérente. Alors $\forall i, j : HV(c_i)[i] \geq HV(c_j)[i]$.
  En effet, l'incrémentation de $HV(c_i)[i]$ ne peut venir que d'un événement local ou résulter d'un message provenant du passé.
- Soit C non cohérente.
  $\Rightarrow$ il existe un événement e_i hors coupe qui est dans le passé causal de C
  $\Rightarrow$ Sur le site i : $HV(c_i)[i] < HV(e_i)[i]$ et $HV(e_i)[i] \leq HV(C)[i]$
  $\Rightarrow HV(C)[i] > HV(c_i)[i]$, donc $HV(C) \neq (HV(c_1)[1], \ldots, HV(c_n)[n])$

Plan

Protocole FIFO

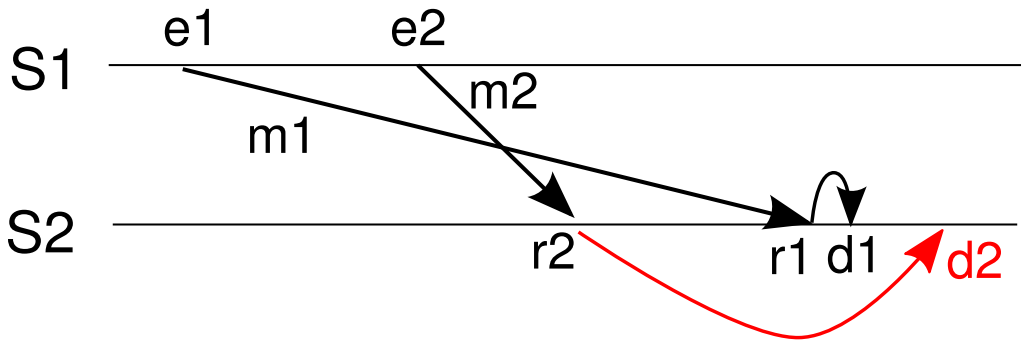


Non FIFO



FIFO

$$\forall m, m' : s_1 \xrightarrow{m} s_2 \wedge s_1 \xrightarrow{m'} s_2 \wedge e(m) \prec e(m') \Rightarrow r(m) \prec r(m')$$

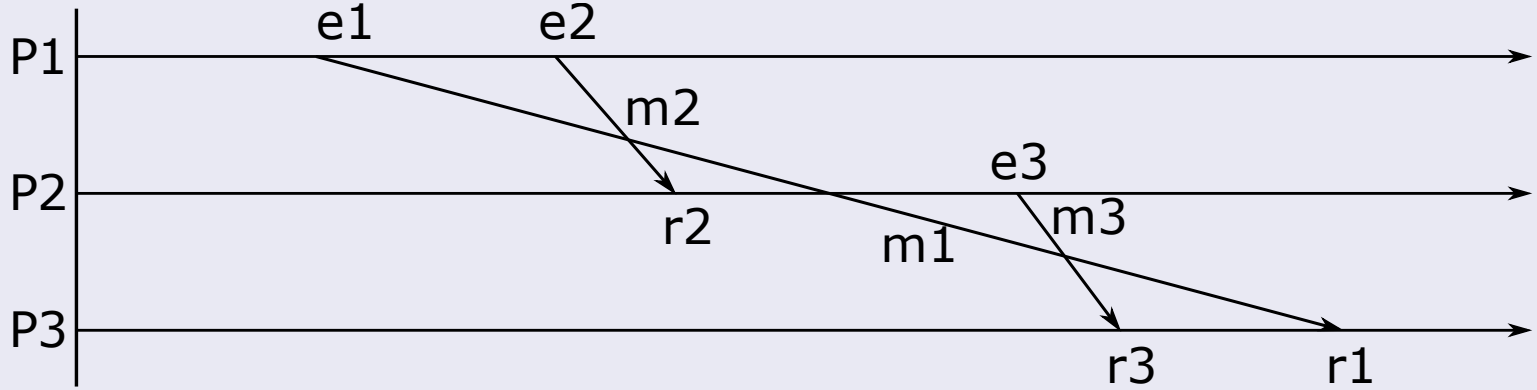


délivrance $\neq$ réception

Protocole causalement ordonné

Objectif : Mettre de l'ordre

Réceptions incohérentes par rapport aux émissions



$r_3 \prec r_1$ alors que $e_1 \prec e_3$

$\Rightarrow$ délivrance $\neq$ réception

1. *Reliable communication in the presence of failures*, Kenneth P. Birman and Thomas A. Joseph. ACM Transactions on Computer Systems, January 1987.

