

30-769

Sistemas Distribuídos

MSc. Fernando Schubert

SINCRONIZAÇÃO EM SISTEMAS DISTRIBUÍDOS

- Além da comunicação, da nomeação temos também a sincronização e coordenação de processos em SDs.
- A sincronização entre processos é importante em diversas situações:
 - Para que vários processos não acessem ao mesmo tempo um recurso compartilhado
 - Processos devem cooperar para garantir um acesso temporário exclusivo exemplo: impressora, arquivos
 - Para que os processos tenham a mesma visão da ordenação de eventos
 - Se uma mensagem $m1$ do processo P foi enviada antes ou depois de $m2$ do processo Q

SINCRONIZAÇÃO EM SISTEMAS DISTRIBUÍDOS

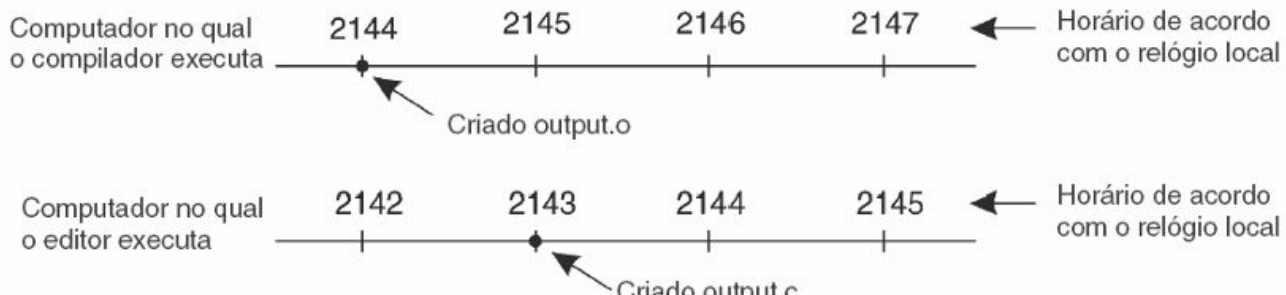
- Sincronização e coordenação são relacionados:
 - Na sincronização de processos é garantido que um processo espera pela finalização de outro processo
 - Na sincronização de dados é garantida que a cópia do dado é consistente
 - Na coordenação o objetivo é gerenciar as interações e dependências entre as atividades de um sistema distribuído

SINCRONIZAÇÃO EM SISTEMAS DISTRIBUÍDOS

- Problema – sincronização em sistemas distribuídos é frequentemente mais difícil comparada com a sincronização em sistemas monoprocessados ou multiprocessados.
- Objetivo: permitir que os processos cooperem entre si para garantir que cada um acesse individualmente e, por um certo período de tempo, o recurso objeto do compartilhamento.

SINCRONIZAÇÃO EM SISTEMAS DISTRIBUÍDOS

- Em um sistema centralizado, o tempo não é ambíguo:
 - Quando um processo quer saber a hora, ele faz uma chamada ao sistema, e o núcleo responde
 - Se A pergunta a hora, e um pouco mais tarde B perguntar a hora, o valor obtido por B será maior do que o obtido por A
 - Em um SD, conseguir acordo nos horários não é trivial
- Exemplo: programa make:



RELÓGIOS FÍSICOS

- Computadores tem circuitos para monitorar a passagem de tempo
- Um temporizador é um cristal de quartzo lapidado, que quando mantidos sob tensão, oscilam em uma frequência bem definida
- Associado a cada cristal
 - Contador
 - Registrador de Retenção

RELÓGIOS FÍSICOS

- Funcionamento do Temporizador:
 - Cada oscilação do cristal reduz uma unidade do contador
 - Quando o contador chega a zero é gerada uma interrupção e o contador é recarregado pelo registrador de retenção
 - É possível programar o temporizador para gerar uma interrupção em uma determinada frequência
 - Cada interrupção é denominada ciclo do relógio
 - A cada ciclo de relógio é somada uma unidade a hora armazenada relógio é mantido atualizado

RELÓGIOS FÍSICOS

- Com um único computador e um único relógio, não há problema se este estiver um pouco defasado
 - Todos os processos da máquina utilizam o mesmo relógio
 - O que importa são os horários relativos
- Com múltiplas CPUs é impossível garantir que todos os cristais funcionem exatamente à mesma frequência
 - A diferença entre os horários é chamada defasagem de relógio
 - Como sincronizar os relógios uns com os outros e com o tempo real?

RELÓGIOS FÍSICOS

- Como o tempo é realmente medido?
 - Até invenção dos relógios mecânicos (século XVII):
 - Tempo medido com auxílio dos astros
 - Sol nasce no horizonte a leste
 - Sol alcança uma altura máxima no céu
 - Sol se põe no horizonte a oeste
 - O ponto mais alto do Sol é chamado trânsito solar
 - Intervalo entre 2 trânsitos solares consecutivos do Sol é um dia solar
- Sendo 24 horas em um dia e cada hora com 3600 segundos
 - Um segundo solar é exatamente $1/86.400$ de um dia solar

RELÓGIOS FÍSICOS

- Em meados de 1940 foi estabelecido que a rotação da Terra não é constante:
 - Devido à desaceleração gradativa resultante de marés e do atrito com atmosfera
 - Acredita-se que há 300 milhões de anos o ano tinha 400 dias!
 - A Terra gira mais devagar, mas não alterou sua órbita
 - Logo, o tamanho do ano aparenta ser o mesmo, mas os dias ficaram mais longos!
 - Também existem variações de curto prazo no comprimento dos dias

RELÓGIOS FÍSICOS

- Em 1948 foi inventado o relógio atômico Tal fato tornou possível:
 - Medir o tempo com maior precisão
 - Medir o tempo independentemente das condições do globo terrestre e da atmosfera]
 - O tempo é calculado através de contagens de transições do átomo de césio 133
 - 1 segundo atômico = 9.192.631.770 transições
 - 1 segundo atômico = 1 segundo solar médio no ano em que foi lançado

RELÓGIOS FÍSICOS

- Vários laboratórios ao redor do mundo possuem relógios de césio 133
- Cada laboratório informa ao BIH (Bureau International de l'Heure) quantas vezes o seu relógio pulsou
- O BIH calcula a média destes valores e produz a hora atômica internacional (TAI)
- A TAI é um número médio de ciclos dos relógios de césio 133 divididos por 9.192.631.770
- A TAI é estável, mas apresenta um problema. Qual?

RELÓGIOS FÍSICOS

- Problema:
 - Dias solares estão aumentando 86.400 segundos
 - TAI equivalem a aproximadamente 3ms a menos que um dia solar médio atual
 - Usar a TAI para medir o tempo significaria que no decorrer dos anos o meio-dia aconteceria sempre mais cedo até que um dia aconteceria de madrugada!
- Solução:
 - O BIH introduz segundos extras sempre que a discrepância entre a TAI e a hora solar alcançar mais de 800ms

RELÓGIOS FÍSICOS

- Com base na TAI com correções de segundos foi estabelecido o sistema UTC (Universal Coordinated Time)
- UTC é a base de toda a moderna medição de tempo
- A maioria das empresas geradoras de energia elétrica sincroniza a temporização de seus relógios com o UTC
- Quando o BIH anuncia segundos extras, essas empresas aumentam a frequência dos seus relógios a fim de alinhar todos os seus relógios em sua área de distribuição
- 30 segundos extras foram introduzidos no UTC até hoje
- O NIST (National Institute of Standard Time) fornece UTC através da operação de uma estação de rádio de ondas curtas cujo prefixo é WWV
- Além disso vários satélites fornecem UTC
- Segundo bissexto

RELÓGIOS FÍSICOS - SINCRONIZAÇÃO

- Se uma das máquinas possui receptor UTC
 - O objetivo é manter todas as outras máquinas sincronizadas com ela
- Se nenhuma possui receptor UTC
 - Cada máquina cuida de seu próprio horário
 - O objetivo passa a ser manter o horário de todas máquinas o mais próximo possível
- Foram propostos vários algoritmos de sincronização de relógios, todos seguindo as mesmas ideias básicas

RELÓGIOS FÍSICOS - SINCRONIZAÇÃO

- Algoritmo:
 - Todas máquinas possuem um temporizador que gera uma interrupção H vezes por segundo
 - Quando o temporizador esgota o tempo fixado o manipulador de interrupção adiciona 1 unidade ao clock C conhecido por todas máquinas do sistema
- Considere que quando o tempo UTC for t o valor de clock de uma máquina p é dado por $C_p(t)$
- Clock skew (defasagem do relógio) denota a magnitude de diferença entre as frequências de dois relógios

RELÓGIOS FÍSICOS - SINCRONIZAÇÃO

- Em condições ideais:
 - $C_p(t) = t$, para qualquer valor de p e t
 - $C'_p(t) = dC/dt = 1$
 - Condições impossíveis de se obter na prática!
 - O erro relativo aproximado dos temporizadores é 10^{-6}

RELÓGIOS FÍSICOS - SINCRONIZAÇÃO

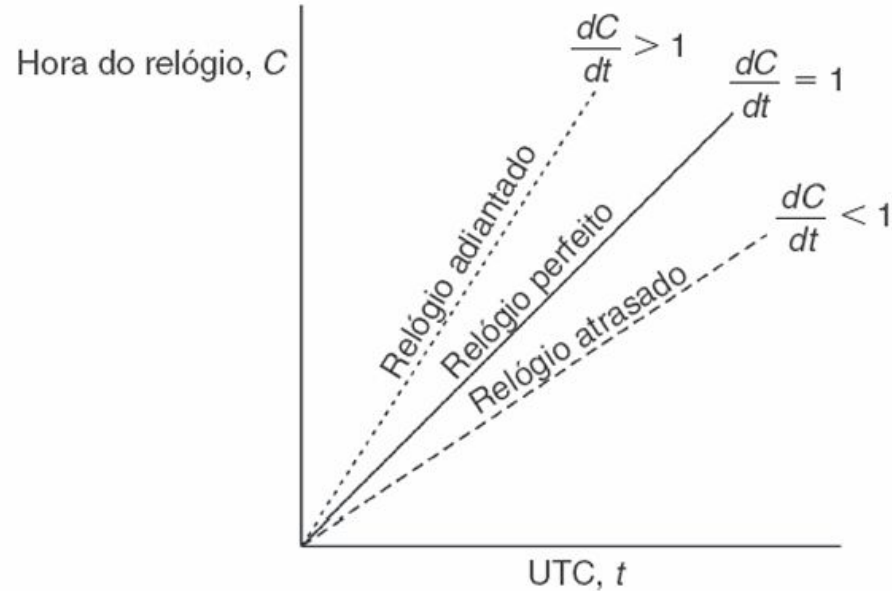


Figura 6.5 Relação entre a hora do relógio e a hora UTC quando as taxas de ciclos de relógios são diferentes.

