Algorithmique répartie

Les Horloges

Cours de Lélia Blin
1er année de Master

Les horloges physiques

- C'est l'horloge physique des machines
- L'horloge qui donne l'heure suivant notre propre temps
- Elles sont cadencées par un mécanisme physique

Les horloges physiques

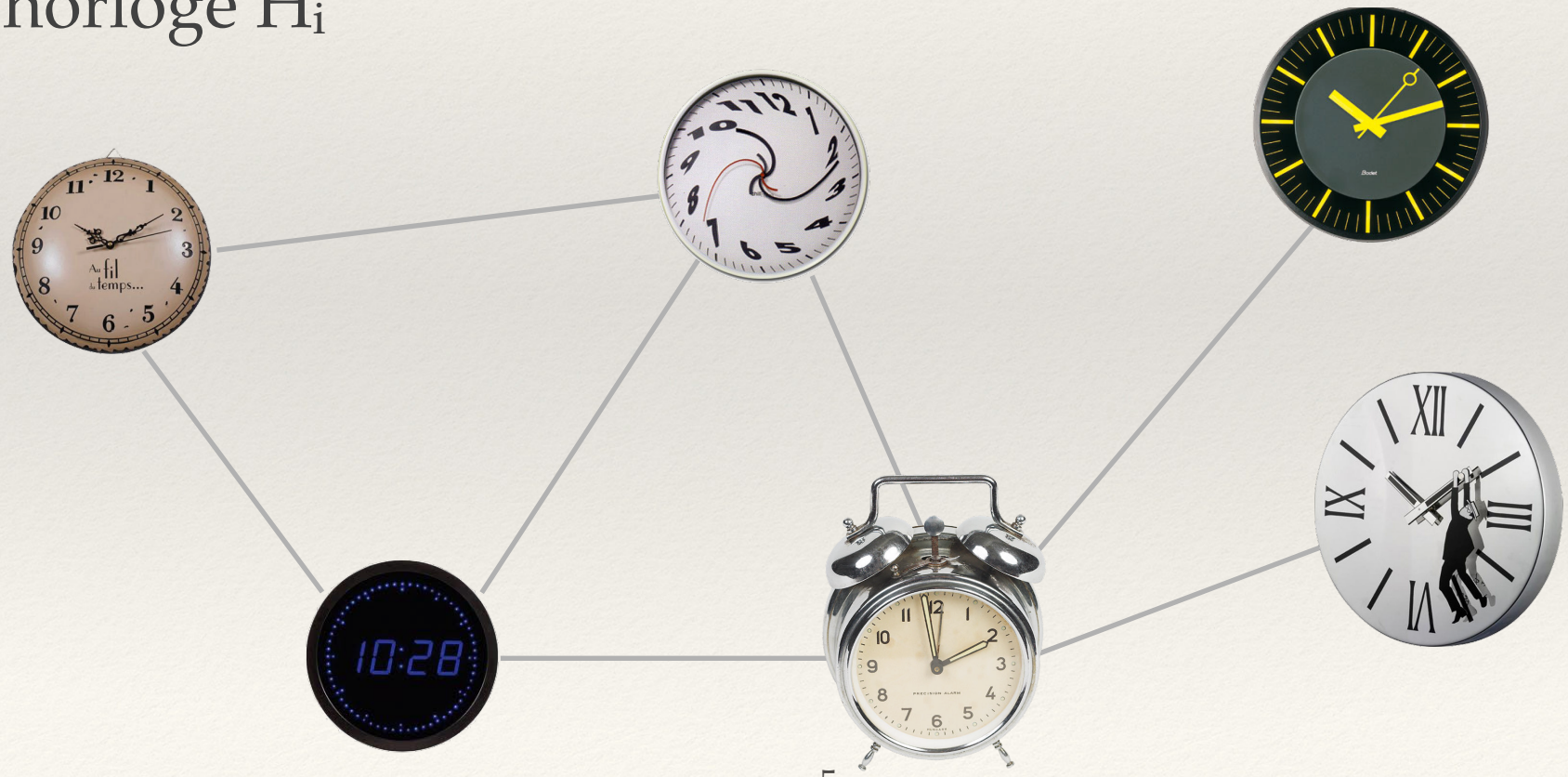
- Un site i qui lit l'heure au temps t
  - obtiendra la valeur $H_i(t)$
- Si l'horloge est parfaite
  - $H_i(t) = t$

Remarques

- Une horloge physique n'est bien sur pas parfaite elle peut subir des variations de précision
  - changement température
  - problèmes d'alimentation électrique
  - .....

Horloge dans les systèmes répartie

- Dans les systèmes répartie chaque site i a sa propre horloge H_i



Mesure du temps

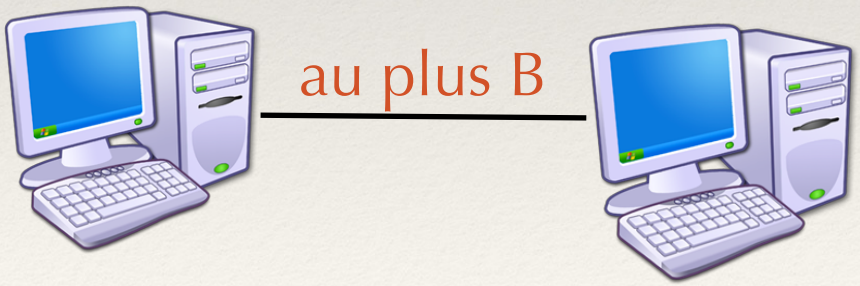
- De nombreux logiciel se basent sur une mesure de temps pour fonctionner
  - compilateur: compilation séparée
  - programmes qui font automatiquement
    - le ménage de fichiers
    - le classement de fichiers
    - l'archivage de fichiers
    - la destruction de fichiers

Mesure du temps

- Une mauvaise mesure du temps peut faire de nombreux dégâts catastrophiques
- Pour pallier à cela une des solutions est:
- De faire en sorte que tous les sites aient le «même» temps
- Cette solution s'appelle ***la synchronisation***

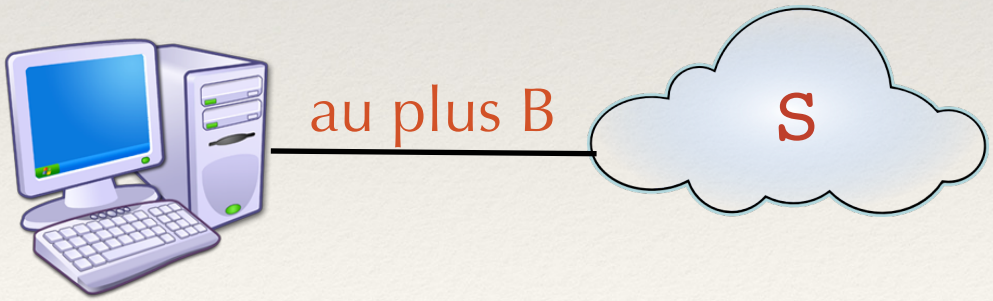
Synchronisation interne

- On veut une borne B
- B est la divergence des horloges entre elles
- Pour tout site i,j et tout temps t on veut:
- $| Hi(t) - Hj(t) | < B$



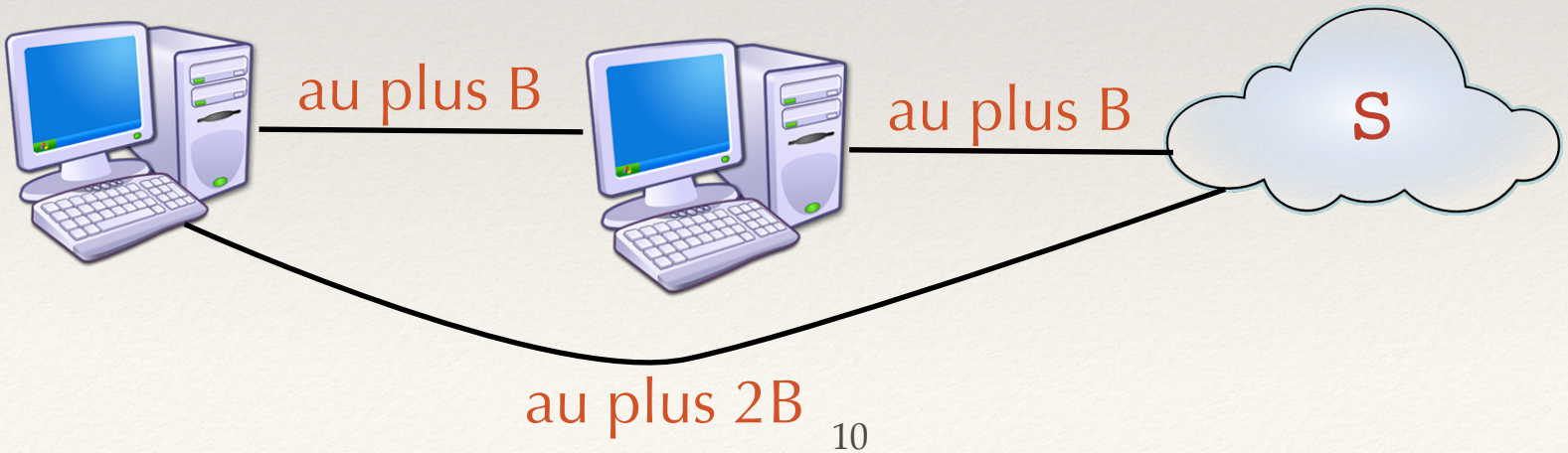
Synchronisation externe

- On veut que toute les horloges soient synchronisées par rapport à une source externe S avec une différence au plus B
- B est la divergence des horloges entre elles
- Pour tout site i et tout temps t on veut:
  - $| Hi(t) - S(t) | < B$



Remarque

- Si les horloges ont une synchronisation externe d'au plus B alors elles ont une synchronisation interne d'au plus 2B



Synchronisation interne de deux horloges dans le cas d'un réseau faiblement synchrone

Borne supérieure

- Dans ce modèle, on connaît une borne supérieure max
- Autrement dit:
  - le temps maximum que met un message pour transiter
- ou
  - le temps maximum que met un message pour aller d'un site vers un autre site

Borne inférieure

- On peut toujours donner une borne inférieure
  - min≥ 0
- Sinon le message n'est pas envoyer
- On peut aussi mesurer de façon empirique
  - En se basant sur les contraintes physiques rencontrées pour communiquer

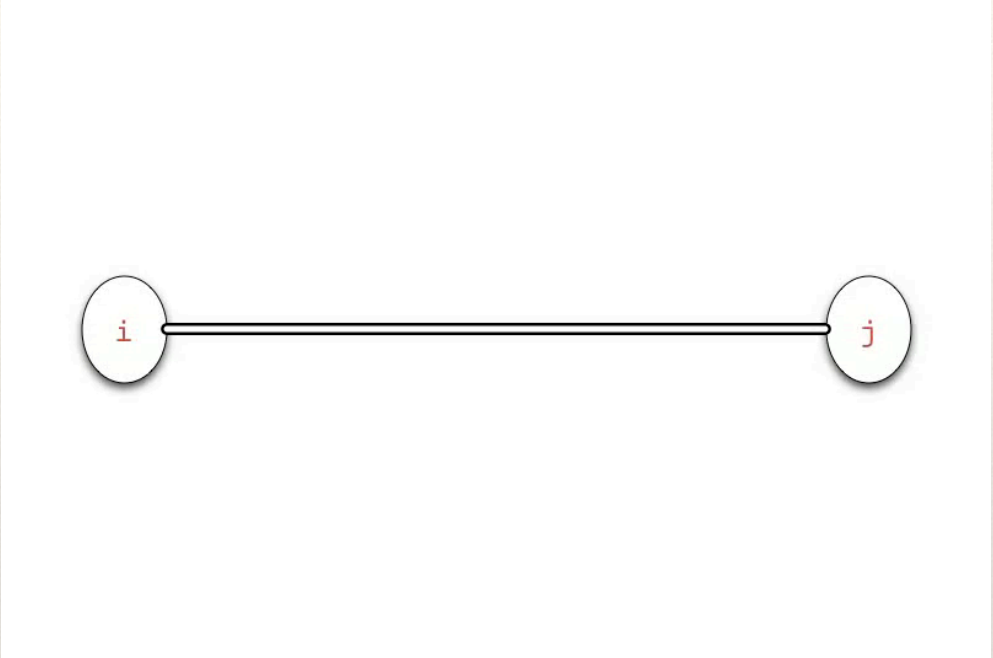
Méthode

- Lorsque le site i veut synchroniser son horloge sur l'horloge du site j
- il envoie un message <HORLOGE> à j

