Instituto Tecnológico de Costa Rica
Programa de Maestría en Ciencia de la Computación
I Semestre 2017
Sistemas Operativos Avanzados
Estudiante: Oscar Rodríguez Arroyo

Tarea 2

1. Determine matemáticamente la propiedad fuerte del reloj para relojes vectoriales

La propiedad fuerte del reloj para relojes vectoriales establece que, si H corresponde a la historia global de un sistema distribuido:

Sean $a, b \in H, sea\ V\ un\ reloj\ vectorial\ de\ a\ y\ W\ un\ reloj\ vectorial\ de\ b, entonces$:

1. $V = W \iff a = b$
2. $V < W \iff a \rightarrow b$
3. $W < V \iff b \rightarrow a$
4. $V \mid\mid W \iff a \mid\mid b$

Se conoce el algoritmo de Fidge para relojes vectoriales:

- Regla 1. Inicialmente todos los valores son 0
- Regla 2. El valor del reloj local se incrementa en 1 antes de cada evento atómico
- Regla 3. El valor actual de todo el vector se adjunta a toda señal que se envía
- Regla 4. Cuando un proceso recibe una señal, éste actualiza su vector con el máximo entre el valor local y el valor adjunto a la señal + 1.
- Regla 5. Los valores de los vectores nunca se decrementan.

Además se sabe que:

$a \rightarrow b \Rightarrow \exists i\ tq\ (a \in i) \wedge (V[i] < W[i])$ (Lampert)

$V = W \iff \forall i, V[i] = W[i]$ (1)

$V \leq W \iff \forall i, V[i] \leq W[i]$ (2)

$V < W \iff V \leq W \wedge \exists i\ tq\ V[i] < W[i]$ (3)

$V \mid\mid W \iff \exists i, j\ tq\ V[i] < W[i] \wedge V[j] > W[j]$ (4)

Demostración:

1. $\boldsymbol{V = W \iff a = b}$
   - $\Longrightarrow$

     Por contradicción: Suponemos a diferente de b

     i. $a \neq b \wedge V = W$
     ii. $\Longrightarrow a \rightarrow b \vee b \rightarrow a$ (i)
     iii. $\Longrightarrow \forall i, V[i] = W[i] \wedge \forall i, V[i] = W[i]$ (i)

iv. $\Longrightarrow \nexists i, V[i] < W[i]$ (ii)
v. $\Longrightarrow \neg(a \rightarrow b)$ (iv), reglas 2 y 4, y por definición de *happened before*
vi. $\Longrightarrow b \rightarrow a$ (ii) y (v)
vii. $\Longrightarrow \nexists i, W[i] < V[i]$ (ii)
viii. $\Longrightarrow \exists i, W[i] < V[i]$ (vi) y Lampert

Hay una contradicción en vii y vii, por lo tanto, el supuesto de que a es diferente de b no es verdadero: $\Longrightarrow a = b$

- $\Longleftarrow$

i. $a = b$
ii. $\Longrightarrow \neg(a \rightarrow b) \wedge \neg(b \rightarrow a)$ (i)
iii. $\Longrightarrow \nexists i, V[i] < W[i] \wedge a \in i$ (ii) y Lampert
iv. $\Longrightarrow \nexists i, V[i] > W[i] \wedge b \in i$ (ii) y Lampert
v. $\Longrightarrow \forall i, V[i] = W[i]$ (iii) y (iv)
$\Longrightarrow V = W$

2. $\boldsymbol{V < W \Leftrightarrow a \rightarrow b}$

- $\Longrightarrow$

$V < W$
$\Longrightarrow V \leq W \ \wedge \exists i \ tq \ V[i] < W[i]$ por (3)
$\Longrightarrow \exists i \ tq \ V[i] < W[i]$

Por reglas 2 y 4 del algoritmo:

$\Longrightarrow (a \ pertenece \ al \ mismo \ proceso \ que \ b \ y \ a \ pasa \ antes \ que \ b)$
$\vee \ (a \ y \ b \ pertenecen \ a \ diferentes \ procesos \ y \ a \ envía \ mensaje \ a \ b)$
$\Longrightarrow a \rightarrow b$ por definición de *happened before*

- $\Longleftarrow$

$a \rightarrow b$

$\Longrightarrow \exists i \ tq \ V[i] < W[i]$ (Lampert)

