CS3.401 Distributed Systems

Name- Vilal Ali

Roll Number: 2024701027

2024701027_Homework_1

Question 1. Consider a system with four processors and a distributed program with about 15 events.

Question 1 (a) - Draw the time diagram for such a program and mark the scalar time of the events. The set of events should include all possible events.

Answer:

Let's consider a distributed system of four processes (P1, P2, P3, P4) with 15 events. Scalar timestamps are assigned to each event using Lamport's logical clocks, where each event within a process receives a timestamp that increases in a sequential manner.

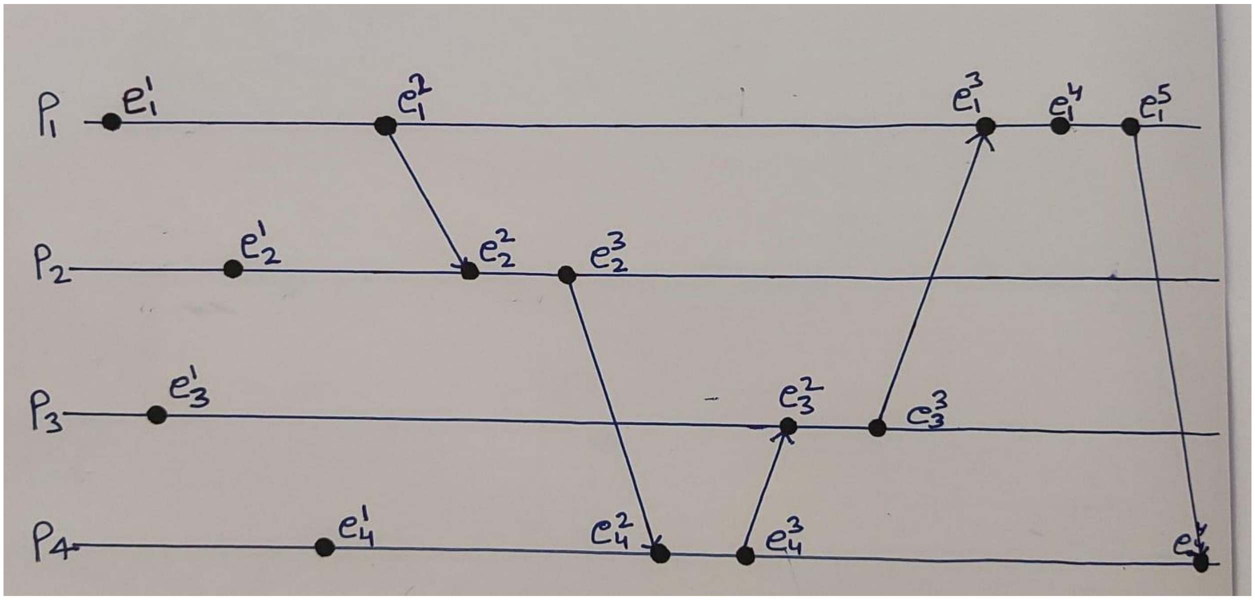

Figure 1.0 The space-time diagram of a distributed execution.

Scalar Time:

Below is the time space diagram created to represent this. Here we are set the increment d to 1 always.

