Synchronization for Distributed Real-time Applications

Ajay D. Kshemkalyani
IBM Corporation, P. O. Box 12195, Research Triangle Park, NC 27709, USA
Email: ajayk@vnet.ibm.com

Abstract

Several distributed applications are characterized by real-time constraints on response times. High-level actions in such distributed applications are modeled by nonatomic events which are collections of atomic events. This paper studies synchronization relations between nonatomic events in such distributed real-time application executions. The relations form a fine-grained hierarchy that can be used to select suitable relations with good properties and clear intuitions, depending on the application. We show the use of the proposed relations in specifying solutions for distributed mutual exclusion and in specifying distributed predicates. As an example of an application, we show how a real-time air defence system that needs to enforce distributed mutual exclusion and specify and evaluate distributed predicates can use the proposed relations.

1 Introduction

Several distributed applications are characterized by real-time constraints on response times. High-level actions in such distributed application executions are realistically modeled by nonatomic events or collections of atomic events. This paper is concerned with synchronization and causality relations between nonatomic events in such distributed application executions. The synchronization and causality relations we consider are useful to sophisticated applications that require a fine level of discrimination of various synchronization relations. This topic is important because distributed real-time applications are getting more complex and sophisticated and thus require a fine level of granularity of the causality and synchronization relations; this theme has not received any attention in the real-time literature, e.g., [10, 22, 24], which has focused on various aspects of strict timing constraints.

1.1 Objectives

The causality relation represents the partial order of events in a system execution [4, 6, 7, 15, 23, 25]. Thus far, the causality relation has been studied primarily between single events in space-time. This is a very simplistic model because in reality, objects or entities that are modeled occupy more than a single point in space. The notion of grouping elementary events in the computation into higher level nonatomic events [12, 13, 14, 16, 17] is useful for event abstraction [16, 17]; it provides simplicity to the programmer and system designer in reasoning at the appropriate level of complexity by reducing the amount of information to be handled. Modeling nonatomic events has many applications identified in Section 1.2. The event abstraction inherent in nonatomic events, however, results in a loss of power to express and reason with various degrees of causality ! It is desirable to have rich expressive power for various degrees of causality between a pair of nonatomic events for sophisticated applications. We propose and examine causality relations between two nonatomic poset events in a distributed system execution without assuming a global time axis.

1.2 Real-time applications

Modeling nonatomic events is very useful in real-time applications [24] such as industrial process control, distributed multimedia support [21], coordination in mobile computing, avionics, terrestrial, undersea and aerial navigation, planning, robotics, virtual reality, and temporal and geographic databases, All these applications deal with entities that occupy more than a single point in space at any time. For these applications, the traditional causality relation [4, 6, 7, 15, 23, 25] defined between individual points in space-time is coarse-grained and inadequate when applied to the abstracted nonatomic events, for the following reasons. (i) The interaction between two nonatomic events cannot be captured at a fine level of discrimination using various degrees of causality, as required for a sophisticated and realistic modeling of these applications. (ii) The synchro-

0-8186-8096-2/97 $10.00 © 1997 IEEE

nization conditions between two nonatomic events cannot be richly specified using various degrees of causality, as required for a sophisticated and realistic modeling of these applications. Therefore, a rich set of causality relations that provide various degrees causality to accurately represent and specify real-life relationships between nonatomic objects/events needs to be defined. It is also important to evaluate the relations efficiently in real-time, particularly as most of the distributed applications are highly time-critical.

