Formal Semantics of Composite Events for Distributed Environments*

Shuang Yang          Sharma Chakravarthy
Database Systems Research and Development Center  
Computer and Information Science and Engineering Department  
University of Florida, Gainesville FL 32611  
email: sharma@cise.ufl.edu

Abstract

Languages for event specification in centralized systems and their semantics have received considerable attention in the literature. In contrast, very little work exists on extending the semantics of event specification languages to distributed environments.

This paper provides a well-defined notion of distributed composite time stamps and their least restricted strict ordering are defined. The ordering is carefully chosen based on mathematical reasoning to ensure the best semantics. The concurrence and weaker-less-than-or-equal temporal relations are also introduced for the expressiveness of ECA rules. Furthermore, a Max operator is introduced for propagating the composite event time stamps. Based on this partial ordering and the Max operator on the time stamps, the semantics of Sentinel composite events is described for distributed event detection.

1. Introduction

The active functionality is necessary to meet the requirements of a large number of real-world applications which requires monitoring and reacting to the internal as well as external changes automatically and without the intervention of the users or the application. Active DBMS enhances the functionality of the conventional DBMS by executing pre-specified operations in response to certain event occurrences or conditions. This active capability is modeled by ECA (Event-Condition-Action) rules. When an event is detected and if the condition specified by the rules is checked to be hold, then an action is executed. Rule definition, event detection and action execution are some of the fundamental active features provided by the active DBMS. Much work has been done on active functionality in the centralized context: Hipac [11], Ode [6], ADAM [4], and SAMOS [5], Sentinel [1, 3].

This paper focuses on deriving a well-defined time stamp and its ordering relation on the distributed environment. We extend the semantics of Sentinel composite events to a distributed system. A distributed composite time stamp is defined to be the set of maximum primitive time stamps from the participating primitive ones. A well-defined partial ordering of the (set of) time stamps are defined which is carefully chosen based on mathematical reasoning for a best semantics. The \( \sim, \prec \) relations are also introduced for dealing with some of the Sentinel operators.

The Max operator on the time stamps is also introduced for propagating the events. Finally, the full semantics of Sentinel distributed composite event detection is discussed. This paper provides a fundamental framework of partial ordered sets relation for distributed event detection. A number of properties concerning the partial ordering is presented and formally proved. Some of them are not so obvious due to the partial ordered property and set representation.

This paper is organized as follows. Section 2 briefly reviews some of the related work, especially Schwiderski’s dissertation [10]. The differences between that work and this paper are discussed. In Section 3, an overview of centralized composite event semantics in Sentinel is presented. Section 4 introduces the distributed time and temporal relation based on the global time and restricted ordering. The definition of composite time stamps and their ordering along with the Max operator is presented in Section 5 and the distributed composite event semantics of Sentinel is derived. Section 6 contains conclusions.

2. Related Work

In additional to Sentinel, there are several efforts attempting to monitor the behavior of the distributed systems. Microsoft’s COM (Component Object Model) [8] and CORBA provide some fundamental distributed event services but none of them have the notion of composite events. Schwiderski’s dissertation [10] presents a general concept of primitive and composite event specification, event semantics and event detection in distributed systems which based on the notion of approximated global time and \( \mathbb{G}_g \) restricted temporal order.

In [10], the syntax of primitive and composite events is derived from the work of both active database systems and distributed debugging systems. The primitive events

---

*This work was supported in part by the Office of Naval Research and the SPAWAR System Center – San Diego, by the Rome Laboratory, DARPA, and the NSF Grant IRI-9528390
are site-related and include time events, data manipulation events, transaction events and abstract events.

When a primitive event occurs, a time stamp is associated to the primitive event which is usually a tuple containing the information of the original site, approximated global time and local time. When a composite event is detected, a set of time stamps are collected corresponding to the time stamps of the constituent primitive and composite events and on the event operators. The temporal ordering relations < and ~ are defined on the primitive and composite time stamps based on the 2gθ− restricted temporal order. The structure and the handling of the time stamps in distributed system are also discussed.

Our approach is similar to [10]. One difference is that we enforce the concurrency and “latest” properties in the definition of the time stamps. The philosophy behind our definition is that only the “latest” time stamps is considered and carried to form the set of the composite time stamp, which is corresponding to the concept of t_occ in centralized systems. Another major difference is the definition of temporal ordering < on the (set of) time stamps of the composite events. Our definition of restricted < ordering satisfies the irreflective and more importantly transitive properties which ensure a well-defined mathematics ordering but their is not. Finally, our time stamps operators Max operator is conceptually similar to their “joining” operators, but defined in a more precise way and is well-integrated with our definition of distributed composite time stamps, which the “latest” and “concurrency” properties are ensured.

3. Centralized Composite Event Semantics

Sentinel is an active object-oriented DBMS which supports ECA rules. In this section, an overview of the event semantics in the centralized environment is presented. Please refer to [1, 3, 2] for detailed explanation.

Time in the centralized systems is totally ordered and can be represented as the (local) clock ticks of the (local) physical clock from some starting point.

In the centralized system, time is totally ordered as non-negative integers representing physical clock ticks [9]. This means for any given two time represented as the time ticks t1 and t2, the temporal relationship between these two can be: t1 < t2, t1 = t2, or t1 > t2.

