Partially Ordered Sets and Logical Clocks for Distributed Systems

Francisco J. Torres-Rojas and José Castro-Mora
{ftorres, jcastro}@itcr.ac.cr
Departamento de Computación, Instituto Tecnológico de Costa Rica
Costa Rica

Abstract. In order to characterize and capture the causal relationships between events in a distributed history, logical clocks have been used in distributed systems with diverse degrees of precision and efficiency (e.g., from efficient but inaccurate Lamport clocks to precise but expensive vector clocks). In this paper, we analyze the representation of logical clocks as mappings between partially ordered sets and relate the accuracy of such clocks with the number of spurious connections manifested in the correspondent Hasse diagrams.

Keywords: Distributed Systems, logical clocks, causality, partially ordered sets.

1 Introduction

In his seminal paper of 1978, Lamport observed that causal relationships among events in a distributed history induce a partial order on them, and proposed the use of logical clocks for ordering these events according to their causality [9]. The implementation of these clocks do not require synchronized physical clocks since they can be accomplished by including additional information with messages exchanged in the system. These logical clocks strive to capture the causality relation, i.e., given the logical timestamps of two events, the clock decides whether they are concurrent or causally related.

Scalar logical clocks can be implemented efficiently [9], but when events are timestamped with these clocks, two events may appear to be ordered even when they are concurrent. On the other hand, vector clocks can precisely order events of a distributed system and detect concurrent events [4, 5, 6, 10], but they require as many entries as sites in the system. Charron-Bost's results [1] discourage any attempt to define a "constant size" clock that completely captures the causality relation. When the number of sites is large, problems of scalability and efficiency arise [13]. Plausible clocks have been proposed as an alternative to vector clocks that, under appropriate circumstances, keep most of the accuracy of vector clocks while allowing efficient implementations [13, 14].

In this paper, we analyze the representation of logical clocks as mappings between partially ordered sets. The accuracy of such clocks is related with the number of "spurious" connections manifested in the correspondent Hasse diagrams. We consider a system where a set of processes communicate exclusively by exchanging messages, all communication is asynchronous and point-to-point, and all messages are delivered correctly. There is neither common memory nor a common physical clock, and the relative speed of the processes is unknown. We briefly present some key background material on partially ordered sets in Section **2**. A general model for logical clocks is explained in Section **3**. Section **4** applies this model to the case of Lamport clocks, and Section **5** does the same for vector clocks. Several plausible clocks (namely, *REV*, *KLA* and *COMB*) are explored in Section **6**. Finally, Section **7** offers the conclusions of this paper.

2 Partially Ordered Sets

In this section, we present some standard definitions from ordered sets theory [2, 3, 7]. Let S be an arbitrary set. A *binary relation* ~ over S is a set $D \subseteq S \times S$. If ($\mathbf{a}$, $\mathbf{b}$) $\in$ D, we denote this as $\mathbf{a} \sim \mathbf{b}$.

Definition 1. A relation ~ over S is:

Reflexive if $\forall\, \mathbf{a},\mathbf{b} \in S$: $\mathbf{a} \sim \mathbf{b} \Rightarrow \mathbf{b} \sim \mathbf{a}$

Irreflexive if $\forall\, \mathbf{a},\mathbf{b} \in S: \mathbf{a} \sim \mathbf{b} \Rightarrow \neg(\mathbf{b} \sim \mathbf{a})$

Transitive if $\forall\, \mathbf{a},\mathbf{b},\mathbf{c} \in S: (\mathbf{a} \sim \mathbf{b}) \wedge (\mathbf{b} \sim \mathbf{c}) \Rightarrow (\mathbf{a} \sim \mathbf{c})$ □

Definition 2. The pair $<S, \sim>$ is a *partially ordered set* (or *poset* for short) if $\sim$ is a binary relation over S that is irreflexive and transitive. We also say that $\sim$ is a *partial order relation* over S. If $\mathbf{a}, \mathbf{b} \in S$ are such that $(\mathbf{a} \neq \mathbf{b}) \wedge \neg(\mathbf{a} \sim \mathbf{b}) \wedge \neg(\mathbf{b} \sim \mathbf{a})$, then **a** and **b** are *non-comparable*. This situation is denoted as $\mathbf{a} \,||\, \mathbf{b}$. □

Posets are usually represented with a *Hasse diagram*, where directly comparable elements are connected with lines (transitive connections are not represented). Besides, if $\mathbf{a} \sim \mathbf{b}$ then element **b** is drawn at a higher position than element **a**; notice, however, that the contrary is not always true, i.e., the fact that element **b** is drawn at a higher position than element **a** does not necessarily imply that $\mathbf{a} \sim \mathbf{b}$. For instance, let S be the set $\{\mathbf{a}, \mathbf{b}, \mathbf{c}, \mathbf{d}, \mathbf{e}, \mathbf{f}\}$ and let $\sim$ be the binary relation given by the set $\{(\mathbf{a}, \mathbf{b}), (\mathbf{a}, \mathbf{c}), (\mathbf{a}, \mathbf{d}), (\mathbf{a}, \mathbf{e}), (\mathbf{a}, \mathbf{f}), (\mathbf{b}, \mathbf{c}), (\mathbf{b}, \mathbf{d}), (\mathbf{b}, \mathbf{e}), (\mathbf{b}, \mathbf{f}), (\mathbf{c}, \mathbf{e}), (\mathbf{c}, \mathbf{f}), (\mathbf{d}, \mathbf{e}), (\mathbf{d}, \mathbf{f}), (\mathbf{e}, \mathbf{f})\}$. Figure 1 presents the corresponding Hasse diagram.

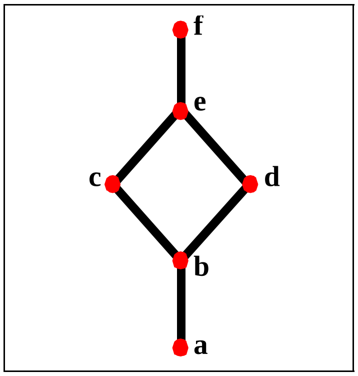

Figure 1. ***Hasse diagram.***

Definition 3. Given two posets $<S, \sim>$ and $<T, \prec>$, we can map the elements of S to the elements of T. As defined in [3], we say that map φ is:

Order-preserving if $\forall\, \mathbf{a}, \mathbf{b} \in S: \mathbf{a} \sim \mathbf{b} \Rightarrow \varphi(\mathbf{a}) \prec \varphi(\mathbf{b})$

Order-embedding if $\forall\, \mathbf{a}, \mathbf{b} \in S: \mathbf{a} \sim \mathbf{b} \Leftrightarrow \varphi(\mathbf{a}) \prec \varphi(\mathbf{b})$

Order-isomorphism if it is an order-embedding that maps S onto T. □

The difference between an order-preserving map and an order-embedding map resides in the double implication of the definition, i.e., if φ is an order-embedding map, then $\varphi(\mathbf{a}) \prec \varphi(\mathbf{b})$ implies that $\mathbf{a} \sim \mathbf{b}, \forall\, \mathbf{a}, \mathbf{b} \in S$, which is not always the case for an order-preserving map. A map φ is an order-isomorphism if, besides of being an order-embedding map, it also happens that every element $\mathbf{b} \in T$ is $\varphi(\mathbf{a})$ for some $\mathbf{a} \in S$.