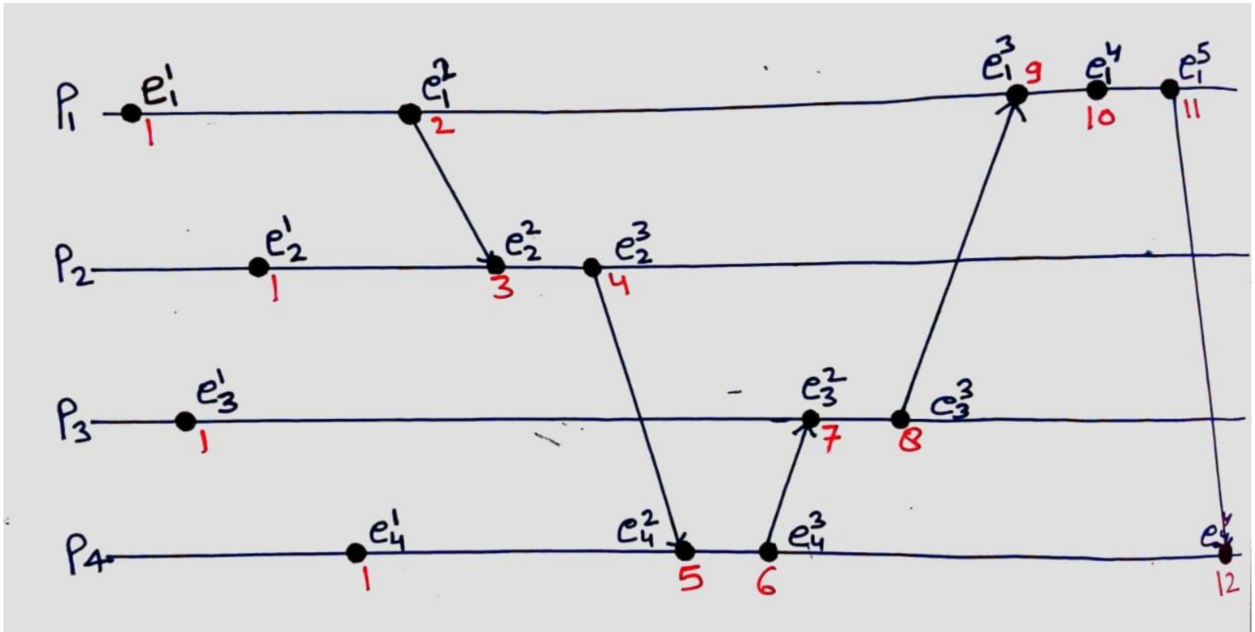

Figure 1.1: The space-time diagram of a distributed execution.

Question 1 (b) - **Identify events that are logically concurrent:**

Answer:

- $\mathbf{e^x}_{\mathbf{i}}$: Represents an event x on process P_i .
- $\mathbf{e^y}_{\mathbf{j}}$: Represents an event y on process P_j.

Two events $\mathbf{e^x}_{\mathbf{i}}$ and $\mathbf{e^y}_{\mathbf{j}}$ are logically concurrent if there is no path (causal relationship) between them.

Logically Concurrent Events Are:

- $e^1{}_1 \,||\, e^1{}_2$, $e^1{}_1 \,||\, e^1{}_3$, $e^1{}_1 \,||\, e^1{}_4$
- $e^1{}_2 \,||\, e^1{}_3$, $e^1{}_2 \,||\, e^1{}_4$
- $e^1{}_3 \,||\, e^1{}_4$, $e^2{}_3 \,||\, e^1{}_4$, $e^2{}_3 \,||\, e^4{}_2$, $e^3{}_3 \,||\, e^4{}_2$

Question 1 (c) - Identify events where strong monotonicity of the scalar time fails.

Answer: Scalar time should preserve the "happens-before" relation (→), meaning that if $\mathbf{e_i \rightarrow e_j}$ then $\mathbf{C(e_i) < C(e_j)}$.

In the diagram, strong monotonicity of scalar time could fail if there is any sequence where an event e^x_i, occurs after another event e_j^y, but C(e^x_i) is less than C(e^y_j). Without actual clock values provided, we can analyze the sequence of events visually:

From Figure (1.1), all the direct causality links (arrows) maintain the order of events without causing any strong monotonicity failures.

Question 2 (a): Draw the time diagram for such a program and mark the vector **time of the events. The set of events should include all possible events.**

Answer: Vector time ensures that the "happens-before" relation is preserved across all processes with vector clocks:

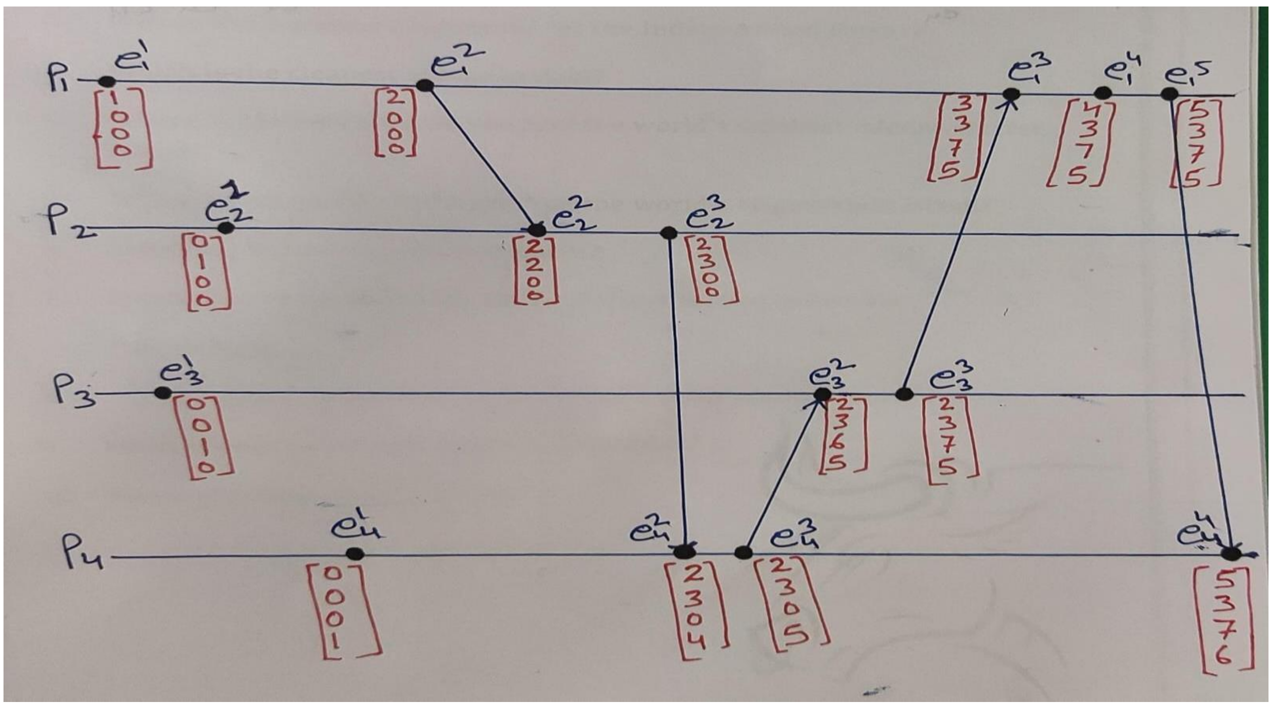

Figure 2.0: The vector space-time diagram of a distributed execution.

- **Initially, the vector clock is set to [0, 0, 0, ..., 0] and d = 1.**
- **Assign timestamps to all the events shown in the diagram:**

For each event, we have the following vector timestamps:

P1	P2	P3	P4
e^1_1 P_1: [1,0,0,0]	e^1_2 P_2: [0,1,0,0]	e^1_3 P_3: [0,0,1,0]	e^1_4 P_4: [0,0,0,1]
e^2_1 P_1: [2,0,0,0]	e^2_2 P_2: [2,2,0,0]	e^2_3 P_3: [2,3,6,5]	e^2_4 P_4: [2,3,0,4]
e^3_1 P_1: [3,3,7,5]	e^3_2 P_2: [2,3,6,5]	e^3_3 P_3: [3,3,7,5]	e^3_4 P_4: [2,3,0,5]
e^4_1 P_1: [4,3,7,5]			e^4_4 P_4: [5,3,7,6]

e^5_1 P_1: [5,3,7,5]			

Table: 1.0: Given Events and Vector Timestamps

Question 2 (b): Identify events that are logically concurrent with respect to vector time.

Answer:

To identify logically concurrent events in the second image, we need to determine pairs of events that have no causal relationship. This means there is no direct or indirect "happens-before" relationship between them.

In the context of vector clocks, two events are considered logically concurrent if their vector timestamps are not comparable, meaning that neither event's timestamp is a component-wise greater than or equal to the other event's timestamp.

Identifying Logically Concurrent Events:

Process P_1:

- e^1_1 is logically concurrent with:
    - e^1_2 [0, 1, 0, 0] (P_2)
    - e^1_3 [0, 0, 1, 0] (P_3)
    - e^1_4 [0, 0, 0, 1] (P_4)
- e^2_1 is logically concurrent with:
    - e^1_2 [0, 1, 0, 0] (P_2)
    - e^1_3 [0, 0, 1, 0] (P_3)
    - e^1_4 [0, 0, 0, 1] (P_4)
- e^3_1 is logically concurrent with:
    - e^2_2 [2, 2, 0, 0] (P_2)
    - e^2_4 [2, 3, 0, 4] (P_4)