3.1. Primitive Events in Centralized Active DBMS

An event is an instantaneous occurrence of interest which occurs at a specific point in time. Primitive events are those that are pre-defined in the system. Each primitive event is associated with a time stamp, which indicates the time ticks of the occurrence of the event. Let e be a primitive event, the time stamp of that event is the time occurrence of the event denoted by T(e), i.e., T(e) = t_occ(e) = clock(e).

Any two primitive events e1 and e2 with the corresponding time stamps T(e1) and T(e2) can be totally ordered based on the ordering of the time stamps. Let e1 and e2 be any primitive events then the temporal order of these two events are defined as follows: e1 is said to be happen-before e2 if T(e1) < T(e2); e1 is said to be simultaneously with e2 if T(e1) = T(e2); e1 is said to be happen-after e2 if T(e1) > T(e2).

Some events are not allowed to occur simultaneously, but there are some events that have to occur simultaneously. The following are the assumption of simultaneity of the events: 1. Each non-temporal event has at least one temporal event happening simultaneously, 2. Each composite event (will be defined later) has at least one primitive event happening simultaneously, 3. No two database events happen simultaneously, 4. No two explicit events happen simultaneously.

3.2. Centralized Sentinel Event Semantics

Primitive events are the basic building blocks for developing an expressive and useful composite event specification language. Composite events are denoted as event expression formed by primitive events associated with event operators. Conceptually, a primitive event type is the name of an interested primitive event while a composite event type is the name of a pattern of the set of interested events (including primitive and composite events) specified by the event expression.

Definition 3.1 Centralized Event An event E (either primitive or composite) is a function from the time domain onto the boolean values, True and False. The negation of the boolean function E is denoted as ¬ E which means given a time point, the non-occurrence of the event at that point. In Sentinel, the composite event expression is defined recursively, by using a set of the primitive events, event operators and the composite events. For a given set of events E1, E2, ..., Em (primitive or composite), the semantics of the composite event operators in Sentinel are defined in [2].

4. Time in Distributed System

The composite event detection relies largely on the comparison of the time occurrence of the events. Because of the lack of global time and lack of total ordering of the global time, a weakened semantics of approximated global time along with their partial ordering in distributed systems, and a semantics of distributed primitive time stamps and the partial ordering [10] is reviewed. Besides, notions of ~, ≈ and “open”, “closed” intervals are introduced for the expressiveness of Sentinel ECA rules. A number of properties about the time stamps and their ordering are discussed and proved for better understanding of the semantics and as a tool for the proofs of composite event ordering.
We assume the definition of (partial) ordering relation to be a well-defined ordering relation (in terms of reflexive, transitive, symmetric, asymmetric, and antisymmetric properties). Also, this definition will serve as a basis when we define a well-defined ordering of given set.

**Definition 4.1 equivalent relation** A relation $=$ on a set $A$ is called a *equivalent relation* if it is transitive, reflexive and symmetric.

**Definition 4.2 partial ordering and total ordering** A relation $<$ on a set $A$ is called a *strict partial ordering* if it is transitive and *irreflexive*. It is called a *strict total ordering* if in addition we have: $(\forall x,y,z \in A)(\text{ only } x < y \text{ or } x = y \text{ or } y < x)$

In centralized time system, the $<$ strict total ordering satisfies *irreflexive, transitive* and *asymmetric*, and the $\leq$ total ordering satisfies *reflexive, transitive*, and *antisymmetric*.

### 4.1. Distributed global time semantics

The notion of physical time is a problem in distributed systems; there is no global time in nature. Each site in a distributed system has a single local physical clock which has its own local clock tick and is converted to local time by some software device. In order to compare the time of occurrence at the remote site, local clocks have to be synchronized. In a distributed system a global time can be achieved with synchronized local clocks through an *approximated global time base* [7]. That is, there exists a unique reference clock $z$ with granularity $g_z$ which is in perfect agreement with the international standard of time. The local clocks can be synchronized by the concept of *precision* $\Pi$, which is the maximum offset of the time difference between two corresponding ticks of any two local clocks observed by the reference clock and measured as the ticks of the reference clock. A global time in Distributed systems can be approximated by adjusting the granularity of local clock measurement to a global clock granularity $g_g$, that is, by selecting a subset of the microticks of each local clock $j$ for generating the local implementation of a global notion of time. We need $g_g > \Pi$ to ensure that two simultaneous events receive time stamps distant at most $\leq 1/g_g$, $g_g$ can be chosen to be just greater than $\Pi$ ($g_g = \Pi + \varepsilon$) and then the global time [10] of each site can be derived from the local clock ticks of the site.

**Definition 4.3 Global time** The global clock granularity $g_g$ is given. The global time $g_k$ of a local clock tick $l_k$ is the local clock ticks expressed according to the standard (Gregorian) calendar with respect to some time zone (e.g. UTC, Universal Time Coordinated) and truncated to a global granularity $g_g$. $g_k(l_k) = TRUNC_{g_g}(clock_k(l_k))$

Here the “TRUNC” function could be *round*, *ceiling* or *floor* depends on the application as long as it is consistent throughout the system. From now on, the “TRUNC” is defined to be integer division. Let $l_1(e)$ be the local ticks of the event $e$ and $g_k(e)$ be the corresponding global time with global granularity $g_g$, then the ordering of the time in distributed system called $2g_g-precedence$ [10] can be derived based on the local and global time of each site.