Por reglas 2, 4 y 5, y definición de *happened before* se sabe que:

$\Longrightarrow \nexists i \ tq \ V[i] > W[i] \Longrightarrow \forall i, V[i] \leq W[i]$

$\Longrightarrow V[i] \leq W[i]$ (2)
$\Longrightarrow V[i] \leq W[i] \wedge \exists i \ tq \ V[i] < W[i]$

$\Longrightarrow V < W$ (3)

3. $\boldsymbol{W < V \Leftrightarrow b \rightarrow a}$

Demostración es igual a la del punto 2 intercambiando los valores de a y b.

4. $\boldsymbol{V} \,||\, \boldsymbol{W} \Leftrightarrow \boldsymbol{a} \,||\, \boldsymbol{b}$

- $\Longrightarrow$

Por contradicción: Suponemos a y b no son concurrentes

i.	$\neg(a \,||\, b) \wedge V \,||\, W$	
ii.	$\Longrightarrow a \rightarrow b \vee b \rightarrow a \vee a = b$ (i)	
iii.	$\Longrightarrow \exists i, j \; tq \; V[i] < W[i] \wedge V[j] > W[j]$	(i) y (4)
iv.	$\Longrightarrow \exists i, V[i] \neq W[i] \Longrightarrow V \neq W \Rightarrow a \neq b$	(iii) y demostración 1.
v.	$\Longrightarrow \neg(V \leq W) \wedge \neg(W \leq V)$	(iii) y (2)
vi.	$\Longrightarrow \neg(a \rightarrow b) \wedge \neg(b \rightarrow a)$	(v) y (Lampert)
vii.	$\Longrightarrow Falso$	(ii), (iv) y (vii)

Se llega a una proposición falsa, por lo tanto el supuesto de que a y b no son concurrentes es falso:

$\Longrightarrow a \,||\, b$

- $\Longleftarrow$

i.	$a \,||\, b$	
ii.	$\Longrightarrow \neg(a \rightarrow b) \wedge \neg(b \rightarrow a) \wedge (b \neq a)$	(i)
iii.	$\Longrightarrow V[i] \geq W[i] \wedge a \in i$	(ii $\neg(a \rightarrow b)$) y Lampert
iv.	$\Longrightarrow W[j] \geq V[i] \wedge b \in j$	(ii $\neg(b \rightarrow a)$) y Lampert
v.	$\Longrightarrow V[i] > W[i] \wedge \; W[j] > V[i]$	(ii $(b \neq a)$) (iii) y (iv)
vi.	$\Longrightarrow \exists i, j \; tq \; V[j] < W[j] \wedge V[i] > W[i]$	(v)
	$\Longrightarrow$ V||W	(vi) y (4)

2. Dado un conjunto arbitrario de relojes vectoriales todos de las mismas dimensiones, podría haber uno o más relojes vectoriales que no pueden venir de la misma historia global que los otros. Dé al menos 2 ejemplos de relojes vectoriales con estas características

Si C es un conjunto de relojes vectoriales donde cada elemento es un vector v con índices i=1, …,n entradas, entonces casos en lo que uno o más vectores podrían no venir de la misma historia global que los otros, son los siguientes:

1. Si se tienen cuatro elementos u, v, w y z que pertenecen a C, tales que v[i] = u[i], y w[i] = z[i] = v[i]+1 = u[i]+1. Y además, no existe j ≠ i talque v[j] = w[j]-1, y u[j] = z[j] - 1.
   En este caso, al menos uno de los cuatro elementos no pertenece a la historia global de los demás elementos de C. Esto porque, dada la definición, solamente los vectores asociados a eventos que pertenecen al proceso i deberían formar una secuencia incremental en la entrada i de cada vector.

