# Sincronização em Sistemas Distribuídos – parte 2

Raquel Mini - raquelmini@pucminas.br

# Relógios Lógicos

Como não podemos sincronizar perfeitamente os relógios em um sistema distribuído, não podemos usar o tempo físico para descobrir a ordem de qualquer par de eventos

Usamos um esquema de causalidade física para ordenar alguns dos eventos que ocorrem em diferentes processos

Do ponto de vista de um único processo, os eventos são ordenados exclusivamente pelos tempos dados pelo relógio local

# Algoritmo de Lamport

Proposta por Lamport em um paper clássico:

- Lamport, L. Time, clocks and the ordering of events in a distributed system, CACM, vol. 21, pp. 558-564, july 1978.

Ideias básicas:

- Definir ordem de eventos apenas entre processos que interagem entre si
- Relógio lógico: o que importa é a ordem dos eventos e não o tempo físico em que os mesmos ocorreram

# Algoritmo de Lamport

Notação: $A \rightarrow B$

- Lê-se que "A ocorreu antes de B"
- Significado: todos os processos do sistema concordam que primeiro ocorreu o evento A e então o evento B

Eventos que não são ordenados pela relação "$\rightarrow$" são ditos concorrentes

- Notação: $A \parallel B$

# Algoritmo de Lamport

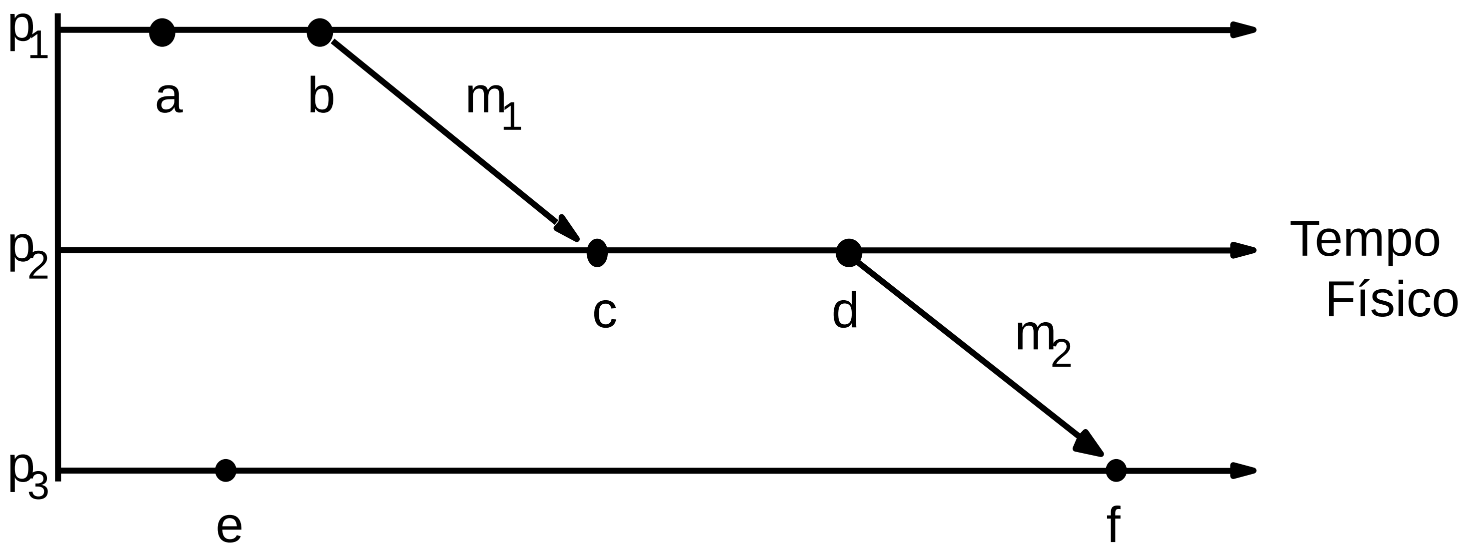
Definição:

- Se A e B são eventos de um mesmo processo e A foi executada antes de B, então $A \rightarrow B$
- Se A consiste no evento de enviar uma mensagem para um processo e B é o evento de recebimento desta mensagem, então $A \rightarrow B$.
- Se $A \rightarrow B$ e $B \rightarrow C$, então $A \rightarrow C$ (transitividade)

# Algoritmo de Lamport

Ocorrência de eventos nos processos p1, p2 e p3

- a →b, c →d, b →c, d →f, a →f
- a||e, e||a (e, a são eventos concorrentes)