**Definition 4.4 $2g_g$-restricted temporal order** The global clock granularity $g_g$ is given. $2g_g$-restricted temporal order $(\rightarrow_{2g_g})$ between primitive events $e_1$ and $e_2$ is defined as follows: 1. If $e_1$ and $e_2$ are primitive events occurring at the same site and $l_1(e_1) < l_1(e_2)$ then $e_1 \rightarrow_{2g_g} e_2$. 2. If $e_1$ and $e_2$ are primitive events occurring at the distinct sites and $g_k(e_1) < g_k(e_2) - 1g_g$, then $e_1 \rightarrow_{2g_g} e_2$.

**Definition 4.5 $2g_g$-restricted concurrency** The global clock granularity $g_g$ is given. $2g_g$-restricted concurrency $(\|_{2g_g})$ between primitive events $e_1$ and $e_2$ is defined as follows: $e_1\|_{2g_g} e_2 \iff \neg(e_1 \rightarrow_{2g_g} e_2) \text{ and } \neg(e_2 \rightarrow_{2g_g} e_1)$

Notice that the $2g_g$-restricted temporal order is *irreflexive* (i.e., $e_1 \rightarrow_{2g_g} e_1$ is never true) and transitive (i.e., if $e_1 \rightarrow_{2g_g} e_2$ and $e_2 \rightarrow_{2g_g} e_3$ then $e_1 \rightarrow_{2g_g} e_3$). Also, the $2g_g$-restricted concurrency relation is not an equivalent relation since it is not transitive. So, the $\rightarrow_{2g_g}$ is a valid strict partial ordering but not totally ordered which is different from the centralized system.

### 4.2. Distributed Primitive Time Stamps

Each event in the centralized system is associated with a time stamp (or even a counter which is advanced at the occurrence of each event) indicating when the event occurs. The detection of composite events is based on the ordering of the time stamp. In distributed systems, the time stamp of a global primitive event is a little more complicated [10]. The information about the site, local time as well as the the global time needs to be captured. Temporal relationship between time stamps can be derived directly from the definition of time stamps and the $2g_g-precedence$.

**Definition 4.6 Time stamps of the global primitive events** A time stamp $T(e)$ of a global primitive event $e$ with event type $E$ is a function $T : E \rightarrow (site, global, local)$, where “site” is the site of occurrence of the primitive event, $local = l_t(e)$ and $global = g_k(e)$. If $T(e) = (site, global, local)$, then we also use the syntax of the object-oriented language to denote site $= T(e).site$, global $= T(e).global$ and local $= T(e).local$.

**Definition 4.7 Temporal Relationship of Global Primitive Events** On the basis of the $2g_g$-precedence time model, the temporal relationship between two time stamps $T(e_1)$ and $T(e_2)$ is defined as follows:

1. Happen-before: $T(e_1) < T(e_2)$ iff $(T(e_1).site \neq T(e_2).site \land T(e_1).local < T(e_2).local) \lor$
(\text{T}(e_1).\text{site} \neq \text{T}(e_2).\text{site} \land \text{T}(e_1).\text{global} < \text{T}(e_2).\text{global} - 1g_g)

2. Simultaneous: \(\text{T}(e_1) = \text{T}(e_2)\) iff \text{T}(e_1).\text{site} = \text{T}(e_2).\text{site}\) and \(\text{\text{T}(e_1).\text{local} = \text{T}(e_2).\text{local}}\)

3. Concurrency: \(\text{T}(e_1) \sim \text{T}(e_2)\) iff \(\neg((\text{T}(e_1) < \text{T}(e_2)) \lor (\text{T}(e_2) < \text{T}(e_1)))\)

Theorem 4.1 strict partial ordering of \(<\) The \(<\) relation defined above is irreflexive and transitive, so it is a strict partial ordering relation on the set of distributed primitive time stamps.

The above theorem ensures a well-defined ordering definition. But that \(\sim\) relationship is not transitive, so it is not an equivalent relation while the simultaneous relation \(=\) is an equivalent relation. The definition of Simultaneous is a special case of Concurrency when sites are same.

The following proposition demonstrates the relationship between the local time and the global time.

Proposition 4.1 Let \(\text{T}(e_1)\) and \(\text{T}(e_2)\) be any two time stamps, then:
- if \(\text{T}(e_1).\text{local} < \text{T}(e_2).\text{local}\) then \(\text{T}(e_1).\text{global} \leq \text{T}(e_2).\text{global}\).
- if \(\text{T}(e_1).\text{local} = \text{T}(e_2).\text{local}\) then \(\text{T}(e_1).\text{global} = \text{T}(e_2).\text{global}\).
- if \(\text{T}(e_1) \sim \text{T}(e_2)\) then \[\text{T}(e_1).\text{global} - \text{T}(e_2).\text{global}\] \(\leq 1g_g\).

Based on the definition of time stamps and their temporal order in distributed time system, the notions of open and closed interval formed by the time stamps can be defined for expressing the event operators in Sentinel.