Some of the above real-time applications are elaborated on next. In industrial process control, events at temperature and pressure sensors at different places may be grouped together as a nonatomic event depending on the complex equations in physical chemistry or thermodynamics that have to be satisfied in industrial manufacturing. When certain predicates, specified by the equations, become true between a pair of such nonatomic events, rules are triggered to perform actions such as changing the temperature and pressure. Rapid real-time responses are also critical for the above processes [10, 24]. Distributed multimedia applications require real-time synchronization within a media, among the media at a site, and among different sites [21]. For multimedia applications such as videoconferencing, it is sufficient to have a part of the coordination among multiple sites based on causality relations. The synchronization conditions can be specified among different sites and on different data streams by viewing aggregates of elementary events as nonatomic events. Planning and navigation in robotics, space navigation, deep-sea navigation and combat are classic examples of mission-critical applications that require the real-time evaluation of causality between complex moving objects, each of which is distributed across space-time. To exercise fine control on the navigation and to track multiple targets and avoid multiple obstacles, it is necessary to specify conditions and reason with a fine-grained suite of causality relations. Similarly, temporal and geographic databases deal with collections of events in space and time. There is a need to evaluate in real-time the causality relations between such collections of events. Virtual reality is another application where there is a need to perform a real-time evaluation of causality between complex objects in a dynamically changing environment, such as a flight cockpit simulator. To track multiple targets and collision courses, it is necessary to deal with fine-grained causality relations to express synchronization conditions. In mobile computing systems, groups of mobile users that are scattered across multiple cells need to coordinate and synchronize their efforts. It is useful to synchronize their actions at a fine degree of causality rather than using the coarser barrier synchronization.

All the above applications benefit by the availability of fine-grained causality relations that are used in synchronization conditions and distributed predicates. The use of the proposed relations for distributed mutual exclusion and for specification of distributed predicates in the context of an air defence system is given in Sections 3.2 and 3.3.

1.3 Model

We use the space-time model for a distributed system execution. This model is a poset event structure model as in [12, 16, 23, 25]. Consider a poset $(E, \prec)$ where $\prec$ is an irreflexive partial ordering. Let $\mathcal{E}$ denote the power set of E. Let $\mathcal{A}$ $(\neq \emptyset) \subseteq (\mathcal{E} - \emptyset)$. Thus, there is an implicit one-many mapping from $\mathcal{A}$ to E. Each element A of $\mathcal{A}$ is a non-empty subset of E, and is termed an *interval* or a *nonatomic event*. (We will use the term "interval" interchangeably with "event" when referring to nonatomic events.) $(E, \prec)$ represents points in space-time which are the most primitive atomic events related by the causality relation. Elements of E are partitioned into local executions at a coordinate in the space dimensions. Each local execution E_i is a linearly ordered set of events in partition i. An event e in partition i is denoted e_i.

For a distributed computer system, points in the space dimension will be the set of processes (also termed *nodes*), and E_i will be the set of events executed by process i. Causality between events at different nodes is imposed by message communication.

1.4 Previous work

There is no well-understood notion of causality between two nonatomic poset events in a distributed system execution, wherein some events in one high-level nonatomic event causally precede some events in the other high-level nonatomic event. Most previous work assumed that the nonatomic events were linearly ordered, e.g., [3, 4, 6, 7] – and confined the study of causality to relations between time durations or linear intervals. [6] includes an exhaustive review of related literature. The causality relations defined in the literature above also assumed that such linear nonatomic events occurred at a single point in space, implying the existence of a global time axis. But in a distributed system, there is no global time axis [1, 23].

The following is the only literature that deals with causality between nonatomic poset events in a distributed system execution and does not assume a global time axis. Two relations $\longrightarrow$ and $--\rightarrow$ between nonatomic elements in a system execution were defined in [16, 17] as follows. Let a nonatomic event be a set of atomic events. For two nonatomic events X and Y, $X \longrightarrow Y$ iff every atomic event in X causally precedes every atomic event in Y. $X --\rightarrow Y$ iff some atomic event in X causally precedes some atomic event in Y. These definitions were used in the context of interprocess communication and mutual exclusion which use

Relation r	Expression for $r(X,Y)$
$R1$	$\forall x \in X \forall y \in Y, x \prec y$
$R1'$	$\forall y \in Y \forall x \in X, x \prec y$
$R2$	$\forall x \in X \exists y \in Y, x \prec y$
$R2'$	$\exists y \in Y \forall x \in X, x \prec y$
$R3$	$\exists x \in X \forall y \in Y, x \prec y$
$R3'$	$\forall y \in Y \exists x \in X, x \prec y$
$R4$	$\exists x \in X \exists y \in Y, x \prec y$
$R4'$	$\exists y \in Y \exists x \in X, x \prec y$

Table 1. Relations in [12].

relation of row header to column header	$R1$	$R2$	$R3$	$R4$
$R1$	$=$	$\sqsubseteq$	$\sqsubseteq$	$\sqsubseteq$
$R2$	$\sqsupseteq$	$=$	$\parallel$	$\sqsubseteq$
$R3$	$\sqsupseteq$	$\parallel$	$=$	$\sqsubseteq$
$R4$	$\sqsupseteq$	$\sqsupseteq$	$\sqsupseteq$	$=$

Table 2. Inclusion relationships between relations, from [12].

nonatomic linear intervals. The above two relations were further studied in [1].