Definition 4. Given two partial orders $<S, \sim>$ and $<T, \prec>$, we say that a map φ from the elements of S to the elements of T is *plausible* if $\forall\, \mathbf{a}, \mathbf{b} \in S$:

$\mathbf{a} = \mathbf{b} \Leftrightarrow \varphi(\mathbf{a}) = \varphi(\mathbf{b})$

$\mathbf{a} \sim \mathbf{b} \Rightarrow \varphi(\mathbf{a}) \prec \varphi(\mathbf{b})$ □

In other words, a map of ordered sets is plausible if it is order-preserving and no two different elements of set S are mapped to the same element of T and vice versa.

Definition 5. Let $<O_1,\sim_1>$, ..., $<O_n,\sim_n>$ be n posets. The *product* of these n posets [3] is the poset $<O_1 \times ... \times O_n, \sim_\times>$, where $\sim_\times$ is the coordinatewise order:

$(x_1, ..., x_n) \sim_\times (y_1, ..., y_n) \Leftrightarrow \forall$ i: $x_i \sim_i y_i$ □

3 Posets, Causality and Logical Clocks

The local history of site *i* is a sequence of events $H_i = \mathbf{e}_{i1}\mathbf{e}_{i2}...$ that are executed at site *i*. The global or distributed history H of a distributed system is the set of all the events occurring at all sites of the system.

Definition 6. Lamport [9] defines the *causality* relation "→" over events as the smallest relation such that:

- If $\mathbf{e}_{ij}$ and $\mathbf{e}_{ik} \in H_i$ and j < k, then $\mathbf{e}_{ij} \rightarrow \mathbf{e}_{ik}$.
- If $\mathbf{e}_{im}$ is **send**(M), $\mathbf{e}_{jn}$ is its corresponding **receive**(M), i.e., M is the same message in both cases, with arbitrary *i*, *j*, m and n, then $\mathbf{e}_{im} \rightarrow \mathbf{e}_{jn}$.
- $\forall\, \mathbf{a},\mathbf{b},\mathbf{c} \in H$ if $\mathbf{a} \rightarrow \mathbf{b}$ and $\mathbf{b} \rightarrow \mathbf{c}$ then $\mathbf{a} \rightarrow \mathbf{c}$.

If neither $\mathbf{a} \rightarrow \mathbf{b}$ and $\mathbf{b} \rightarrow \mathbf{a}$ holds between two different events **a** and **b**, then **a** and **b** are *concurrent*. This situation is denoted as $\mathbf{a} \,\|\, \mathbf{b}$. □

Since the causality relation → is irreflexive and transitive, it defines a poset $<H, \rightarrow>$ over the events of the system (a pair of concurrent events are non-comparable). We say that $\mathbf{a} \leftarrow \mathbf{b} \Leftrightarrow \mathbf{b} \rightarrow \mathbf{a}$. Figure 2 shows an execution of a distributed system with 4 sites. The arrows indicate the sending and receiving of messages. Since $<H, \rightarrow>$ is a poset, Figure 3 presents the Hasse diagram corresponding to the same execution.

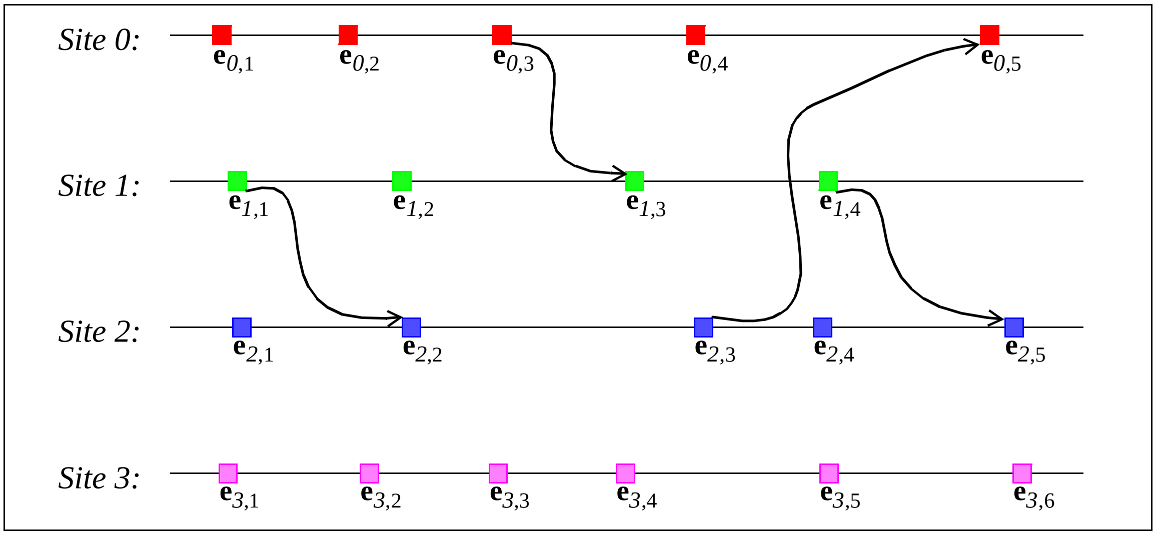

Figure 2. Distributed History.

Lamport proposed the use of *logical clocks* for ordering events in a distributed system according to their causal relationships [9]. These clocks do not need synchronized physical clocks since they can be implemented by including additional information with messages exchanged in the system. These clocks strive to capture the causal relationships between events in a distributed history.

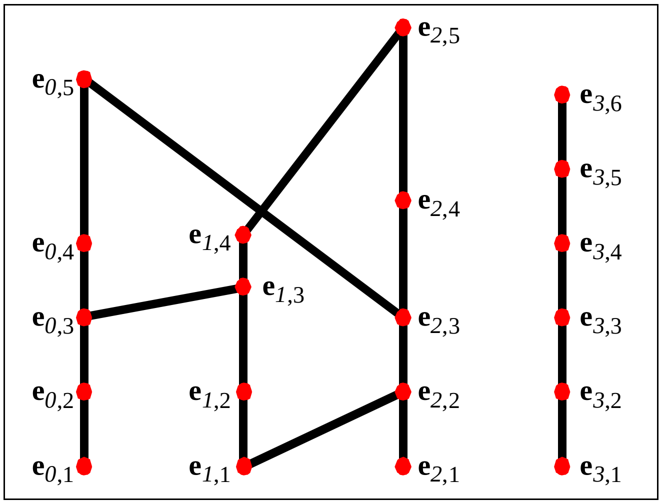

Figure 3. ***Hasse diagram*** **corresponding to the execution of Figure 2.**

Definition 7. Let a *timestamp* be a structure that represents an instant in time as observed by some site. The particular details of this structure are left open. For a distributed system with global history H, a *logical clock* X (also called a *Time Stamping System* X in [14]) is a pair ($<S_X, \xrightarrow{X}>$, X.**stamp**), where:

- S_X is a set of timestamps.
- $\xrightarrow{X}$ is an irreflexive and transitive relation defined on the elements of S_X.
- $<S_X, \xrightarrow{X}>$ is a poset.
- X.**stamp** is the *timestamping function* mapping H to S_X.