Protocole FIFO

♪♪♪

Objectif

Garantir la cohérence des réceptions sur un même site par rapport à leur éventuelle émission depuis un même site

$\Rightarrow$ réordonner les messages reçus sur un site

- Trois événements au lieu de deux par message :
  - l'émission e,
  - la réception r,
  - la délivrance d.
  - Causalité : $e \prec r \prec d$
- S'exprime par la propriété :

$$\forall s_1, s_2, m, m' \ : \ s_1 \xrightarrow{m} s_2 \ \wedge \ s_1 \xrightarrow{m'} s_2 \ \wedge \ e(m) \prec e(m') \Rightarrow d(m) \prec d(m')$$

Protocole FIFO

Réalisation : il suffit de numéroter les messages pour chaque canal (couple site d'émission, site de réception)

Récepteur pour un canal :

```
type Message = ⟨contenu, numéro⟩;
int prochain = 0;
SortedSet<Message> enAttente;  // trié par numéro
while (true) {
    recevoir m;
    enAttente.add(m);
    while (enAttente.first().numéro == prochain) {
        m ← enAttente.removeFirst();
        délivrer m.contenu
        prochain++;
    }
}
```

Protocole causalement ordonné

Objectif

Garantir la cohérence des réceptions sur un même site par rapport à leur causalité éventuelle en émission

$\Rightarrow$ réordonner les messages reçus sur un site

- Trois événements au lieu de deux par message :
  - l'émission e,
  - la réception r,
  - la délivrance d.
  - Causalité : $e \prec r \prec d$
- S'exprime par la propriété :

$$\forall s, m, m' \ : \ \_ \xrightarrow{m} s \ \wedge \ \_ \xrightarrow{m'} s \ \wedge \ e(m) \prec e(m') \Rightarrow d(m) \prec d(m')$$

[Précis 3.2 pp.47–50]

Histoire causale

Histoire causale d'un message $H_c(m)$

L'histoire causale $H_c(m)$ d'un message m est l'ensemble des messages qui ont leurs événements d'émission précédant causalement l'émission de m :