# Algoritmo de Lamport

Utiliza o conceito de relógio lógico (L):

- Inteiro monotonicamente crescente armazenado em cada nó
- Incrementado a cada evento
- Os processos transmitem os valores de seus relógios lógicos em mensagens

Implementação do conceito de relógio lógico:

- Se $A \rightarrow B$ então $L(A) < L(B)$

# Algoritmo de Lamport

O valor do relógio lógico de um processo P, Lp, é atualizado da seguinte forma:

- Lp é incrementado antes de cada evento de P
- Quando P envia uma mensagem M para Q, ele insere na mesma um *timestamp* (t= Lp)
- Quando Q recebe (M, t), então

    Lq := max (Lq, t)

    Lq := Lq+1

    (garante-se assim que Lp < Lq)

# Algoritmo de Lamport

Em qualquer sequência de eventos relacionando dois eventos a e b que

- Se $a \rightarrow b$ então $L(a) < L(b)$

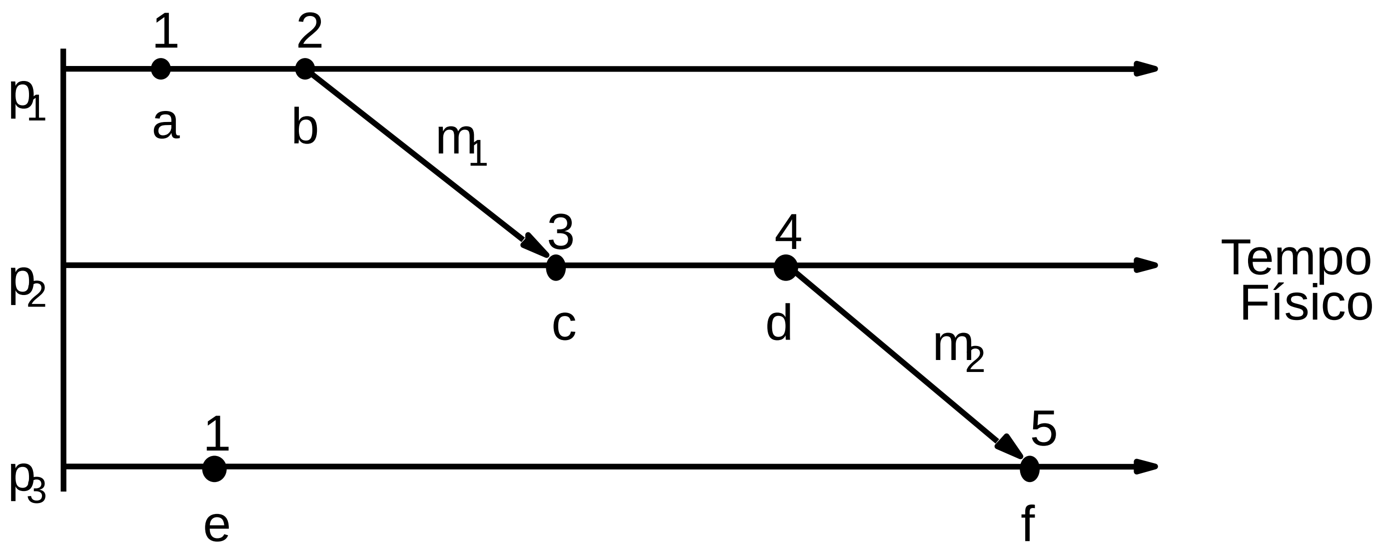
O inverso não é verdadeiro

- Se $L(a) < L(b)$ então não podemos inferir que $a \rightarrow b$

# Algoritmo de Lamport

Rótulos de tempo do Algoritmo de Lamport para ordenação de eventos

- L(b) > L(e), mas b||e



# Algoritmo de Lamport

Relógio lógico dá origem a uma relação de ordem parcial

- Alguns pares de eventos distintos, gerados por diferentes processos, têm carimbos de tempo de Lamport numericamente idênticos

Ordenação total: adicionar o identificador de cada processo no final do relógio lógico

- Exemplo
  - Processo 1: relógio lógico = 40.1
  - Processo 2: relógio lógico = 40.2

# Relógios Vetoriais

Se L(a) < L(b) não podemos concluir que a→b

Um relógio vetorial para um sistema de N processos é um vetor de N inteiros

- Cada processo mantém seu próprio relógio vetorial $V_i$ o que utiliza para gerar carimbos de tempo dos eventos locais
- Os processos levam de carona os carimbos de tempo vetoriais nas mensagens que trocam entre si

# Relógios Vetoriais

Para um relógio vetorial $V_i$, $V_i[i]$ é o número dos eventos em que $p_i$ indicou o carimbo de tempo e $V_i[j]$ ($j \neq i$) é o número dos eventos ocorridos em $p_j$ nos quais $p_i$ potencialmente foi afetado

RV1: Inicialmente, $V_i[j] = 0$, para $i$, $j = 1, 2..., N$.