Definition 4.8 weakened Less-than-equal relation for Primitive Time stamps Let \(\text{T}(e_1), \text{T}(e_2)\) be the time stamps of global primitive event \(e_1\) and \(e_2\). \(\text{T}(e_1)\) is said to be weakened-less-than-equal to \(\text{T}(e_2)\) denoted \(\text{T}(e_1) \preceq \text{T}(e_2)\) iff \(\text{T}(e_1) < \text{T}(e_2)\) or \(\text{T}(e_1) \sim \text{T}(e_2)\).

Notice again that the \(\preceq\) relation is not a partial ordering relation because it does not satisfy the transitivity property which is due to the non-transitiveness of \(\sim\) relation. The reason \(\sim\) instead of \(=\) be chosen is because we want any give primitive time stamps be able to compare by \(\preceq\) in distributed system, instead of requiring the sites be same.

Definition 4.9 Open Interval of Primitive Time stamps in Distributed Systems Let \(\text{T}(e_1), \text{T}(e_2)\) be the time stamps of global primitive event \(e_1\) and \(e_2\). An event \(e\) with time stamp \(\text{T}(e)\) is said to be in the open interval formed by \(\text{T}(e_1)\) and \(\text{T}(e_2)\) with \(\text{T}(e_1) < \text{T}(e_2)\) denoted \(\text{T}(e) \in (\text{T}(e_1), \text{T}(e_2))\) iff \(\text{T}(e_1) < \text{T}(e) < \text{T}(e_2)\).

Suppose \(\text{T}(e_1), \text{T}(e_2)\) are in different sites, then \(\text{T}(e_1) < \text{T}(e) < \text{T}(e_2)\) implies:

\[\text{T}(e_1).\text{global} < \text{T}(e).\text{global} - 1g_g\text{ and}\]
\[\text{T}(e).\text{global} < \text{T}(e_2).\text{global} - 1g_g\]
\[\implies \text{T}(e_1).\text{global} < \text{T}(e_2).\text{global} - 3g_g\]

for a non-empty open interval. Intuitively, the open interval

\[(\text{T}(e_1).\text{global}, \text{T}(e_2).\text{global}) = \{\text{T}(e_1).\text{global} + 2g_g, \text{T}(e_1).\text{global} + 3g_g, \ldots, \text{T}(e_2).\text{global} - 3g_g, \text{T}(e_2).\text{global} - 2g_g\}\]

Definition 4.10 Closed Interval of Primitive Time stamps in Distributed Systems Let \(\text{T}(e_1), \text{T}(e_2)\) be the time stamps of global primitive event \(e_1\) and \(e_2\). A time stamp \(\text{T}(e)\) is said to be in the closed interval formed by \(\text{T}(e_1)\) and \(\text{T}(e_2)\) (requires \(\text{T}(e_1) \sim \text{T}(e_2)\)) denoted \(\text{T}(e) \in [\text{T}(e_1), \text{T}(e_2)]\) iff \(\text{T}(e_1) \sim \text{T}(e) \sim \text{T}(e_2)\).

Suppose \(\text{T}(e_1), \text{T}(e_2)\) are in different sites, then \(\text{T}(e_1) \preceq \text{T}(e) \preceq \text{T}(e_2)\) implies:

\[|\text{T}(e_1).\text{global} - \text{T}(e).\text{global}| \leq 1g_g\text{ and}\]
\[|\text{T}(e).\text{global} - \text{T}(e_1).\text{global}| \leq 1g_g\]
\[\implies |\text{T}(e_1).\text{global} - \text{T}(e_2).\text{global}| \preceq 1g_g\text{ or} \text{T}(e_1) \sim \text{T}(e_2)\text{ for a non-empty closed interval}\]

Intuitively, the closed interval

\[\tilde{[} \text{T}(e_1).\text{global}, \text{T}(e_2).\text{global}] = \{\text{T}(e_1).\text{global} - 1g_g, \text{T}(e_1).\text{global}, \ldots, \text{T}(e_2).\text{global}, \text{T}(e_2).\text{global} + 1g_g\}\]

The open and closed intervals of given two time stamps of events is showed in the Figure 1. The following proporportion provide a better understanding about the \(\sim, <, and \preceq\) relationship. Some of the properties are used for the proofs of the distributed composite time stamps defined next section. The differences between \(=\) and \(\sim\) relationship can be observed.

Proposition 4.2 Let \(\text{T}(e_1), \text{T}(e_2)\) and \(\text{T}(e_3)\) be any three primitive time stamps, then:

1. (asymmetric) If \(\text{T}(e_1) < \text{T}(e_2)\), then \(\neg(\text{T}(e_2) < \text{T}(e_1))\).
2. (antisymmetric) If $T(e_1) \preceq T(e_2)$ and $T(e_2) \preceq T(e_1)$ then $T(e_1) \sim T(e_2)$.