RELÓGIOS FÍSICOS - SINCRONIZAÇÃO

- Proposto por Cristian(1989), baseia-se em clientes consultarem um servidor de tempo
 - Funcionamento:
 - Cada máquina envia uma mensagem para o servidor de tempo (máquina com receptor WWV ou relógio de precisão), perguntando pelo tempo corrente
 - Servidor de tempo responde o mais rápido possível, com uma mensagem contendo o tempo corrente *C-UTC*
 - Quando o transmissor obtém uma resposta, ajusta seu clock.

RELÓGIOS FÍSICOS - SINCRONIZAÇÃO

- Proposto por Cristian(1989), baseia-se em clientes consultarem um servidor de tempo
 - Funcionamento:
 - Cada máquina envia uma mensagem para o servidor de tempo (máquina com receptor WWV ou relógio de precisão), perguntando pelo tempo corrente
 - Servidor de tempo responde o mais rápido possível, com uma mensagem contendo o tempo corrente *C-UTC*
 - Quando o transmissor obtém uma resposta, ajusta seu clock.
 - O problema é que quando se contata o servidor, os atrasos nas mensagens farão que a hora fornecida esteja desatualizada!

RELÓGIOS FÍSICOS - SINCRONIZAÇÃO

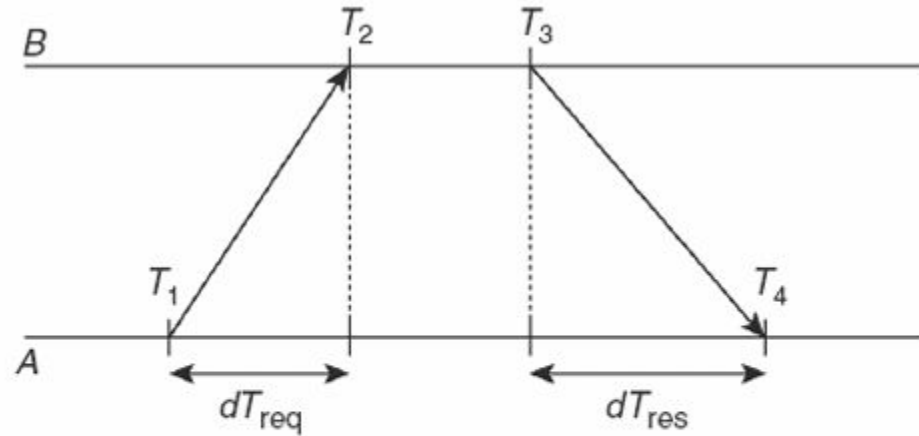


Figura 6.6 Obtenção da hora corrente por meio de um servidor de tempo.

RELÓGIOS FÍSICOS - SINCRONIZAÇÃO

- Após calcular o atraso no envio das mensagens, o relógio de A deve ser sincronizado com o valor enviado por B
- Se o relógio de A estiver adiantado o tempo não pode retroceder
- Uma solução é ajustar o tempo gradativamente
 - Suponha que o relógio de A esteja programado para gerar cem interrupções por segundo
 - Cada interrupção soma 10ms ao relógio
 - Atrasa somando a cada interrupção apenas 9ms ao invés de 10ms
 - O mesmo pode ser empregado para adiantar o relógio somar 11ms ao invés de 10ms

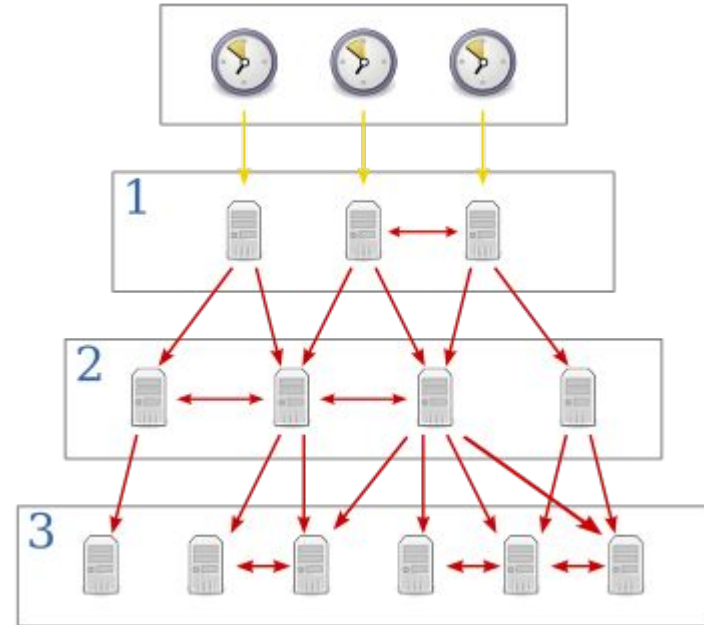
RELÓGIOS FÍSICOS - SINCRONIZAÇÃO

- Protocolo de Tempo de Rede – NTP (Network Time Protocol)
 - Neste protocolo, o relógio é ajustado entre pares de servidores
 - A consulta B e B consulta A
 - Qual horário prevalece?
 - Qual relógio possui melhor precisão?
- Cada computador possui seu “estrato”
- Um servidor que possui um receptor WWV ou um relógio atômico é conhecido como servidor de estrato 1

RELÓGIOS FÍSICOS - SINCRONIZAÇÃO

Hierarquia de servidores organizada em estratos (strata)

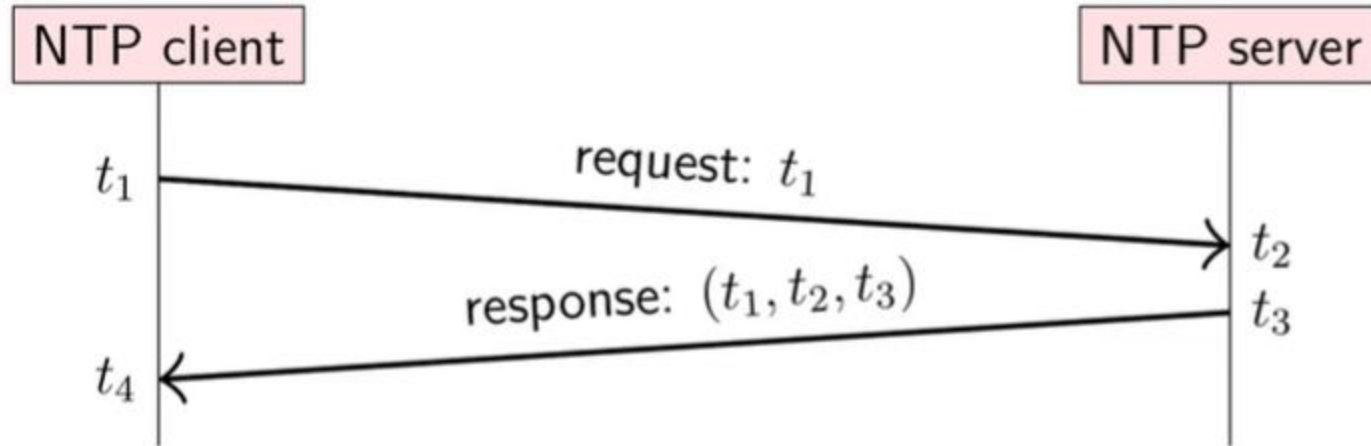
- Stratum 0: servidor é sincronizado diretamente de um relógio atômico ou receptor GPS
- Stratum 1: servidor é sincronizado com o servidor de stratum 0
- Stratum 2: servidor é sincronizado com servidor de stratum 1.



RELÓGIOS FÍSICOS - SINCRONIZAÇÃO

- Protocolo de Tempo de Rede – NTP
 - Quando A contata B só ajustará seu horário se seu estrato for maior do que B
 - Prevalece o horário do servidor com menor estrato
- Para aumentar a precisão no cálculo do atraso de envio entre os servidores, o NTP também se baseia em médias já calculadas anteriormente.
- O cliente efetua múltiplas requisições com o intuito de reduzir o erro médio
- O NTP alcança uma precisão em âmbito mundial na faixa de 1 a 50 ms

RELÓGIOS FÍSICOS - SINCRONIZAÇÃO



Delay da rede $dT = (t_4 - t_1) - (t_3 - t_2)$

Hora no servidor em t_4 $tS = t_3 + dT/2$

Clock skew: $CS = t_3 + tS/2 - t_4$

RELÓGIOS FÍSICOS - SINCRONIZAÇÃO

- Ajuste do relógio no NTP:
 - Clock skew entre 0 e 125ms: acelerar ou lentamente reduzir o relógio do computador até convergência com o relógio remoto (slew clock skew)
 - Clock skew entre 125ms e 1000s: ajusta o relógio automaticamente para o tempo remoto (step clock skew)
 - Clock skew é maior que 1000s: erro - não faz nada, espera ação humana.

```
// BAD:  
long startTime = System.currentTimeMillis();  
doSomething();  
long endTime = System.currentTimeMillis();  
long elapsedMillis = endTime - startTime;  
// elapsedMillis may be negative!
```

RELÓGIOS FÍSICOS - SINCRONIZAÇÃO

```
// BAD:  
long startTime = System.currentTimeMillis();  
doSomething();  
long endTime = System.currentTimeMillis();  
long elapsedMillis = endTime - startTime;  
// elapsedMillis may be negative!
```

```
// GOOD:  
long startTime = System.nanoTime();  
doSomething();  
long endTime = System.nanoTime();  
long elapsedNanos = endTime - startTime;  
// elapsedNanos is always >= 0
```

RELÓGIOS FÍSICOS - SINCRONIZAÇÃO

- Algoritmo de Berkeley:
 - No NTP o servidor de tempo é passivo
 - No algoritmo Berkeley o servidor consulta todas as máquinas de tempos em tempos, obtendo o horário de cada máquina
 - Gera uma média de todas as horas, e informa a todos os computadores o deslocamento de tempo a ser feito
 - Este método é adequado quando nenhuma das máquinas possui um receptor WWV

RELÓGIOS FÍSICOS - SINCRONIZAÇÃO

- Algoritmo de Berkeley:

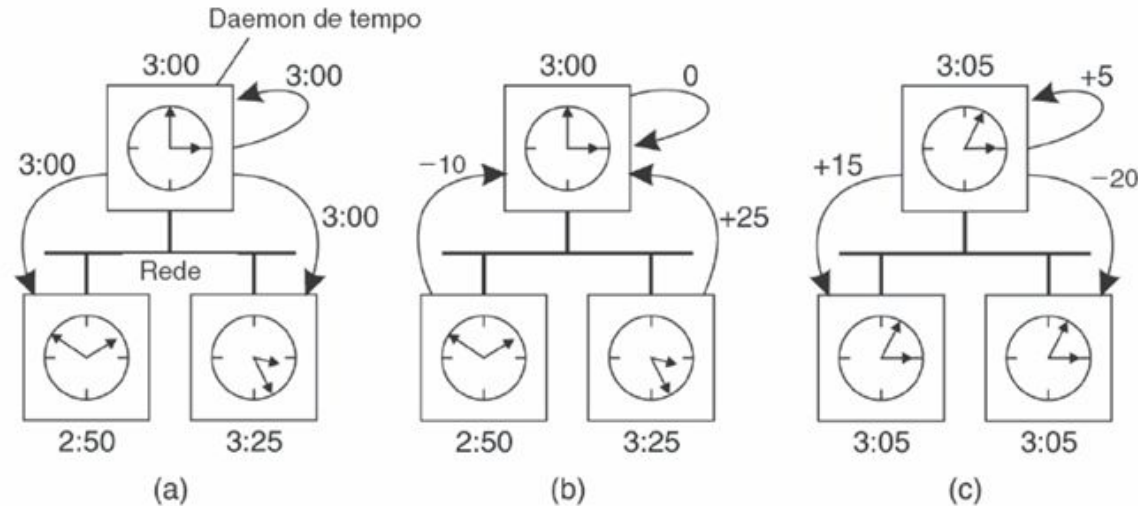
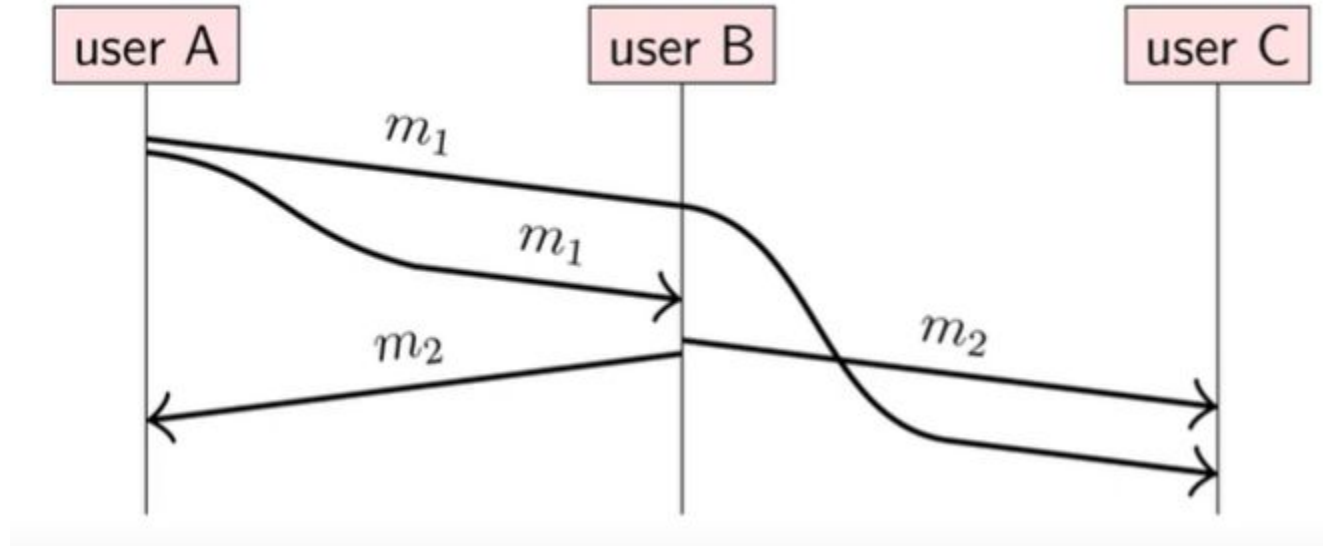


Figura 6.7 (a) O daemon de tempo pergunta a todas as outras máquinas os valores marcados por seus relógios. (b) As máquinas respondem. (c) O daemon de tempo informa a todas como devem ajustar seus relógios.

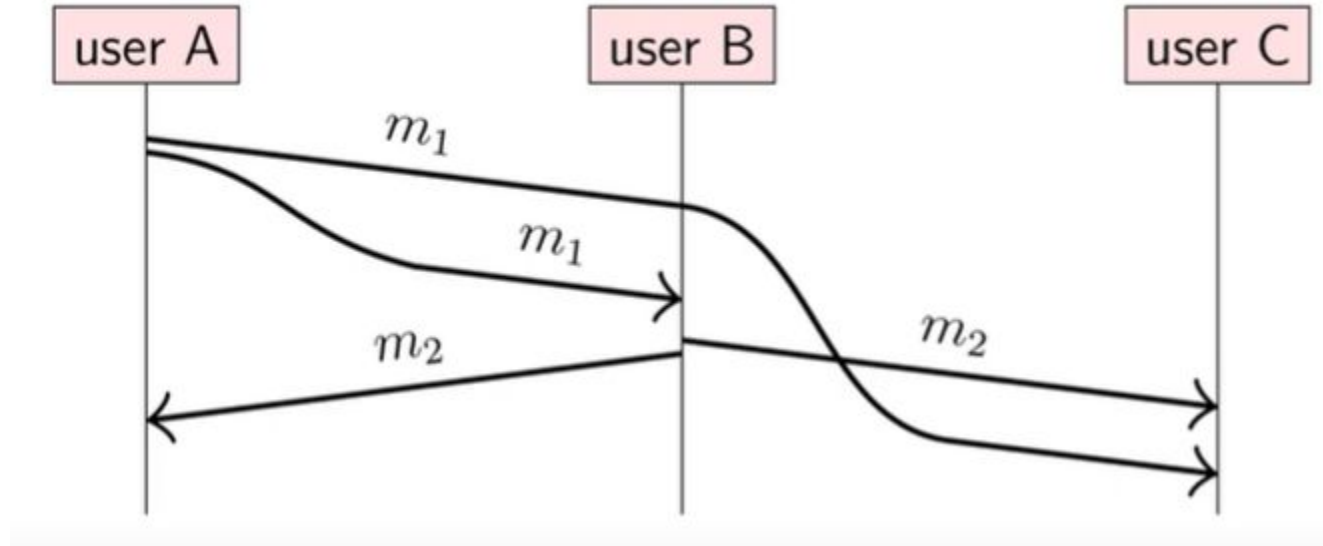
ORDENAÇÃO DE EVENTOS



m_1 : a terra é plana!

m_2 : não é não!

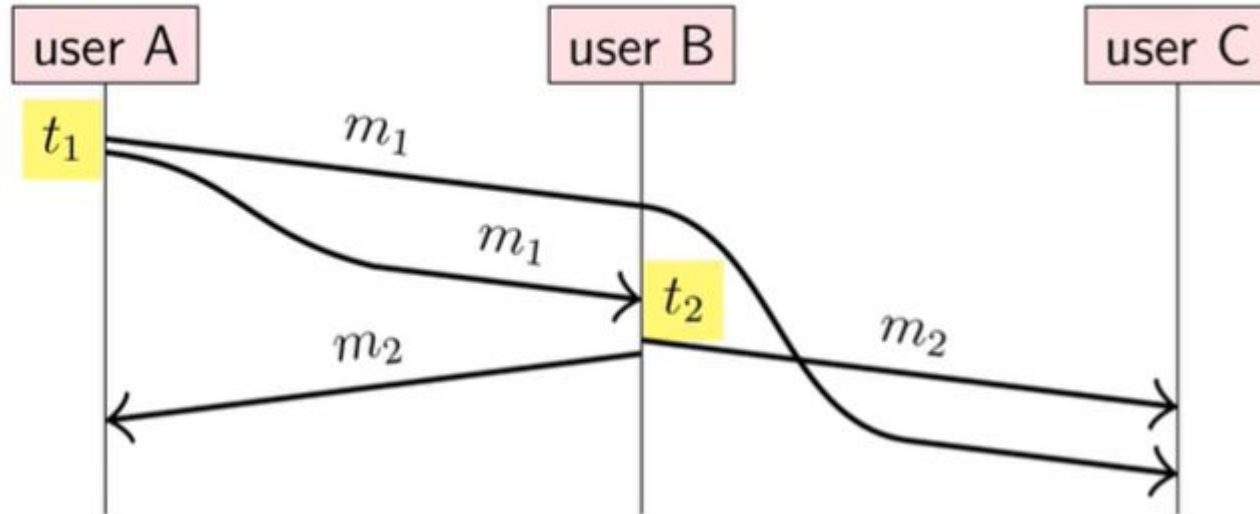
ORDENAÇÃO DE EVENTOS



m_1 : a terra é plana!

m_2 : não é não!

ORDENAÇÃO DE EVENTOS



Problema: mesmo com sincronização de relógios com, por exemplo NTP, não podemos garantir a correta ordenação dos eventos e t_2 pode ser menor que t_1

ORDENAÇÃO DE EVENTOS

Problema: mesmo com sincronização de relógios com, por exemplo NTP, não podemos garantir a correta ordenação dos eventos e t_2 pode ser menor que t_1

RELÓGIOS LÓGICOS

- Até o momento foi considerada a sincronização de relógios como naturalmente relacionada com a hora real
- Entretanto pode ser suficiente que cada nó concorde com a hora corrente sem que esta seja igual a hora real
- Relógios Lógicos
 - Em 1978 Lamport mostrou que, embora a sincronização de relógios seja possível, não precisa ser absoluta:
 - Se dois processos não interagem entre si, não é necessário que seus relógios sejam sincronizados
 - Não é necessário que todos os processos concordem com a hora exata mas com a ordem em que os eventos ocorrem