   Por ejemplo:

   Suponga que se tienen los siguientes vectores de dos dimensiones:

   A=[30] C=[31] E=[22]

   B=[20] D=[21] F=[01]

   Para i =1 y j=2, se tiene que los elementos A, B, C y D cumplen con las condiciones:

   B[1] = D[1] = 2

   A[1] = C[1] = B[1]+1 = D[1]+1 = 3

   B[2] ≠A[2]-1 ∧ D[2] ≠C[2]-1

   De esta manera, cumpliendo con la definición y observando los valores secuenciales, se tendría que A, B, C y D pertenecen a eventos del proceso 1. Sin embargo, esto no es posible pues tendremos dos opciones:

   - A y B pertenecen al proceso 1, mientras que C y D no forman parte de la historia global de los demás
   - C y D pertenecen a proceso 1, pero A y B no forman parte de la historia global de los demás vectores

2. Si existen al menos dos elementos de C, u y v, para los cuales existe un i talque v[i]=u[i]=1 y para todo j diferente de i se cumple que v[j]=u[j]=0. Entonces, uno de los dos vectores no corresponde a la historia global de los demás. En otras palabras, dadas las reglas 1 y 2 del algoritmo para relojes vectoriales, se pueden identificar elementos que no comparten la historia global de los demás si existe más de un vector que puede ser asociado al mismo evento, correspondiente al primero de un proceso. Esto porque los vectores de los eventos que ocurren primero en cada proceso deberían ser únicos.

Por ejemplo:
En el siguiente conjunto de vectores de 3 dimensiones:

A=[0,2,3]	D=[0,2,3]
B=[1,0,1]	E=[0,1,0]
C=[0,1,0]	F=[1,2,4]

Inmediatamente podemos inferir que ya sea C ó E no pertenecen a la misma historia global de los demás.

3. Investigue la técnica conocida como "relojes plausibles". Explique su propósito, implementación y casos donde son utilizables.

Los relojes plausibles corresponden a una solución propuesta por Francisco Torres-Rojas y Mustaque Ahamad cuyo propósito es lograr un alto nivel de precisión en el ordenamiento de eventos en sistemas distribuidos, tal como lo hacen los relojes vectoriales, pero sin los problemas de escalabilidad que estos últimos presentan. Los relojes plausibles logran esto debido a que el número de componentes necesarios en las estructuras de datos es independiente de la cantidad de nodos en el sistema distribuido, además pueden ser implementados para ordenar un número pequeño de eventos concurrentes sin afectar de forma significativa el rendimiento de los algoritmos que usan dichos relojes.

La idea de los relojes plausibles es utilizar un timestamp, el cual se define como una estructura que representa un instante en el tiempo observado por cierto nodo. De esta forma, si se tiene una historia global H en un sistema distribuido, un *Time Stamping System* (TSS) X es un par $\left(\langle S, \overset{X}{\rightarrow}\rangle, X.stamp\right)$ donde:

- S es un conjunto de timestamps
- $\overset{X}{\rightarrow}$ es una relación irreflexiva y transitiva definida sobre los elementos de S
- $\langle S, \overset{X}{\rightarrow}\rangle$ es un orden parcial estricto
- X.stamp es la función que mapea los timestamps de H a S.

También se definen tres relaciones adicionales. Sean v y w timestamps que pertenecen a a S, entonces:

- $v \underset{=}{\overset{X}{}} w \Leftrightarrow v = w$
- $v \overset{X}{\leftarrow} w \Leftrightarrow w \overset{X}{\rightarrow} v$
- $v \parallel w \Leftrightarrow \neg(v \underset{=}{\overset{X}{}} w) \wedge \neg(v \overset{X}{\rightarrow} w) \wedge \neg(v \overset{X}{\leftarrow} w)$

La relación causal expresada por la función X entre dos eventos a y b (timestamps X(a) y X(b)) no necesariamente es correcta. Sin embargo, se puede comprobar una definición equivalente a la propiedad fuerte del reloj donde se tiene que, un TSS X = $\left(\langle S, \overset{X}{\rightarrow}\rangle, X.stamp\right)$ determina causalidad si para todo a y b que pertenecen a H:

- $a = b \Leftrightarrow a \underset{=}{\overset{X}{}} b$
- $a \rightarrow b \Leftrightarrow a \overset{X}{\rightarrow} b$
- $a \parallel b \Leftrightarrow a \underset{\parallel}{\overset{X}{}} b$

Y de esta forma, se tiene que un TSS P = $\left(\langle S, \overset{P}{\rightarrow}\rangle, P.stamp\right)$ es plausible si para todo a,b que pertenecen a H:

- $a = b \Leftrightarrow a \underset{=}{\overset{P}{}} b$
- $a \rightarrow b \Longrightarrow a \overset{P}{\rightarrow} b$

