Computación distribuida.

Aplicaciones de Relojes Vectoriales.

Integrantes:
Aguilera Moreno Adrian
Torres Valencia Kevin Jair
Pérez Romero Natalia Abigail

Facultad de Ciencias, UNAM

Tabla de contenido.

Los relojes vectoriales son un tipo de reloj lógico propuesto de manera independiente por *Colin J. Fidge* y *Friedemann Mattern* en 1988.

Esta técnica consiste en un mapeo entre eventos en una historia distribuida y vectores enteros.

Definiciones: Vector Tiempo.

Sistema Vectorial de Relojes. Es un mecanismo capaz de caracterizar estados locales (en adelante, eventos) en un sistema distribuido, asociando un valor vectorial a cada estado.

Tiempo Vectorial. Es la noción de tiempo capturada por los relojes vectoriales.

Caracterización Formal del Tiempo Vectorial. Sea `date(e)` la caracterización asociada a un evento `e`, de tal manera que se cumple:

1. $\forall_{e_1,e_2} : (e_1 \rightarrow e_2) \Leftrightarrow \texttt{date}(e_1) < \texttt{date}(e_2)$.
2. $\forall_{e_1,e_2} : (e_1 \mid\mid e_2) \Leftrightarrow \texttt{date}(e_1) \mid\mid \texttt{date}(e_2)$.

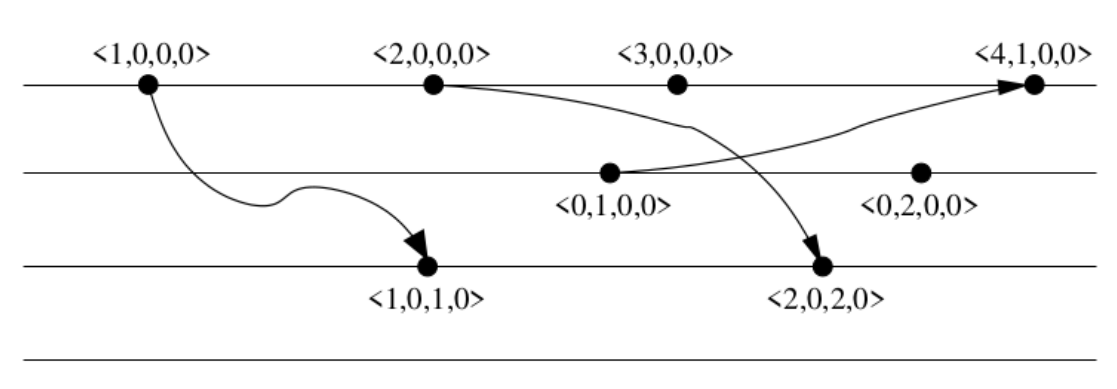


Figure: Reloj Vectorial con tiempos locales.

Reloj Vectorial. La implementación del tiempo vectorial requiere que cada proceso, en el sistema, mantenga un vector de enteros positivos $Vc_i[1, \cdots, n]$ con valores inicialmente $[0, \cdots, 0]$. Este vector debe cumplir con

1. $Vc_i[i]$ cuenta el número de eventos producidos por p_i.
2. $Vc_i[j], j \neq i$, nos dice cuántos eventos conoce p_i producido por p_j.

De manera formal, sea e un evento producido por p_i, tenemos que

$$Vc_i[k] = |\{f | (f \text{ se produjo por } p_k) \wedge (f \rightarrow e)\}| + 1(k, i).$$

Reloj Vectorial: Algoritmo.

Una primera aproximación. Inicialmente todos los procesos disponen de un vector con entradas igual al número total de procesos, este vector debe estar inicializado en 0 para cada entrada. A continuación se describe el algoritmo:

- Si p_i produce un evento, entonces:
  (1) $Vc_i[i] \leftarrow Vc_i[i] + 1$;
  (2) Produce un evento e caracterizado por $Vc_i[1, \cdots, n]$.
- Cuando p_i envia un mensaje a p_j, entonces:
  (3) $Vc_i[i] \leftarrow Vc_i[i] + 1$;
  (4) `send(⟨msj,` Vc_i`[1, ···, n] ⟩)` a p_j.
- Cuando p_j recibe un mensaje, entonces:
  (3) $Vc_j[j] \leftarrow Vc_j[j] + 1$;
  (4) $Vc_j[1, \cdots, n] \leftarrow \forall_{k \in [1, \ldots, n]} \texttt{max}(Vc_i[k], Vc_j[k])$.

Algoritmo: Propagación del tiempo vectorial.