RELÓGIOS LÓGICOS DE LAMPORT

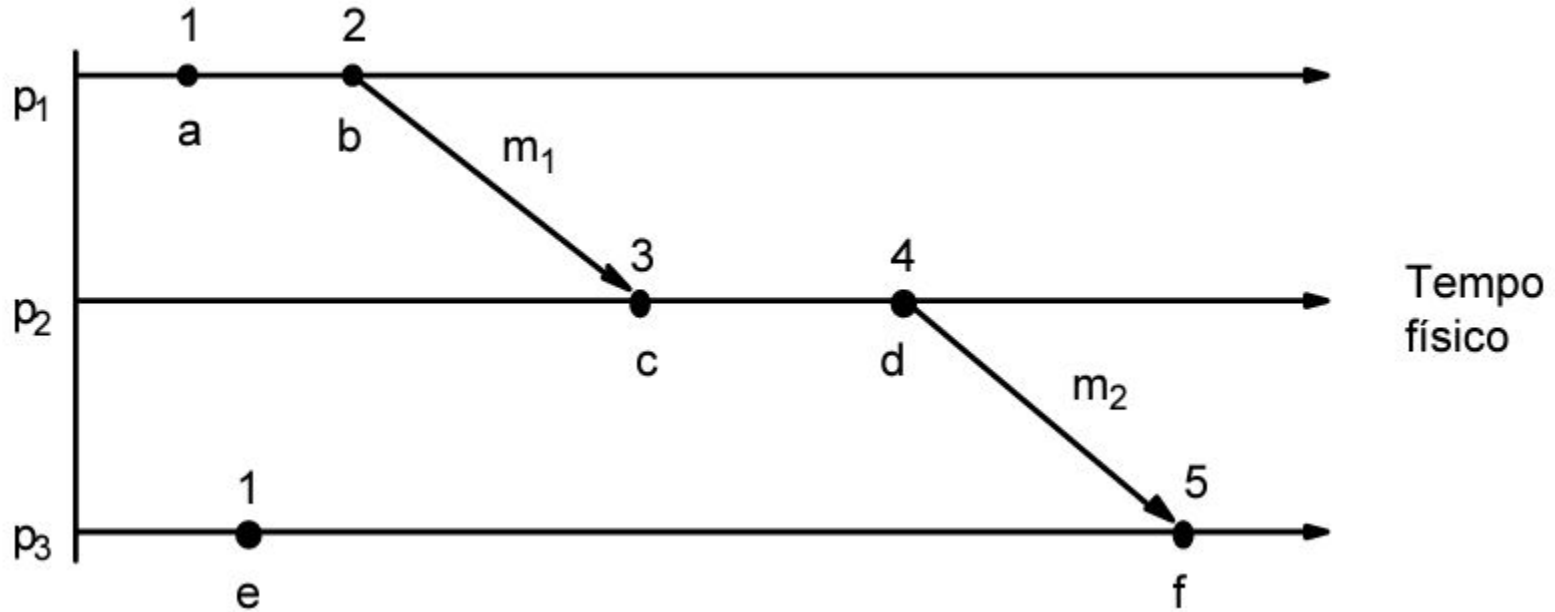
- Definida a relação acontece antes:
 - Se a e b são eventos do mesmo processo:
 - Se a acontece antes de b : $a \rightarrow b$
 - Sendo a o envio da mensagem e b o recebimento da mensagem então: $a \rightarrow b$
 - Se $a \rightarrow b$ e $b \rightarrow c$, então $a \rightarrow c$ (propriedade transitiva)
 - Se x e y acontecem em processos diferentes e tanto $x \rightarrow y$ quanto $y \rightarrow x$ são falsas, os processos x e y são ditos concorrentes

RELÓGIOS LÓGICOS DE LAMPORT

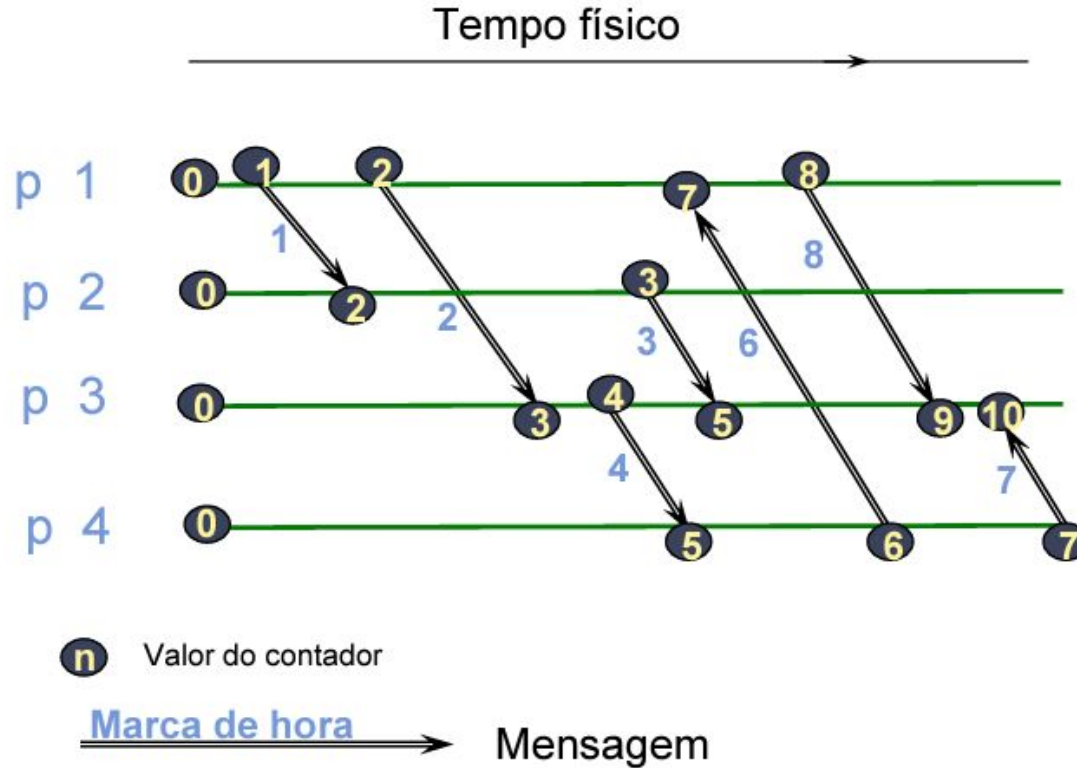
- Cada processo tem um contador (relógio lógico)
- Inicialmente o relógio lógico tem valor 0
- Processo incrementa seu contador quando um evento de envio ou processamento é realizado
- Contador é atribuído a um evento como seu marcador de hora
- Um evento de envio de mensagem carrega seu marcador de hora
- Em um evento de recebimento de mensagem o contador é atualizado por:

$max(contador\ local, marcador\ de\ hora\ da\ mensagem) + 1$

RELÓGIOS LÓGICOS DE LAMPORT



RELÓGIOS LÓGICOS DE LAMPORT



RELÓGIOS LÓGICOS DE LAMPORT

○ Arquitetura

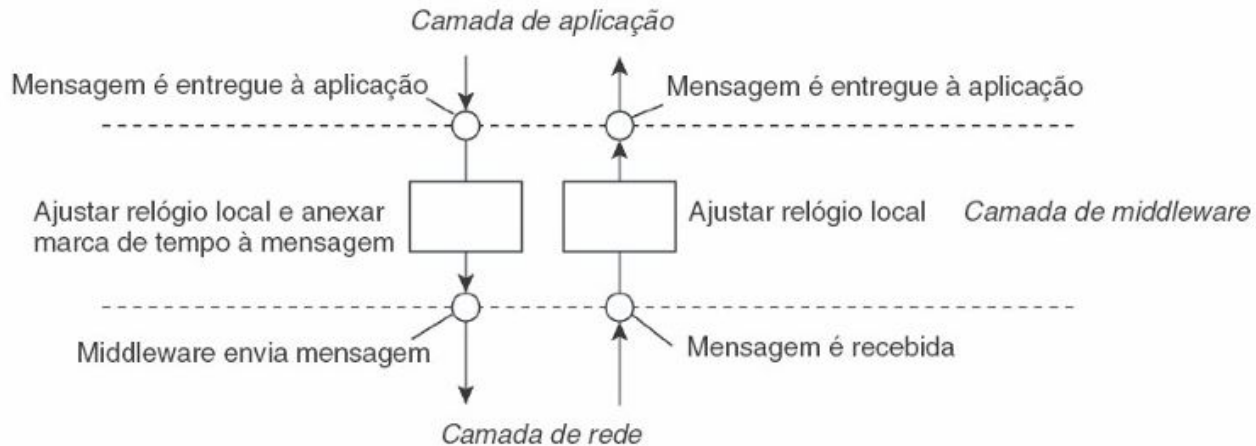
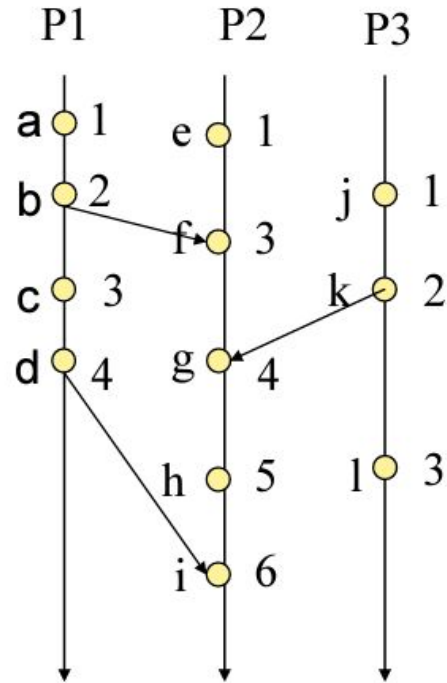


Figura 6.10 Posicionamento de relógios lógicos de Lamport em sistemas distribuídos.

RELÓGIOS LÓGICOS DE LAMPORT



Assumindo que o relógio lógico de cada processo começa em 0

RELÓGIOS LÓGICOS DE LAMPORT

- A partir do diagrama de tempo anterior, o que se pode falar da relação entre os seguintes eventos?

a e b: $a \rightarrow b$

b e f: $b \rightarrow f$

e e k: concorrentes

c e h: concorrentes

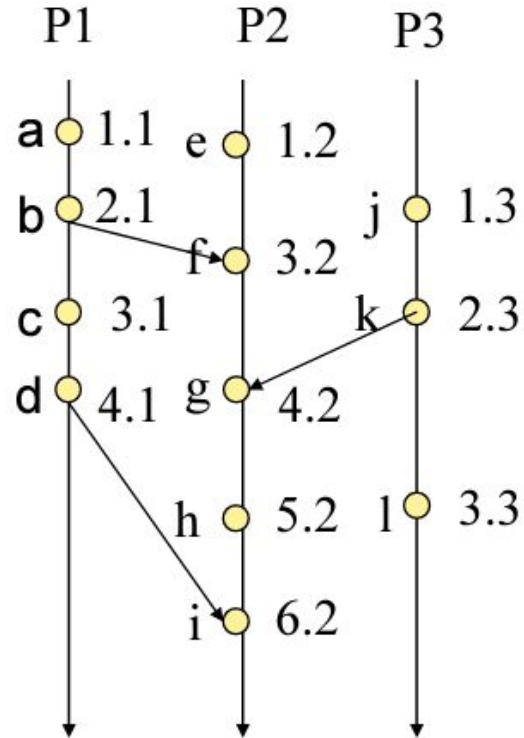
k e h: $k \rightarrow h$

RELÓGIOS LÓGICOS DE LAMPORT

- **Eventos totalmente ordenados**

- Um marcador de hora igual a 1 está associada aos eventos a, e, j nos processos $P1, P2, P3$ respectivamente.
- Um marcador de hora igual a 2 está associada aos eventos b, k nos processos $P1, P3$ respectivamente.
- As marcadores de horas são os mesmos, mas os eventos são distintos.
- Seria bom criar uma ordem total dos eventos i.e. para qualquer evento a, b poder dizer que $a \rightarrow b$ ou $b \rightarrow a$
- Cria uma ordem total atribuindo um número de processo ao evento.
- P_i atribui o marcador de hora $C_i(e).i$ para o evento e
- Então dizemos que $C_i(a).i$ acontece antes de $C_j(b).j$ se e somente se:
 - $C_i(a) < C_j(b)$; ou
 - $C_i(a) = C_j(b)$ e $i < j$

RELÓGIOS LÓGICOS DE LAMPORT



RELÓGIOS LÓGICOS DE LAMPORT

- **Exemplo – Multicast totalmente ordenado**
 - Considere que um banco de dados foi replicado
 - O problema:
 - As operações de atualização devem ser executadas na mesma ordem em cada cópia
 - Em geral, situações como estas exigem um ordenação total de envio
 - Os relógios lógicos de Lamport podem ser usados para implementar envio totalmente ordenado de forma distribuída

RELÓGIOS LÓGICOS DE LAMPORT

- Suponha que um cliente em San Francisco quer adicionar \$100 à sua conta, que contém atualmente \$1000
- Ao mesmo tempo, um funcionário de banco em NY inicia uma atualização pela qual a conta do cliente recebe um acréscimo de 1% de juros
- Ambas as atualizações devem ser efetuadas em duas cópias do banco de dados
- Operação de atualização do cliente é realizada em San Francisco antes da atualização de juros de 1%.
- Por outro lado, a cópia da conta em NY é primeiro atualizada com os juros de 1% e depois com o depósito de \$ 100.