En otras palabras, un TSS plausible satisface la condición débil del reloj. Además, si P es plausible, entonces para todo a y b que pertenece a H se cumple que $a \parallel b \Rightarrow a \overset{P}{\underset{\parallel}{}} b$

Para la implementación de relojes plausibles, los autores proponen tres grupos:

1. R-Entries Vector TSS (REV)

   Es una variante de los relojes vectoriales donde cada vector de un tamaño fijo R<=N. El valor de R es independiente del número de nodos en el sistema. En este caso, cada vector tiene menos entradas que el total de nodos, por lo que los nodos deberán compartir entradas en el vector. Esto se logra con una función de mapeo (Por ejemplo, módulo R). Los métodos para comparación de timestamps son prácticamente idénticos a los utilizados en el algoritmo de relojes vectoriales.

2. K-Lamport Clocks (KLA)

   Estos se implementan utilizando las mismas estructuras de datos que en REV (una identificación del nodo y un vector de enteros), sin embargo, se utilizan diferentes reglas para actualizar el vector. Suponga que se tiene un sistema distribuido con K nodos. Cada nodo contiene además un reloj de Lamport y una variable h con el máximo timestamp de entre todos aquellos mensajes que fueron recibidos por él mismo y por los K-2 nodos que anteriormente se hayan comunicado directa o indirectamente con el nodo actual.

   Lo anterior permite identificar si dos eventos a y b tienen una relación causal, de forma que, aunque el valor de los relojes de Lamport indique que sí hay relación, el nodo puede verificar el valor de h para comprobar si en efecto la relación causal entre a y b existe.

3. Combined TSS (Comb)

   Es una implementación que combina tanto REV como KLA de la siguiente forma:

   Se define Comb = $\left(\langle S, \overset{Comb}{\underset{\rightarrow}{}} \rangle, Comb.stamp\right)$, donde

   - S es un conjunto de elementos de la forma $(\langle i, V_i \rangle, \langle i, W_i \rangle)$ donde i es un entero que identifica cada nodo en el sistema distribuido, V_i es un vector de enteros de R dimensiones y W_i es un vector de enteros de K dimensiones.
   - Para todo a que pertenece a una historia global H:
     Comb.stamp(a) = (REV.stamp(a), KLA.stamp(a))

Los casos en los cuales los relojes plausibles pueden ser utilizados, radican en aquellas aplicaciones donde el hecho de imponer un orden sobre un par de eventos concurrentes no va a afectar la exactitud de la aplicación. Algunos de estos casos son:

- **Medidas de concurrencia:** Permiten establecer qué tan concurrente es un sistema, de forma que se toman en cuenta las relaciones causales entre los eventos que pertenecen a una historia finita y cuántos eventos pueden estarse ejecutando concurrentemente en un instante dado.
- **Uso compartido de recursos distribuidos:** Al igual que Lamport al demostrar que los relojes lógicos pueden utilizarse para resolver un problema de sincronización para manejo de recursos

compartidos en un sistema, se propone otro algoritmo (más eficiente que el de Lamport) para garantizar exclusión mutua en el acceso a recursos mediante el uso de relojes plausibles.

- **Consistencia de objetos:** Se pueden utilizar relojes plausibles para garantizar consistencia cuando la información compartida en un sistema distribuido es encapsulada en objetos, los cuales deben replicarse y guardarse en cache para lograr disponibilidad y rendimiento.

Referencias

Fidge, C. J. (1988), 'Timestamps in message-passing systems that preserve the partial ordering', Proceedings of the 11th Australian Computer Science Conference 10 (1) , 56–66 .

Lamport, L. (1978), 'Time, clocks, and the ordering of events in a distributed system', Commun. ACM 21 (7) , 558--565 .

Torres-Rojas, F. J.; Ahamad, M. & Raynal, M. (1999), Timed consistency for shared distributed objects, in 'PODC '99: Proceedings of the eighteenth annual ACM symposium on Principles of distributed computing' , ACM, New York, NY, USA , pp. 163--172 .

Torres-Rojas, F. J. & Ahamad, M. (1999), 'Plausible Clocks: Constant Size Logical Clocks for Distributed Systems.', Distributed Computing 12 (4) , 179-195 .