$$H_c(m) = \{m' : e(m') \prec e(m)\}$$

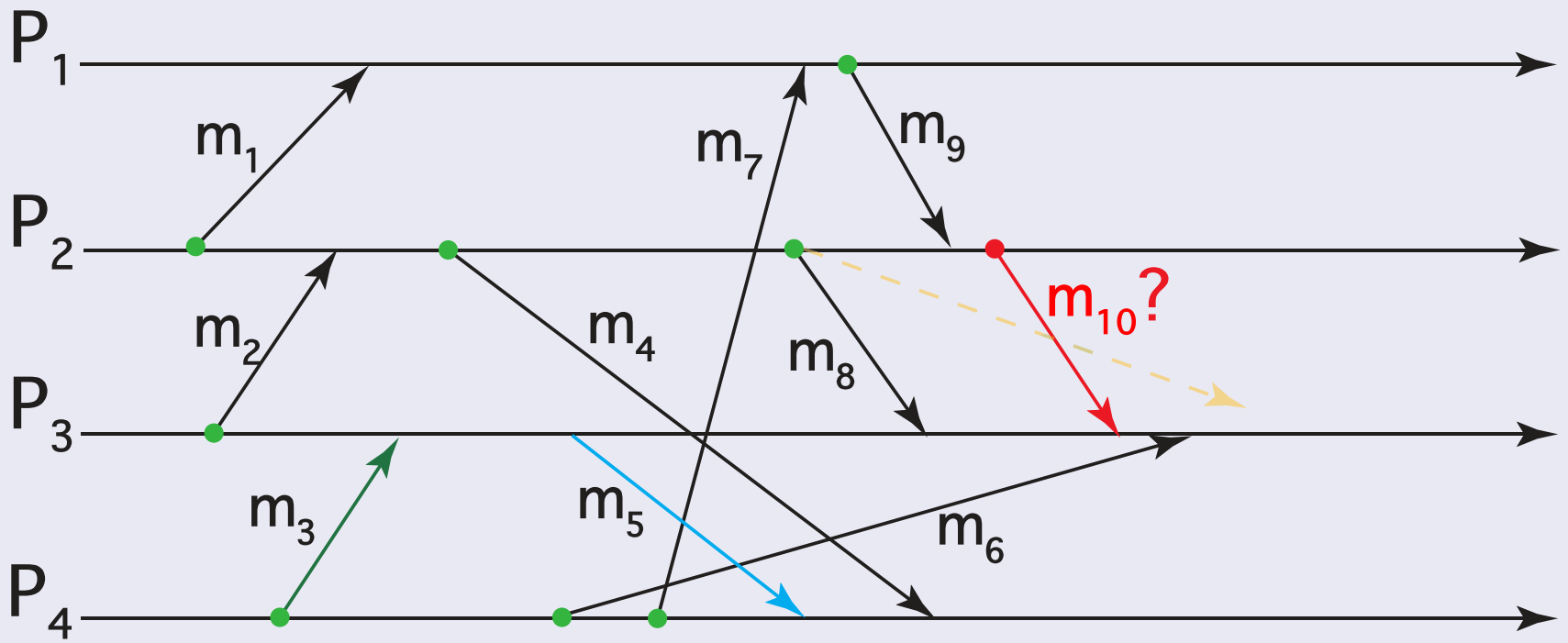
Critère de délivrance d'un message

Un message m est délivré sur un site s ssi tous les messages de son histoire causale ayant aussi s comme site de destination ont été déjà délivrés :

$$\forall s, m' \in H_c(m) : \_ \xrightarrow{m} s \ \wedge \ \_ \xrightarrow{m'} s \wedge \ \Rightarrow d(m') \prec d(m)$$

Histoire causale : exemple

Exemple



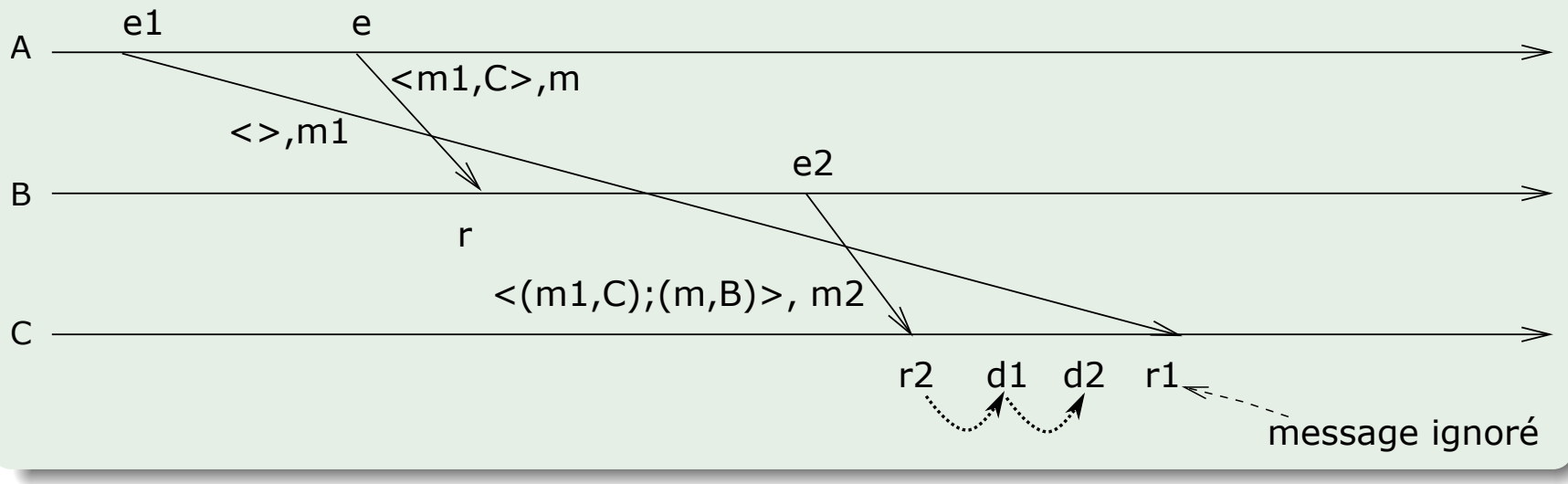
$H_c(m_{10}) = \{m_8, m_4, m_9, m_7, m_1, m_6, m_3, m_2\}$
$H_c(m_6) = \{m_3\}$
$H_c(m_8) = \{m_4, m_1, m_2\}$

