CS 3551 DISTRIBUTED COMPUTING

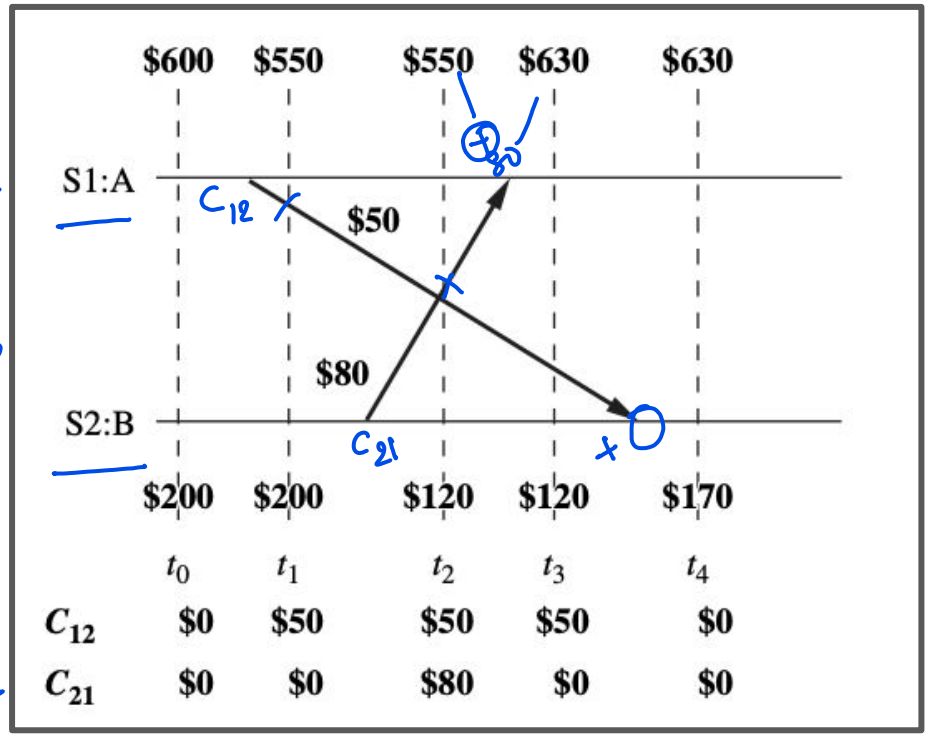
UNIT II LOGICAL TIME AND GLOBAL STATE 10

Logical Time: Physical Clock Synchronization: NTP – A Framework for a System of Logical Clocks – Scalar Time – Vector Time; Message Ordering and Group Communication: Message Ordering Paradigms – Asynchronous Execution with Synchronous Communication – Synchronous Program Order on Asynchronous System – Group Communication – Causal Order – Total Order; Global State and Snapshot Recording Algorithms: Introduction – System Model and Definitions – Snapshot Algorithms for FIFO Channels.

Global State - **Overall state of the distributed system**

- Why finding global state is difficult?
  - No shared memory between systems.
  - No common global clock.
  - Difficult to synchronize different local clocks.
- How global state is computed?(Process states+ Channel States)
  - Process state - Local Memory + Historical Activity
  - Channel state - Msg sent/received
  - **All these should be computed at same time instant**
- What is the need of global states?
  - Deadlock Detection
  - Termination
  - Failure Recovery
  - Debugging

What happens if state is computed at different time instant?