For all timestamps $v, w \in S_X$, we define the additional relations:

$$v \overset{X}{=} w \quad \Leftrightarrow v = w$$

$$v \xleftarrow{X} w \quad \Leftrightarrow w \xrightarrow{X} v$$

$$v \overset{X}{\parallel} w \quad \Leftrightarrow \neg(v \overset{X}{=} w) \wedge \neg(v \xrightarrow{X} w) \wedge \neg(v \xleftarrow{X} w) \qquad \square$$

X.**stamp** assigns timestamps to each event of H. It can be expressed as a series of rules for updating the logical clock of a site before assigning a timestamp to an event of H. More interestingly, it can also be understood as a mapping between posets: the poset $<H, \rightarrow>$ is mapped to the poset $<S_X, \xrightarrow{X}>$. When it is clear from the context, we just use $X(\mathbf{a})$ instead of $X.\mathbf{stamp}(\mathbf{a})$, $\forall\, \mathbf{a} \in H$.

Since $\xrightarrow{X}$ is irreflexive, the relations $\xrightarrow{X}$, $\xleftarrow{X}$, $\overset{X}{=}$ and $\overset{X}{\parallel}$ are mutually disjoint. Their purpose is to reflect causality, equality and concurrency from the point of view of X. These relations are defined over timestamps, but we allow them to directly compare events in H. Consider $\mathbf{a},\mathbf{b} \in H$ with timestamps $X(\mathbf{a})$ and $X(\mathbf{b})$, respectively. X reports the causal relationship (not necessarily correct) between **a** and **b**, in this way:

$\mathbf{a} \overset{X}{=} \mathbf{b} \Leftrightarrow X(\mathbf{a}) \overset{X}{=} X(\mathbf{b}) \Leftrightarrow X$ "believes" that **a** and **b** are the same event.
$\mathbf{a} \xrightarrow{X} \mathbf{b} \Leftrightarrow X(\mathbf{a}) \xrightarrow{X} X(\mathbf{b}) \Leftrightarrow X$ "believes" that **a** causally precedes **b**.
$\mathbf{a} \xleftarrow{X} \mathbf{b} \Leftrightarrow X(\mathbf{a}) \xleftarrow{X} X(\mathbf{b}) \Leftrightarrow X$ "believes" that **b** causally precedes **a**.
$\mathbf{a} \overset{X}{\parallel} \mathbf{b} \Leftrightarrow X(\mathbf{a}) \overset{X}{\parallel} X(\mathbf{b}) \Leftrightarrow X$ "believes" that **a** and **b** are concurrent.

Definition 8. A logical clock $X = (<\mathcal{S}_X, \xrightarrow{X}>, X.\mathbf{stamp})$ *is consistent with causality* [11] if $\forall\, \mathbf{a},\mathbf{b} \in \mathcal{H}$:

$$\mathbf{a} \rightarrow \mathbf{b} \;\Rightarrow \mathbf{a} \xrightarrow{X} \mathbf{b} \qquad \square$$

The previous definition is also known as the *weak clock condition* [9]. It is easy to notice the similarity of this definition with the first clause of Definition 3. Thus, we say that a logical clock $X = (<\mathcal{S}_X, \xrightarrow{X}>$, $X.\mathbf{stamp})$ is consistent with causality if the map $X.\mathbf{stamp}$ is an order-preserving map.

Definition 9. A logical clock $X = (<\mathcal{S}_X, \xrightarrow{X}>, X.\mathbf{stamp})$ *characterizes causality* [11] if $\forall\, \mathbf{a},\mathbf{b} \in \mathcal{H}$:

$$\mathbf{a} = \mathbf{b} \;\Leftrightarrow \mathbf{a} \overset{X}{=} \mathbf{b}$$
$$\mathbf{a} \rightarrow \mathbf{b} \;\Leftrightarrow \mathbf{a} \xrightarrow{X} \mathbf{b}$$
$$\mathbf{a} \parallel \mathbf{b} \;\Leftrightarrow \mathbf{a} \overset{X}{\parallel} \mathbf{b} \qquad \square$$

This is also called the *strong clock condition* [11]. Once again, Definition 3 can be used and, thus, a logical clock $X = (<\mathcal{S}_X, \xrightarrow{X}>, X.\mathbf{stamp})$ characterizes causality if the map between the poset $<\mathcal{H}, \rightarrow>$ and the poset $<\mathcal{S}_X, \xrightarrow{X}>$ is an order-isomorphism.

4 Lamport Clocks

Lamport Clocks are efficient scalar logical clocks that are consistent with causality [9]. These clocks map the set of events to a set of integers, in such a way that if event **a** is causally before event **b**, then **a** receives a lower timestamp than **b**. We define the logical clock *Lamport* as the pair $L=(<\mathcal{S}_L, \xrightarrow{L}>, L.\mathbf{stamp})$, where $\mathcal{S}_L$ is a set of positive integers greater than zero and the order relation $\xrightarrow{L}$ is trivially defined as:

$$\mathbf{x} \xrightarrow{L} \mathbf{y} \Leftrightarrow \mathbf{x} < \mathbf{y}\ ,\forall\, \mathbf{x}, \mathbf{y} \in \mathcal{S}_L$$

In order to implement the map $L.\mathbf{stamp}$, each site i maintains an integer counter $\mathbf{L}_i$ that initially has a value of zero, and every message sent by site i includes a timestamp which is the value of $\mathbf{L}_i$ when the message was sent. Before an event (other than a **receive**) is executed at site i $\mathbf{L}_i$ is increased by 1, when a message with timestamp $\mathbf{D}$ is received by site i, then $\mathbf{L}_i$ becomes max $(\mathbf{L}_i, \mathbf{D}) + 1$. If **a** is an event of $\mathcal{H}_i$, then $L(\mathbf{a})$ is the value of $\mathbf{L}_i$ when **a** is executed. Lamport clocks exhibit the weak clock condition, this is, $\forall\, \mathbf{a},\mathbf{b} \in \mathcal{H}$:

$$\mathbf{a} \rightarrow \mathbf{b} \;\Rightarrow L(\mathbf{a}) \xrightarrow{L} L(\mathbf{b}) \Leftrightarrow L(\mathbf{a}) < L(\mathbf{b})$$

Lamport clocks capture the order between causally related events but they do not detect concurrency between events and just by inspecting two timestamps, it cannot be decided if the associated events are causally related or concurrent. Figure 4 shows the same distributed execution of Figure 2, but now each event has been timestamped with its corresponding Lamport clock. Figure 5 shows the Hasse diagram induced by the ordering of these logical clocks. By comparing Figures 3 and 5, we notice that these clocks are not an order-embedding map, but just an order-preserving map. A number of inexistent causal connections are reported (in fact, every pair of concurrent events is falsely ordered). For instance, it is reported that $\mathbf{e}_{0,2}$ is causally before than $\mathbf{e}_{3,5}$ which is false.