Méthode

- Lorsque j reçoit ce message
  - il renvoie le message <HORLOGE,Hj(t)> à i
- Lorsque i reçoit ce message
  - i donne à son horloge l'heure suivante
  - $H_i(t) = H_j(t)+(max+min)/2$

Exemple



min= 2s
max= 10s
$H_i(t)=2'34''+(2''+10'')/2=2'40''$

Soit d le temps réel de parcours
Donc $H_j(t)=2'34''+ d$
avec $min \leq d \leq max$

Les cas extrêmes: exemple

- Cas très rapide:
  - d=2″
  - $H_i(t)$=2′40″ et $H_j(t)$=2′36″
  - différence =4″
- Cas très lent
  - d=10″
  - $H_i(t)$=2′40″ et $H_j(t)$=2′44″
  - différence =4″

Cas très rapide

- d=min
- $H_i(t)= H_j(t)+(max +min)/2$ et
- $H_j(t)= H_j(t)+min$
- différence = $[H_j(t)+(max +min)/2]-[H_j(t)+min]$
- différence =(max-min)/2

Cas très lent

- d=max
- $H_i(t)= H_j(t)+(max +min)/2$ et
- $H_j(t)= H_j(t)+max$
- différence = $[H_j(t)+max]-[H_j(t)+(max +min)/2]$
- différence =(max-min)/2

Borne

- B = (max-min)/2

Synchronisation externe

- On va considérer un réseaux asynchrone
- Dans ce cas la borne max n'existe pas
- Le réglage va se faire en mesurant empiriquement la durée d'aller retour
- Le réglage va se faire par rapport à cette durée mesurée

Méthode

- Lorsque le site i veut synchroniser son horloge sur l'horloge d'une source externe S
  - il envoie un message <HORLOGE> à S
  - et
  - Il déclenche en même temps un chronomètre

Méthode

- Lorsque le site i reçoit le message de la source:
  - Il arête son chronomètre
  - T est le temps mesurée
- Le site i met son horloge à **S(t)+T/2**

Inconvénients

- Dans se cas on considère que:
- Les temps aller retour ont étaient les mêmes.
- Ce qui est rarement le cas

Message rapide à l'aller

- Considérons le cas d'un message rapide à l'aller (temps 0s)
- Et un message lent au retour (temps T).
- Quand i change son horloge a S(t)+T/2
- S à une horloge de S(t)+ T
- Donc il y a une différence de T/2

Message rapide au retour

- Considérons le cas d'un message lent à l'aller (temps T)
- Et un message rapide au retour (temps 0s)
- quand i change son horloge a S(t)+T/2
- S à une horloge de S(t)+ 0
- Il y a donc une différence de T/2

Commentaires

- En pratique cette mesure est appliquée de nombreuse fois et il est fait des **statistiques** sur les temps d'aller retour.
- Ces statistiques sont utilisées pour synchronisée l'horloge.
- Ce réglage est bon si les variations aller retour sont faibles ce qui n'est pas toujours le cas.

Questions

- A t-on besoin pour chaque tâche d'avoir une mesure de temps exacte?
- Exemples:
  - Dans une conversations par emails:
    - L'heure exacte d'envoie des email est-il plus important que l'ordre des emails dans la conversations?
  - Dans la gestion des fichiers:
    - L'heure exacte de la sauvegarde externe est-elle importante pour pouvoir effacer des donner locales?
    - Ou est-il seulement important de savoir que la sauvegarde externe a été faite avant d'effacer localement?

Horloges logiques

Constat

- Les horloges physiques sont difficiles à synchroniser finement.
- Toutes les applications n'ont pas besoin d'avoir une mesure de temps très précise
- Il suffit dans bien des cas de savoir si un événement à eu lieu **avant** ou **après** tel autre.

Constat

- Nous allons voir des horloges suffisamment précises pour distinguer l'avant de l'après
- Mais insuffisamment précises pour mesurer un temps physique

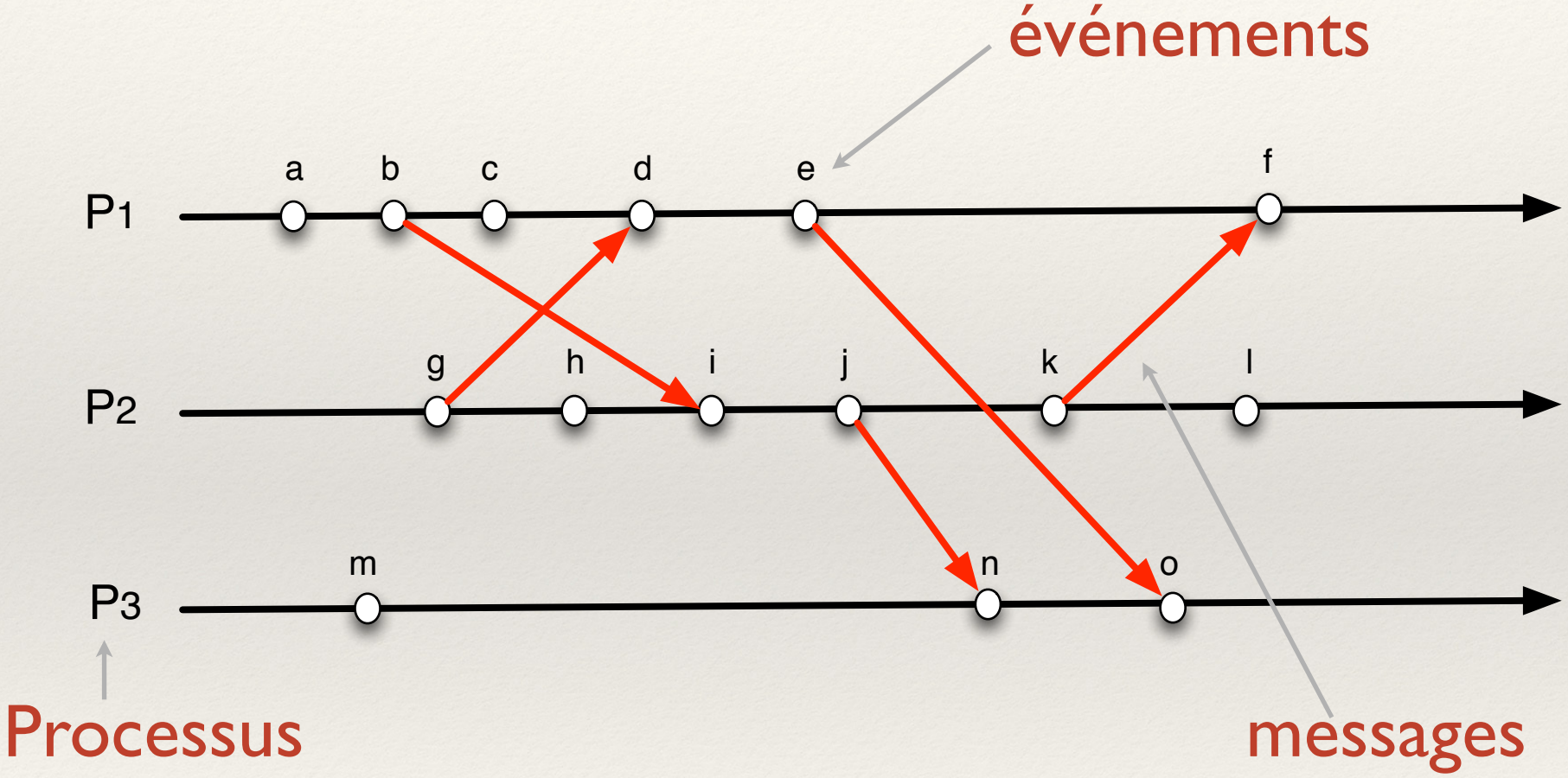
L'ordre causal

- Rappel:
  - Les systèmes répartis n'ont ni mémoire commune ni horloge commune.
- On veut pouvoir capturer:
  - Un comportement global correct du système
  - Ordonner les envois et réceptions de messages
- Le problème c'est que lorsque on regarde au niveau de chaque machine cela est très compliqué

Evénements locaux

- Un événement **e** est :
  - soit l'envoie d'un message.
  - soit la réception d'un message.
  - soit un calcul interne.

Graphe de précédence immédiate



Un ordre causal

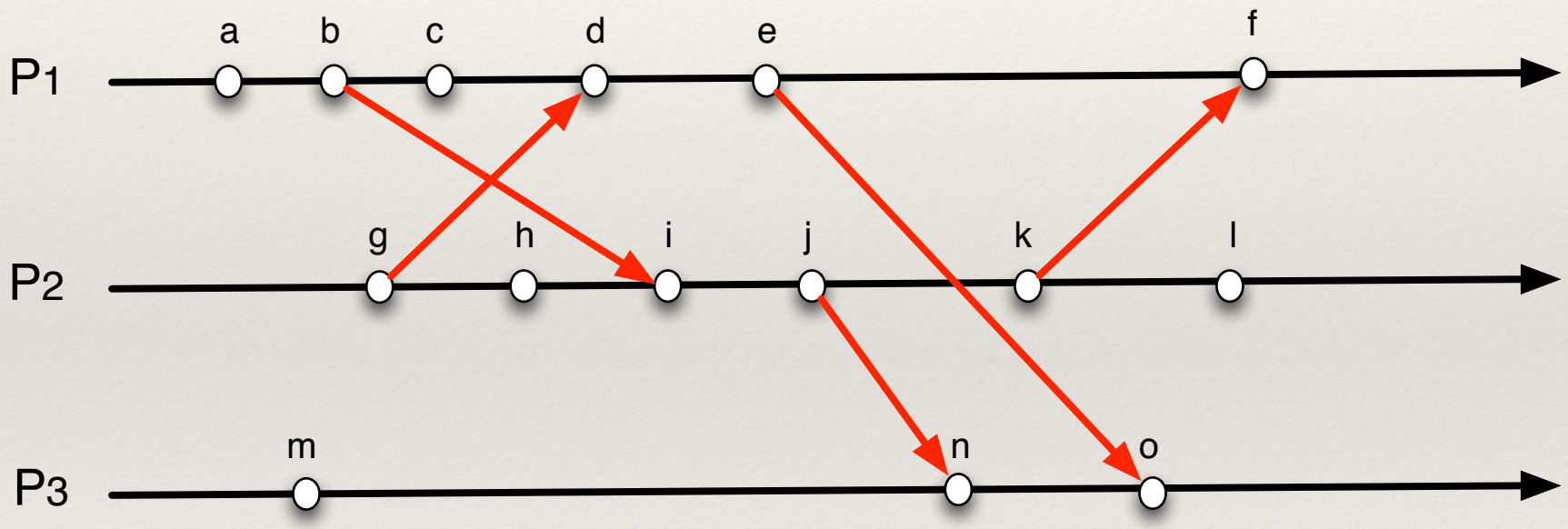
- Un ordre causal est un **ordre partiel** sur les événements.
- Si e et e' sont sont deux événements
  - on note e → e' (e précède e' suivant l'ordre)
  - si et seulement si une des trois conditions suivantes est vraie:

1. e et e' ont lieu sur le même site avec e avant e'
2. e =Envoyer(<M>) et e'=Recevoir(<M>)
   - M est le même message
3. Il existe un événement e'' tel que e → e'' et e''→ e'

