Distributed Computing Module 2 -Lecture 2 & 3

Mahesh C.
Centre for High Performance Computing
FISAT

Scalar Time - Proposed by Lamport in 1978

- Time domain in this representation is the set of non-negative integers.
- The logical local clock of a process p_i and its local view of the global time are squashed into one integer variable Ci

Rules

- R1: Before executing an event (send, receive, or internal), process pi executes the following:

$$C_i := C_i + d \quad \text{where } (d > 0)$$

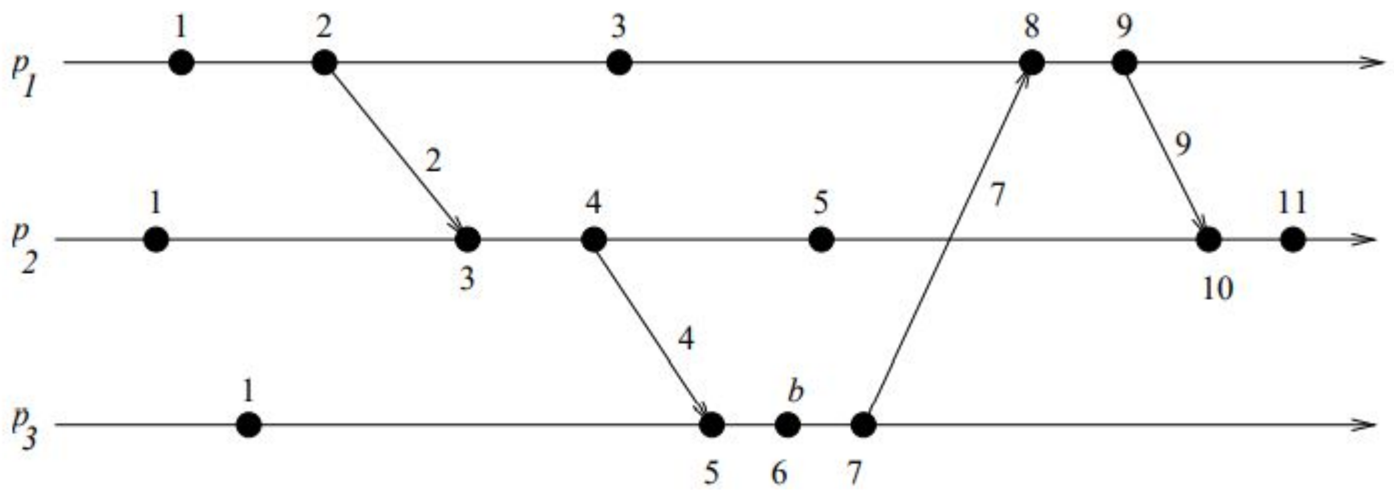
Scalar Time - Proposed by Lamport in 1978

- R2: Each message piggybacks the clock value of its sender at sending time. When a process p_i receives a message with timestamp C_{msg}, it executes the following actions:

  1) $C_i := \max(C_i , C_{msg})$

  2) Execute R1.

Scalar Time - Evolution of time in Distributed System



Scalar Time - Basic Properties

- Consistency Property

Clearly, scalar clocks satisfy the monotonicity and hence the consistency property:

for two events e_i and e_j, $e_i \rightarrow e_j =: \mathrm{C}(e_i) < \mathrm{C}(e_j)$.

Scalar Time - Basic Properties

Total Ordering

- Scalar clocks can be used to totally order events in a distributed system.
- The main problem in totally ordering events is that two or more events at different processes may have identical timestamp.
- A tie-breaking mechanism is needed to order such events. A tie is broken as follows:

Process identifiers are linearly ordered and tie among events with identical scalar timestamp is broken on the basis of their process identifiers.

The lower the process identifier in the ranking, the higher the priority.

The timestamp of an event is denoted by a tuple (t, i) where t is its time of occurrence and i is the identity of the process where it occurred.

The total order relation $\prec$ on two events x and y with timestamps (h,i) and (k,j), respectively, is defined as follows:

$$x \prec y \Leftrightarrow (h < k \textbf{ or } (h = k \textbf{ and } i < j))$$

Scalar Time - Basic Properties

Event counting

- If the increment value d is always 1, the scalar time has the following interesting property: if event e has a timestamp h, then h-1 represents the minimum logical duration, counted in units of events, required before producing the event e.
- We call it the height of the event e.
- In other words, h-1 events have been produced sequentially before the event e regardless of the processes that produced these events.