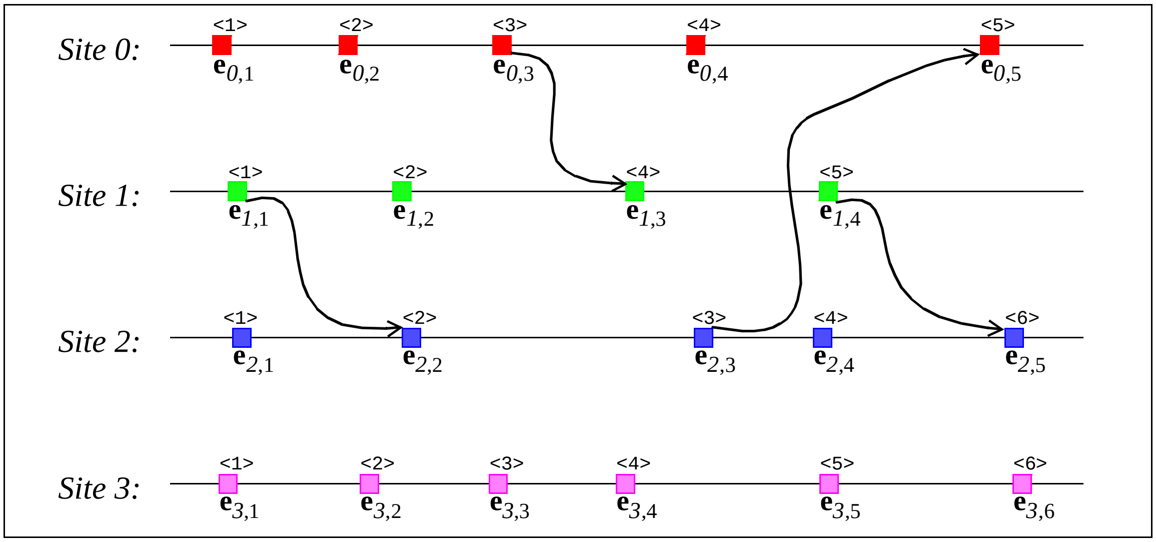
Site 0:
<1> <2> <3> <4> <5>
$\mathbf{e}_{0,1}$ $\mathbf{e}_{0,2}$ $\mathbf{e}_{0,3}$ $\mathbf{e}_{0,4}$ $\mathbf{e}_{0,5}$
Site 1:
<1> <2> <4> <5>
$\mathbf{e}_{1,1}$ $\mathbf{e}_{1,2}$ $\mathbf{e}_{1,3}$ $\mathbf{e}_{1,4}$
Site 2:
<1> <2> <3> <4> <6>
$\mathbf{e}_{2,1}$ $\mathbf{e}_{2,2}$ $\mathbf{e}_{2,3}$ $\mathbf{e}_{2,4}$ $\mathbf{e}_{2,5}$
Site 3:
<1> <2> <3> <4> <5> <6>
$\mathbf{e}_{3,1}$ $\mathbf{e}_{3,2}$ $\mathbf{e}_{3,3}$ $\mathbf{e}_{3,4}$ $\mathbf{e}_{3,5}$ $\mathbf{e}_{3,6}$

Figure 4. Execution timestamped with Lamport Clocks.

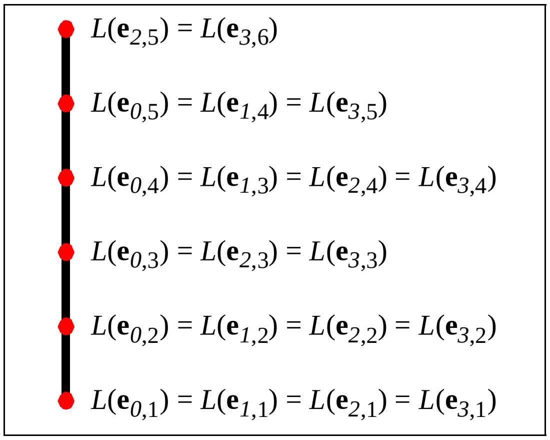
$L(\mathbf{e}_{2,5}) = L(\mathbf{e}_{3,6})$
$L(\mathbf{e}_{0,5}) = L(\mathbf{e}_{1,4}) = L(\mathbf{e}_{3,5})$
$L(\mathbf{e}_{0,4}) = L(\mathbf{e}_{1,3}) = L(\mathbf{e}_{2,4}) = L(\mathbf{e}_{3,4})$
$L(\mathbf{e}_{0,3}) = L(\mathbf{e}_{2,3}) = L(\mathbf{e}_{3,3})$
$L(\mathbf{e}_{0,2}) = L(\mathbf{e}_{1,2}) = L(\mathbf{e}_{2,2}) = L(\mathbf{e}_{3,2})$
$L(\mathbf{e}_{0,1}) = L(\mathbf{e}_{1,1}) = L(\mathbf{e}_{2,1}) = L(\mathbf{e}_{3,1})$

Figure 5. Hasse diagram corresponding to Lamport Clocks

5 Vector Clocks

Fidge [4, 5, 6] and Mattern [10] independently proposed the technique of vector clocks that permits a complete characterization of causality. It consists of a mapping from events in the distributed history to integer vectors. We define *vector clocks* as the logical clock $V = (<S_V, \xrightarrow{V}>, V.\mathbf{stamp})$, where S_V is a set of N-dimensional vectors of integers (N is the number of sites in the distributed system). Each site i keeps an integer vector $\mathbf{V}_i$ of N entries, where N is the number of sites in the distributed system. Initially, this vector is filled up with zeroes. Site i keeps its own logical clock in $\mathbf{V}_i[i]$, i.e., before any event (other than a **receive**) is executed at site i, $\mathbf{V}_i[i]$ becomes $\mathbf{V}_i[i] + 1$. On the other hand, $\mathbf{V}_i[j]$ represents the knowledge that site i has of the activity at site j. All messages include the timestamp of their corresponding **send** event. Thus, when a message with timestamp $\mathbf{W}$ is received by site i, the local clock is updated in this way:

$$0 \le j \le N\text{-}1\text{: } \mathbf{V}_i[j] = \max(\mathbf{V}_i[j], \mathbf{W}[j])$$