Remarques

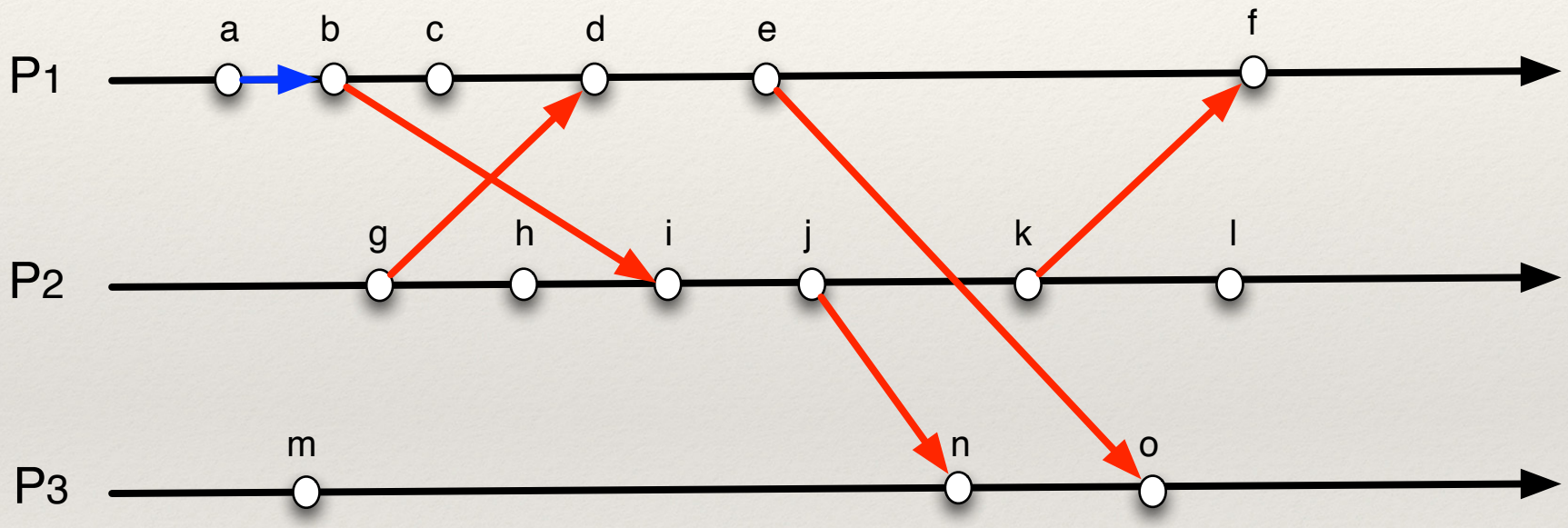
- La condition 3 est la clôture transitive de la relation $\rightarrow$
- Le graphe de la relation $\rightarrow$ n'a aucun circuit
  - on ne remonte pas le temps
- Certains sommets (événements) ont des prédécesseurs, d'autres pas.

Graphe de précédence immédiate



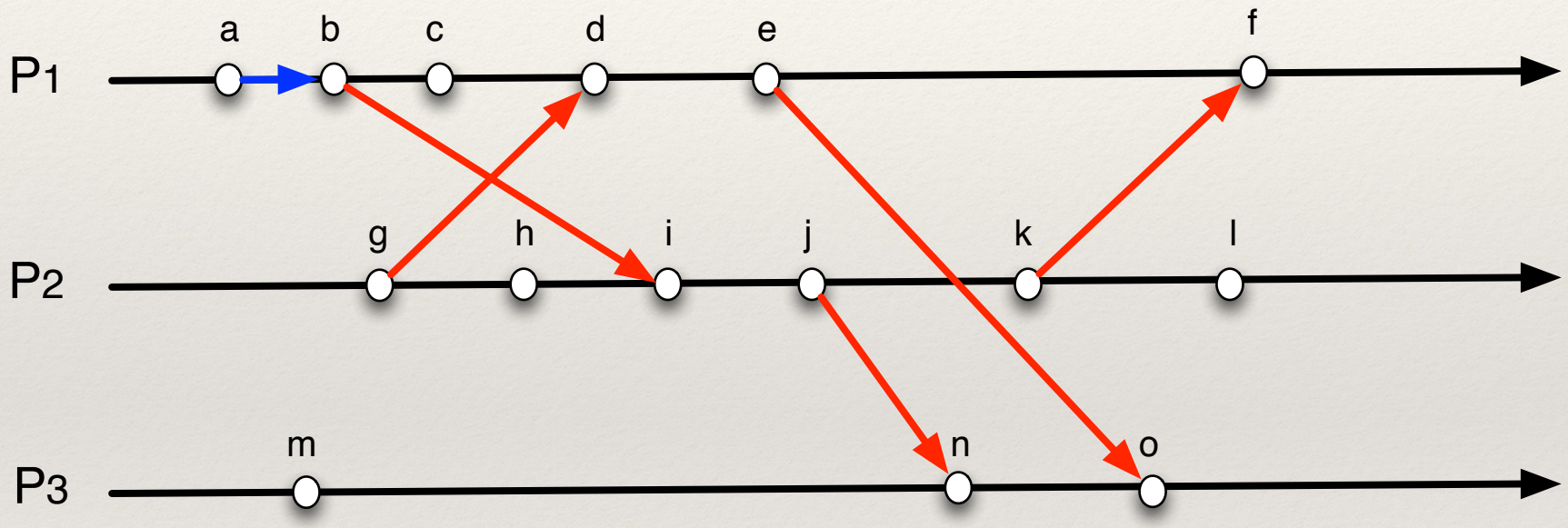
Existe t'il un ordre causal entre a et b

Graphe de précédence immédiate



Existe t'il un ordre causal entre a et b
a→b

Graphe de précédence immédiate

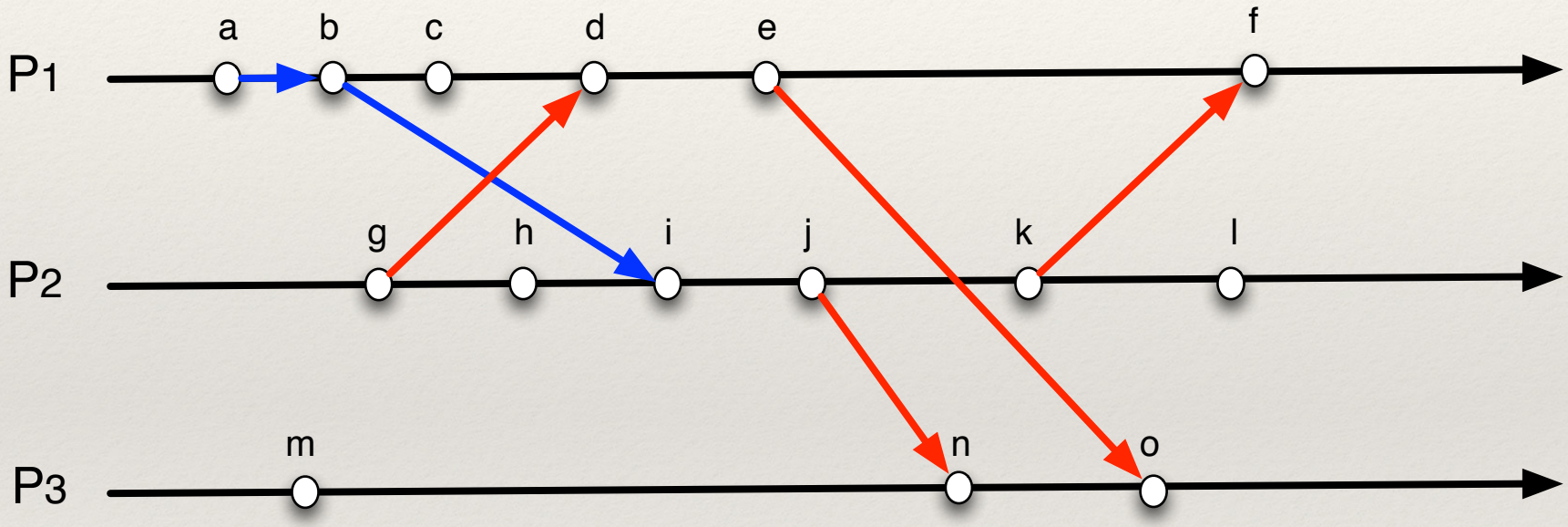


Existe t'il un ordre causal entre a et b

a→b

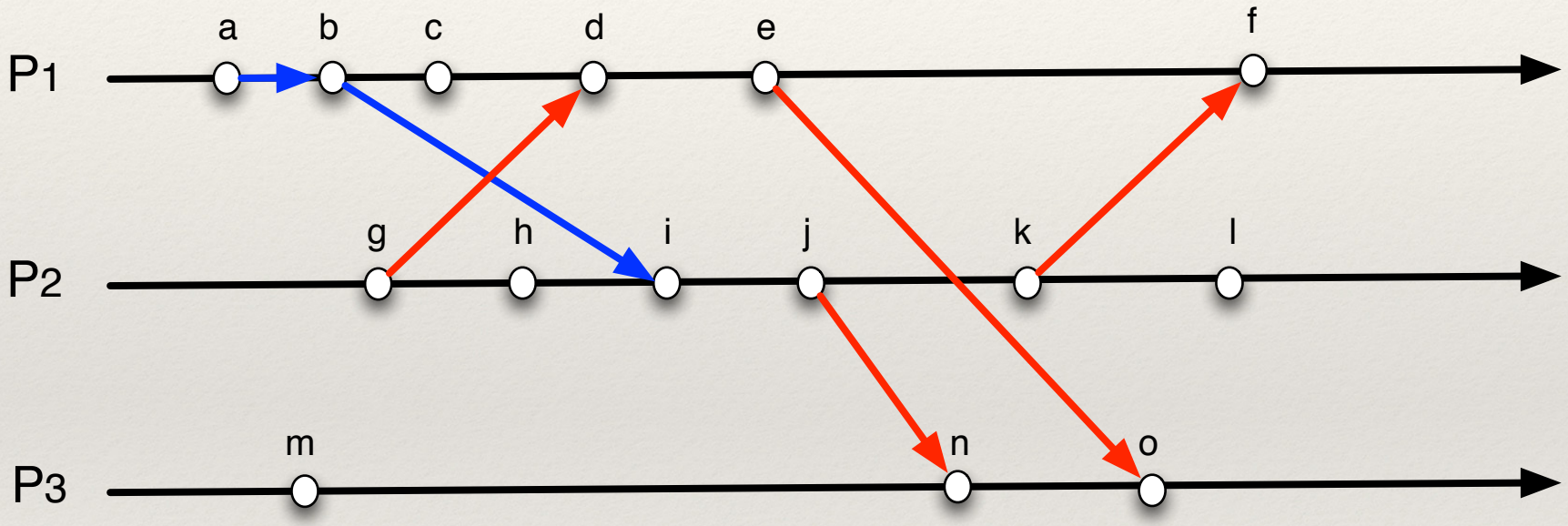
Existe t'il un ordre causal entre a et i?

Graphe de précédence immédiate



Existe t'il un ordre causal entre a et b
a→b
Existe t'il un ordre causal entre a et i?
a→b→i

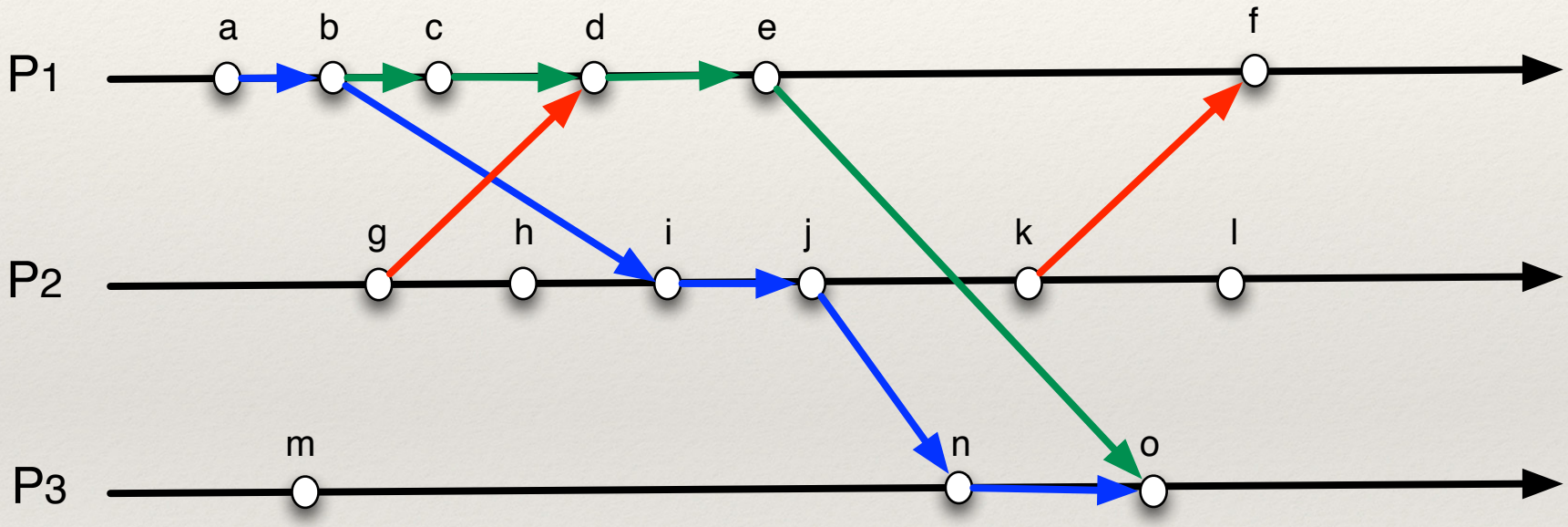
Graphe de précédence immédiate



Existe t'il un ordre causal entre a et b
a→b
Existe t'il un ordre causal entre a et i?
a→b→i

Existe t'il un ordre causal entre a et o?