Notación: Para, cualesquiera, dos vectores Vc_1 y Vc_2 del tamaño n. Tenemos que

- $Vc_1 \leq Vc2 =_{def.} \left(\forall_{k \in \{1,\cdots,n\}} : Vc_1[k] \leq Vc_2[k]\right)$;
- $Vc_1 < Vc2 =_{def.} (Vc_1[k] \leq Vc_2[k]) \wedge (Vc_1[k] \neq Vc_2[k])$;
- $Vc_1 || Vc_2 =_{def} \neg(Vc_1 \leq Vc_2) \wedge \neg(Vc_2 \leq Vc_1)$.

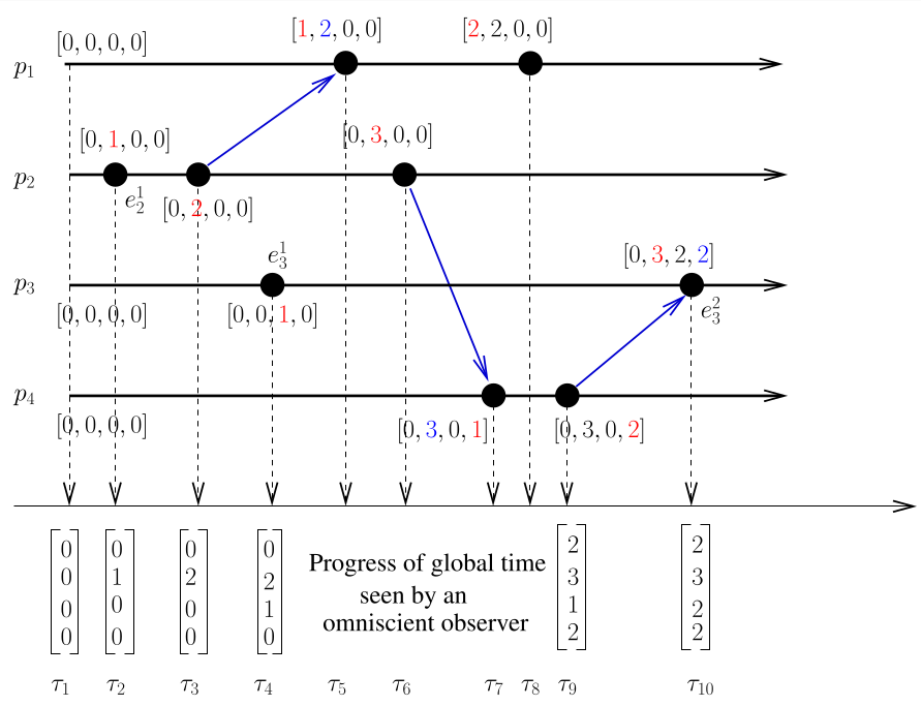


Figure: Ejemplo de propagación en un reloj Vectorial.

Def. Sea $e.Vc$ el vector asociado al evento e.

Teo 1. Por el algoritmo mencionado tenemos que, para cualesquiera e_1 y e_2 distintos tenemos que

a) $(e_1 \to e_2) \Leftrightarrow (e_1.Vc < e_2.Vc)$;

b) $(e_1 || e_2) \Leftrightarrow (e_1.Vc || e_2.Vc)$.

Cor 1. Dadas dos caracterizaciones a eventos (fechas), determinar si estos eventos están relacionados o no, puede requerir hasta n comparaciones de enteros.

Teo 2. Sean dos eventos e_1 y e_2 con tuplas $\langle e_1.Vc, i\rangle$ y $\langle e_2.Vc, j\rangle$ de manera respectiva y $i \neq j$. Entonces

a) $(e_1 \to e_2) \Leftrightarrow (e_1.Vc[i] \leq e_2.Vc[i])$;

b) $(e_1 || e_2) \Leftrightarrow ((e_1.Vc[i] > e_2.Vc[i]) \wedge (e_2.Vc[j] > e_1.Vc[j]))$.

Algoritmo: Reducción de costo en la comparación de dos vectores.

Mejora en la complejidad en tiempo. Hasta el momento la complejidad en tiempo para combinar 2 eventos nos toma $\mathcal{O}(n)$, con n el número de procesos en el sistema.

Por el *Teo. 2*, sabemos que basta con verificar dos entradas para saber como es un evento respecto al otro. Así, basta comparar dos entradas para combinar la caracterización de 2 eventos esto nos toma $\mathcal{O}(1)$.

Algoritmo: Relación del tiempo vectorial y estados globales.