Scalar Time - Basic Properties

No Strong Consistency

- The system of scalar clocks is not strongly consistent; that is, for two events e_i and e_j , if $C(e_i) < C(e_j)$ does not always guarantee that e_i is happened before e_j

Vector Time

- The system of vector clocks was developed independently by Fidge, Mattern and Schmuck
- In the system of vector clocks, the time domain is represented by a set of n-dimensional non-negative integer vectors
- Each process p_i maintains a vector vt_i [1..n], where vt_i [i] is the local logical clock of p_i and describes the logical time progress at process p_i .

Vector Time

- $vt_i\,[j]$ represents process p_i's latest knowledge of process p_j local time.
- If $vt_i\,[j]=x$, then process p_i knows that local time at process p_j has progressed till x.

Rules

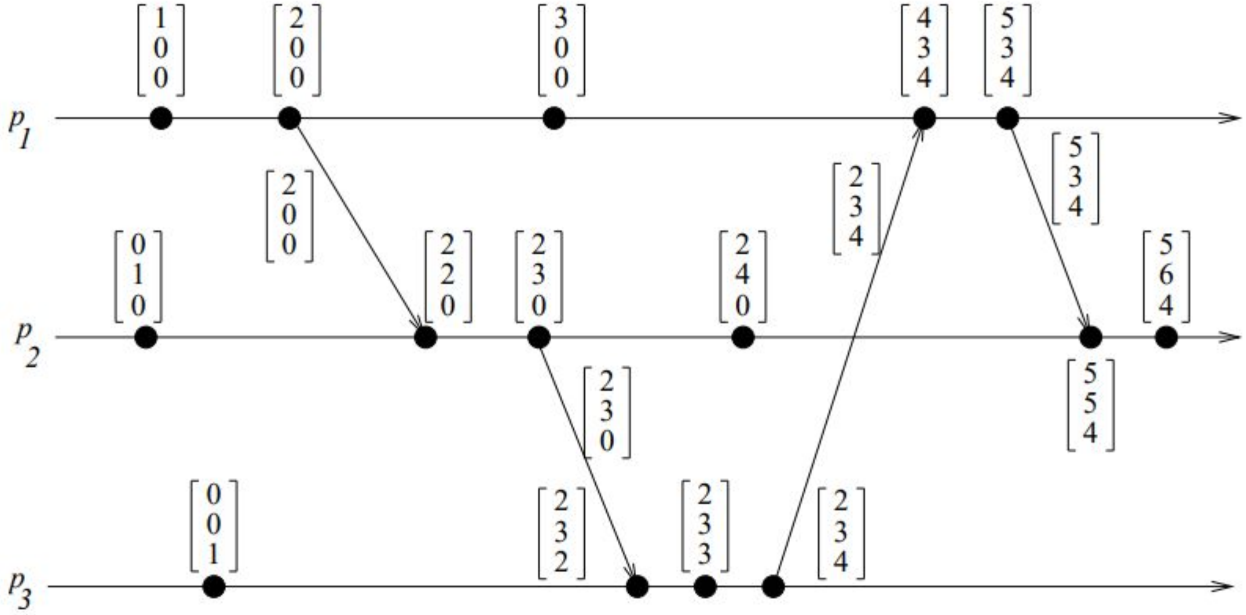
R1: Before executing an event, process p_i updates its local logical time as follows:

$$vt_i\,[i] := vt_i\,[i] + d \quad \text{where } (d > 0)$$

R2: Each message m is piggybacked with the vector clock vt of the sender process at sending time. On the receipt of such a message (m,vt), process pi executes the following sequence of actions .

1) $1 \leq k \leq n : \; vt_i[k] := \max(vt_i[k], vt[k])$
2) Execute R1

Vector Time - Evolution of vector time



Vector Time - Basic Properties

Isomorphism

- If events in a distributed system are time stamped using a system of vector clocks, we have the following property. If two events x and y have timestamps vh and vk, respectively, then

$$x \rightarrow y \quad \Leftrightarrow \quad vh < vk$$
$$x \parallel y \quad \Leftrightarrow \quad vh \parallel vk.$$

- Thus, there is an isomorphism between the set of partially ordered events produced by a distributed computation and their vector timestamps.

Vector Time - Basic Properties

Strong Consistency

- The system of vector clocks is strongly consistent; thus, by examining the vector timestamp of two events, we can determine if the events are causally related.

Event Counting

- If d=1 (in rule R1), then the i th component of vector clock at process p_i , vt_i [i], denotes the number of events that have occurred at p_i until that instant.