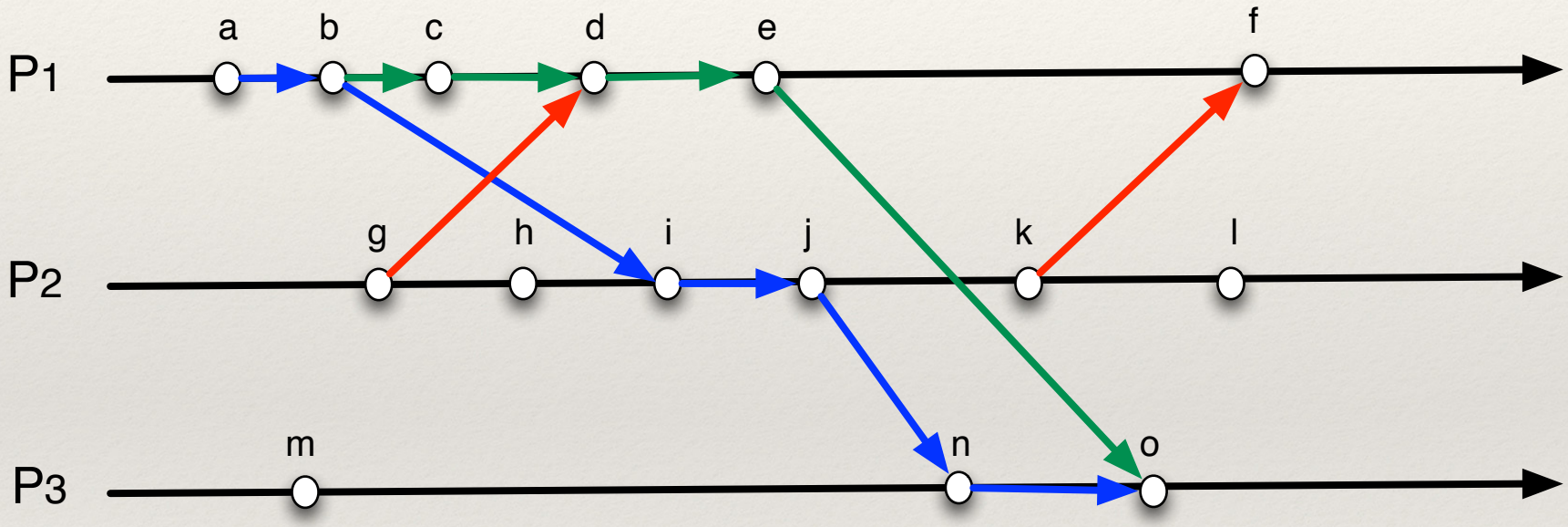
Graphe de précédence immédiate



Existe t'il un ordre causal entre a et b
a→b
Existe t'il un ordre causal entre a et i?
a→b→i

Existe t'il un ordre causal entre a et o?
a→b →i →j →n →o ou
a→b →c→d→e→o

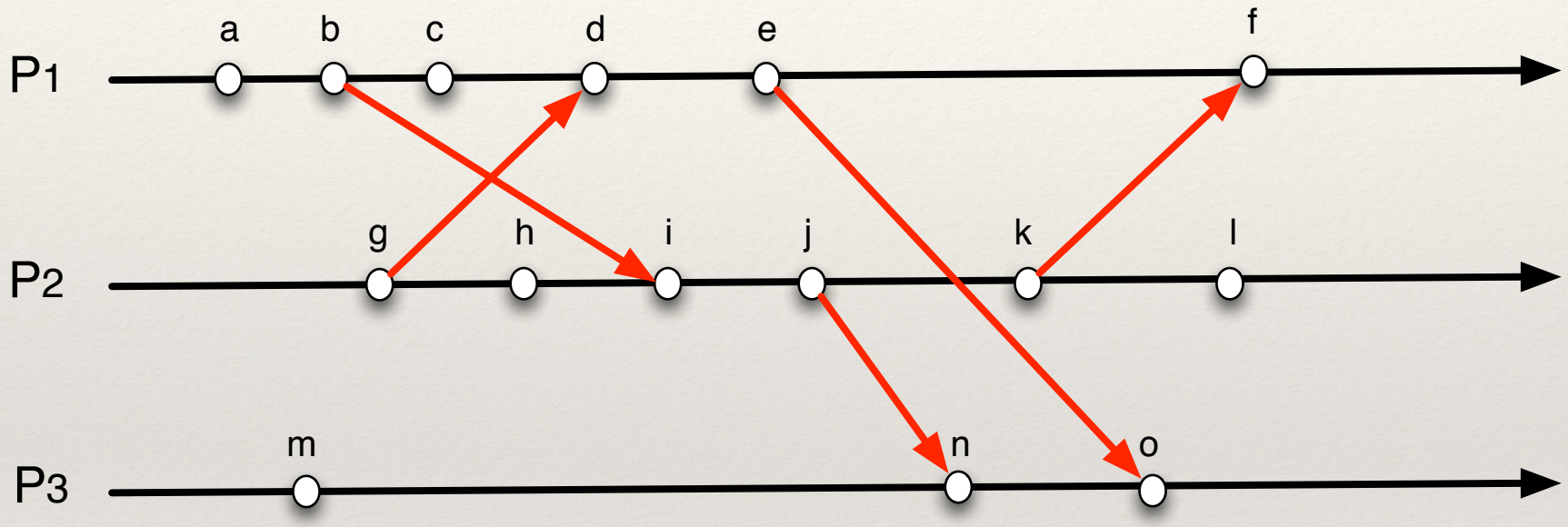
Graphe de précédence immédiate



Existe t'il un ordre causal entre a et b
a→b
Existe t'il un ordre causal entre a et i?
a→b→i

Existe t'il un ordre causal entre a et o?
a→b →i →j →n →o ou
a→b →c→d→e→o
Existe t'il un ordre causal entre c et n?

Graphe de précédence immédiate



Existe t'il un ordre causal entre a et b
a→b
Existe t'il un ordre causal entre a et i?
a→b→i

Existe t'il un ordre causal entre a et o?
a→b →i →j →n →o ou
a→b →c→d→e→o
Existe t'il un ordre causal entre c et n?
non

Horloge logique de Lamport

- Lamport a proposé un algorithme pour étiqueter les événements d'un système.
- Cet algorithme respecte l'ordre causal
- Si on note L(e) l'étiquette d'un événement e alors:
  - $e \rightarrow e' \Rightarrow L(e) < L(e')$

Algorithme de Lamport

- Chaque site i gère un compteur d'entier h_i
- Ce compteur est initialisé à 0
- Il est modifié de la façon suivante

Algorithme de Lamport

- si e est un événement interne alors faire
  - $h_i := h_i + 1$ et $L(e) = h_i$

Algorithme de Lamport

- Si e est l'envoie d'un message M alors faire
  - $h_i := h_i + 1$ et $L(e) = h_i$
- On envoie le message M avec l'estampille L(e)
  - Envoyer ($<M, h_i>$)

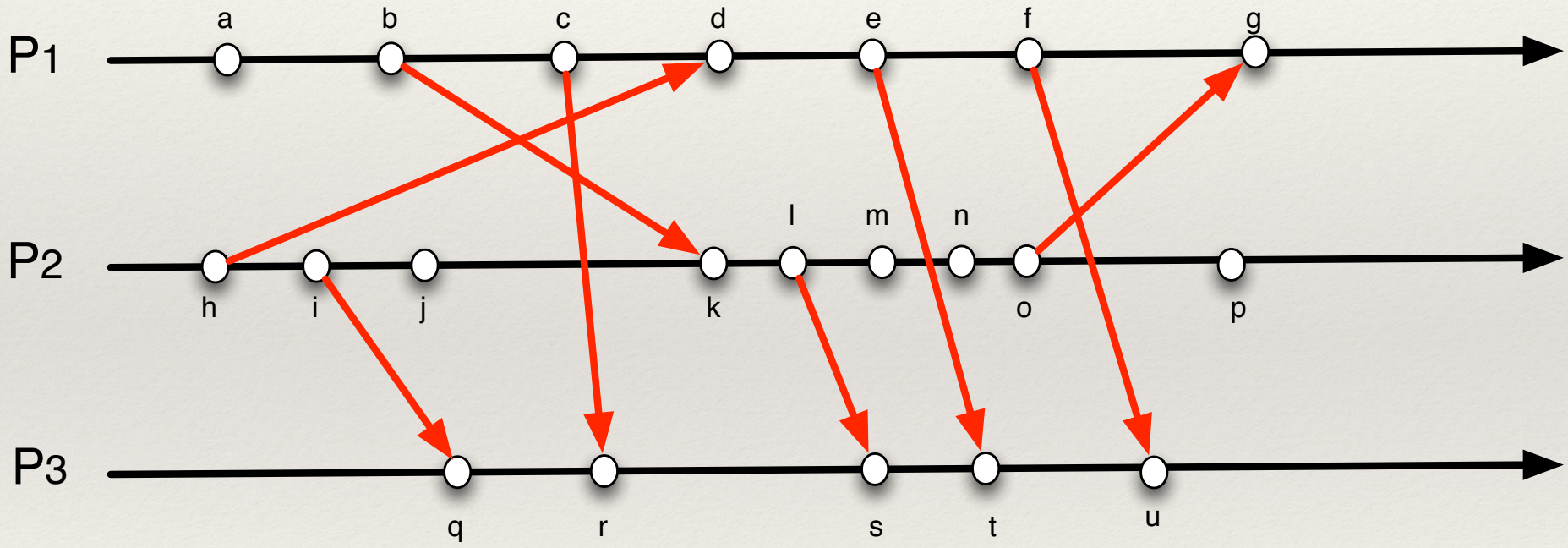
Algorithme de Lamport

- Si e est la réception d'un message Recevoir (<M, h >) alors faire
  - $h_i := \max(h,h_i)+1$ et $L(e)= h_i$