Approche par surcharge (piggybacking) ♪♪♪

Principe

- Surcharger chaque message avec l'histoire des messages qui le précèdent causalement
- Dans le contexte du courrier électronique : Approche similaire du « réexpédier » avec copie de ce que l'on a reçu

Exemple



Approche par surcharge (piggybacking) ♪♪♪

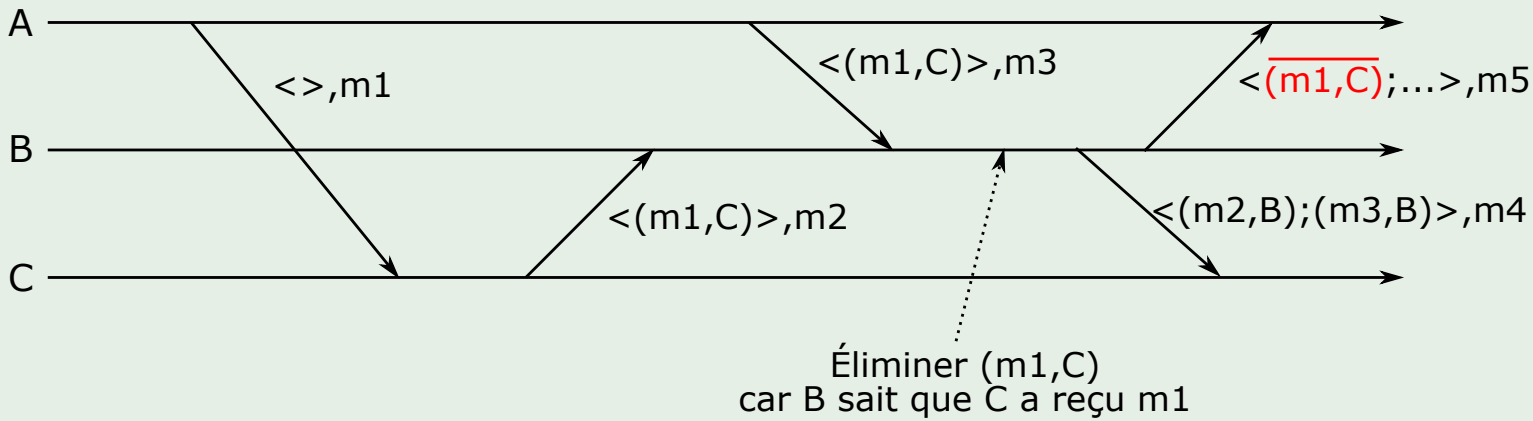
Mise en œuvre

☺ Simple et apport d'une certaine tolérance aux pertes de messages par redondance

☹ Messages de + en + longs

⇒ Quand, comment réduire les histoires ?

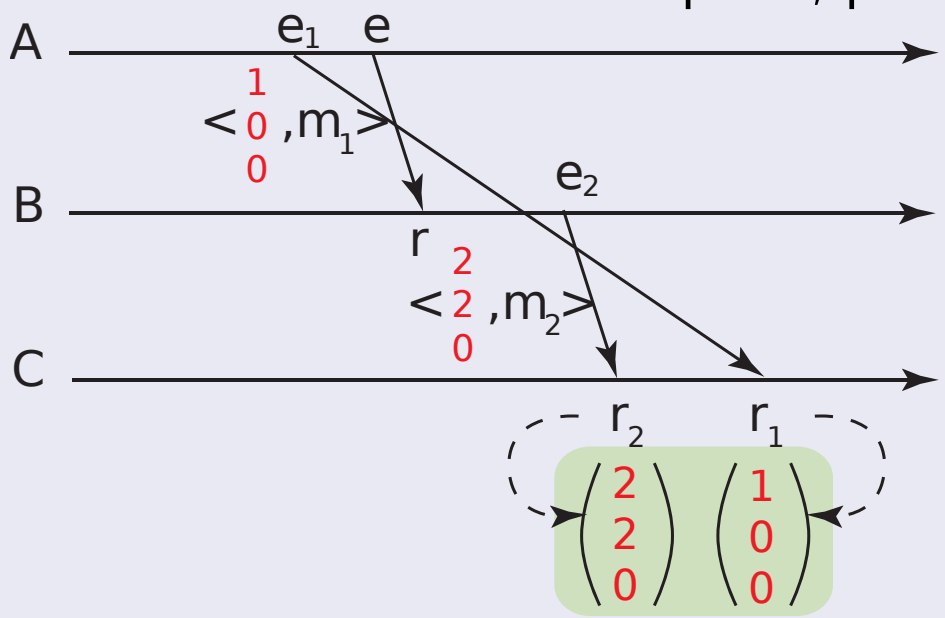
Exemple



Datation, premier essai...

