

## **Facultad de Ciencias, UNAM**

**Materia:** Computación Distribuida

**Profesor:** Mauricio Riva Palacio Orozco

**Ayudantes:** Alan Alexis Martínez López y Adrián Felipe Fernández Romero

### **Práctica 5 - Relojes Lógicos**

#### **Encabezado**

Esta práctica tiene como objetivo que los alumnos comprendan el concepto de reloj en un sistema distribuido asíncrono.

Deberán implementar dos relojes **Lamport y vectorial** sobre los algoritmos desarrollados previamente.

#### **Desarrollo**

En esta práctica se convertirá el sistema síncrono en uno parcialmente asíncrono.

Esto se logrará introduciendo tiempos de espera aleatorios entre cada envío y recepción de mensajes.

Como dichos valores estarán acotados, el sistema no será completamente asíncrono.

Para el reloj de Lamport, se utilizará como base el algoritmo de Broadcast.

Deberán adaptar el archivo entregado en esa práctica para que cada nodo:

- opere de forma asíncrona, y
- cuente con una variable de reloj.

Para el reloj vectorial, se tomará como base el algoritmo DFS.

El procedimiento será el mismo que con Lamport, pero en este caso el reloj será un arreglo de enteros.

(Nota: será necesario modificar el constructor del nodo para indicar cuántos nodos hay en la gráfica).

## Facultad de Ciencias, UNAM

**Materia:** Computación Distribuida

**Profesor:** Mauricio Riva Palacio Orozco

**Ayudantes:** Alan Alexis Martínez López y Adrián Felipe Fernández Romero

### Práctica 5 - Relojes Lógicos

#### Desarrollo

En ambos algoritmos deberán mantener una lista de eventos (envíos y recepciones).

Cada elemento de la lista debe incluir:

- el valor del reloj al enviar o recibir el mensaje,
- el tipo de evento ( $E$  para envío,  $R$  para recepción),
- el mensaje,
- el nodo emisor y el nodo receptor (en ese orden).

Además, deberán implementar la clase `NodoConsenso` para la ejecución del algoritmo.

En resumen, se deberán entregar las clases `NodoBroadcast` y `NodoDFS` actualizadas

#### Prerrequisitos

Para esta práctica no se proporcionará un esqueleto, ya que usarán los algoritmos previamente desarrollados. Solo se les dará el archivo `test.py`, que deberán colocar en la carpeta donde se encuentren los archivos de sus nodos.

Para ejecutar las pruebas:

```
pytest -q test.py
```

#### Observaciones

1.- Lo fundamental es asegurarse de que los relojes asignen valores temporales coherentes con el orden causal de los eventos.

2.- Para generar números aleatorios pueden usar la biblioteca `random` de Python

**Facultad de Ciencias, UNAM**

**Materia:** Computación Distribuida

**Profesor:** Mauricio Riva Palacio Orozco

**Ayudantes:** Alan Alexis Martínez López y Adrián Felipe Fernández Romero

**Práctica 5 - Relojes Lógicos**

**Lineamientos de entrega**

Como se reutiliza código anterior, se asume que ya está bien documentado.

Sin embargo, deberán documentar las modificaciones realizadas en esta práctica para los relojes.

Dentro de la entrega deberán incluir un archivo README con:

- el número y nombre de la práctica (Práctica 5),
- los nombres de los integrantes con su número de cuenta, y una breve descripción de la implementación de cada algoritmo.

Solo un integrante deberá subir la práctica; el otro deberá marcarla como entregada antes de la fecha límite.

El archivo final deberá subirse comprimido en formato .zip con el nombre:

**Practica5-ApellidoPaterno1ApellidoMaterno1ApellidoPaterno2ApellidoMaterno2...**