RELÓGIOS LÓGICOS DE LAMPORT

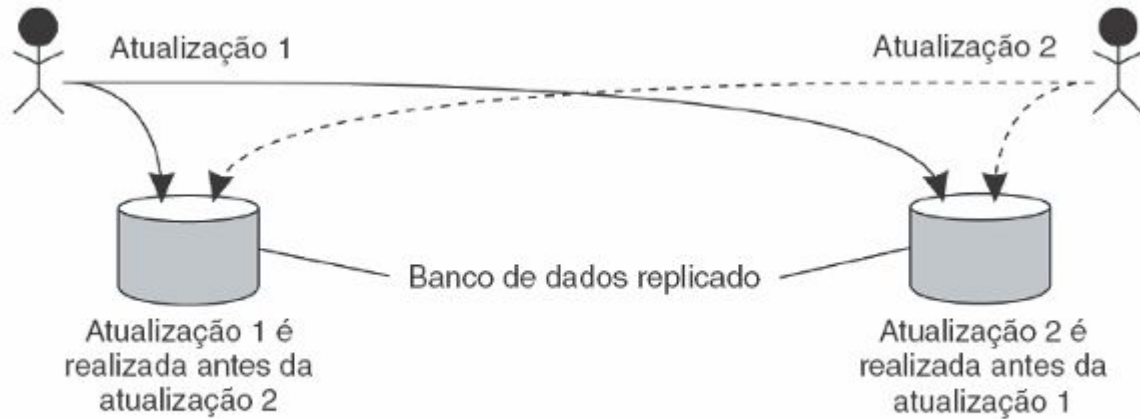


Figura 6.11 Atualização de banco de dados replicado que o deixa em estado inconsistente.

RELÓGIOS LÓGICOS DE LAMPORT

- Consequentemente:
 - No banco de dados de San Francisco será gravado um montante total de \$1111
 - Já, no banco de dados de NY será gravado o total de \$1110
 - Para manter a consistência, as duas operações de atualização deveriam ter sido realizadas na mesma ordem em ambas as cópias
 - Neste caso, deve ser usado um multicast totalmente ordenado
- Mensagens de atualização recebem o marcador de hora com a hora lógica que a atualização foi solicitada pelo cliente
- A mensagem de atualização é enviada para todos os processos que devem realizar a atualização (inclusive para eles mesmos)

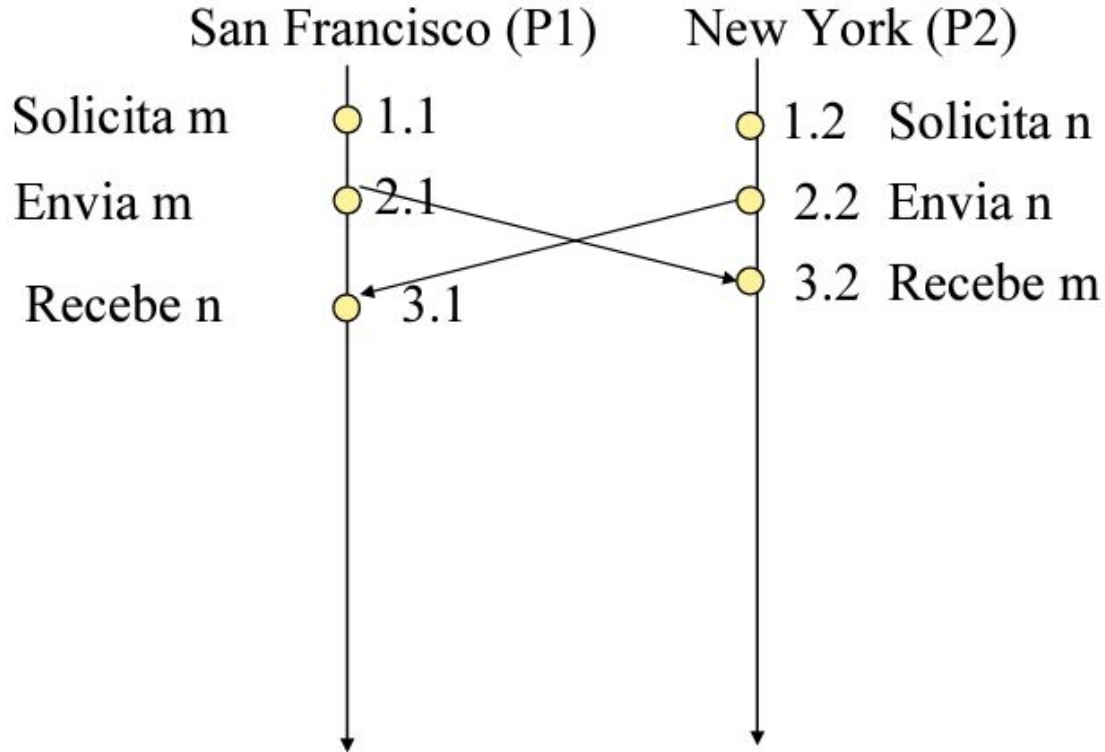
RELÓGIOS LÓGICOS DE LAMPORT

- Quando uma mensagem de atualização é recebida:
 - Ela é colocada em uma fila local que é ordenada de acordo com o seu marcador de hora
 - O processo P_i envia uma mensagem de reconhecimento de uma mensagem da fila local somente quando:
 - Não há pedido de atualização em P_i ou
 - Identificador do processo P_i é maior ou igual que identificador do processo P_j da mensagem ou
 - A atualização solicitada em P_i já foi realizada
- A atualização contida em uma mensagem vai ser executada em um processo quando:
 - A mensagem que contém a atualização é a cabeça da fila
 - A mensagem foi reconhecida por todos os processos

RELÓGIOS LÓGICOS DE LAMPORT

- Considere que m corresponde à atualização “Adicionar \$100” e n corresponde à atualização “Adicionar juros de 1%”.
- Quando uma mensagem de atualização for enviada (e.g., m, n) a mensagem vai incluir o marcador de hora gerado quando a atualização foi solicitada pelo cliente.

RELÓGIOS LÓGICOS DE LAMPORT



RELÓGIOS LÓGICOS DE LAMPORT

- O envio da mensagem m consiste no envio da operação de atualização m junto com o marcador de hora em que esta atualização foi solicitada que é 1.1
- O envio da mensagem n consiste no envio da operação de atualização n junto com o marcador de hora em que esta atualização foi solicitada que é 1.2
- Mensagens são enviadas para todos os processos do grupo incluindo eles mesmos
 - Assuma que a mensagem enviada para um processo para si mesmo é recebida pelo processo quase que imediatamente.
 - Para os outros processos, pode ter um atraso no recebimento.

RELÓGIOS LÓGICOS DE LAMPORT

- Neste ponto, temos as seguintes filas locais para cada processo:
 - P1: (m,1.1), (n,1.2)
 - P2: (m,1.1), (n,1.2)
- P1 vai enviar uma mensagem de reconhecimento de (m,1.1) mas não de (n,1.2)
 - Porque? Identificador de P1 que é 1 é igual ao identificador do processo em (m,1.1), mas é menor que o identificador do processo em (n,1.2)
- P2 vai enviar uma mensagem de reconhecimento para todos de (m,1.1) e (n,1.2)
 - Porque? Identificador de P2 que é 2 é maior que o identificador do processo em (m,1.1), e igual ao identificador do processo em (n,1.2)

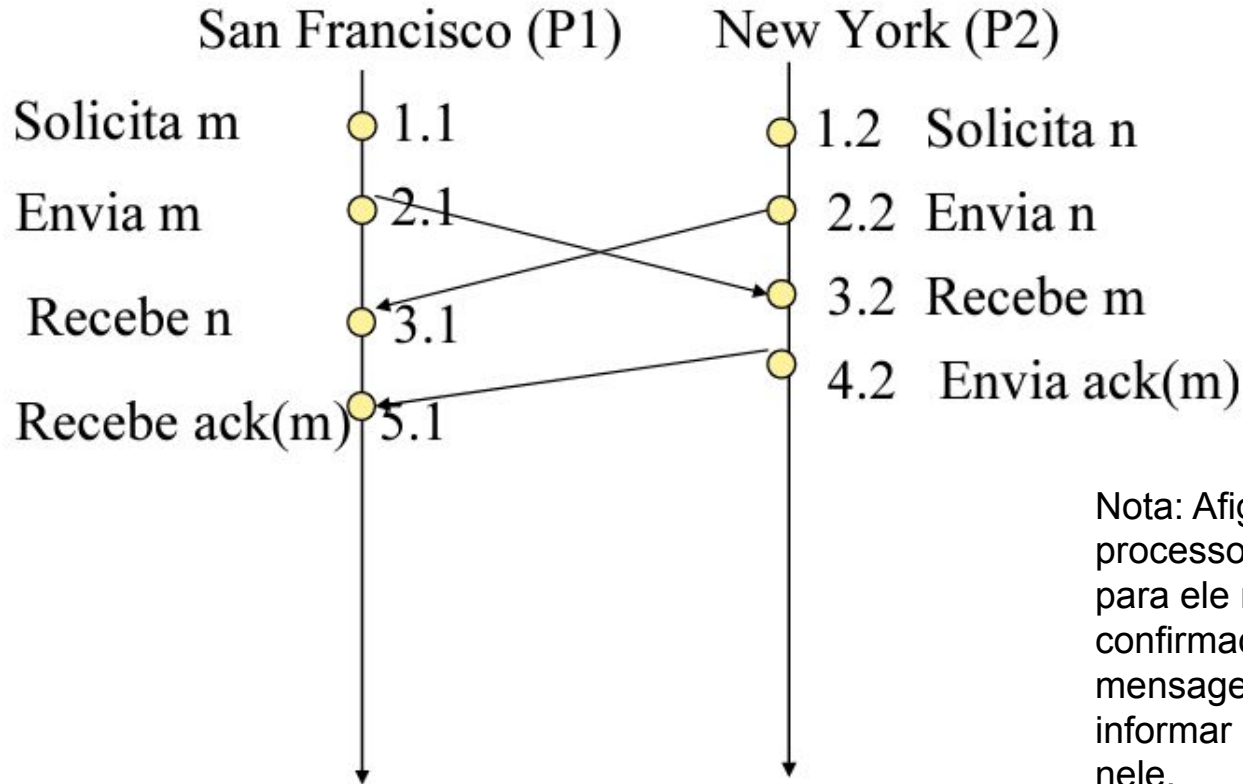
RELÓGIOS LÓGICOS DE LAMPORT

- P1 não envia a mensagem de reconhecimento de $(n, 1.2)$ até que atualização m tenha sido processada.
- Nota: O evento de recebimento da mensagem $(n, 1.2)$ por P1 recebe o marcador de hora 3.1.
- Nota: O evento de recebimento da mensagem $(m, 1.1)$ por P2 recebe o marcador de hora 3.2.