A - 600
B - 200
800

A $\xrightarrow{50}$ B

B $\xrightarrow{80}$ A

① A - 600 - 50 = 550

B - 200 + 50 = 250

800

② B - A 250 - 80 = 170

A → 550 + 80 = 630

800

$t_0 \Rightarrow 600 - A$

$t_2 \Rightarrow B - 120$

$C_{12} - 50$

$C_{21} - 80$

$ 800

$t_3 - A - 630$

$B - 120$

$c_{12} - 50$

$c_{21} - 0$

800

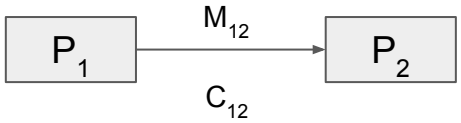
LIKE

COMMENT

SHARE

SUBSCRIBE

System model



- The system consists of a collection of n processes p_1, p_2, ..., p_n that are connected by channels.
- There are no globally shared memory and physical global clock and processes communicate by passing messages through communication channels.
- C_{ij} denotes the channel from process p_i to process p_j and its state is denoted by SC_{ij}.
- The actions performed by a process are modeled as three types of events: Internal events,the message send event and the message receive event.
- For a message m_{ij} that is sent by process p_i to process p_j, let $send(m_{ij})$ and $rec(m_{ij})$ denote its send and receive events.

System model

P_1 :- $e_1 e_2 e_4 e_8 e_{10}$ LS

Chennai - Jny - Mad - Jir - Kany

- At any instant, the state of process p_i, denoted by LS_i, is a result of the sequence of all the events executed by p_i till that instant.
- For an event e and a process state LS_i, $e \in LS_i$ iff e belongs to the sequence of events that have taken process p_i to state LS_i.
- For an event e and a process state LS_i, $e \notin LS_i$ iff e does not belong to the sequence of events that have taken process p_i to state LS_i.
- For a channel C_{ij}, the following set of messages can be defined based on the local states of the processes p_i and p_j

Kany

$p_i \rightarrow p_j$

Transit: $transit(LS_i, LS_j) = \{m_{ij} \mid send(m_{ij}) \in LS_i \bigwedge rec(m_{ij}) \notin LS_j\}$

Models of communication

Recall, there are three models of communication: FIFO, non-FIFO, and Co.

- In FIFO model, each channel acts as a first-in first-out message queue and thus, message ordering is preserved by a channel.
- In non-FIFO model, a channel acts like a set in which the sender process adds messages and the receiver process removes messages from it in a random order.
- A system that supports causal delivery of messages satisfies the following property: "For any two messages m_{ij} and m_{kj}, if $send(m_{ij}) \longrightarrow send(m_{kj})$, then $rec(m_{ij}) \longrightarrow rec(m_{kj})$".

Consistent global state

C1 Sent ✓ ⇒ channel or receiver

① $P_1 \xrightarrow{m_{12}} P_2$ Send ✓

C2 − receive + sent

② $P_1 \rightarrow P_2$ m_{12}

- The global state of a distributed system is a collection of the local states of the processes and the channels.
- Notationally, global state GS is defined as,

$$GS = \{\bigcup_i LS_i, \bigcup_{i,j} SC_{ij}\}$$

- A global state GS is a *consistent global state* iff it satisfies the following two conditions :

① C1: $\text{send}(m_{ij}) \in LS_i \Rightarrow m_{ij} \in SC_{ij} \oplus \text{rec}(m_{ij}) \in LS_j$. ($\oplus$ is Ex-OR operator.)

② C2: $\text{send}(m_{ij}) \notin LS_i \Rightarrow m_{ij} \notin SC_{ij} \wedge \text{rec}(m_{ij}) \notin LS_j$.

Cuts- A cut is a line joining an arbitrary point on each process line that slices the space–time diagram into a PAST and a FUTURE