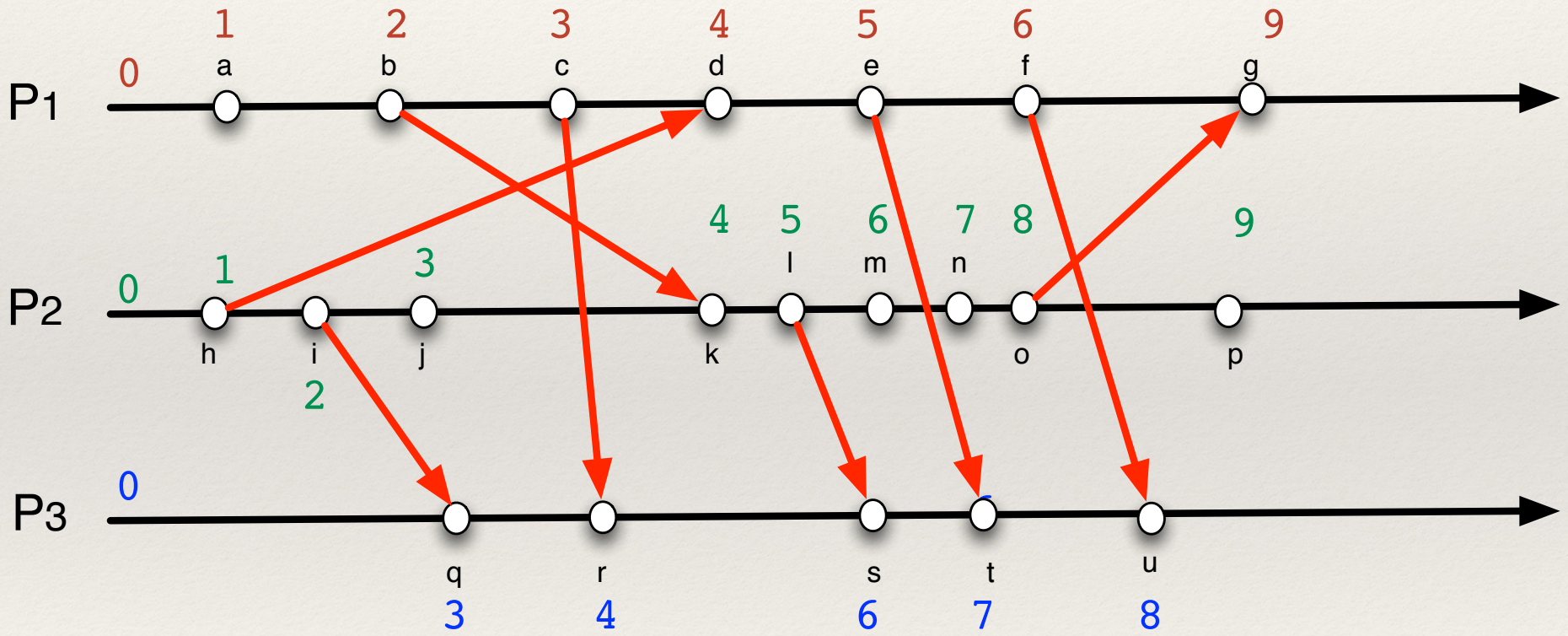
Algorithme de Lamport

- Si e est un événement interne alors faire
  - $h_i := h_i + 1$ et $L(e) = h_i$
- Si e est l'envoie d'un message M alors faire
  - $h_i := h_i + 1$ et $L(e) = h_i$
  - on envoie le message M avec l'estampille L(e)
  - Envoyer (<M, h_i >)
- Si e est la réception d'un message Recevoir (<M, h >) alors faire
  - $h_i := \max(h, h_i)+1$ et $L(e) = h_i$

Etiquetage de Lamport



Etiquetage de Lamport

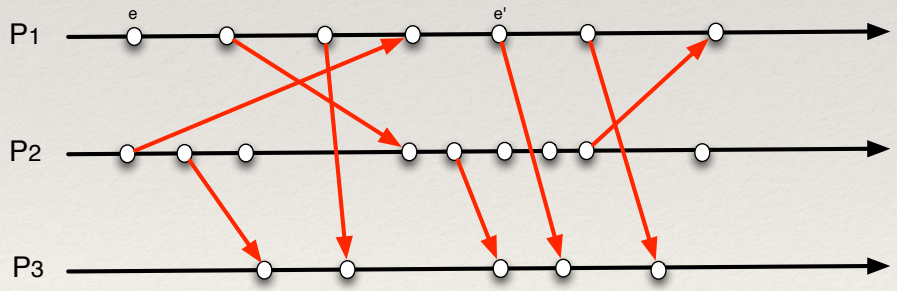


Théorème

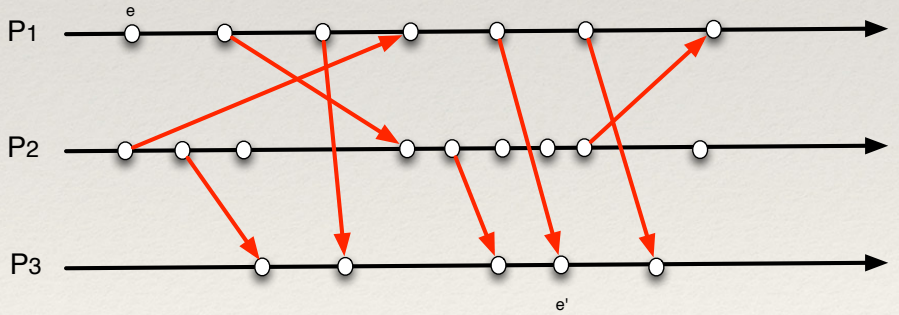
- Théorème: $e \rightarrow e' \Rightarrow L(e) < L(e')$

Preuve

- Considérons deux événements e et e' tel que e→e'
- Il faut considérer deux cas:



e et e' ont lieu sur le même site.



e et e' ont lieu sur des sites différents.

Preuve

- 1er cas: e et e'ont lieu sur le même site
- L'algorithme incrémente les événement qui sont sur un même sites
- l'étiquette de e' sera plus grande que celle de e
  - donc L(e)<L(e')

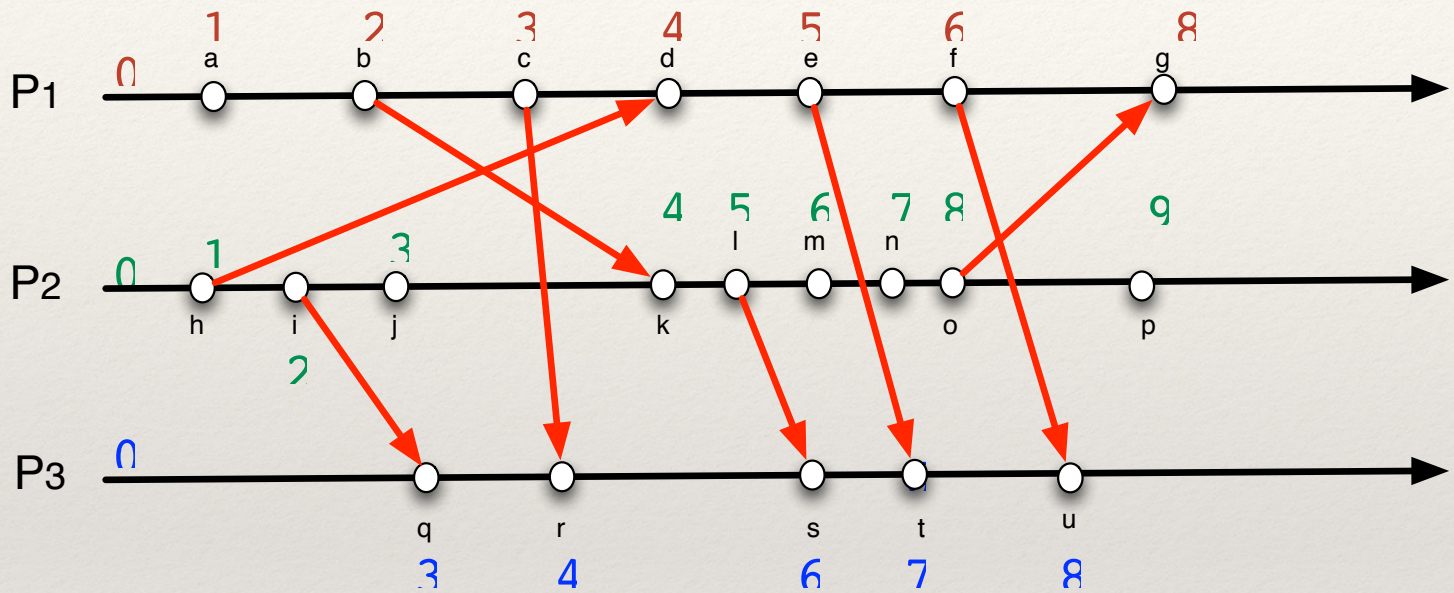
Preuve

- 2eme cas: e et e’ sont deux événements qui ont lieu sur deux sites différents i et j.
- Dans se cas il est nécessaire d’avoir eu au moins un message M entre les deux sites:
  - e →Envoyer(<M>)→Recevoir(<M>)→ e’
- Il est facile de voir que
  - L(e’)>L(Envoyer(<M>))≥L(e)

Rendre les horloges de Lamport deux à deux comparable

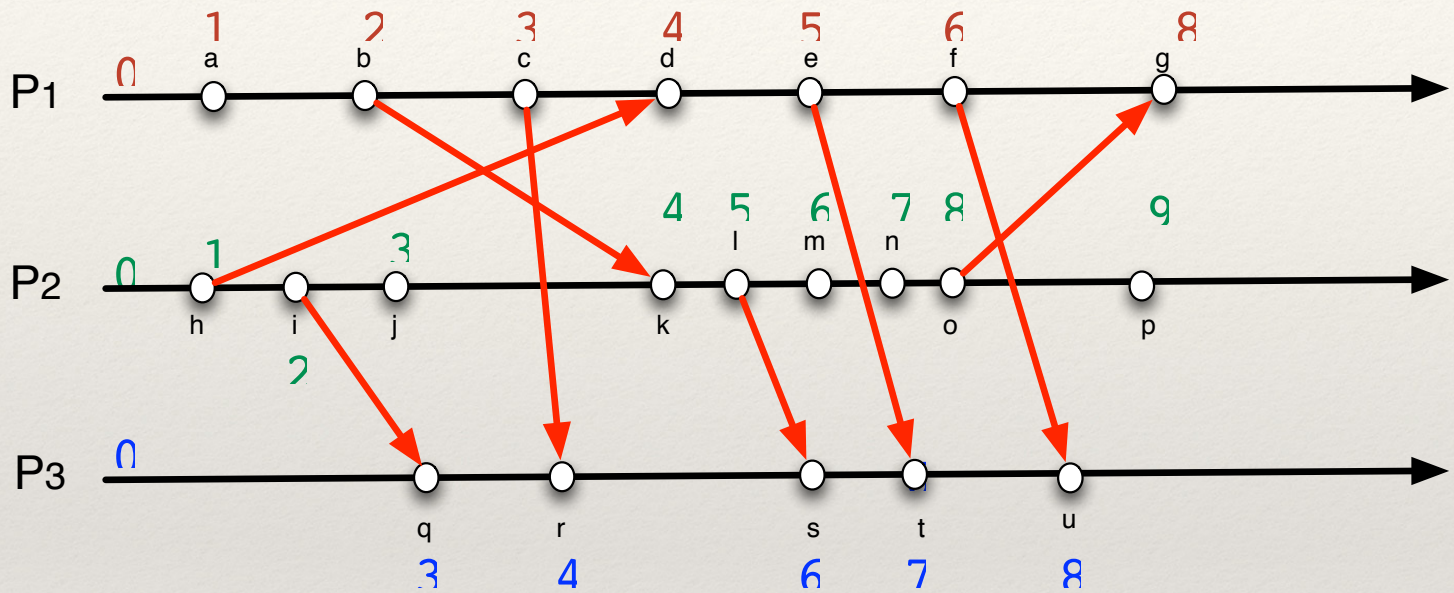
- Il est souhaitable de pouvoir comparer sans ambiguïté deux événements e et e'même si ceux-ci sont incomparables par l'ordre causal.
- Pour cela il suffit d'étiqueter un événement e ayant lieu sur le site i par : $L(e)=(h_i,i)$
- La gestion de l'étiquette reste la même, par abus de langage ont utilisera
  - $(h_i,i) < (h_j,j) \Leftrightarrow (h_i< h_j$ ou $(h_i =h_j$ et $i<j)$

Etiquetage de Lamport



Ordre total*:

Etiquetage de Lamport



Ordre total*:

- Le problème c'est qu'avec les horloges de Lamportla propriété n'est vérifiée que dans un sens.
- Il est facile de construire un exemple dans lequel
  - $L(e)<L(e')$ mais $e \nrightarrow e'$
- Pour que l'implication inverse soit satisfaite nous allons voir des étiquettes plus précises mais plus grosses 

Horloges Vectorielles

Définition

- Chaque site i va maintenir localement un vecteur V_i à n composantes entières.
- Au départ chaque vecteur est initialisé à zéro: pour tout i, V_i=(0,0,0,....,0)
- Ce vecteur est gérer de la façon suivante

Définition

- si e est un événement interne alors faire
  - $V_i[i] := V_i[i] + 1$ et $V(e) = V_i$