RELÓGIOS LÓGICOS DE LAMPORT

- Se P2 recebe $(n, 1.2)$ antes de $(m, 1.1)$ ele envia a mensagem de reconhecimento de $(n, 1.2)$ para todos?
 - Sim
- Então como P2 sabe que tem outras atualizações que devem ser realizadas na frente da atualização que ele recebeu do cliente (n) ?
 - Ele não sabe;
 - Ele não executa a atualização especificada em $(n, 1.2)$ até que ele receba um reconhecimento de todos os processos que neste caso é ele mesmo e P1
- P2 envia uma mensagem de recebimento quando recebe $(m, 1.1)$? Sim pois seu identificador (2) é maior que o identificador 1 da mensagem.

RELÓGIOS LÓGICOS DE LAMPORT

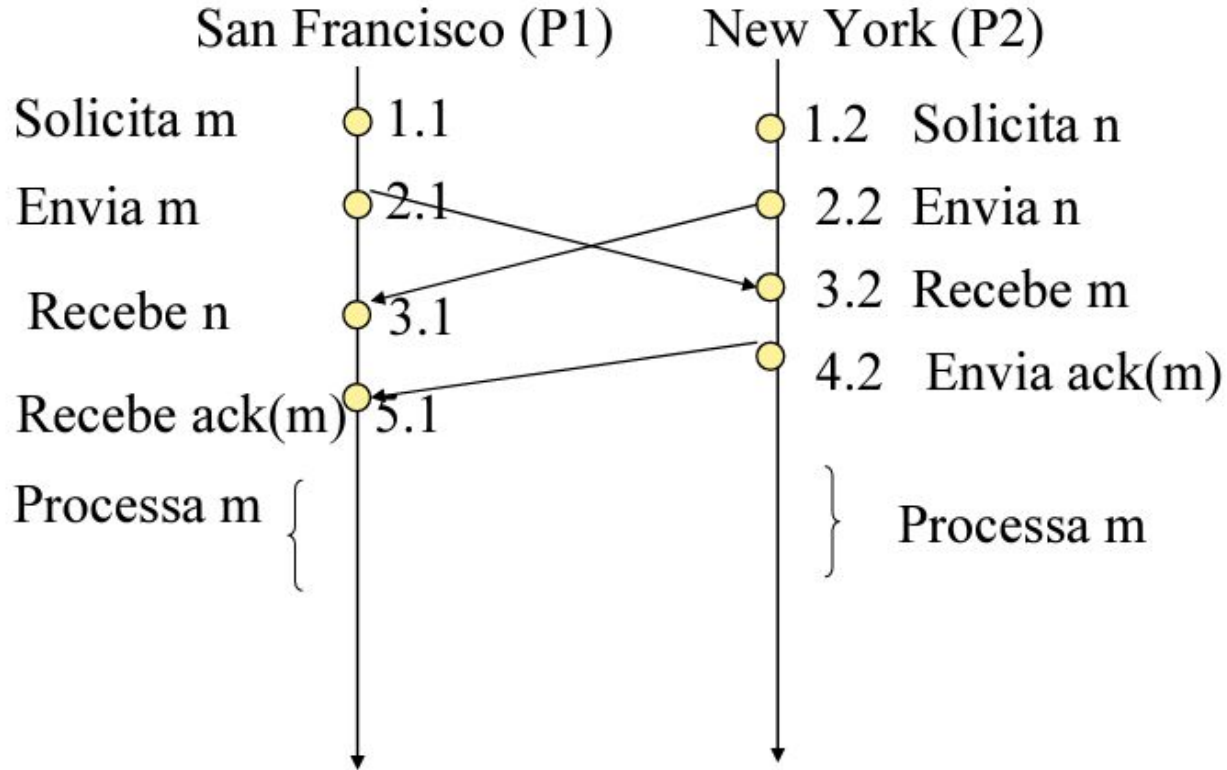


Nota: Afigura não mostra um processo enviando a mensagem para ele mesmo ou as confirmações de recebimentos das mensagens que ele envia para informar uma atualização solicitada nele.

RELÓGIOS LÓGICOS DE LAMPORT

- Resumindo, o processamento das mensagens:
 - P1e P2 necessitam realizar operações de atualização
 - P1 enviou uma mensagem de reconhecimento de recebimento de $(m, 1.1)$.
 - P2 enviou uma mensagem de reconhecimento de recebimento de $(m, 1.1)$ e de $(n, 1.2)$.
- P1e P2 receberam mensagens de reconhecimento de recebimento de $(m, 1.1)$ de todos os processos.
- Então, a atualização representada por m pode ser realizada por P1 e P2

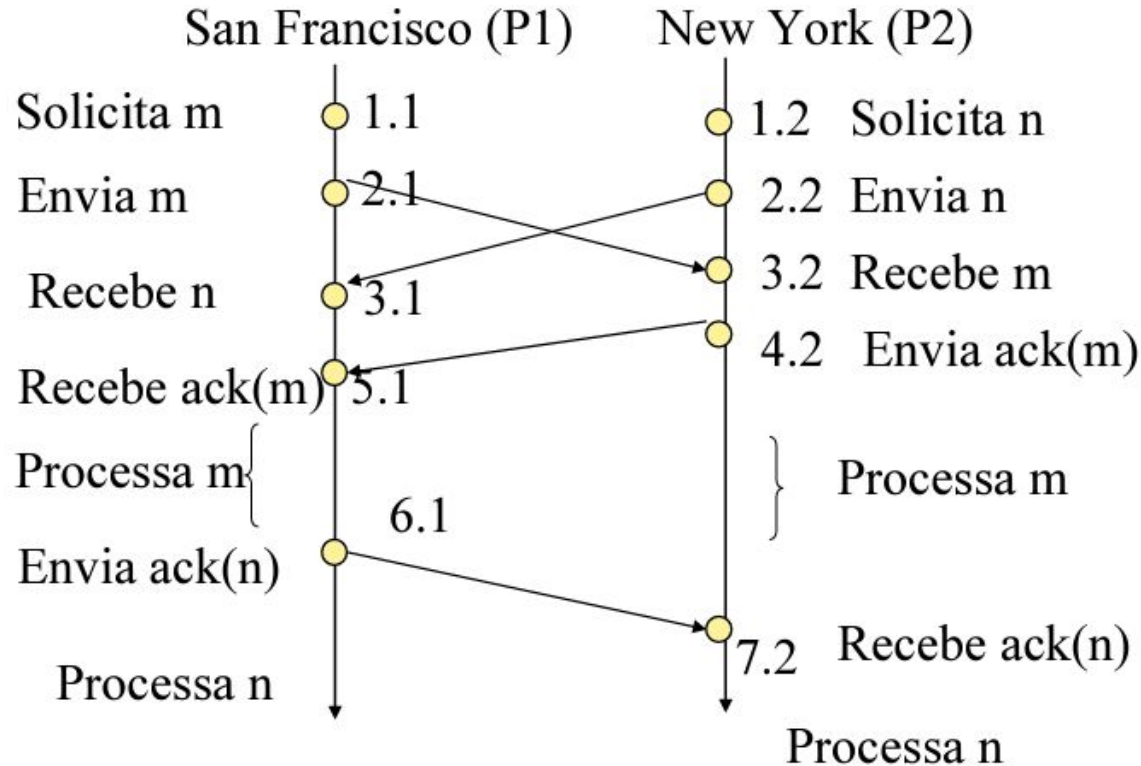
RELÓGIOS LÓGICOS DE LAMPORT



RELÓGIOS LÓGICOS DE LAMPORT

- Quando P1 finaliza a atualização m , ele pode enviar a mensagem de reconhecimento de recebimento de $(n, 1.2)$.
- Quando P1 e P2 tiverem recebido esta mensagem de reconhecimento, ambos processos receberam todas as mensagens de reconhecimento de recebimento de $(n, 1.2)$ de todos os processos.
- Neste ponto, P1 e P2 podem realizar a atualização representada por n

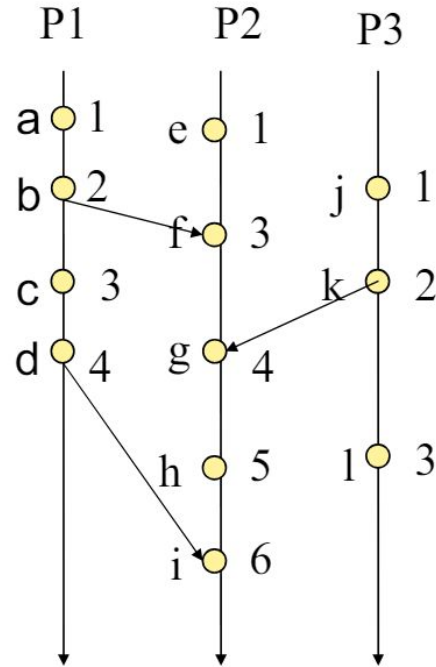
RELÓGIOS LÓGICOS DE LAMPORT



RELÓGIOS LÓGICOS DE VETORES

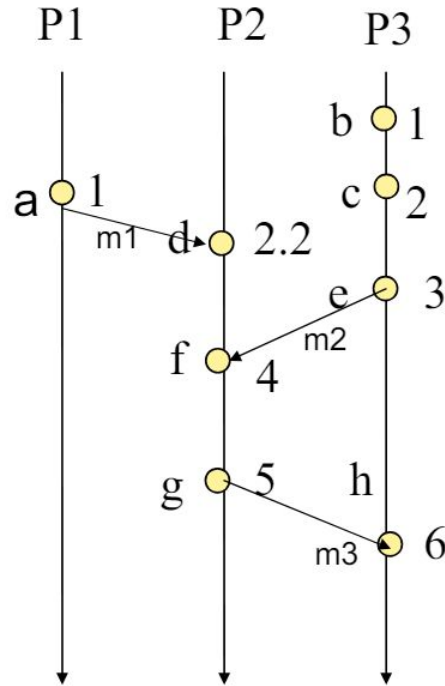
- Usando simplesmente os relógios lógicos de Lamport, nada pode ser dito sobre a relação de causalidade entre dois eventos a e b , apenas comparando seus valores de tempo $C(a)$ e $C(b)$.
 - Se $C(a) < C(b)$ não é verdadeiro implica que a aconteceu antes de b não é verdadeiro.
 - Se $C(a) < C(b)$ é verdadeiro, isso não implica necessariamente que a aconteceu antes de b .