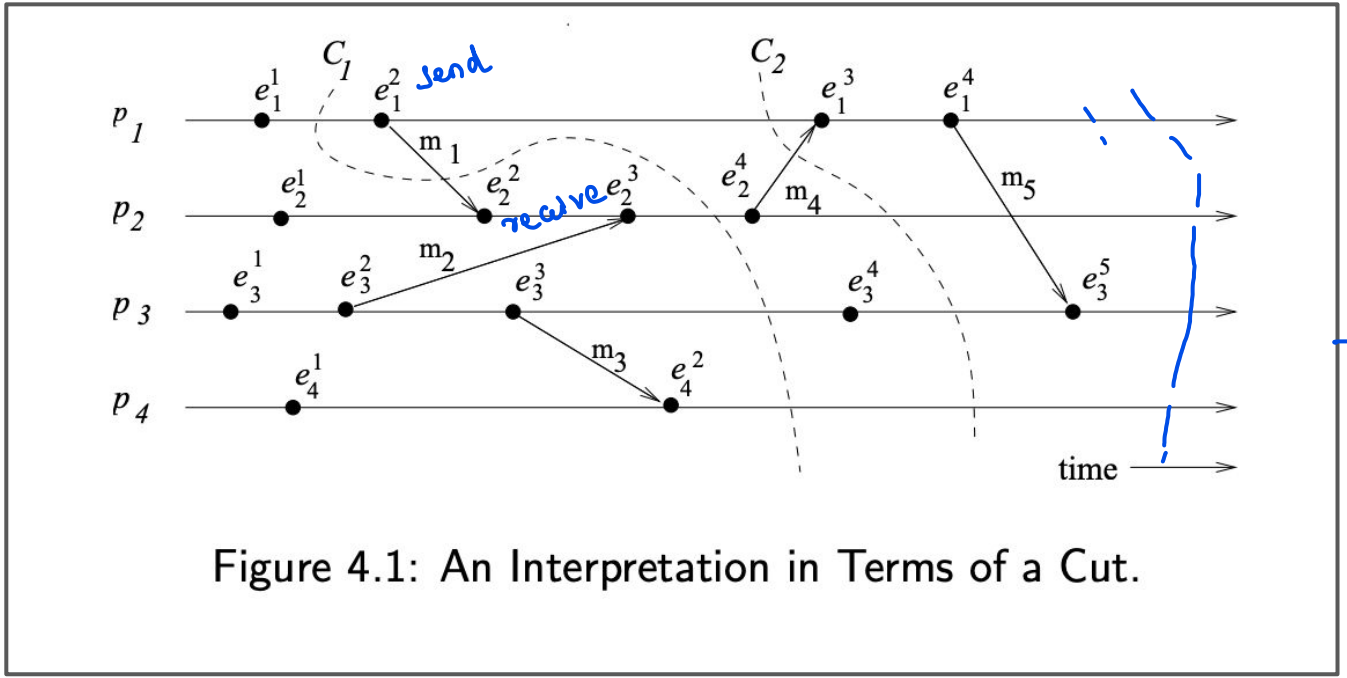


Figure 4.1: An Interpretation in Terms of a Cut.

C_1 - consistent ✗

C_2

$m_4 \in C_{21}$

Key Points

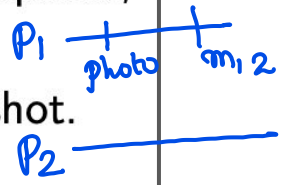
- A cut in a space-time diagram is a line joining an arbitrary point on each process line that slices the space-time diagram into a PAST and a FUTURE.
- A consistent global state corresponds to a cut in which every message received in the PAST of the cut was sent in the PAST of that cut.
- Such a cut is known as a *consistent cut*.
- For example, consider the space-time diagram for the computation illustrated in Figure 4.1.
- Cut C1 is inconsistent because message m1 is flowing from the FUTURE to the PAST.
- Cut C2 is consistent and message m4 must be captured in the state of channel C_{21}.

Issues in recording a global state

The following two issues need to be addressed:

I1: How to distinguish between the messages to be recorded in the snapshot from those not to be recorded.

-Any message that is sent by a process before recording its snapshot, must be recorded in the global snapshot (from **C1**).
-Any message that is sent by a process after recording its snapshot, must not be recorded in the global snapshot (from **C2**).



I2: How to determine the instant when a process takes its snapshot.

-A process p_j must record its snapshot before processing a message m_{ij} that was sent by process p_i after recording its snapshot.