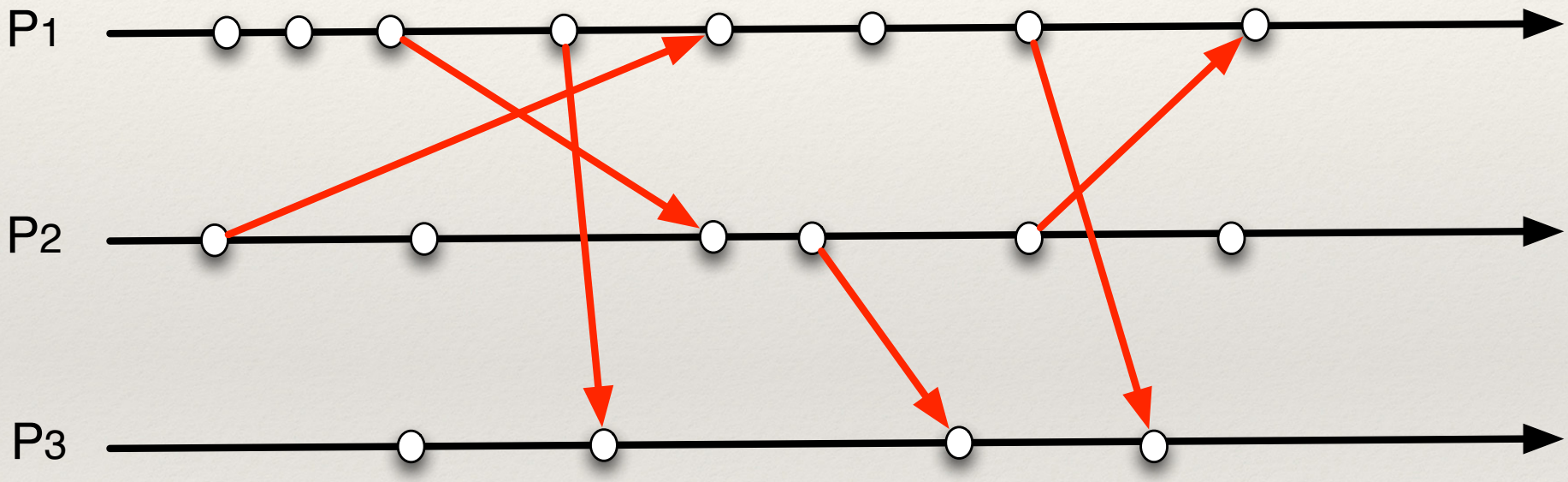
Définition

- Si e est l'envoie d'un message M alors faire
  - $V_i[i] := V_i[i] + 1$ et $V(e) = V_i$
  - on envoie le message M avec l'estampille V(e)
  - Envoyer (<M, V_i >)

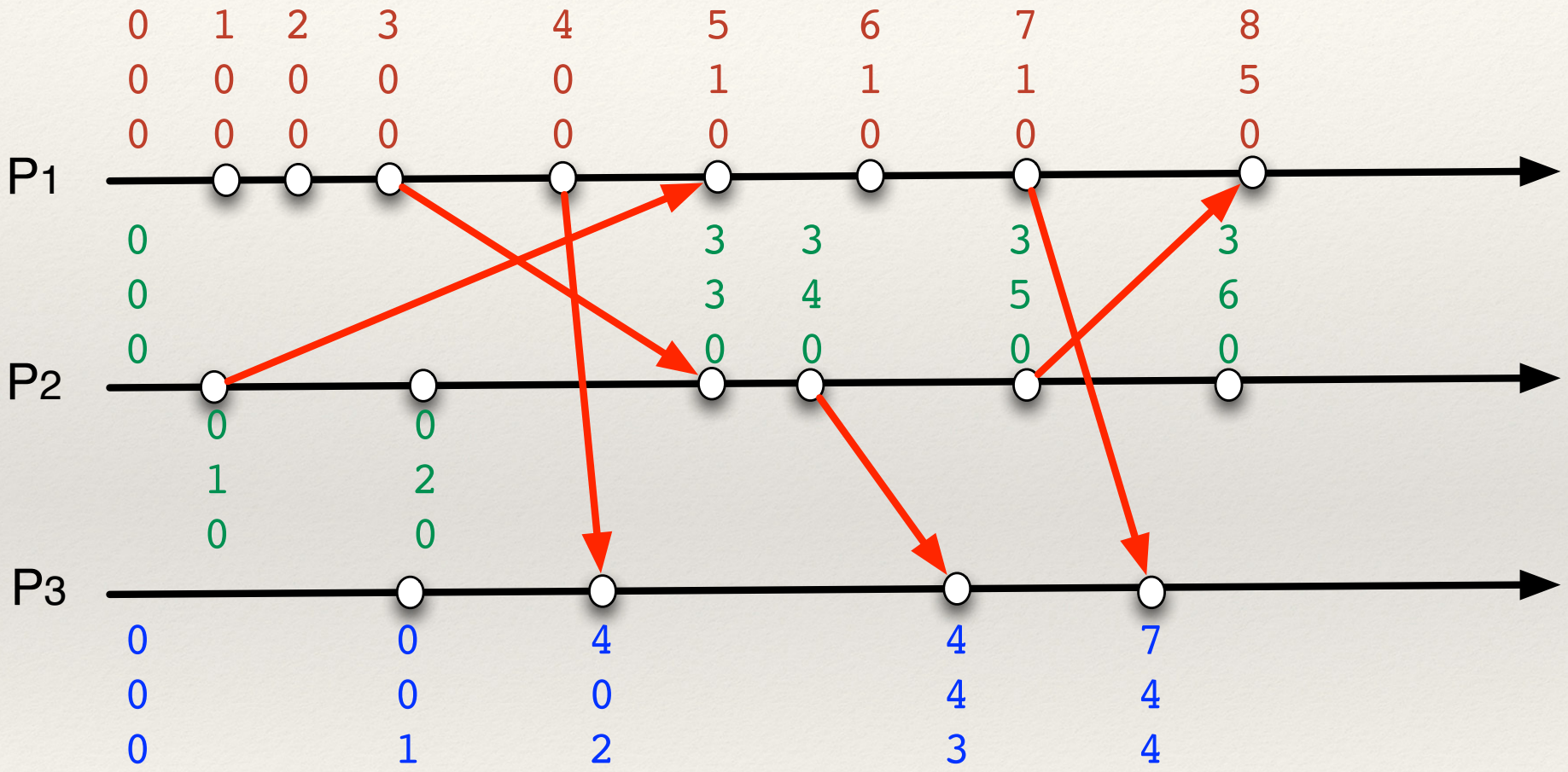
Définition

- si e est la réception d'un message Recevoir (<M, V >) alors faire
  - $V_i[i] := V_i[i] + 1$
  - et $V_i[j] := \max \{V_i[j], V[j]\}$ pour tout j puis $V(e) = V_i$

Etiquetage vectoriel



Etiquetage vectoriel



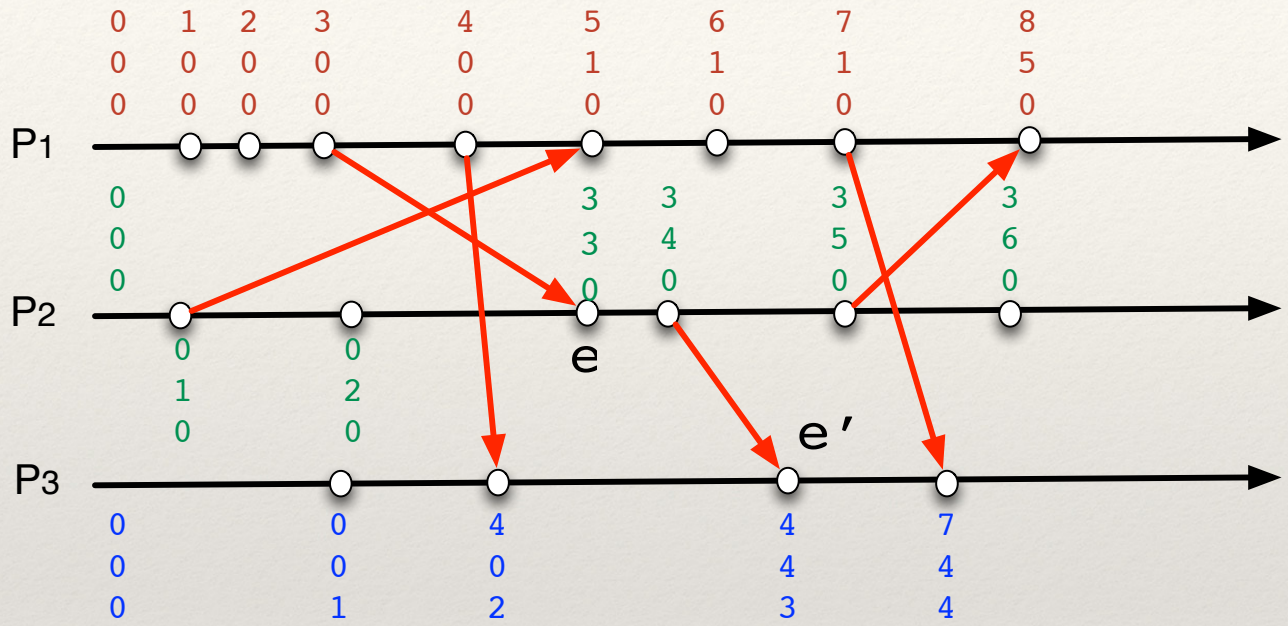
Comparaison des horloges vectorielles

- Grâce à ce mécanisme, chaque événement e reçoit une étiquette V(e)
- Pour comparer deux événements il faut comparer leurs étiquettes
- Il nous faut donc une fonction de comparaison
  - noter $<_v$

Comparaison des horloges vectorielles

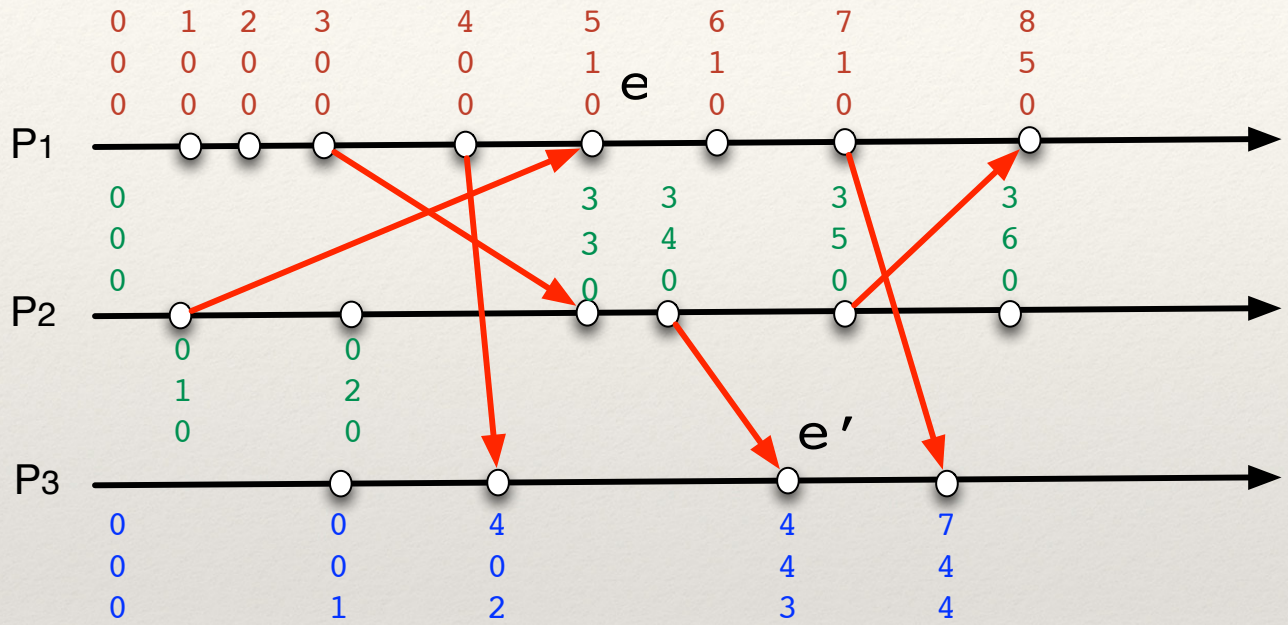
- Considérons deux vecteur d'entiers
  - $V_1 = (x_1,...,x_n)$ et $V_2 = (y_1,...,y_n)$
- on a:
  - $V_1 <_v V_2 \Leftrightarrow \forall i=1,...,n\ V_1[i] \leq V_2[i]$
  - On notera que cet ordre n'est pas total