$$\mathbf{V}_i[i] = \mathbf{V}_i[i] + 1$$

If $\mathbf{a} \in H_i$, then $V(\mathbf{a})$ is the value of $\mathbf{V}_i$ when $\mathbf{a}$ is executed. Let $\mathbf{V}$ and $\mathbf{W} \in S_V$, the partial order $\xrightarrow{V}$ and its companion relations $\overset{V}{=}$, $\xleftarrow{V}$ and $\overset{V}{\parallel}$ can be defined like:

$$\mathbf{V} \overset{V}{=} \mathbf{W} \Leftrightarrow 0 \le j \le N\text{-}1 : \mathbf{V}[j] = \mathbf{W}[j]$$

$$\mathbf{V} \overset{V}{\rightarrow} \mathbf{W} \Leftrightarrow 0 \le j \le N\text{-}1 : \mathbf{V}[j] \le \mathbf{W}[j] \text{ and } \exists k \text{ such that } \mathbf{V}[k] < \mathbf{W}[k]$$

$$\mathbf{V} \overset{V}{\leftarrow} \mathbf{W} \Leftrightarrow \mathbf{W} \overset{V}{\rightarrow} \mathbf{V}$$

$$\mathbf{V} \overset{V}{\|}| \mathbf{W} \Leftrightarrow \exists k \text{ such that } \mathbf{V}[k] < \mathbf{W}[k] \text{ and } \exists j \text{ such that } \mathbf{V}[j] > \mathbf{W}[j].$$

Mattern [10] and Fidge [6] proved that the map *V*.**stamp** between the poset $<H, \rightarrow>$ and the poset $<S_V, \overset{V}{\rightarrow}>$ is an order-isomorphism[1]. Vector clocks satisfy the strong clock condition and, therefore, they characterize causality (see Definition 9). Figure 6 shows the same execution previously presented in Figure 2. Each event has been timestamped with its respective vector clock. The causal relations between any pair of events are correctly established with the tests presented in this section The Hasse diagram corresponding to the ordering of these vector clocks would be identical to the Hasse diagram presented in Figure 3.

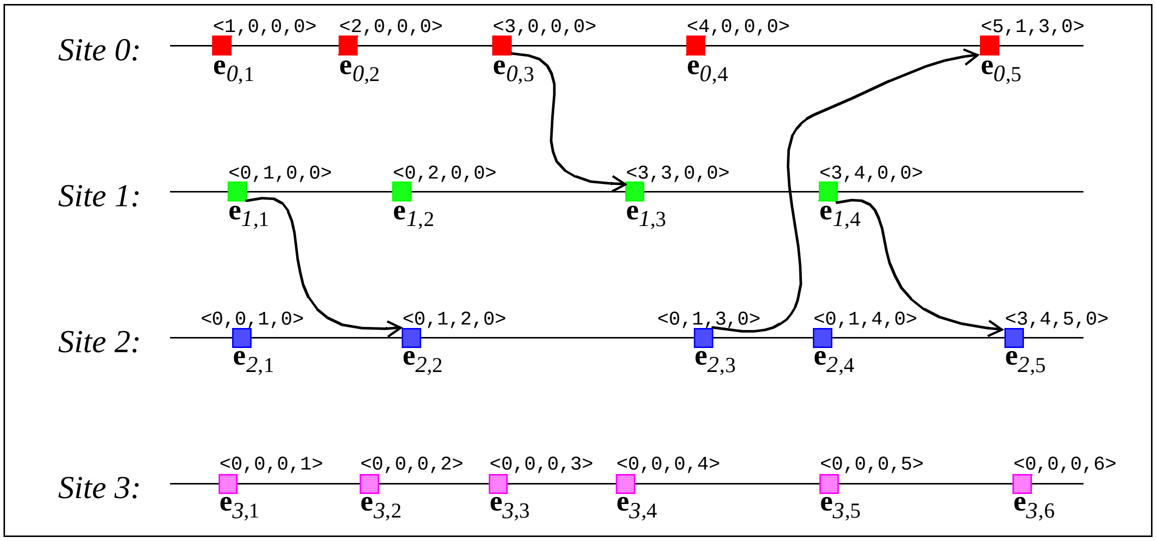

Figure 6. Vector Clock timestamps of events in a distributed execution.

6 Plausible Clocks

The vector clocks technique provides a logical clock that captures completely the causality relation between events in *H*. However, vector clocks require one entry for each one of the N sites of the system. If N is large, several scalability and efficiency problems arise [11, 13]. There are growing storage costs because each site must reserve space to keep its local version of the vector clock and, depending on the particular system, vector times associated with certain events and data structures must be stored as well. Every message must be tagged with the current vector clock of the sender site. Since it can be expected that the total number of messages exchanged increases when the number of sites in the system gets larger, there is a considerable added overhead to the communications of such systems. Charron-Bost proved in [1] that given a distributed system with N sites, it is always possible to find a distributed history whose causality can only be captured by vector clocks with at least N entries. *Plausible clocks* [14] do not characterize causality completely, but they can be constructed with a small and constant number of elements and yet they can decide the causal relationship between arbitrary pairs of events with an accuracy close to vector clocks, in particular when the computation follows a Client/Server communication pattern.

[1] Actually, the order-isomorphism is between vector clocks assigned to events and the causality relation among these events. For instance, the “concurrency bubble” concept presented in [12] describes the existence of vector timestamps for which there cannot be a corresponding event in certain distributed histories.

Definition 10. A logical clock $P = (<S_P, \xrightarrow{P}>, P.\mathbf{stamp})$ is *plausible* if it is a plausible map from the poset $<H, \rightarrow>$ to the poset $<S_P, \xrightarrow{P}>$ (see Definition 4). This is, P is plausible if $\forall \mathbf{a}, \mathbf{b} \in H$:

$$\mathbf{a} = \mathbf{b} \iff \mathbf{a} \overset{P}{=} \mathbf{b}$$
$$\mathbf{a} \rightarrow \mathbf{b} \Rightarrow \mathbf{a} \xrightarrow{P} \mathbf{b}$$

□

A plausible clock P assigns unique timestamps to each event. Besides, P never confuses the direction of causality between any two ordered events (i.e., it is an order-preserving map). Thus, if in fact **a** causally precedes **b**, P will always report $\mathbf{a} \xrightarrow{P} \mathbf{b}$, or if **b** causally precedes **a**, P will always report $\mathbf{a} \xleftarrow{P} \mathbf{b}$. If P states that $\mathbf{a} \overset{P}{\parallel} \mathbf{b}$, this necessarily is correct, because if the actual causal relation were $\mathbf{a} = \mathbf{b}$, $\mathbf{a} \rightarrow \mathbf{b}$ or $\mathbf{a} \leftarrow \mathbf{b}$, it would have been reported as $\mathbf{a} \overset{P}{=} \mathbf{b}$, $\mathbf{a} \xrightarrow{P} \mathbf{b}$ or $\mathbf{a} \xleftarrow{P} \mathbf{b}$, respectively [14]. Vector clocks are plausible clocks, but not every plausible clock P characterizes causality since it is possible that $\mathbf{a} \parallel \mathbf{b}$, but instead P reports $\mathbf{a} \xrightarrow{P} \mathbf{b}$ or $\mathbf{a} \xleftarrow{P} \mathbf{b}$. Several examples of plausible clocks are presented in [13, 14]. In this paper, we consider the plausible clocks *R-Entries Vector* (*REV*), *K-Lamport* (*KLA*) and *Combined* (*COMB*).