Datation causale (horloge vectorielle)

Permet la détection des anomalies en réception, pas leur prévention



- Lors de r_2, C sait qu'il y a 2 événements sur A et 2 sur B (dont e_2) dans le passé de m_2, mais le concernent-ils ?
- Lors de r_1, C découvre qu'un événement de A, causalement antérieur à r_2, le concerne.

Approche par matrice

♪♪♪

Structures de données

Représenter l'histoire causale de chaque message

Chaque site S_s gère :

- MP_s : une matrice de précédence causale
  $MP_s[i, j]$ = nombre de messages émis de S_i vers S_j, connu de S_s
- $Dernier_s$: un vecteur de compteurs des messages reçus de chaque site
  $Dernier_s[i]$ = nombre de messages reçus du site S_i sur S_s
- Tout message est surchargée par une copie de la matrice MP du site émetteur

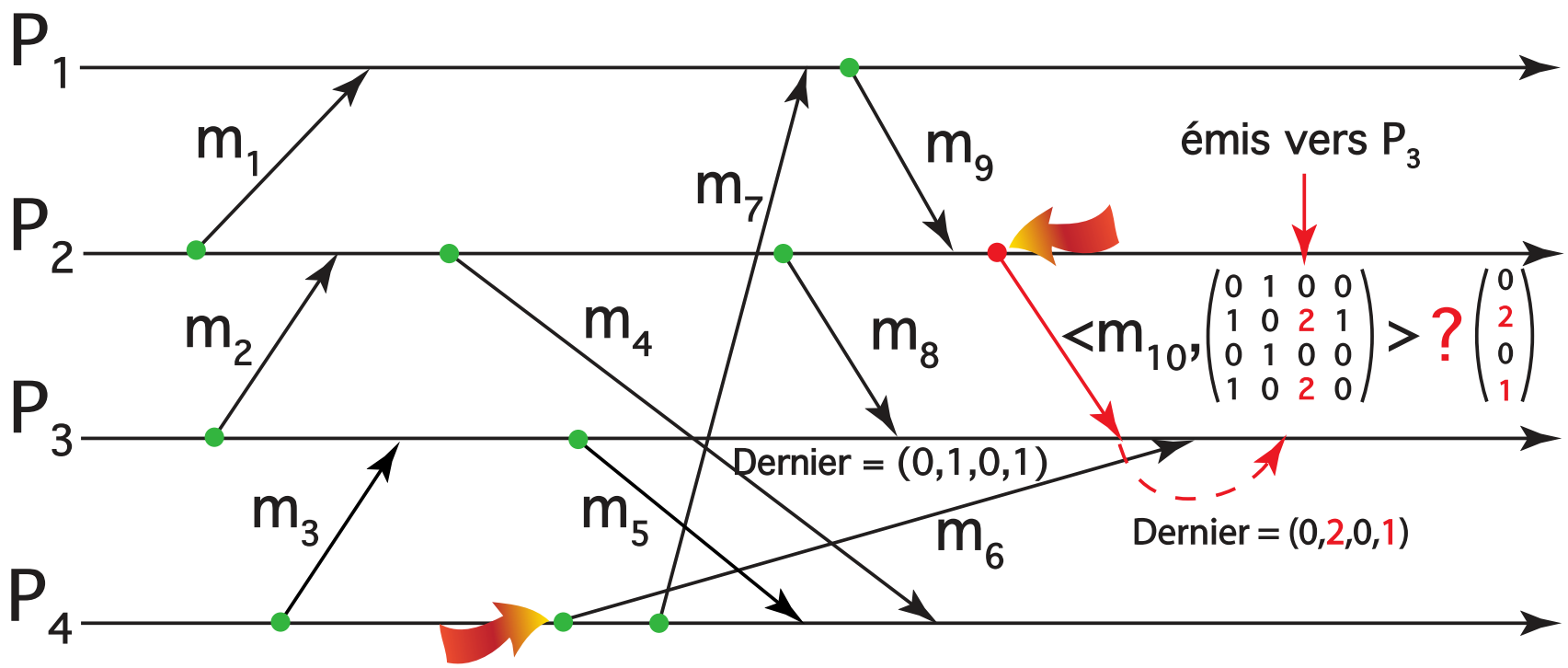
1. *The Causal Ordering Abstraction and a Simple Way to Implement it*, Michel Raynal, André Schiper and Sam Toueg. Information Processing Letters, 1991.