RELÓGIOS LÓGICOS DE VETORES



$C(j) = 1$ and $C(b) = 2$; então $C(j) < C(b)$ mas j não aconteceu antes de b

RELÓGIOS LÓGICOS DE VETORES



$C(d) < C(g)$ e d e g são eventos que estão em P_2 , então pode indicar que envio de m_3 pode depender do que foi recebido em m_1

$C(d) < C(e)$ mas parece que o envio de m_2 não tem nada a ver com o recebimento de m_1

RELÓGIOS LÓGICOS DE VETORES

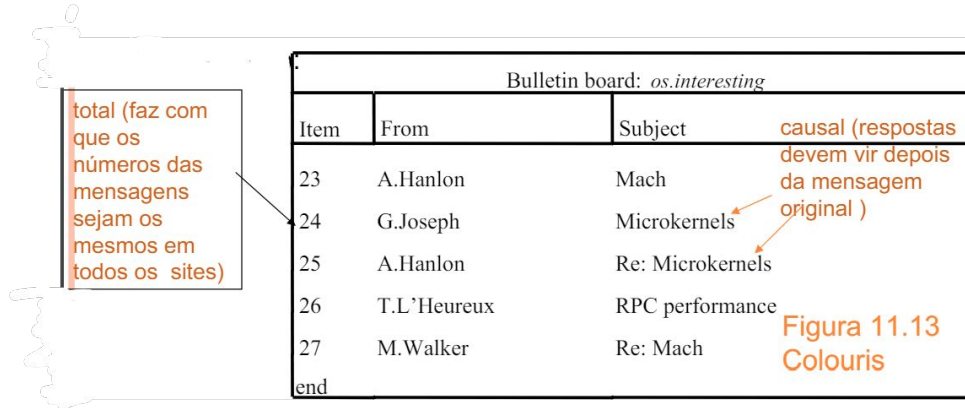
- Anteriormente vimos um ambiente ***totalmente ordenado*** onde as mensagens são processadas da mesma maneira em cada processo
- Pode-se ter um ambiente de ***qualquer ordem*** no qual todas as mensagens são recebidas por todos os processos, mas podem chegar em ordens diferentes para cada processo
- ***Ordem causal*** é usada quando uma mensagem recebida por um processo pode potencialmente afetar qualquer mensagem enviada posteriormente pelo processo. Estas mensagens devem ser recebidas na mesma ordem em todos os processos. Mensagens que não se relacionam entre elas podem ser entregues em qualquer ordem.

RELÓGIOS LÓGICOS DE VETORES

- Um serviço de boletim eletrônico da Internet
- Usuários (processos) se juntam a grupos específicos (grupos de discussão).
- Postagem, sejam novas mensagens ou respostas a mensagens, são enviadas a todos os membros do grupo.
- Pode ser usado um esquema totalmente ordenado

RELÓGIOS LÓGICOS DE VETORES

- Um serviço de boletim eletrônico da Internet
- Usuários (processos) se juntam a grupos específicos (grupos de discussão).
- Postagens, sejam novas mensagens ou respostas a mensagens, são enviadas a todos os membros do grupo.
- Pode ser usado um esquema totalmente ordenado



total (faz com que os números das mensagens sejam os mesmos em todos os sites)

Bulletin board: <i>os.interesting</i>		
Item	From	Subject
23	A.Hanlon	Mach
24	G.Joseph	Microkernels
25	A.Hanlon	Re: Microkernels
26	T.L'Heureux	RPC performance
27	M.Walker	Re: Mach
end		

causal (respostas devem vir depois da mensagem original)

Figura 11.13
Colouris

RELÓGIOS LÓGICOS DE VETORES

- Em um ambiente totalmente ordenado, caso uma mensagem B seja entregue depois de uma mensagem A, isso não significa que B é uma reação à A.
- Não se necessita de um ambiente tão rígido como o totalmente ordenado
- A recepção de um artigo precede causalmente a postagem de uma reação ao artigo.
- A recepção da reação a um artigo deve sempre seguir a recepção do artigo.
- No exemplo do boletim, os itens 26 e 27 podem ser apresentados em ordens diferentes em sites diferentes.
- Os itens 25 a 26 podem ser apresentados em ordens diferentes em sites diferentes.

RELÓGIOS LÓGICOS DE VETORES

- O problema é que relógios lógicos de Lamport não capturam causalidade
- Algo mais é necessário:
 - Relógios Lógicos Vetoriais tentam solucionar problemas de causalidade apresentados nos relógios de Lamport
- Um relógio vetorial **$VC(a)$** designado a um evento **a** , tem a seguinte propriedade: se **$VC(a) < VC(b)$** para algum evento **b** , então **a** precede por causalidade o evento **b**

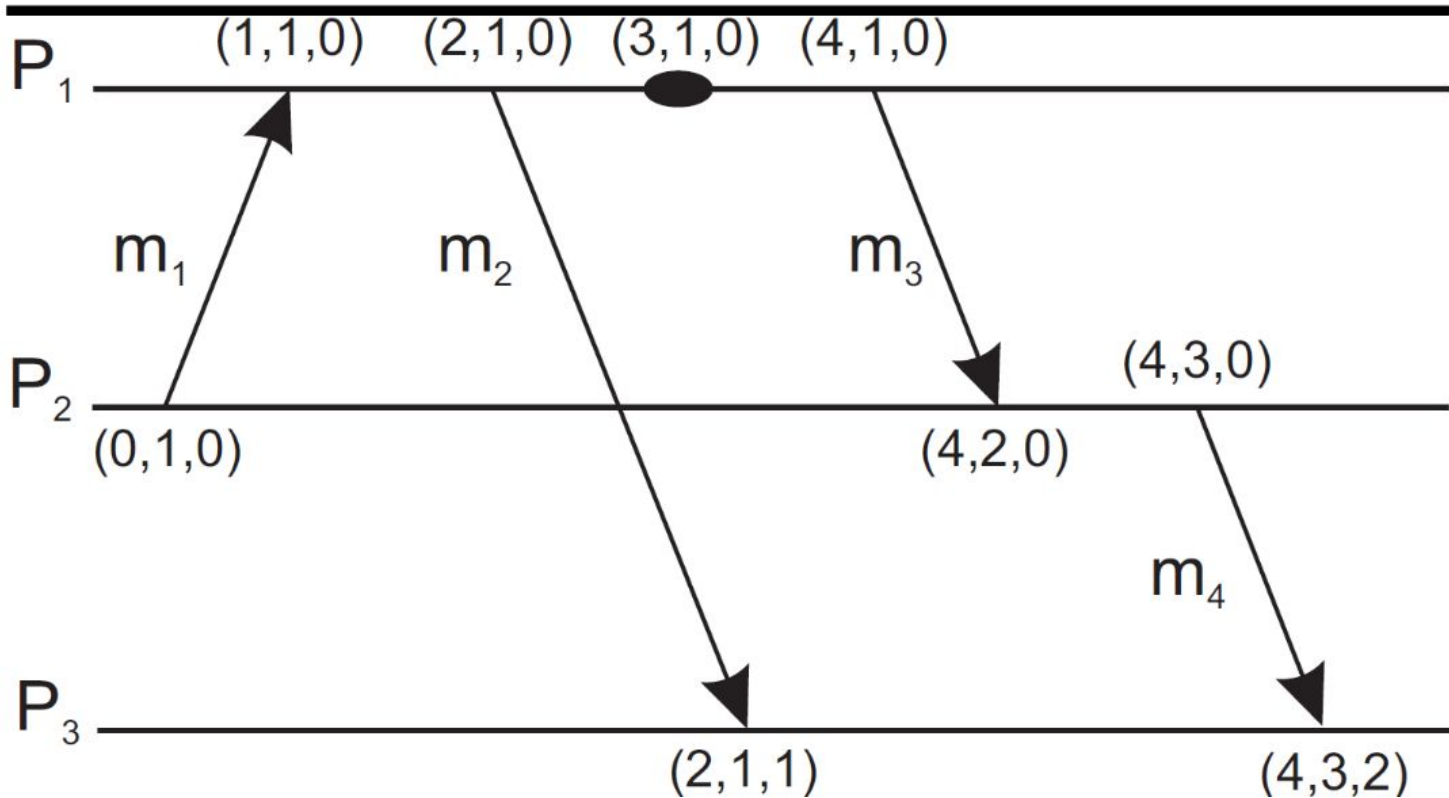
RELÓGIOS LÓGICOS DE VETORES

- Cada processo P_i mantém um vetor VC_i com as propriedades:
 - 1. $VC_i[i]$ é o número de eventos que ocorreram em P_i até o instante em questão. $VC_i[i]$ é o relógio lógico local em P_i
 - 2. Se $VC_i[j] = k$, então P_i sabe que k eventos ocorreram em P_j . Portanto, P_i conhece o tempo local de P_j
- A primeira propriedade é mantida incrementando $VC_i[i]$ na ocorrência de cada evento em P_i
- A segunda propriedade é mantida por meio de caronas que os vetores pegam com as mensagens que são enviadas.