3. Either $T(e_1) < T(e_2)$, or $T(e_2) < T(e_1)$ or $T(e_1) \sim T(e_2)$ but no more than two of them holds.

4. Either $T(e_1) \preceq T(e_2)$, or $T(e_2) \preceq T(e_1)$ or both.

5. If $T(e_1) \sim T(e_2)$ and $T(e_1).site = T(e_2).site$ then $T(e_1) = T(e_2)$.

6. If $T(e_1) = T(e_2)$ and $T(e_1) < T(e_3)$ then $T(e_2) < T(e_3)$ regardless the sites of the events. Compare with the following: If $T(e_1) \sim T(e_2)$ and $T(e_1) < T(e_3)$ then $T(e_2) < T(e_3)$ does not hold. If $T(e_1) \sim T(e_2)$ and $T(e_2) \sim T(e_3)$ then $T(e_1) \sim T(e_3)$ does not hold. $(T(e_1).global = 1, T(e_2).global = 2, T(e_3).global = 3$ can be served as the counterexample of the both two cases).

7. If $T(e_1) < T(e_2)$ and $T(e_2) \sim T(e_3)$ then $T(e_1) \preceq T(e_3)$.

8. If $T(e_1) \sim T(e_2)$ and $T(e_2) < T(e_3)$ then $T(e_1) \preceq T(e_3)$.

9. If $\neg(T(e_1) < T(e_2))$ then $T(e_2) \preceq T(e_1)$.

10. If $\neg(T(e_1) < T(e_2))$ and $\neg(T(e_2) < T(e_1))$ then $T(e_1) \sim T(e_2)$.

5. Distributed Composite Event Semantics

The time stamp of composite events is different from the primitive events in the sense that the time stamps of a composite event may be a set of time stamps instead of just one time stamp in the primitive events. In the centralized system, the time stamp of a composite event is defined as the latest time occurrence of participating primitive events. Because of the partial ordered property of the global time, the “latest” is not uniquely defined. That is, there exists more than one global primitive time stamp which is the “latest” and these “latest” ones are concurrent to each other. This gives rise to multiple time stamps. We define the “maximum” time stamp in a set of time stamps to be the ones (again may not unique) that is not less than any other time stamp in the set. Those time stamps from a set called the “max” of the given set of time stamps and the time stamps in the set of “max” is proven to be concurrent to each other.

**Definition 5.1 Set of maximum Time Stamps** Given a set of time stamps $ST$, a time stamp $t \in ST$ is called a **maximum of ST** iff ($\forall t_1 \in ST, t < t_1$). The set of maximum time stamps in $ST$ is defined as $\max(ST) = \{ t \in ST : t \text{ is a maximum of ST} \}$

**Theorem 5.1** Given a set of primitive time stamps, $ST$, the time stamps in the max set $\max(ST)$ are concurrent to each other, i.e., $\forall t_1, t_2 \in \max(ST), t_1 \sim t_2$ where $\sim$ is the primitive concurrency relation defined last section.

**Definition 5.2 Time Stamp of the Distributed Composite Event** A time stamp of the distributed composite event $e$ denoted $T(e)$ is a set of triples ($site, global, local$). Each triple is a maximum of the set of time stamps of the constituent primitive event collected when the composite event occurs.

We automatically have the property that the primitive time stamps in a composite event time stamps are concurrent to each other, i.e., given $T(e)$ a composite time stamp, then $\forall t_1, t_2 \in T(e), t_1 \sim t_2$ where $\sim$ is the primitive concurrency relation defined last section.

Note that our definition of composite time stamps is different from [10]. In our definition, the concept of the “latest” is stressed and enforced by the definition and the concurrency within the time stamps is ensured by the theorem. Our definition is more precise in capturing the characteristics of the composite event time stamps.

5.1. Partial Ordering on the Distributed Composite Time Stamps

Suppose $\prec_p$ is the strict partial order temporal relation on the set of distributed composite time stamps. In order for $\prec_p$ to make sense, we have the following requirements:

1. If $T(e_1) \prec_p T(e_2)$ then at least $\exists t_1 \in T(e_1), \exists t_2 \in T(e_2)$ such that $t_1 < t_2$ where $<$ is the strict partial order temporal relation on primitive time stamps defined last section.

2. $\prec_p$ satisfies the irreflexivity and transitivity, i.e. it is a well-defined strict partial ordering.

3. $\prec_p$ is the “least” restricted in the sense that no existence ordering $\prec_q$ which is more restricted than $\prec_p$, i.e., $\forall T(e_1), \forall T(e_2), T(e_1) \prec_q T(e_2)$ if $T(e_1) \prec_p T(e_2)$ then $T(e_1) \prec_q T(e_2)$.

A definition that satisfies requirements 1 and 2 is valid but may not be acceptable. For example, if we define: $T(e_1) \prec_p T(e_2)$ iff $\forall t_1 \in T(e_1), \forall t_2 \in T(e_2), t_1.global < t_2.global - 10g_2$. The above definition satisfies 1 and 2 but is not acceptable because it is too “restricted” in the sense that there are a lot of time stamps that cannot be compared using the definition. It requires that all the constituent primitive time stamps of $T(e_2)$ to be at least $10g_2$ greater than all the ones in $T(e_1)$, which is too restricted and does not make sense. The third requirement is added to ensure that most pairs of time stamps can be compared.
Let \(<_p\) be a partial ordering satisfies the requirements, we have:

\[ T(e1) <_p T(e2) \Rightarrow \exists t1 \in T(e1), \exists t2 \in T(e2), t1 < t2 \]
\[ T(e2) <_p T(e3) \Rightarrow \exists t2 \in T(e2), \exists t3 \in T(e3), t2 < t3 \]

by our requirement 1. Also, the transitivity needs to be satisfied.