Etiquetage vectoriel



$$L(e) = \begin{pmatrix} 3 \\ 3 \\ 0 \end{pmatrix} \text{ et } L(e') = \begin{pmatrix} 4 \\ 4 \\ 3 \end{pmatrix} \text{ comme } \begin{matrix} 3 < 4 \\ 3 < 4 \\ 0 < 3 \end{matrix} \Rightarrow V(e) <_v V(e')$$

Etiquetage vectoriel



$$L(e) = \begin{pmatrix} 5 \\ 1 \\ 0 \end{pmatrix} \text{ et } L(e') = \begin{pmatrix} 4 \\ 4 \\ 3 \end{pmatrix} \text{ comme } \begin{matrix} 5 > 4 \\ 1 < 4 \\ 0 < 3 \end{matrix} \Rightarrow V(e) \text{ et } V(e') \text{ sont incomparables}$$

Théorème

- $e \rightarrow e' \Leftrightarrow V(e) <_v V(e')$

Preuve du théorème

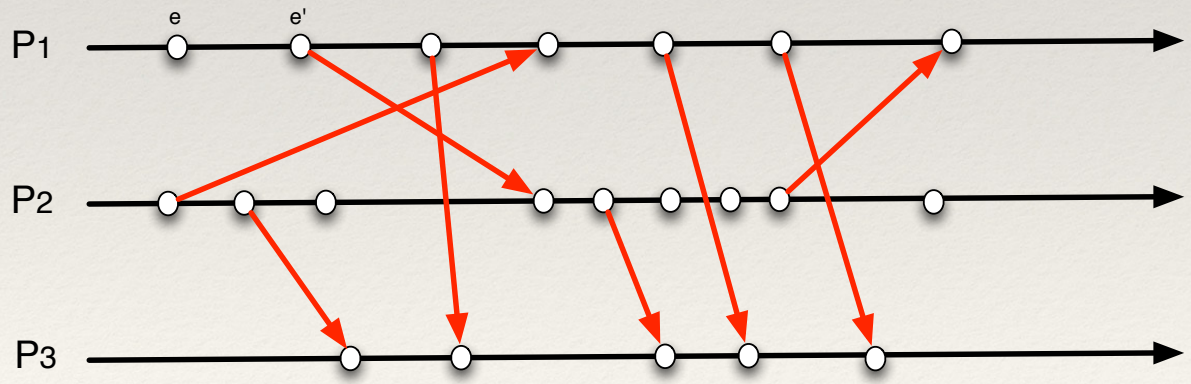
- Théorème: $e \rightarrow e' \Leftrightarrow V(e) <_v V(e')$
- Pour montrer cela on va utiliser deux lemmes
- Le lemme 1 va montrer directement la moitié du théorème (partie $\Rightarrow$).
- Le lemme 2 va être utiliser pour montrer l'autre implication (partie$\Leftarrow$).

Lemme 1

- Lemme 1: e $\rightarrow$ e' alors V(e) $<_v$ V(e')
- Preuve:
  - On montre cela par récurrence sur la longueur d'un chemin causal

Lemme 1- Preuve

- Cas de base: Considérons un chemin causal de longueur 1 de e vers e',
- C'est-à-dire que e précède immédiatement e', si e et e' ont lieu sur le même site i



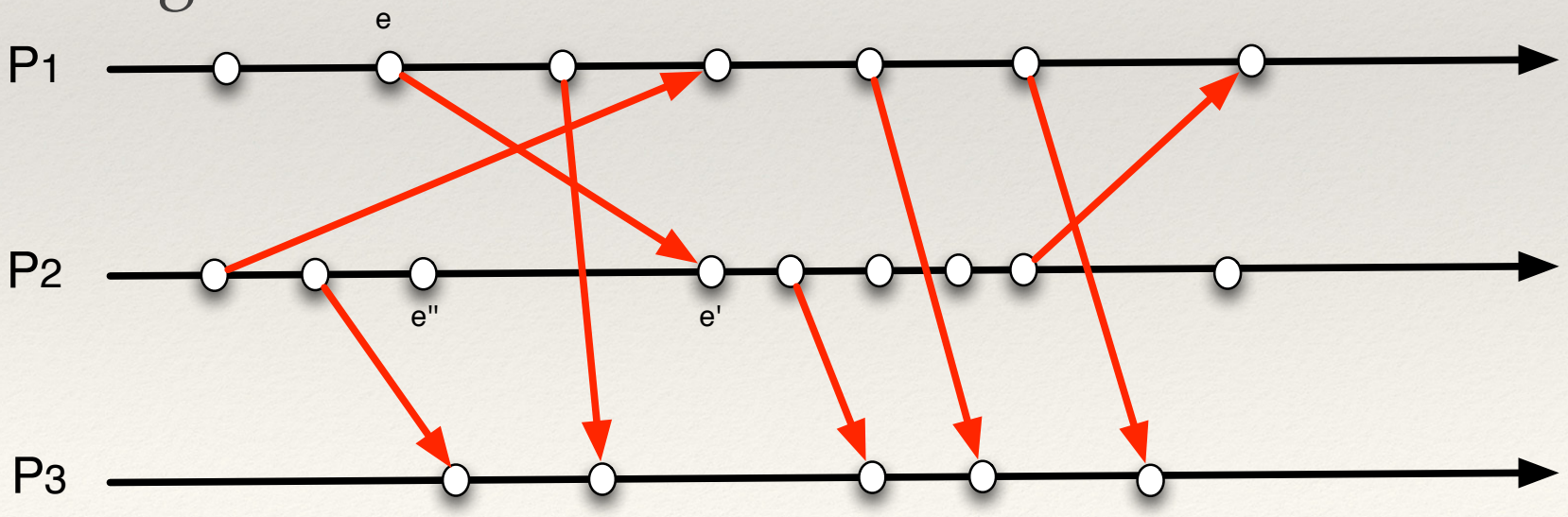
Lemme 1- Preuve

- Cas de base: Considérons un chemin causal de longueur 1 de e vers e', c'est-à-dire que e précède immédiatement e'.
- Si e et e' ont lieu sur le même site i alors les composantes des deux vecteurs sont les mêmes, sauf pour la composante i.

$$L(e) = \begin{pmatrix} e_i \\ e_j \\ e_k \end{pmatrix} \text{ et } L(e') = \begin{pmatrix} e_i + 1 \\ e_j \\ e_k \end{pmatrix} \text{ comme } \begin{matrix} e_i < e_i + 1 \\ e_j = e_j \\ e_k = e_k \end{matrix} \Rightarrow V(e) <_v V(e')$$

Lemme 1- Preuve

- Si e et e' sont sur 2 sites i et j différent le chemin causal est de longueur 1.
- Donc e et e' sont l'envoie et la reception d'un même message M.



Lemme 1- Preuve

- Si e et e' sont sur 2 sites i et j différent le chemin causal est de longueur 1.
- Donc e et e' sont l'envoie et la reception d'un même message M.

$$L(e) = \begin{pmatrix} e_i \\ e_j \\ e_k \end{pmatrix} \text{ et } L(e') = \begin{pmatrix} e''_i \\ e''_j \\ e''_k \end{pmatrix} \text{ on aura } L(e') = \begin{pmatrix} e'_i = max\{e_i, e''_i\} \\ e'_j = e''_j + 1 \\ e'_k = max\{e_k, e''_k\} \end{pmatrix} => V(e) <_v V(e')$$

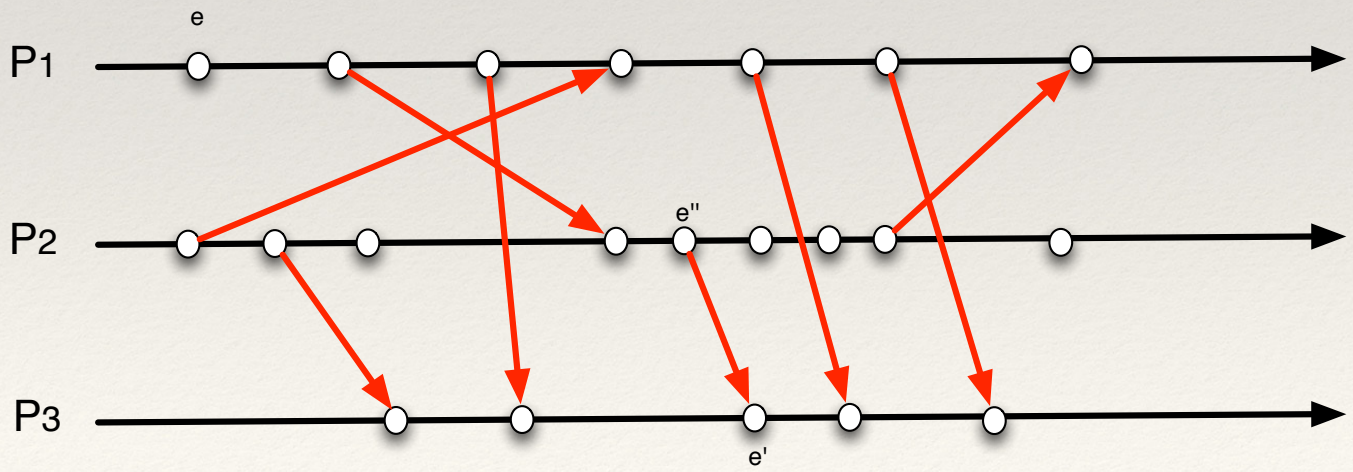
- Le lemme est donc montré pour tout chemin causal de longueur 1

Lemme 1- Preuve

- Hypothèse de récurrence:
- La propriété est vraie pour tout chemin causal de longueur au plus L
- Considérons maintenant un chemin de longueur L +1 entre e et e'.
  - Celui-ci peut-être décomposée en un chemin causal de longueur L de e vers e'', puis un chemin de longueur 1 de e'' vers e'.

Lemme 1- Preuve

- Hypothèse de récurrence:
- La propriété est vraie pour tout chemin causal de longueur au plus L
- Considérons maintenant un chemin de longueur L +1 entre e et e'.
  - Celui-ci peut-être décomposée en un chemin causal de longueur L de e vers e'', puis un chemin de longueur 1 de e'' vers e'.