6.1 R-Entries Vector (*REV*)

The plausible logical clock $REV = (<S_{REV}, \xrightarrow{REV}>, REV.\mathbf{stamp})$ uses vectors of a fixed size R ≤ N, which is independent of the number of sites in the distributed system. Site *i* updates entry *i* modulo R, since R may be less than N then multiple sites share the same entry in the vector (other mappings between sites and entries of the vector are possible). A similar technique is proposed by Haban and Weigel [8], where processes that are executed at the same site share an entry in the vector clock; *REV* does not have this restriction and allows processes running on different sites to share an entry in the vector. The mechanisms for timestamp comparison and assignment of timestamps to events (i.e., *REV*.**stamp**) are similar to the ones defined for vector clocks (see Section **5**). The elements of S_{REV} are of the form $<s, \mathbf{V}>$ where s uniquely identifies each site, and **V** is a R-dimensional vector of integers.

Definition 11. Let $v = <s_v, \mathbf{V}>$ and $w = <s_w, \mathbf{W}> \in S_{REV}$, then:

- $v \overset{REV}{=} w \iff (s_v = s_w \wedge \mathbf{V}[s_v \text{ modulo R}] = \mathbf{W}[s_w \text{ modulo R}])$
- $v \xrightarrow{REV} w \iff (s_v = s_w \wedge \mathbf{V}[s_v \text{ modulo R}] < \mathbf{W}[s_w \text{ modulo R}]) \vee$
  $(\neg(s_v = s_w) \wedge \mathbf{V} < \mathbf{W} \wedge \mathbf{V}[s_w \text{ modulo R}] < \mathbf{W}[s_w \text{ modulo R}])$
- $v \overset{REV}{\parallel} w \iff \neg(s_v = s_w) \wedge \neg(v \xrightarrow{REV} w) \wedge \neg(v \xleftarrow{REV} w)$

Vectors **V** and **W** are compared using the tests previously presented in Section **5**, with the only difference that they have R entries instead of N. □

Figure 7 shows the same execution presented in Figure 2, but using timestamps from *REV* (R = 2). In order to simplify, the part of the timestamp corresponding to the site identification has been omitted. This clock establishes correctly the causal relationship between 320 out of the 400 possible pairs of events. For instance, *REV* recognizes that $\mathbf{e}_{3,1} \parallel \mathbf{e}_{0,2}$, but mistakenly reports $\mathbf{e}_{3,1} \xrightarrow{REV} \mathbf{e}_{1,2}$, when the true is that $\mathbf{e}_{3,1} \parallel \mathbf{e}_{1,2}$. Figure 8 presents the Hasse diagram corresponding to the timestamps that *REV* generates for this execution. Notice that, albeit there are extra connections in Figure 8, the general structure of this diagram is similar to the one in Figure 3 (this is a consequence of the requirements of plausible clocks). The bold lines represent the actual causal relationships between events, while the thin lines are spurious connections that are artifacts of the imperfection of *REV*. Notice that the direct connection between $\mathbf{e}_{2,3}$ and $\mathbf{e}_{0,5}$ has been omitted because it can be obtained by transitivity through $\mathbf{e}_{2,4}$.

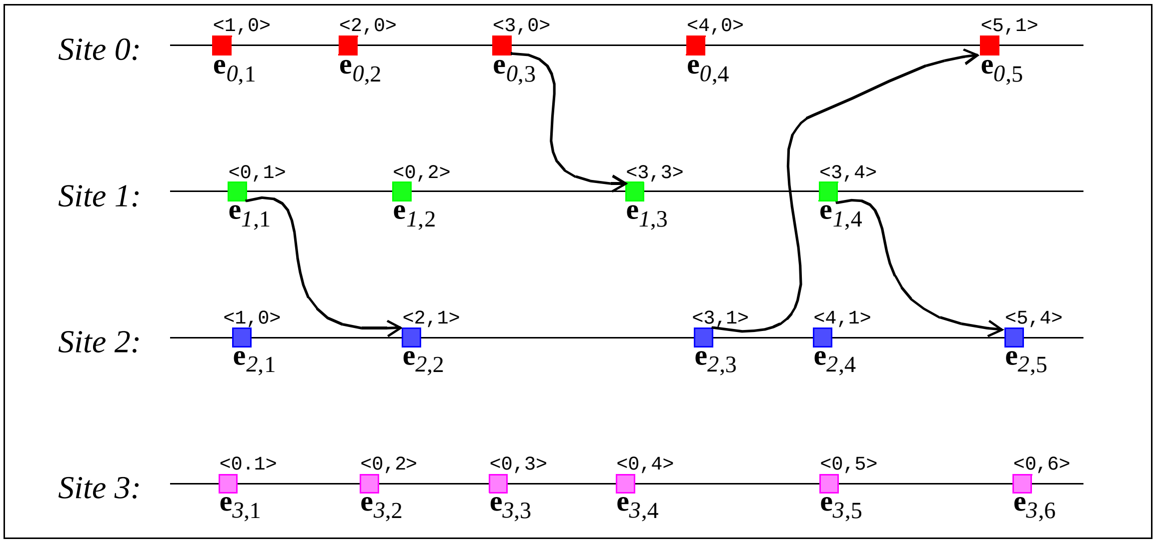

Figure 7. Execution timestamped with *REV* (R=2)

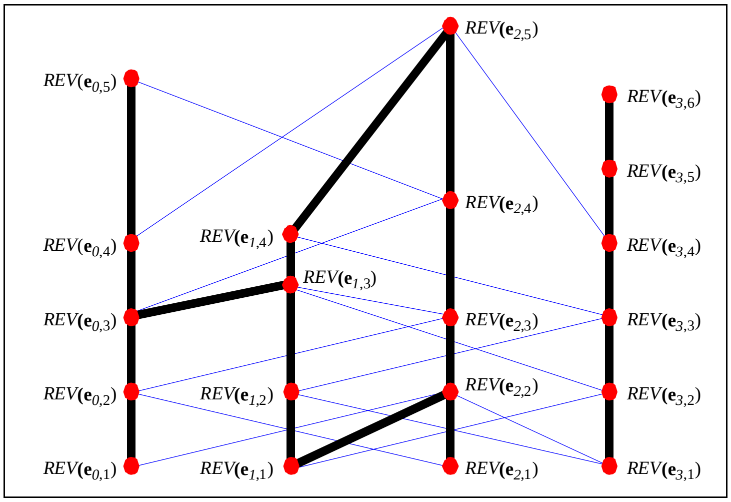

Figure 8. Hasse diagram for execution timestamped with *REV* (R=2).