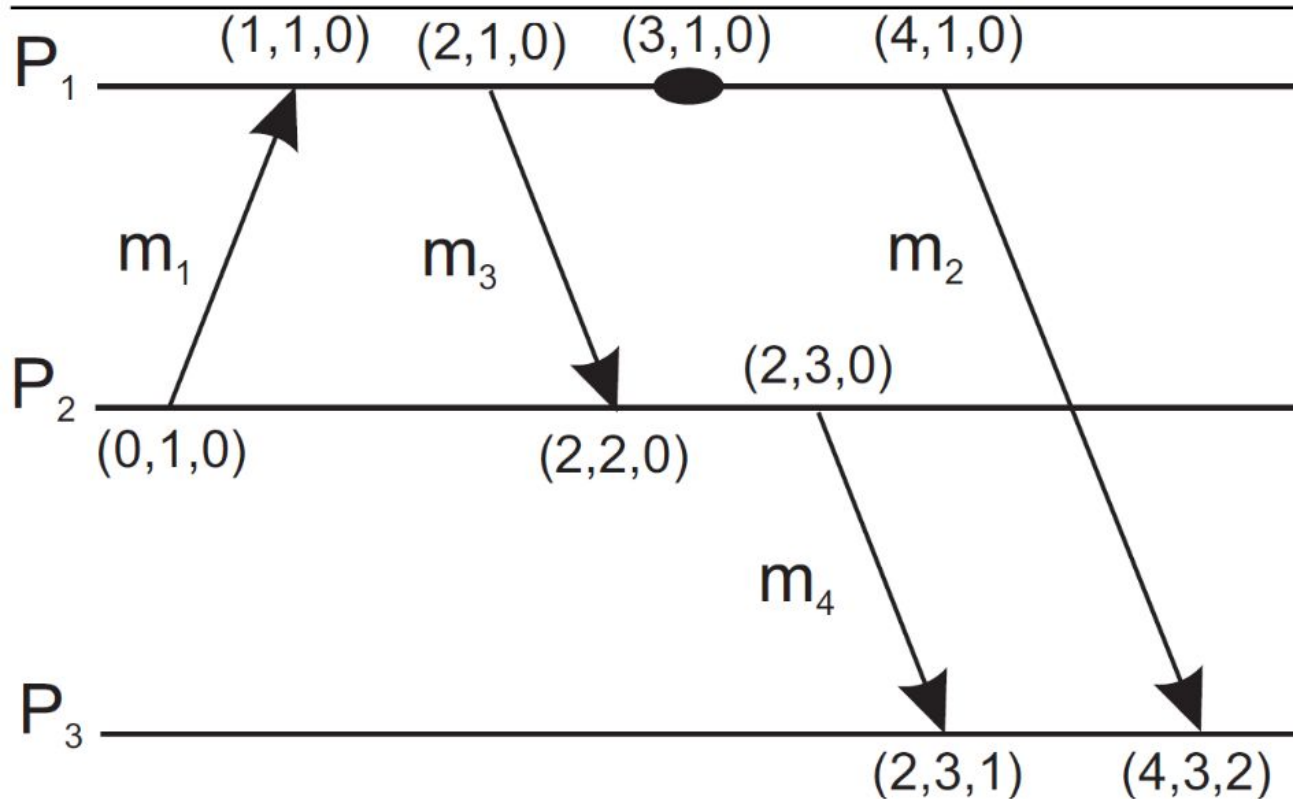
RELÓGIOS LÓGICOS DE VETORES

- Ocorrem as seguintes etapas
 - Antes de executar um evento, P_i faz $VC_i[i] \leftarrow VC_i[i] + 1$
 - Quando o processo P_i envia uma mensagem m a P_j , ele iguala a etiqueta de tempo de m , ou seja $ts(m) \leftarrow VC_i[i]$
 - Quando o processo P_j recebe m , ajusta seu próprio vetor definindo cada entrada $V_j[k]$ para $\max \{V_j[k], ts(m)[k]\}$
- Desta forma, um receptor é informado sobre o número de eventos que ocorreram em outros processos antes de P_i enviar a mensagem m

RELÓGIOS LÓGICOS DE VETORES



RELÓGIOS LÓGICOS DE VETORES



RELÓGIOS LÓGICOS DE VETORES

EXEMPLO: BOLETIM

Analysis

Situation	$ts(m_2)$	$ts(m_4)$	$ts(m_2) < ts(m_4)$	$ts(m_2) > ts(m_4)$	Conclusion
(a)	(2, 1, 0)	(4, 3, 0)	Yes	No	m_2 may causally precede m_4
(b)	(4, 1, 0)	(2, 3, 0)	No	No	m_2 and m_4 may conflict

RELÓGIOS LÓGICOS DE VETORES

EXEMPLO: BOLETIM

- Os marcadores de hora do vetor são atualizados somente quando postam ou recebem artigos i.e., quando uma mensagem é enviada ou recebida.
- V_q é colocada na mensagem enviada pelo processo q para o processo p
- Quando p recebe uma mensagem, ele executa:
 - *For $i = 1$ to n do*
 - $V_p[i] = \max(V_p[i], V_q[i])$;
- Quando p envia uma mensagem, ele executa:
 - $V_p[p] = V_p[p] + 1$;

RELÓGIOS LÓGICOS DE VETORES

EXEMPLO: BOLETIM

- Quando um processo P_i publica um artigo, ele envia para o grupo uma mensagem com o artigo e o vetor de marcadores de hora. Esta mensagem vai ser chamada de a . Assuma que o vetor desta mensagem seja V_i .
- O processo P_j posta uma reação e envia uma mensagem com a reação e o vetor V_j chamada de r .
- Note que $V_j > V_i$
- Mensagem r pode chegar em P_k antes da mensagem a .

RELÓGIOS LÓGICOS DE VETORES

EXEMPLO: BOLETIM

- **P_k** vai adiar enviar a reação **r** para o display até que todas a mensagens que precedem casualmente **r** também tenham sido recebidas.
- A mensagem **r** é enviada para o display se somente se ocorrem as seguintes condições:
 - $V_j[j] = V_k[j] + 1$
 - Isso mostra que **r** é a próxima mensagem que **P_k** estava esperando do processo **P_j**
 - $V_j[i] \leq V_k[i]$ para todo **i** não igual a **j**
 - Isso mostra que **P_k** viu pelo menos o mesmo número de mensagens vistas pelo processo **P_j** quando ele enviou a mensagem **r** .

RELÓGIOS LÓGICOS DE VETORES

EXEMPLO: BOLETIM

- Inicialmente $V_j[i] = 0$ e $V_k[i] = 0$ para todo i . Faz sentido pois nenhuma mensagem foi enviada.
- Suponha que P_j envia uma mensagem com $V_j[j] = 1$. Isso implica que esta é a primeira mensagem enviada por P_j
- Como $V_k[j]$ é 0, P_k vai aceitar a mensagem enviada por P_j pois ele está esperando a primeira mensagem de P_j .
- Suponha que P_j envia uma mensagem com $V_j[j] = 5$ (indicando que é a quinta mensagem enviada) e que $V_k[j]$ é 3, o que indica que P_k está esperando pela quarta mensagem enviada por P_j . Isso indica que a mensagem recebida deve ser atrasada por P_k para ser mostrada no display

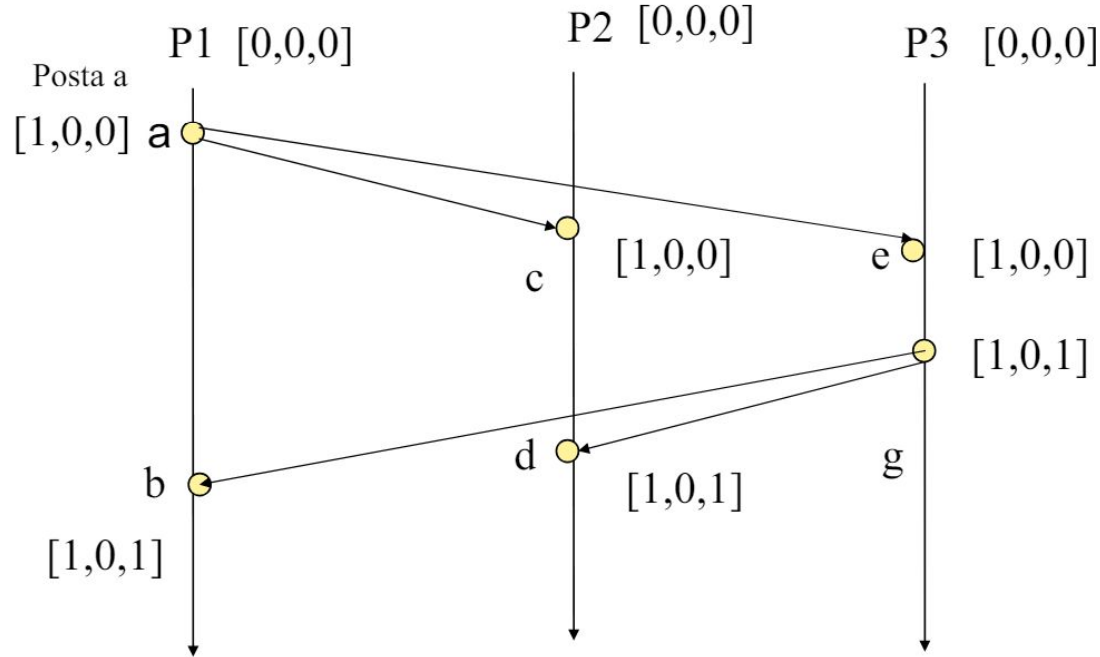
RELÓGIOS LÓGICOS DE VETORES

EXEMPLO: BOLETIM

- Assuma que $V_j[i] \leq V_k[i]$ não seja verdadeiro para algum i . Ou seja, $V_j[i] > V_k[i]$ para algum i . Isso indica que P_i enviou uma mensagem que foi recebida por P_j mas não por P_k .
- Neste caso, P_k não vai enviar a mensagem para o display até que ele receba a mensagem que está faltando

RELÓGIOS LÓGICOS DE VETORES

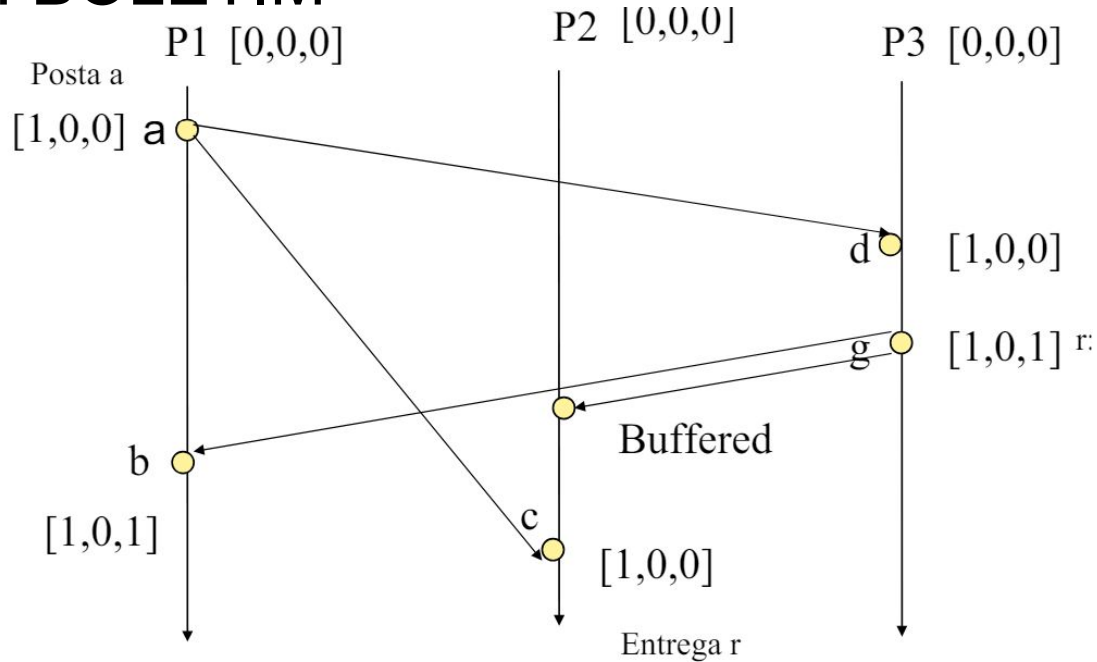
EXEMPLO: BOLETIM



Mensagem *a* chega em P2 antes da resposta *r* de P3

RELÓGIOS LÓGICOS DE VETORES

EXEMPLO: BOLETIM



A mensagem *a* chega em P2 depois da resposta de P3;
 A resposta não é entregue logo que é recebida.