Actions associées aux événements

Type d'événement sur un site s	Action sur le site s
Émission sur s de m vers s'	$MP_s[s, s'] + +$ envoi de $\langle MP_s, M \rangle$
Réception sur s de $\langle MP, m \rangle$ issu de s'	$MP_s \leftarrow max(MP, MP_s)$ $Dernier_s[s'] + +$
Délivrance sur s de $\langle MP, m \rangle$ issu de s'	Délivrable(m) $\stackrel{\Delta}{=}$ $Dernier_s[s'] = MP[s', s]$ $\wedge \forall i \neq s : Dernier_s[i] \geq MP[i, s]$

m délivrable $\stackrel{\Delta}{=}$ FIFO entre s' et s et il ne précède pas les messages dont l'émission le précède causalement.

Contrôle des délivrances



Horloges de plus en plus précises

Horloges de Lamport

Passé de l'ensemble du système, connu de s, réduit à la longueur de la plus longue chaîne causale aboutissant à l'événement.

Horloges vectorielles

Connaissance de premier ordre : passé de s' que s connaît, connaissance par s que s' a eu un certain nombre d'événements.

Horloges matricielles

Connaissance de second ordre : connaissance par s de la connaissance par s' du passé de s''.

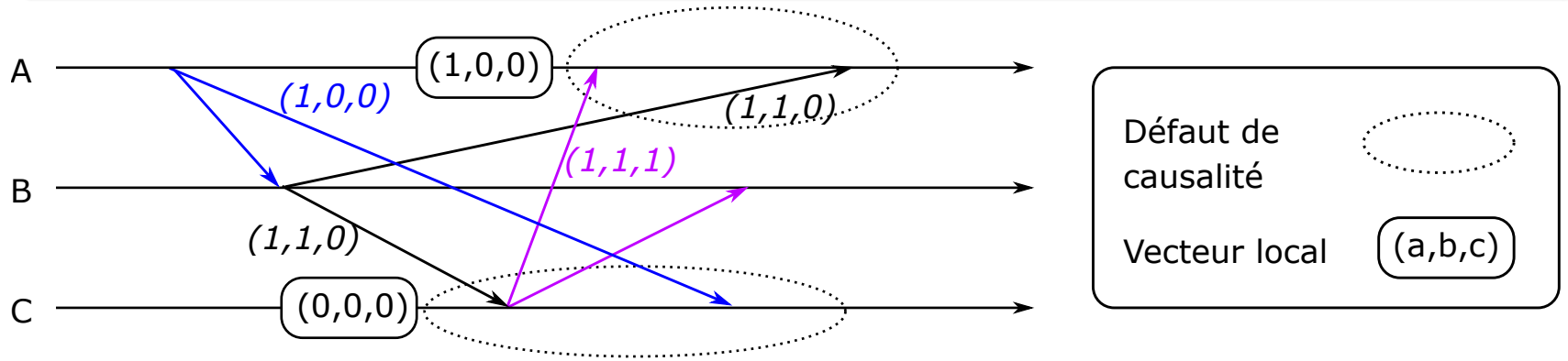
Par exemple, permet de savoir que *tous les sites* connaissent un événement donné.

Diffusion causalement ordonnée ♪♪♪

La diffusion ordonnée est plus simple que la communication point-à-point !

Approche par matrice causale

- Il suffit de gérer un vecteur d'émission au lieu d'une matrice
- Toutes les colonnes sont identiques : un processus envoie le même nombre de messages à tous



[Précis 3.2.2 pp.50–51]

1. *Lightweight Causal and Atomic Group Multicast*, Kenneth P. Birman, André Schiper and Pat Stephenson. ACM Trans. on Computer Systems, 1991.

Conclusion

- Datation logique des événements
- Protocoles de communication ordonnés
- Distinction réception / délivrance