**Clase del día - 17/03/2021**

El tiempo es una referencia que se utiliza para establecer un orden en una secuencia de eventos.

Como vimos anteriormente, si dos computadoras no interactúan entonces no es necesario que sus relojes estén sincronizados.

Por otra parte, si dos computadoras interactúan, en general no es importante que coincidan en el tiempo real sino en el orden en que ocurren los eventos.

**Happens-before**

En el artículo [Time, Clocks, and the Ordering of Events in Distributed Systems](http://lamport.azurewebsites.net/pubs/time-clocks.pdf) (1978) Leslie Lamport define la relación A→B (se lee, A happens-before B) de la siguiente manera:

1. Si A y B son eventos del mismo proceso y A ocurre antes que B, entonces A → B
2. Si A es el envío de un mensaje y B la recepción del mensaje, entonces A → B

La relación happens-before tiene las siguientes propiedades:

Transitiva:            Si A → B y B → C entonces A → C

Anti-simétrica:    Si A → B entonces no(B → A)

Irreflexiva:           no(A → A) para cada evento A

**Relojes lógicos**

Se define un **reloj lógico Ci** para un procesador Pi como una función Ci(A) la cual asigna un número al evento A.

Un reloj lógico se implementa como un contador sin una relación directa con un reloj físico, como es el caso de los contadores de "ticks" de las computadoras digitales.

Dados los eventos A y B, si el evento A ocurre antes que el evento B, entonces Ci(A)<Ci(B), por tanto:

Si A → B entonces Ci(A)<Ci(B)

Esto significa que si A happens-before B, entonces el evento A ocurre en un tiempo lógico menor al tiempo lógico en que ocurre el evento B.

**Algoritmo de sincronización de relojes lógicos de Lamport**

Ahora utilizaremos la relación happends-before definida por Lamport para sincronizar relojes lógicos en diferentes procesadores (computadoras).

Supongamos que tenemos los procesadores P1, P2 y P3. Cada procesador tiene un reloj lógico (contador) que se incrementa periódicamente mediante un thread.

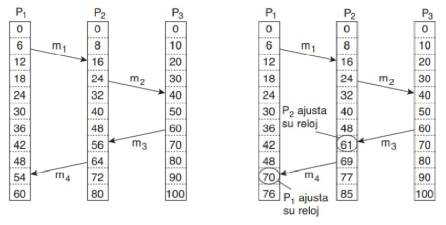
El reloj lógico del procesador P1 se incrementa en 6, el reloj lógico del procesador P2 se incrementa en 8 y el reloj lógico del procesador P3 se incrementa en 10.

¿Cómo sincronizar los relojes lógicos de los tres procesadores de manera que los eventos que ocurren en los procesadores puedan ordenarse?

Lamport resuelve este problema utilizando la relación happends-before para sincronizar los relojes lógicos de diferentes procesadores. Explicaremos el algoritmo de sincronización de relojes lógicos de Lamport con un ejemplo.

Supongamos que al tiempo **6** el procesador P1 envía el mensaje m1 al procesador P2, este procesador recibe el mensaje al tiempo **16**. Al tiempo **24** el procesador P2 envía el mensaje m2 al procesador P3, este procesador recibe el mensaje al tiempo **40**.

Hasta ahora todo es correcto, debido a que el mensaje m1 es enviado al tiempo 6 y recibido al tiempo 16, y el mensaje m2 es enviado al tiempo 24 y recibido al tiempo 40, es decir, el tiempo de envío es menor al tiempo de recepción.



Fuente: Sistemas Distribuidos Principios y Paradigmas 2a. Ed. Andrew S. Tanenbaum

Al tiempo **60** el procesador P3 envía el mensaje m3 al procesador P2, este procesador recibe el mensaje al tiempo **56**. Lo anterior contradice la definición de la relación happends-before, ya que la recepción de un mensaje debe ocurrir después del envío del mismo mensaje.

Entonces lo que se hace es ajustar el reloj lógico del procesador P2, asignando el tiempo lógico del procesador P3 cuando envía el mensaje m3 más uno, es decir, se modifica el reloj lógico del procesador P2 a 61 cuando recibe el mensaje m3.

Al tiempo 69, el procesador P2 envía el mensaje m4 al procesador P1, este procesador recibe el mensaje al tiempo 54, lo cual contradice la relación happends-before.

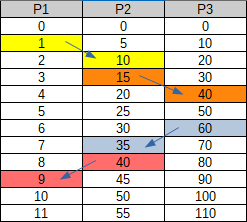
Entonces se aplica el mismo procedimiento para el ajuste del reloj lógico del procesador P1, por tanto el reloj lógico de este procesador se modifica 70 cuando recibe el mensaje m4.

**Actividades individuales a realizar**

Considere el ejemplo que vimos sobre el algoritmo de Lamport. Suponiendo que los relojes lógicos de los tres procesadores inician en cero, y el reloj lógico del procesador P1 se incrementa en 1, el reloj lógico del procesador P2 se incrementa en 5 y el reloj lógico del procesador P3 se incrementa en 10.

Suponga que se envían los siguientes mensajes sin sincronizar los relojes lógicos:

* El procesador P1 envía el mensaje m1 al tiempo 1, y el procesador P2 lo recibe al tiempo 10.
* El procesador P2 envía el mensaje m2 al tiempo 15, y el procesador P3 lo recibe al tiempo 40.
* El procesador P3 envía el mensaje m3 al tiempo 60, y el procesador P2 lo recibe al tiempo 35.
* El procesador P2 envía el mensaje m4 al tiempo 40, y el procesador P1 lo recibe al tiempo 9



Si aplica el algoritmo de Lamport para sincronizar los relojes lógicos ¿Qué tiempos lógicos tendrán los procesadores P1 y P2 cuando el procesador P3 tenga el tiempo 110?