Process P_2:

- e^1_2 is logically concurrent with:
  - e^1_1 [1, 0, 0, 0] (P_1)
  - e^1_3 [0, 0, 1, 0] (P_3)
  - e^1_4 [0, 0, 0, 1] (P_4)
- e^2_2 is logically concurrent with:
  - e^3_1 [3, 0, 0, 0] (P_1)
  - e^2_3 [2, 3, 5, 0] (P_3)
  - e^2_4 [2, 3, 0, 4] (P_4)

Process P_3:

- e^1_3 is logically concurrent with:
  - e^1_1 [1, 0, 0, 0] (P_1)
  - e^1_2 [0, 1, 0, 0] (P_2)
  - e^1_4 [0, 0, 0, 1] (P_4)
- e^2_3 is logically concurrent with:
  - e^2_4 [2, 3, 0, 4] (P_4)
- e^3_3 is logically concurrent with:
  - e^4_1 [3, 3, 7, 5] (P_1)
  - e^3_4 [2, 3, 0, 5] (P_4)

Process P_4:

- e^1_4 is logically concurrent with:
  - e^1_1 [1, 0, 0, 0] (P_1)
  - e^1_2 [0, 1, 0, 0] (P_2)
  - e^1_3 [0, 0, 1, 0] (P_3)
- e^2_4 is logically concurrent with:
  - e^3_1 [3, 0, 0, 0] (P_1)
  - e^2_2 [2, 2, 0, 0] (P_2)
  - e^2_3 [2, 3, 5, 0] (P_3)

Summary of Logically Concurrent Events:

- e^1_1 with e^1_2, e^1_3, e^1_4

- e^2_1 with e^1_2, e^1_3, e^1_4
- e^3_1 with e^2_2, e^2_4
- e^1_2 with e^1_1, e^1_3, e^1_4
- e^2_2 with e^3_1, e^2_3, e^2_4
- e^1_3 with e^1_1, e^1_2, e^1_4
- e^2_3 with e^2_4
- e^3_3 with e^4_1, e^3_4
- e^1_4 with e^1_1, e^1_2, e^1_3
- e^2_4 with e^3_1, e^2_2, e^2_3

These events are logically concurrent because they have no causal dependency on each other, as indicated by their vector timestamps.

Question 3: Show that the timestamps assigned by vector time are strongly consistent.

Answer: To demonstrate that the timestamps assigned by vector time are strongly consistent, we need to show that for any two events e_i and e_j in a distributed system:

If $e_i \rightarrow e_j$ (i.e., event ei happens-before event e_j), then the vector timestamp of e_i is less than or equal to the vector timestamp of e_j in every component.

Note: This is the condition of strong consistency in the context of vector clocks.

Analysis of Event Causality: Using the Table 1.0

1. **Causality within a Process:**
   - **Within any single process P_i, events are ordered in time by their index. For example:**
     - $e^1_1 \rightarrow e^2_1 \rightarrow e^3_1 \rightarrow e^4_1 \rightarrow e^5_1$
   - **This implies that the vector timestamps must be monotonically non-decreasing:**
     - **[1, 0, 0, 0] ≤ [2, 0, 0, 0] ≤ [3, 0, 0, 0] ≤ [3, 3, 7, 5] ≤ [5, 3, 7, 5]**
   - **We see that this holds true for each process.**

2. **Causality across Processes:**
   - **Consider the causal relationships between events across different processes, as indicated by the arrows in the diagram. For example:**
     - $e^2_1 \rightarrow e^2_2$
       - **Vector timestamps: [2, 0, 0, 0] ≤ [2, 2, 0, 0]**
     - $e^2_2 \rightarrow e^2_3$
       - **Vector timestamps: [2, 2, 0, 0] ≤ [2, 3, 5, 0]**
     - $e^3_2 \rightarrow e^3_3$
       - **Vector timestamps: [2, 3, 6, 5] ≤ [2, 3, 7, 5]**
     - $e^2_4 \rightarrow e^3_4$
       - **Vector timestamps: [2, 3, 0, 4] ≤ [2, 3, 0, 5]**

These relations hold true according to the vector timestamps. This guarantees that vector time is strongly consistent, meaning it correctly captures the causal relationships between events in a distributed system.