\[ T(e1) <_p T(e2), T(e2) <_p T(e3) \Rightarrow T(e1) <_p T(e3) \]

which implies that we need:

\[ \exists t1 \in T(e1), \exists t2 \in T(e2), t1 < t2 \quad (1) \]
\[ \exists t2 \in T(e2), \exists t3 \in T(e3), t2 < t3 \quad (2) \]
\[ \Rightarrow \exists t1 \in T(e1), \exists t3 \in T(e3), t1 < t3 \quad (3) \]

As the value of \(t2\) in (1) may be different from the value of \(t2\) in (2), we cannot conclude that the third equation holds, and hence we need at least one of the existential quantifiers (2) to be changed to the universal quantifier (\(\forall\)) in the above three equations. If not, there will always exist cases when the transitivity does not hold, i.e., the transitivity may not be guaranteed. So, a definition of \(\langle T(e1) <_p T(e2) \iff (\forall t2 \in T(e2), \exists t1 \in T(e1))(t1 < t2)\rangle\) or \(\langle T(e1) <_g T(e2) \iff (\forall t1 \in T(e1), \exists t2 \in T(e2))(t1 < t2)\rangle\) becomes the only two valid definitions with the “least restricted” strict ordering definition that satisfies the requirements. Any other valid definitions will need at least one \(\forall\) and one \(\exists\) to ensure the transitivity which implies a more restricted case. If we denote: “\(\langle T(e1) >_p T(e2) \iff (\forall t2 \in T(e2), \exists t1 \in T(e1))(t1 > t2)\rangle\)” and “\(\langle T(e1) >_g T(e2) \iff (\forall t1 \in T(e1), \exists t2 \in T(e2))(t1 > t2)\rangle\)” then \(T(e1) <_g T(e2) \iff T(e2) >_g T(e1)\) and \(T(e1) <_p T(e2) \iff T(e2) >_p T(e1)\). So, \(\langle <_p, >_g \rangle\) and \(\langle <_g, >_p \rangle\) are two dual pairs of ordering relation satisfying the above three requirements. In this paper, \(\langle <_p, >_p \rangle\) is chosen to be our definition of strict ordering of the composite event timestamps. Notice that the definition of \(T(e1) <_{p1} T(e2) \iff (\exists t1 \in T(e1), \exists t2 \in T(e2))(t1 < t2)\) is not valid since it is not transitive.

There are some other interesting definitions that are valid but not “least” restricted:

1. \(T(e1) <_{p2} T(e2) \iff (\forall t1 \in T(e1), \forall t2 \in T(e2))(t1 < t2)\).

This one is valid but is more restricted than \(<_p\). Example: \(T(e1) = \{(site1, 8, 80), (site2, 7, 70)\}\) and \(T(e2) = \{(site3, 9, 90)\}\) satisfies \(<_p\) relation but not the above \(<_{p2}\) relation.

2. Let \(\min\) be the time stamp in \(T(e1)\) with minimum global time, \(T(e1) <_{p3} T(e2) \iff \forall t2 \in T(e2), (\min t1 < t2)\).

This one is also valid but is more restricted than \(<_p\). Example: \(T(e1) = \{(site1, 8, 80), (site2, 7, 70)\}\) and \(T(e2) = \{(site1, 8, 81), (site2, 7, 71)\}\) satisfies \(<_p\) relation but not this \(<_{p2}\) one, since \((site1, 8, 81)\) in \(T(e2)\) is \(\not<_{p2}\) \((site2, 7, 70)\) in \(T(e1)\) which has the minimal global time.

Based on the above analysis, we have the following definition:

**Definition 5.3 Temporal Relationship on Composite Time Stamps** Let \(T(e1)\) and \(T(e2)\) be the two distributed composite event time stamps. Then the temporal relationship between \(T(e1)\) and \(T(e2)\) is defined as the following:

1. Concurrency: \(T(e1) \sim T(e2) \iff (\forall t1 \in T(e1), \forall t2 \in T(e2))(t1 \sim t2)\)

2. Happen before: \(T(e1) < T(e2) \iff (\forall t2 \in T(e2), \exists t1 \in T(e1))(t1 < t2)\)

3. Incomparable: \(T(e1) \not\sim T(e2) \iff (T(e1) < T(e2) \lor (T(e1) > T(e2)) \lor (T(e1) \sim T(e2)))\)

**Theorem 5.2** The \(<\) defined above is irreflexive and transitive; hence it is a well-defined strict partial ordering on the set of all distributed composite event time stamps.

A semantics of \(\tilde{\sim}\) can be defined as follows. Interestingly enough, this definition can be proved to be consistent with the definition of primitive \(\tilde{\approx}\) as well.

**Definition 5.4 weaker-less-than-or-equal-to** \(T(e1) \tilde{\sim} T(e2) \iff (\forall t1 \in T(e1) \forall t2 \in T(e2))(t1 \tilde{\sim} t2)\).