6.2 K-Lamport (*KLA*)

Timestamps in the logical clock KLA = (<S_{KLA}, $\xrightarrow{KLA}$>, KLA.**stamp**) are of the form <s, **V**> where s uniquely identifies each site, and **V** is a K-dimensional vector of integers. Each site keeps a Lamport clock together with the maximum timestamp of any message received by itself and by the K-2 previous sites that directly or indirectly have had communications with this site. When a message with timestamp <s_j, **W**> is received at a site whose current clock is <s_i, **V**>, **V**[0] becomes *max*(**V**[0], **W**[0]) + 1. Entries **V**[1] to **V**[K-1] are *max*-ed with entries **W**[0] to **W**[K-2], respectively.

Definition 12. Let $\mathbf{v}$ = <s_v, **V**> and $\mathbf{w}$ = <s_w, **W**> ∈ S_{KLA}, then:

- $\mathbf{v} \overset{KLA}{=} \mathbf{w} \Leftrightarrow (s_v = s_w \wedge \mathbf{V}[0] = \mathbf{W}[0])$
- $\mathbf{v} \xrightarrow{KLA} \mathbf{w} \Leftrightarrow (s_v = s_w \wedge \mathbf{V}[0] < \mathbf{W}[0]) \vee$
  $(\neg(s_v = s_w) \wedge \mathbf{V}[0] \le \mathbf{W}[1] \wedge \mathbf{V}[1] \le \mathbf{W}[2] \wedge \ldots \wedge \mathbf{V}[K\text{-}2] \le \mathbf{W}[K\text{-}1])$
- $\mathbf{v} \overset{KLA}{\parallel} \mathbf{w} \Leftrightarrow \neg(s_v = s_w) \wedge \neg(\mathbf{v} \xrightarrow{KLA} \mathbf{w}) \wedge \neg(\mathbf{v} \xleftarrow{KLA} \mathbf{w})$

□

Every case where **a** || **b** that is recognized by (*K-1*)*LA*, is also recognized by *KLA*, but the converse is not always true. Thus, *KLA* can provide higher ordering accuracy than (*K-1*)*LA*. Figure 9 shows the same execution presented in Figure 2, but using timestamps from *KLA* (K = 3). This clock fails to establish the causal relationship between 46 out of 400 possible pairs of events. For instance, it detects correctly that $\mathbf{e}_{3,1} \parallel \mathbf{e}_{1,2}$, but fails when it reports $\mathbf{e}_{1,3} \xleftarrow{KLA} \mathbf{e}_{3,3}$, since actually $\mathbf{e}_{1,3} \parallel \mathbf{e}_{3,3}$.

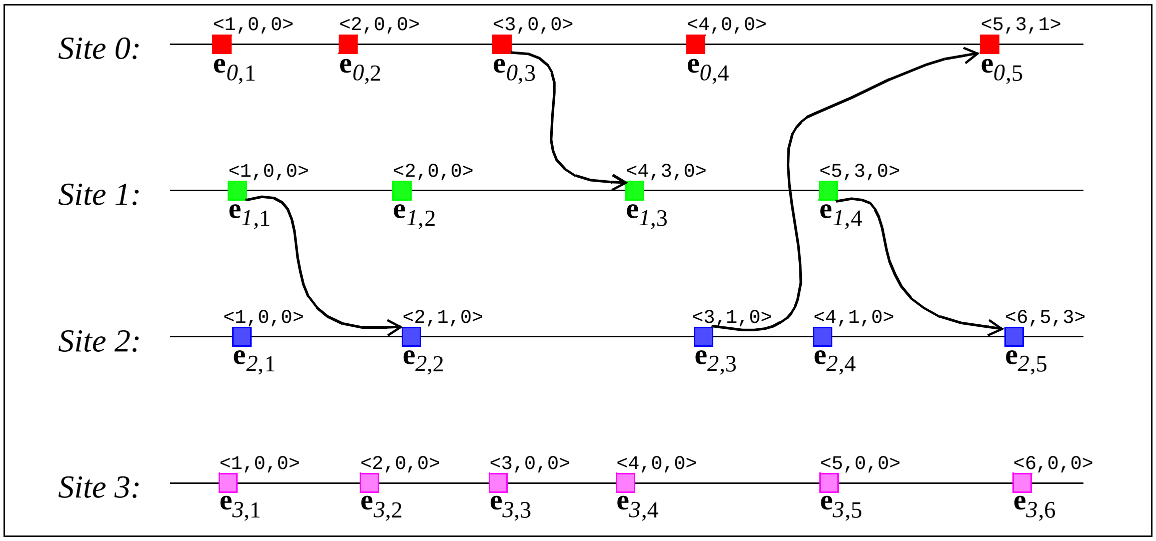

Figure 9. Execution timestamped with *KLA* (K = 3).

Figure 10 presents the Hasse diagram corresponding to the timestamps generated by *KLA* for this execution. Notice that the number of false connections is less than the ones produced by *REV* (see Figure 8).

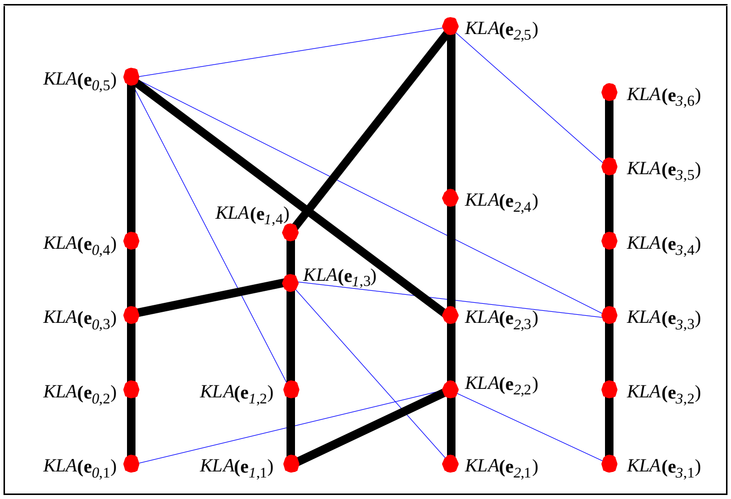

Figure 10. Hasse diagram for *KLA* (K = 3)

6.3 Combined Clock (*Comb*)

A *combination* of plausible clocks allows that each event has several timestamps coming from different plausible clocks. These clocks are used to compare their respective timestamps, and a causal relationship is reported only if all the clocks agree on the same result. Whenever that at least one clock disagrees on the possible causal relationship between events **a** and **b**, this means that **a || b**. This "rule of contradiction of plausible clocks” is proved in [14]. Besides, it is established that a combination will always be at least as accurate as any of its components and it is very likely that it will be better. The product of ordered sets, as presented in Definition 5, is equivalent to the combination of plausible clocks:

Definition 13. Let P_1=(<S_1,$\xrightarrow{1}$>, P_1.**stamp**), ... , P_n=(<S_n,$\xrightarrow{n}$>, P_n.**stamp**) be plausible clocks. The plausible clock $P_\times$=(<$S_1 \times ... \times S_n$,$\xrightarrow{\times}$>, $P_\times$.**stamp**) is a *combination* of P_1, ... , P_n iff:

$\forall \mathbf{a} \in H$: $P_\times$.**stamp(a)**= (P_1.**stamp(a)**, ... , P_n.**stamp(a)**)

$(\boldsymbol{v}_1, ... , \boldsymbol{v}_n) \xrightarrow{\times} (\boldsymbol{w}_1, ... , \boldsymbol{w}_n) \Leftrightarrow \forall$ i: $\boldsymbol{v}_i \xrightarrow{i} \boldsymbol{w}_i$ □

The logical clock *Comb*, as defined in [14], is a combination of the clocks *REV* and *KLA*. As an illustration, if the clocks *REV* and *KLA* used to timestamp the execution presented in Figures 7 and 9 were combined to timestamp the same execution history, the number of errors is reduced to 38 out of 400 pairs of events. Figure 11 presents the Hasse diagram for the timestamps generated by *Comb*.

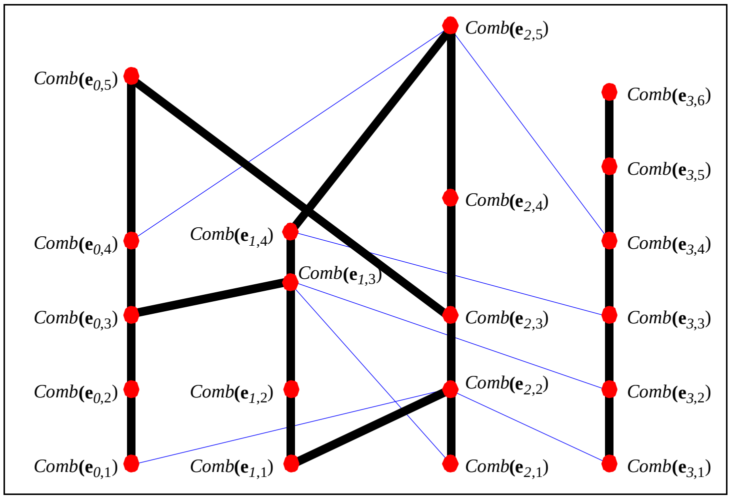

Figure 11. Hasse diagram for *Comb***.**

7 Conclusions

In order to analyze and understand the behavior of a distributed system, the causality among events in the corresponding distributed history must be captured. This relationship, which defines a partially ordered set over these events [9], can be represented by using logical clocks. These clocks establish a particular format of logical timestamp, assign these timestamps to events, and, finally, define rules to compare these logical timestamps. Since a logical clock induces a partially ordered set on its corresponding timestamps, its use can be understood as a mapping between the poset induced by the causality relationship and the poset induced by the rules of the specific logical clock.

Lamport clocks can be implemented efficiently [9], but they induce a total order of the events and, as the Hasse diagram in Figure 5 shows, every single pair of concurrent events is wrongly ordered. This comes from the fact that the mapping between events in the distributed history and Lamport clocks is just an order-preserving map.

The mapping with vector clocks is an order-isomorphism [4, 5, 6, 10], and, therefore, these clocks can precisely order events of a distributed system and detect concurrent events (see Figure 6). However, as it is proved in [1], they require as many entries as sites in the system. When the number of sites is large, serious problems of scalability and efficiency arise [13].

Plausible clocks have been proposed as an alternative to vector clocks that, under appropriate circumstances, keep most of the accuracy of vector clocks while allowing efficient implementations [13, 14]. A mapping of posets is said to be plausible if it is an order-preserving that assigns unique images to each element of the original set and vice versa. Thus, a plausible clock satisfies the weak clock condition and assigns unique timestamps to each event. There are many possible implementations of plausible clocks. For the purposes of this paper, we presented *REV* (variant of vector clocks where R-entries vectors are used; since R is less than the number of sites, several entries are shared by more than one site of the system and therefore a mapping between sites and entries in the vector must be defined), *KLA* (extension of Lamport clocks where each site keeps a standard Lamport clock together with a collection of the maximum timestamp of any message received by itself and by the K - 2 previous sites that directly or indirectly have had communications with this site), and *Comb* (combination or product of the posets [3] induced by *REV* and *KLA*, which can be proved to be at least as good, and possibly better than, as any of its components). The Hasse diagrams corresponding to each one of these clocks (see Figures 8, 10 and 11) exhibit a number of false or spurious connections that explain the imperfection of these clocks.

References

1 B. Charron-Bost, “Concerning the size of logical clocks in Distributed Systems”, Information Processing Letters 39, pp. 11-16. 1991.

2 P. Crawley and R. P. Dilworth, “Algebraic Theory of Lattices”, Prentice-Hall Inc., 1973.

3 B.A. Davey and H.A. Priestley, “Introduction to Lattices and Order”, Cambridge University Press, 1990.

4 C.J. Fidge, “Timestamps in message-passing systems that preserve the partial ordering”, Proc. 11th Australian Comp.Science Conference, Univ. of Queensland, pp. 55-66, 1988.

5 C.J. Fidge, “Logical Time in Distributed Computing Systems”, Computer, vol 24, No. 8, pages 28-33, August 1991.

6 C.J. Fidge, “Fundamentals of Distributed Systems Observation”, IEEE Software, vol 13, No. 6, November 1996.

7 P. C. Fishburn, “Interval Orders and Interval Graphs”, John Wiley and Sons, New York, 1985.

8 D. Haban and W. Weigel, “Global Events and Global Breakpoints in Distributed Systems”, Proceedings of the 21st Hawaii International Conference on Systems Sciences, January 1988.

9 L. Lamport, “Time, clocks and the ordering of events in a Distributed System”, Communications of the ACM, vol 21, pp. 558-564, July 1978.

10 F. Mattern, “Virtual Time and Global States in Distributed Systems”, Conf. (Cosnard et al (eds)) Proc. Workshop on Parallel and Distributed Algorithms, Chateau de Bonas, Elsevier, North Holland, pp. 215-226. October 1988.

11 R. Schwarz and F. Mattern, “Detecting causal relationships in distributed computations: in search of the holy grail”, Distributed Computing, Vol. 7, 1994.

12 F. Torres-Rojas, “Efficient Time Representation in Distributed Systems”, MSc. Thesis, College of Computing, Georgia Institute of Technology, 1995.

13 F. Torres-Rojas, “Scalable Approximations to Causality and Consistency of Distributed Objects”, Ph.D. dissertation, College of Computing, Georgia Institute of Technology. July 1999.

14 F. Torres-Rojas and Mustaque Ahamad,“Plausible Clocks: Constant Size Logical Clocks for Distributed Systems”, Distributed Computing, pp. 179-195, December 1999.