LIKE

COMMENT

SHARE

SUBSCRIBE

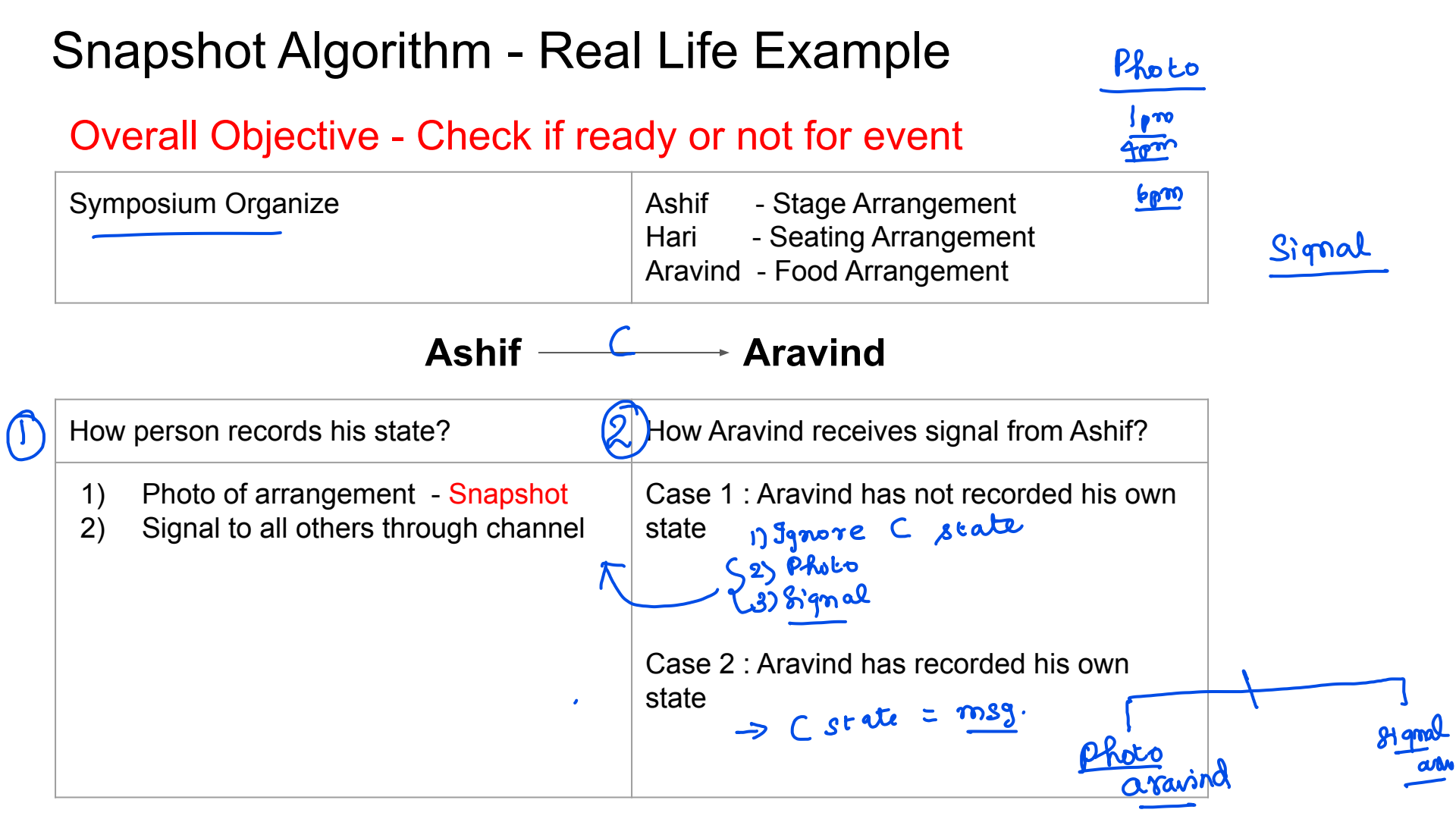
Snapshot Algorithm - Real Life Example

Overall Objective - Check if ready or not for event

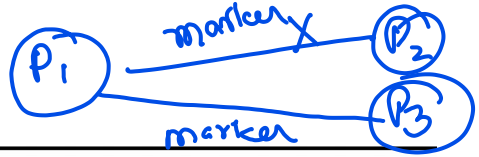
Symposium Organize	Ashif - Stage Arrangement Hari - Seating Arrangement Aravind - Food Arrangement

Ashif ——C——→ **Aravind**

(1) How person records his state?	(2) How Aravind receives signal from Ashif?
1) Photo of arrangement - Snapshot 2) Signal to all others through channel	Case 1 : Aravind has not recorded his own state Case 2 : Aravind has recorded his own state



Chandy Lamport Algorithm



(1) **Marker Sending Rule** for process i

1. Process i records its state.
2. For each outgoing channel C on which a marker has not been sent, i sends a marker along C before i sends further messages along C.