**Theorem 5.3** \(T(e1) \tilde{\sim} T(e2) \iff T(e1) \sim T(e2)\) or \(T(e1) < T(e2)\).

Our definition is different from that of [10]. The “happen before” definition in [10] is not transitive although it is indicated as such in the thesis. The following example provides a counter example: Let \(T(e1) = \{(site1, 8, 80), (site2, 2, 80)\}\), \(T(e2) = \{(site1, 9, 90), (site2, 8, 80)\}\), and \(T(e3) = \{(site2, 9, 90)\}\). It is easy to see that \(T(e1) \sim T(e2), T(e2) < T(e3)\) by their definition but \(T(e1) \sim T(e3)\). So, their definition of \(<\) is not a well-defined strict partial order temporal relation. On the other hand our definition depends only on the definition of the distributed primitive event time stamp. The concepts of “sites”, “global time” and “local time” are embedded in the definition of primitive event time stamps and become transparent to composite event. One other obvious advantage is that our definition is well-defined mathematically and hence ensure to be the “best” by the above mathematical reasoning. A semantics of open and closed interval which is needed for the complexity of Sentinel composite event operator can be derived easily from the \(<\) ordering. Also, the temporal relation between any two distributed time stamps is very easy to visualize via our graph presentation shown below.

The time stamps in distributed system can be represented as a two dimensional grid with the X-axis being the global time embedded with the local time and the Y-axis being
Let \( T(e_1), T(e_2) \) be the sites in the distributed environment. The following is an example indicating the composite time stamp and its \( \sim, <, \triangleq \) area. Let the global distributed composite event be \( T(e) = \{(Site3, 8, 81), (Site6, 7, 72)\}. \) For any composite time stamp \( T(e_1), T(e_1) \sim T(e) \) iff \( T(e_1) \) lies between Line2 and Line3; \( T(e_1) < T(e) \) iff \( T(e_1) \) lies before Line1; \( T(e) < T(e_1) \) iff \( T(e_1) \) lies after Line4; \( T(e_1) \triangleq T(e) \) iff \( T(e_1) \) lies before Line3; and \( T(e) \triangleq T(e_1) \) iff \( T(e_1) \) lies after Line2; See Figure 2. A time stamp going across those lines indicates an incomparable situation.

Analogously, the notation of Open and Closed interval can defined as follows:

**Definition 5.5 Open Interval of Composite Time stamps in Distributed Systems** Let \( T(e_1), T(e_2) \) be the time stamps of global composite event \( e_1 \) and \( e_2 \). An event \( e \) with time stamp \( T(e) \) is said to be in the open interval formed by \( T(e_1) \) and \( T(e_2) \) with \( T(e_1) < T(e_2) \) denoted \( T(e) \in (T(e_1), T(e_2)) \) iff \( T(e_1) < T(e) < T(e_2) \).

**Definition 5.6 Closed Interval of Composite Time stamps in Distributed Systems** Let \( T(e_1), T(e_2) \) be the time stamps of global composite event \( e_1 \) and \( e_2 \). A time stamp \( T(e) \) is said to be in the closed interval formed by \( T(e_1) \) and \( T(e_2) \) (requires \( T(e_1) \triangleq T(e_2) \)) denoted \( T(e) \in [T(e_1), T(e_2)] \) iff \( T(e_1) \triangleq T(e) \triangleq T(e_2) \).

Below, we give an example that demonstrates the temporal relationship of global event time stamps. Let \( k, l, m \) be a set of physical clocks in different sites of the Distributed system with granularity \( g = 1/100s \). Let \( z \) be the reference clock with granularity \( g_z = 1/1000s \). Assume the physical clocks are synchronized with precision \( \Pi < 1/10s \) and the global granularity is chosen to be \( g_g = 1/10s \). Let \( T(e_1), \ldots, T(e_5) \) be the sets of time stamps of the global composite events \( e_1, \ldots, e_5 \):

- \( T(e_1) = \{(k, 9154827, 91548276), (m, 9154827, 91548277)\} \)
- \( T(e_2) = \{(l, 9154827, 91548276), (k, 9154827, 91548277)\} \)
- \( T(e_3) = \{(m, 9154827, 91548276), (l, 9154827, 91548277)\} \)
- \( T(e_4) = \{(k, 9154828, 91548288), (l, 9154827, 91548277)\} \)
- \( T(e_5) = \{(k, 9154829, 91548289), (l, 9154828, 91548287)\} \)

Then, \( T(e_1) \cap T(e_2) \cap T(e_3), T(e_4) \sim T(e_3) \) and \( T(e_3) < T(e_5) \)

**5.2. Distributed Composite Time Stamp Operation**

In centralized systems, when a composite event is detected, a new time stamp \( t_{occ} \) indicating the latest time stamp (that made the composite event occur) along with the event name, and event parameters are propagated to the parents node(if any). Similarly, in distributed systems, when the composite event is detected, a set of time stamps indicating the “latest” time stamps will be generated and sent to the parent node with the event type and parameters. This procedure of generating a set of “latest” time stamps is defined as a Max operator. The resulting composite time stamp generated by the Max operator also needs to satisfies the definition of the distributed composite time stamps. The Max operator can be specified is several ways depending on the relationship between two time stamps. Before defining the Max of two sets of time stamps, the joining of concurrent and incomparable time stamps are defined as follows:

**Definition 5.7 Joining procedure concurrent time stamps** Let \( T(e_1) \) and \( T(e_2) \) be two global time stamps such that \( T(e_1) \sim T(e_2) \). Then the joining time stamp of \( T(e_1) \) and \( T(e_2) \) denoted \( T(e_1) \cup T(e_2) \) is defined as:

\[ T(e_1) \cup T(e_2) = \{ts|ts \in T(e_1) or ts \in T(e_2)\} \]

**Definition 5.8 Joining procedure on incomparable time stamps** Let \( T(e_1) \) and \( T(e_2) \) be two global time stamps such that \( T(e_1) \cap T(e_2) \). Then the joining time stamp of \( T(e_1) \) and \( T(e_2) \) denoted \( T(e_1) \cap T(e_2) \) is defined as:

\[ T(e_1) \cap T(e_2) = \{ts|ts \in T(e_1) such that \exists ts_2 \in T(e_2), ts < ts_2\} \]

\[ \cup \{ts|ts \in T(e_2) such that \exists ts_1 \in T(e_1), ts < ts_1\} \]

The “joining” of \( T(e_1) \sim T(e_2) \) is simply merging the corresponding components and eliminating the duplicates which is the same as the join operation defined on sets (or relations). The concurrency property is ensured (see [12] for proofs of theorems and additional propositions). The “joining” of \( T(e_1) \cap T(e_2) \) is to keep the “latest” information of the two sets of time stamps. The resulting composite time stamps after joining operation can be proved to satisfy the definition of the composite time stamps. The above two
joining procedure are conceptually same as the joining in [10]. Our definition is more precise and easy to understand.

Definition 5.9 the Maximum of time stamps Let \( T(e_1) \) and \( T(e_2) \) be two sets of time stamps of distributed composite events, then the maximum of \( T(e_1) \) and \( T(e_2) \) denoted \( \text{Max}(T(e_1), T(e_2)) \) is defined as follows:

\[
\text{Max}(T(e_1), T(e_2)) = \begin{cases} 
T(e_1) & \text{if } T(e_2) < T(e_1) \\
T(e_2) & \text{if } T(e_1) < T(e_2) \\
T(e_1) \cup T(e_2) & \text{if } T(e_1) \text{ and } T(e_2) \text{ are concurrent or incomparable}
\end{cases}
\]

Theorem 5.4 If \( T(e) = \text{Max}(T(e_1), T(e_2)) \) where \( T(e_1) \) and \( T(e_2) \) are any time stamps then \( T(e) \) is a distributed composite time stamp with the primitive components from \( T(e_1) \) and \( T(e_2) \), i.e., let \( ST = T(e_1) \cup T(e_2) \) where \( \cup \) is the ordinary mathematics union operator, then \( T(e) = \text{max}(ST) \).

5.3. Distributed Composite event Semantics

A distributed event \( E \) (either primitive or composite) is a function from the time stamp domain onto the boolean values, True or False. \( E : TS \rightarrow \{\text{True}, \text{False}\} \) given by:

\[
E(t_s) = \begin{cases} 
T(\text{true}) & \text{if an event of type E occurs} \\
F(\text{false}) & \text{otherwise}
\end{cases}
\]

Here the time stamp \( t_s \) could be a set of time stamps formed by the \( \text{Max} \) operator by collecting time stamps from the corresponding primitive events if \( E \) is a global composite event. Based on the time stamp set operation defined in last section, the semantics of Snoop operators for detecting composite event expressions in a distributed environment can be defined as follows. We elaborate on a few of them for lack of space (see [12] for the others):

AND (\( \land \)): Conjunction of two events \( E_1 \) and \( E_2 \):

\[
(E_1 \land E_2)(t_s) = (\exists t_1, t_2)(E_1(t_1) \land E_2(t_2))
\]

Sequence operator (\( ; \)):

\[
(E_1; E_2)(t_s) = (\exists t_1, t_2)(E_1(t_1) \land E_2(t_2)) \land (t_1 < t_2)
\]

Aperiodic Operators (\( A, A^* \)). Non-cumulative variant of an aperiodic event:

\[
A(E_1, E_2, E_3)(t_s) = (\exists t_1)(\forall t_2)(t_1 < t_2 \land E_1(t_1) \land E_2(t_2) \land E_3(t_2))
\]

The cumulative version:

\[
A^*(E_1, E_2, E_3)(t_s) = (\exists t_1, t_1 < t_2)(E_1(t_1) \land E_2(t_2) \land E_3(t_2))
\]

The not operator (\( \neg \)):

\[
\neg(E_2)(E_1, E_3)(t_s) = (\exists t_1)(\forall t_2)(t_1 < t_2 \land (E_1(t_1) \land E_3(t_2))
\]

6. Conclusions

This paper presents a mathematical framework of ordered sets relation for distributed composite event detection. A new definition of distributed composite time stamps is defined ensuring the “maximum” and “concurrency” properties of the constituent primitive time stamps. The strict ordering relation on the time stamps for the comparison in a distributed system is chosen carefully to have the “least restrictive” property and is well-defined. A Max operator which is well-integrated with the definition of time stamps is introduced for handling and propagating the time stamps. A number of properties are presented (rigorously proved in [12] for providing a better understanding of the semantics of distributed time stamps and the partial ordering. Finally, using the formalism developed, the semantics of Sentinel composite event operators is extended to the distributed environment.

References