RV2: Imediatamente antes de $p_i$ gerar o carimbo de tempo de um evento, ele configura $V_i[i] := V_i[i] + 1$.

RV3: $p_i$ inclui o valor $t = V_i$ em cada mensagem que envia.

RV4: Quando $p_i$ recebe um carimbo de tempo $t$ em uma mensagem, ele configura $V_i[j] := max(V_i[j], t[j])$, para $j = 1, 2..., N$. Considerar o máximo de duas componentes de carimbos de tempo vetoriais dessa maneira é conhecido como operação de *merge* (integração).

# Relógios Vetoriais

Podemos comparar carimbos de tempo vetoriais como segue:

$$V = V' \ sse \ V[j] = V'[j] \text{ para } j = 1, 2\ldots, N$$

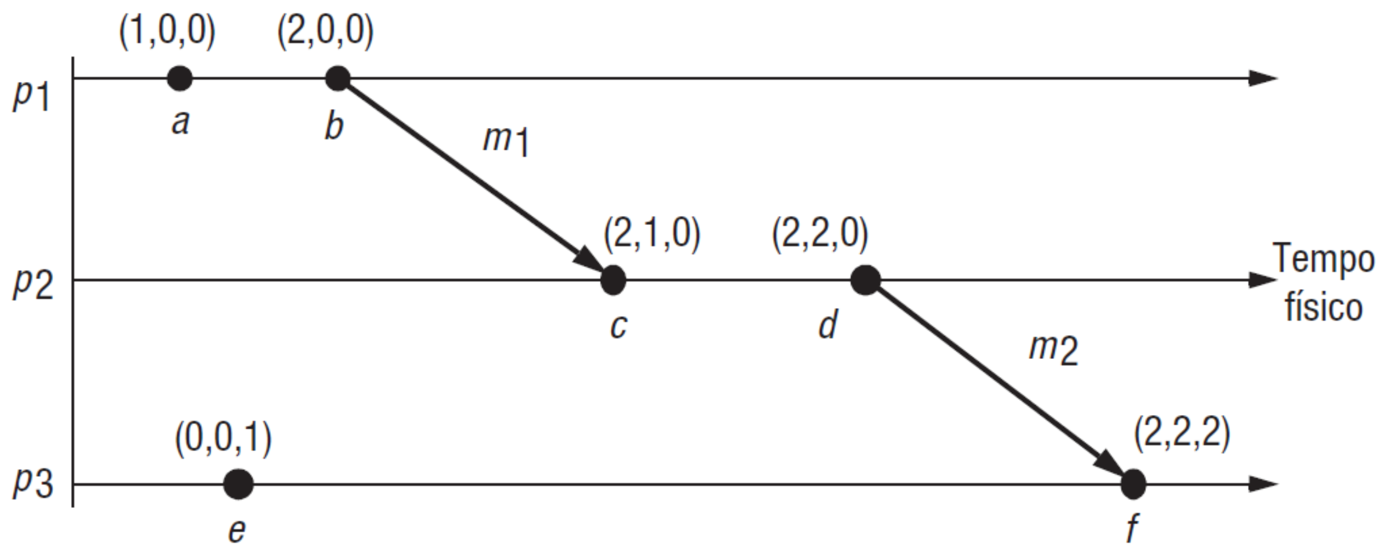
$$V \leq V' \ sse \ V[j] \leq V'[j] \text{ para } j = 1, 2\ldots, N$$

$$V < V' \ sse \ V \leq V' \wedge V \neq V'$$

# Relógios Vetoriais

Carimbos de tempo vetoriais

- V(a)<V(f) pois a →f
- c||e pois nem V(c) ≤ V(e) nem V(e) ≤ V(c)



# Relógios Vetoriais

Desvantagem em relação aos carimbos de tempo de Lamport

- Ocupam espaço de armazenamento e carga útil de mensagem proporcional a N (número de processos)

# Tarefa 5 – postar no Canvas até 21/03/2021

1. Mostre que, se os eventos a e b são concorrentes, então nem $V(a) \leq V(b)$, nem $V(b) \leq V(a)$. Assim, mostre que, se $V(a) < V(b)$ então $a \longrightarrow b$.