(2) **Marker Receiving Rule** for process j

On receiving a marker along channel C:

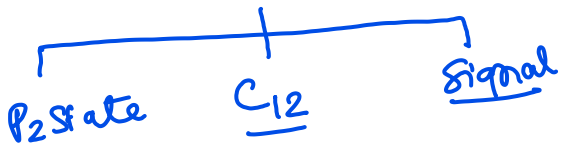
if j has not recorded its state **then**

Record the state of C as the empty set

Follow the "Marker Sending Rule"

else

Record the state of C as the set of messages received along C after j's state was recorded and before j received the marker along C

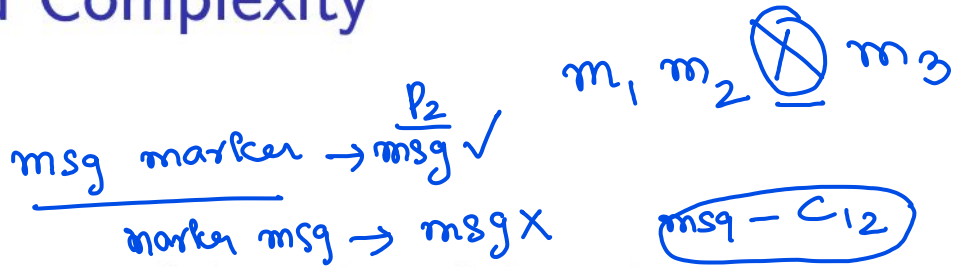


Key Points

- The Chandy-Lamport algorithm uses a control message, called a *marker* whose role in a FIFO system is to separate messages in the channels.
- After a site has recorded its snapshot, it sends a *marker*, along all of its outgoing channels before sending out any more messages.
- A marker separates the messages in the channel into those to be included in the snapshot from those not to be recorded in the snapshot.
- A process must record its snapshot no later than when it receives a marker on any of its incoming channels.

- The algorithm can be initiated by any process by executing the "Marker Sending Rule" by which it records its local state and sends a marker on each outgoing channel.
- A process executes the "Marker Receiving Rule" on receiving a marker. If the process has not yet recorded its local state, it records the state of the channel on which the marker is received as empty and executes the "Marker Sending Rule" to record its local state.
- The algorithm terminates after each process has received a marker on all of its incoming channels.
- All the local snapshots get disseminated to all other processes and all the processes can determine the global state.