Lemme 1- Preuve

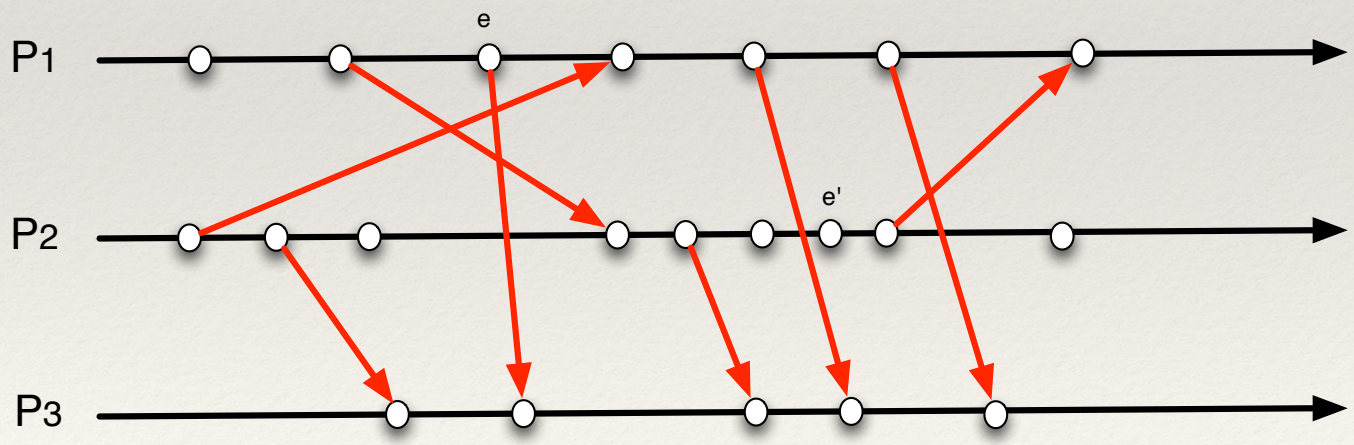
- Hypothèse de récurrence:
- La propriété est vraie pour tout chemin causal de longueur au plus L
- Considérons maintenant un chemin de longueur L +1 entre e et e'.
  - Celui-ci peut-être décomposée en un chemin causal de longueur L de e vers e'', puis un chemin de longueur 1 de e'' vers e'.
- Grâce à la preuve du cas de base et à l'hypothèse de récurrence on obtient:
  - $V(e) <_v V(e'') <_v V(e')$

Lemme 2

- Si e et e’ sont causalement indépendant alors V(e) et V(e’) sont incomparables par $<_v$

Lemme 2 - Preuve

- «Grandes lignes de la preuve»
- soient e un événement du site i et e' un événement du site j sinon e et e' seraient causalement dépendants

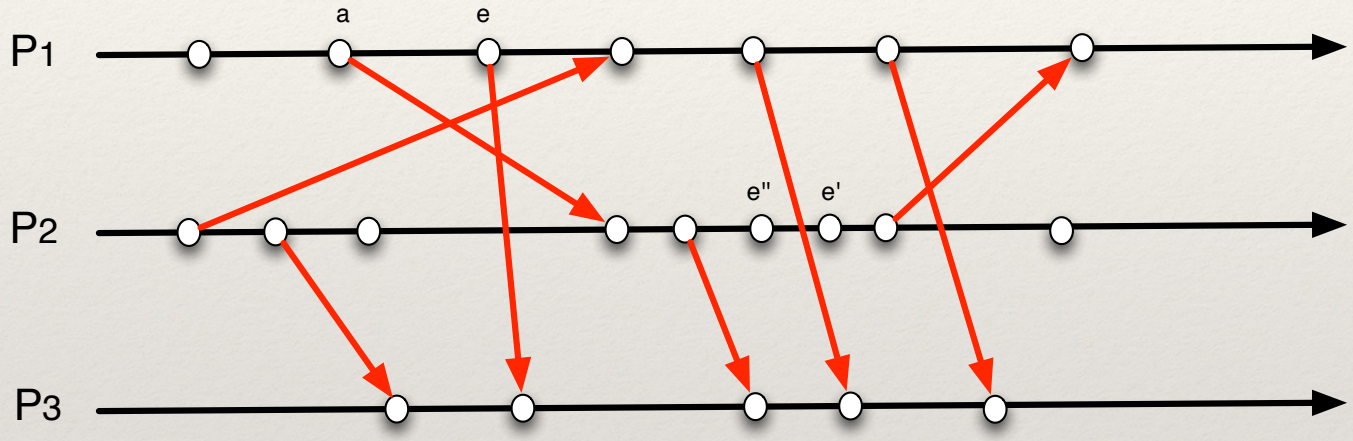


Lemme 2 - Preuve

- Plusieurs cas sont à examiner:

Lemme 2 - Preuve

- Considérons a le dernier événement du site i qui précède causalement e': si a existe.



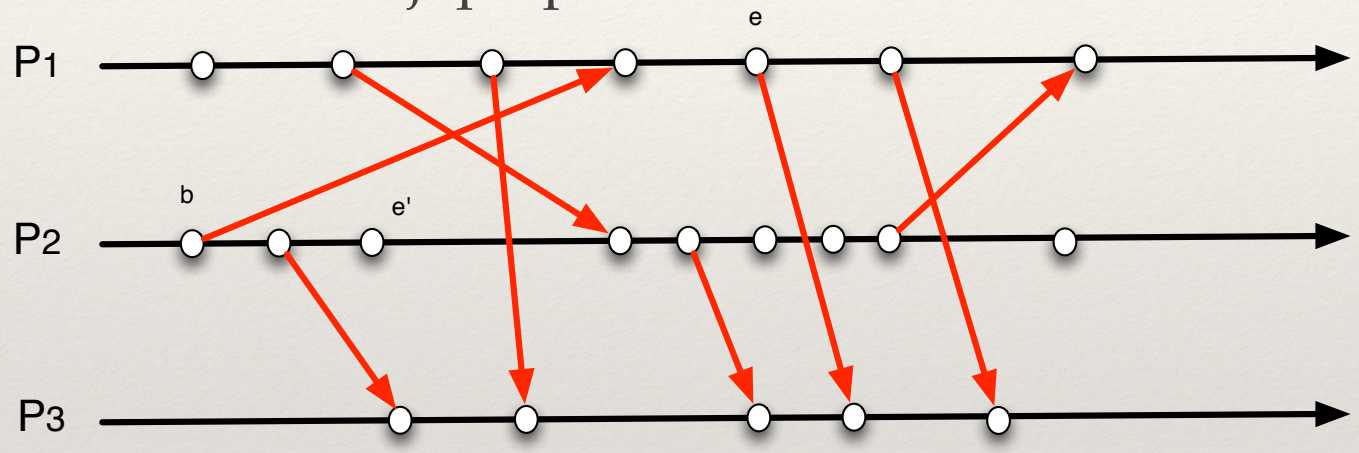
Comme $a \rightarrow e'$ on a $v(a) = \begin{pmatrix} a_i \\ a_j \\ a_k \end{pmatrix}$ et $v(e') = \begin{pmatrix} a_i \\ e'_j > a_j \\ max\{e''_k, a_k\} \end{pmatrix}$ on a $=> V(a) <_v V(e')$

Comme $a \rightarrow e$ on a $v(a) = \begin{pmatrix} a_i \\ a_j \\ a_k \end{pmatrix}$ et $v(e) = \begin{pmatrix} a_i + 1 \\ a_j \\ a_k \end{pmatrix}$ on a $=> V(a) <_v V(e)$

Conclusion $v(e)$ et $v(e')$ sont incomparables

Lemme 2 - Preuve

- De manière symétrique on peut considérer (s'il existe) b le dernier événement sur le site j qui précède causalement e.



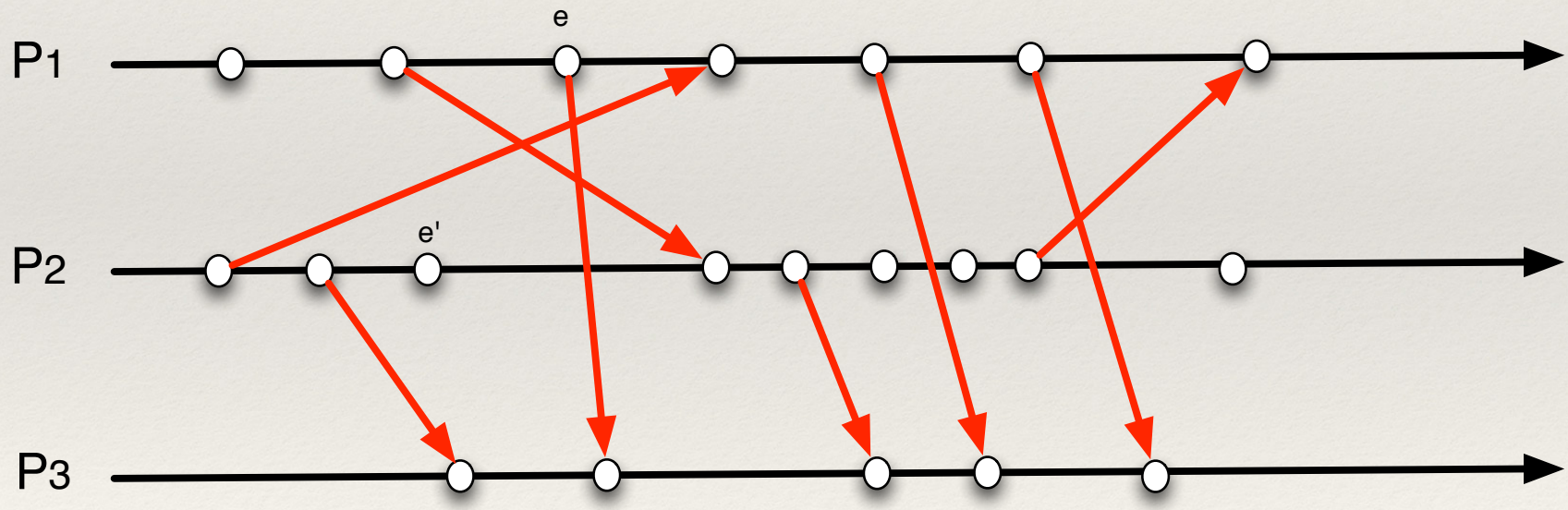
$$\text{Comme } b \to e' \text{ on a } v(b) = \begin{pmatrix} b_i \\ b_j \\ b_k \end{pmatrix} \text{ et } v(e') = \begin{pmatrix} b_i \\ b_j + k \\ b_k \end{pmatrix} \Rightarrow V(b) <_v V(e')$$

$$\text{Comme } b \to e \text{ on a } v(b) = \begin{pmatrix} b_i \\ b_j \\ b_k \end{pmatrix} \text{ et } v(e') = \begin{pmatrix} b_i + k' \\ b_j \\ b_k \end{pmatrix} \Rightarrow V(b) <_v V(e)$$

$$V(e) \text{ et } V(e') \text{ sont incomparables}$$

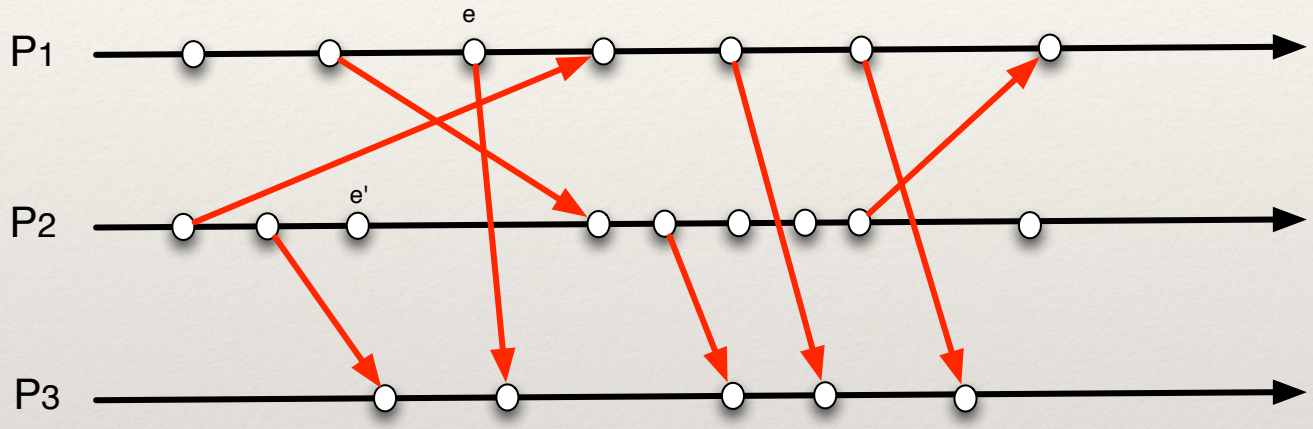
Lemme 2 - Preuve

- Le dernier cas à considérer est le cas où ni a ni b n'existent.



Lemme 2 - Preuve

- Le dernier cas à considérer est le cas où ni a ni b n'existent.



- Dans ce cas, on a V(e')[i]=0 et V(e)[i]>0. mais aussi V(e)[j]=0 et V(e')[j]>0.
- Donc V(e) et V(e') sont incomparables.

Preuve Théorème

- Théorème: $e \rightarrow e' \Leftrightarrow V(e) <_v V(e')$
- Le premier lemme montre directement la moitié (partie⇒) du théorème.

Preuve Théorème

- Montrons l'autre implication ($\Leftarrow$): $V(e) <_v V(e') \Rightarrow e \rightarrow e'$
- Considérons e et e' tels que $V(e) <_v V(e')$.
- Le lemme 2 nous dit que e et e'sont causalement dépendant sinon on aurait pas $V(e) <_v V(e')$
- Or si on avait $e' \rightarrow e$, grâce au lemme 1 on aurait $V(e') <_v V(e)$ ce qui contre dirait $V(e) <_v V(e')$.
- Ainsi on a bien $e \rightarrow e'$.