Consideremos

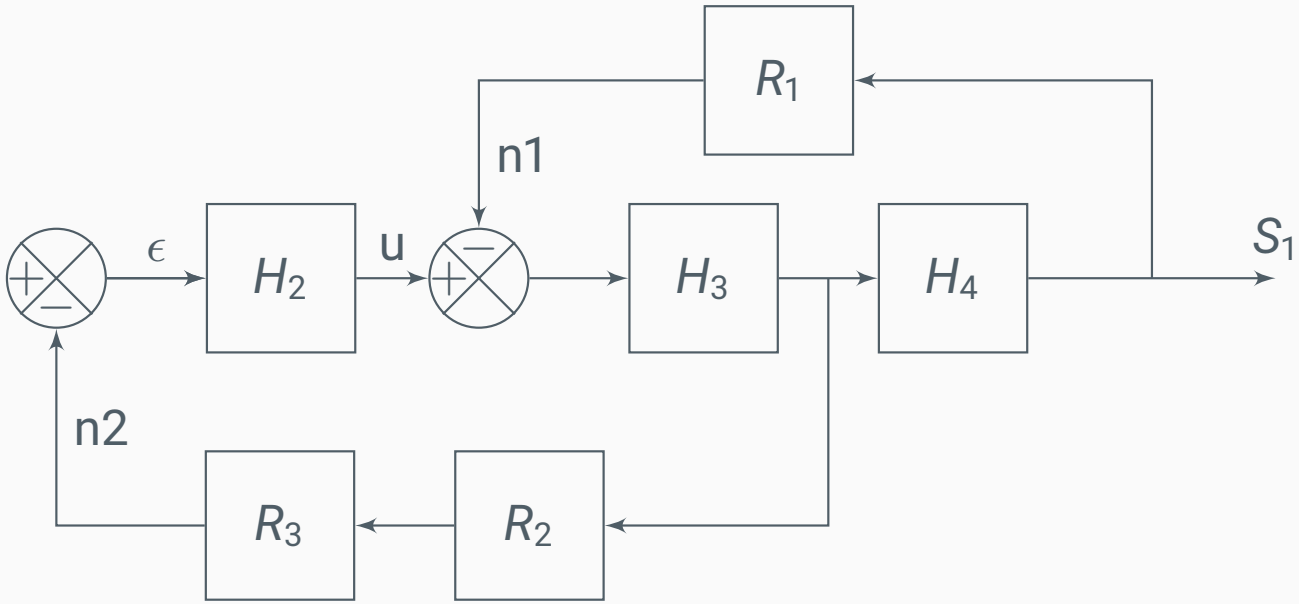
Relojes Vectoriales: Desventajas.

Esta mejora a los relojes lógicos de Lamport tiene un problema de implementación, que en un momento será más evidente.

Desventaja

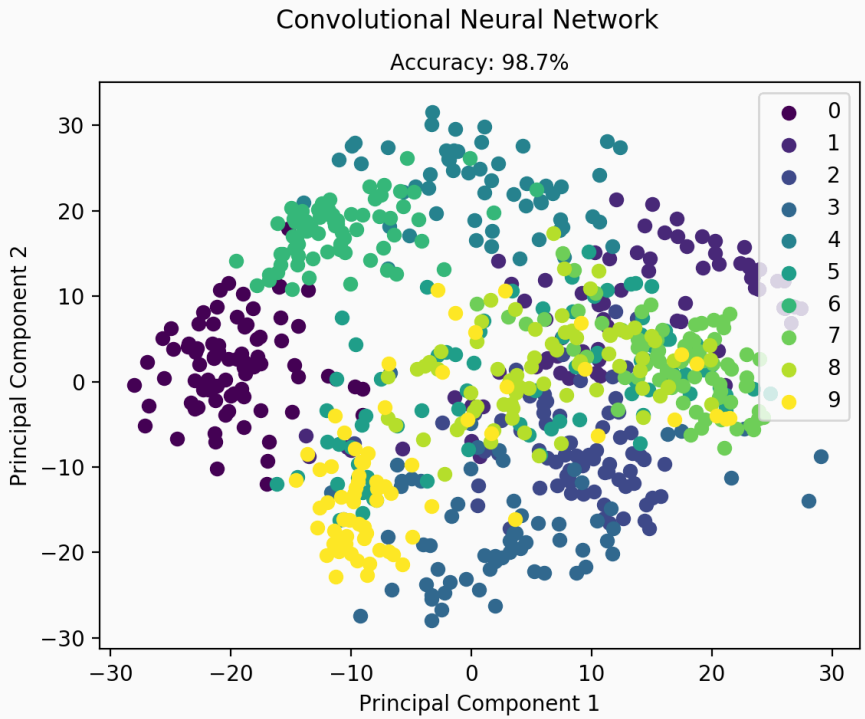
Esta desventaja es que cada proceso tiene que cargar con espacio igual al número de procesos en el sistema y cada intercambio entre eventos es de este tamaño.

Flux diagrams



Bloc diagram example from **texample.net**

Flux plots



Binary Softmax classifier

$$\sigma(\sum_i w_i x_i + b)$$

Loss function

$$L_i = -log(\frac{e^{f_{y_i}}}{\sum_j e^{f_j}})$$

cross entropy

This is a plain frame.
Use it to display full page images.

References I

Flux license

Flux is licensed under GNU General Public License v3.

http://www.gnu.org/licenses

Inspired by **Metropolis** theme from Matthias Vogelgesang.
https://github.com/matze/mtheme