Correctness and Complexity



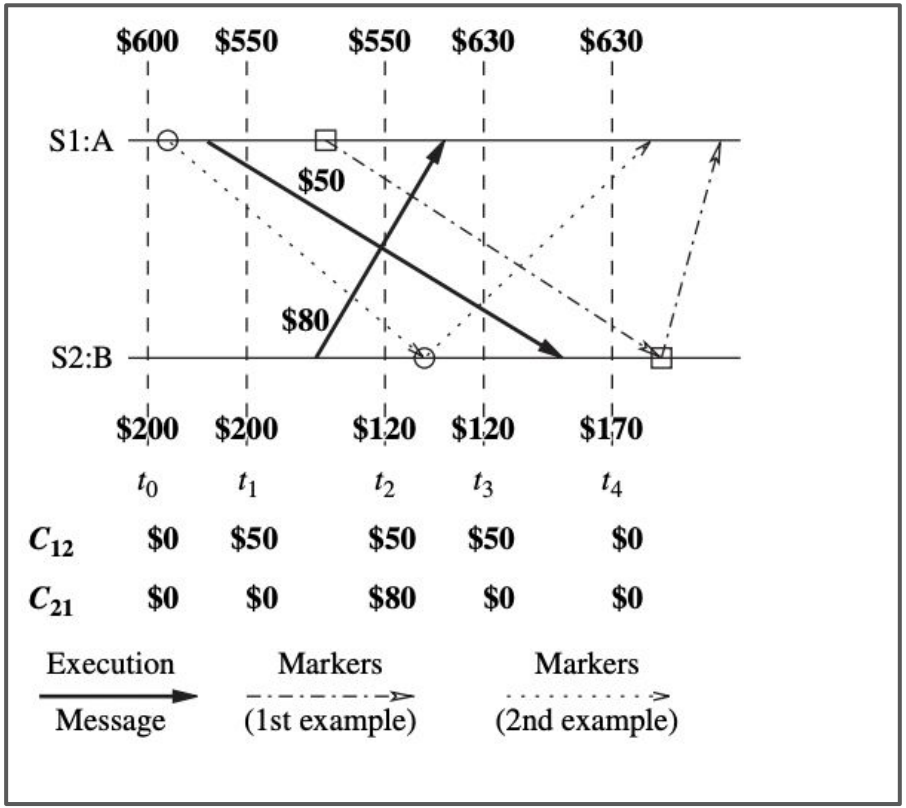
Correctness

- Due to FIFO property of channels, it follows that no message sent after the marker on that channel is recorded in the channel state. Thus, condition **C2** is satisfied.
- When a process p_j receives message m_{ij} that precedes the marker on channel C_{ij}, it acts as follows: If process p_j has not taken its snapshot yet, then it includes m_{ij} in its recorded snapshot. Otherwise, it records m_{ij} in the state of the channel C_{ij}. Thus, condition **C1** is satisfied.

Complexity

- The recording part of a single instance of the algorithm requires $O(e)$ messages and $O(d)$ time, where e is the number of edges in the network and d is the diameter of the network.

Properties of Recorded Global State



□ → A - 550 ✓
C_{AB} - 0
B → 170
C_{BA} → marker
C_{BA} → 80

800$

○ → A - 600
C_{AB} - 0
B → 120
C_{BA} - marker
C_{BA} - 80

800

Explanation of Example

1. (Markers shown using dashed-and-dotted arrows.) Let site S1 initiate the algorithm just after t_1. Site S1 records its local state (account A = \$550) and sends a marker to site S2. The marker is received by site S2 after t_4. When site S2 receives the marker, it records its local state (account B = \$170), the state of channel C_{12} as \$0, and sends a marker along channel C_{21}. When site S1 receives this marker, it records the state of channel C_{21} as \$80. The \$800 amount in the system is conserved in the recorded global state,

$$A = \$550, B = \$170, C_{12} = \$0, C_{21} = \$80.$$

2. (Markers shown using dotted arrows.) Let site S1 initiate the algorithm just after t_0 and before sending the \$50 for S2. Site S1 records its local state (account A = \$600) and sends a marker to site S2. The marker is received by site S2 between t_2 and t_3. When site S2 receives the marker, it records its local state (account B = \$120), the state of channel C_{12} as \$0, and sends a marker along channel C_{21}. When site S1 receives this marker, it records the state of channel C_{21} as \$80. The \$800 amount in the system is conserved in the recorded global state,

$$A = \$600, B = \$120, C_{12} = \$0, C_{21} = \$80.$$

Properties of the recorded global state

- The recorded global state may not correspond to any of the global states that occurred during the computation.
- This happens because a process can change its state asynchronously before the markers it sent are received by other sites and the other sites record their states.
  - But the system could have passed through the recorded global states in some equivalent executions.
  - The recorded global state is a valid state in an equivalent execution and if a stable property (i.e., a property that persists) holds in the system before the snapshot algorithm begins, it holds in the recorded global snapshot.
  - Therefore, a recorded global state is useful in detecting stable properties.

LIKE

COMMENT

SHARE

SUBSCRIBE