In an earlier paper [12], we showed that the two relations defined in [16, 17] are not sufficient to capture the essential temporal properties of system executions and specify causality constraints between nonatomic events in distributed systems. In [12], we proposed a set of new relations between nonatomic events in a distributed system to capture a spectrum of causality specifications, without assuming a global time axis. These relations $R1$ - $R4$ and $R1'$ - $R4'$ from [12] are expressed in terms of the quantifiers over X and Y in Table 1.

The following is an interpretation of the relations in Table 1 introduced in [12].

R1 : Same as $\longrightarrow$. $R1(X,Y)$ iff every event in interval X causally precedes every event in interval Y.

$R1$ signifies that interval Y begins after it fully knows the result of interval X.

R2 : $R2(X,Y)$ iff every event in interval X causally precedes some event in interval Y. (See [12].)

$R2$ signifies that interval Y completes after it fully knows the result of interval X. However, no one event in Y may be able to take actions based on the complete result of X.

R3 : $R3(X,Y)$ iff some event in interval X causally precedes every event in interval Y. (See [12].)

$R3$ signifies that interval Y can be fully controlled by some common input from X. However, the complete input from X may not be received by all of Y or even some of Y.

R4 : Same as $-- \rightarrow$. $R4(X,Y)$ iff some event in interval X causally precedes some event in interval Y.

$R4$ signifies that interval Y completes after it receives some input from interval X.

Note that all the relations in Table 1 are not independent relations. Table 2 gives the inclusion relationship of the causality relations $R1$ - $R4$. Each cell in the grid indicates the relationship of the row header to the column header. The notation for the inclusion relationship between causality relations on nonatomic events is as follows. The inclusion relation "is a subrelation of" is denoted '$\sqsubseteq$'. '$\sqsupseteq$' is the inverse of $\sqsubseteq$. '$=$' stands for equality between relations in addition to its standard usage as the equality in other contexts. For two causality relations r_1 and r_2, we define $r_1 \parallel r_2$ to be $(r_1 \not\sqsubseteq r_2 \bigwedge r_2 \not\sqsubseteq r_1)$. The relations $\{ R1, R2, R3, R4 \}$ form a lattice hierarchy ordered by $\sqsubseteq$.

Table 1 also defined relations $R1'$, $R2'$, $R3'$, and $R4'$, for which the order of quantifiers was reversed from the order in $R1$, $R2$, $R3$, and $R4$, respectively. Observe that the relations $R2'$ and $R3'$ are different from relations $R2$ and $R3$, respectively, when applied to posets. However, for a linear interval, they are the same as $R2$ and $R3$, respectively. $R1'$ and $R4'$ are the same as $R1$ and $R4$, respectively.

Action refinement of posets has been extensively studied [9, 20] along with related work in Petri nets [18]. In these areas, there is no known work that addresses specific causality relations between nonatomic poset events.

1.5 Relation of contribution to previous work

The set of relations proposed in [12] formed an exhaustive set of causality relations to express all possible interactions between a pair of linear intervals and extended the incomplete hierarchy of relations of [16, 17]. However, when the relations of [12] are applied to a pair of poset intervals, the hierarchy they form is incomplete. We formulate causality relations between a pair of nonatomic poset intervals along the lines of [11, 12] by extending these results to nonatomic poset events. The relations form an "exhaustive" set of causality relations between nonatomic poset events using first-order predicate logic and fill in the existing partial hierarchy of causality relations between nonatomic poset events, formed by relations in [12, 16, 17].

Organization: Section 2 discusses how the proposed relations fill in the existing partial hierarchy of causality relations. Section 3 gives the significance of the proposed relations and shows their use in distributed mutual exclusion and specification of distributed predicates. The results of this paper are included in [11].

2 Causality relations between nonatomic poset events

Causality relations between poset intervals capture interactions between arbitrary and complex groupings of events. The proposed causality relations are defined using first-order predicate logic over $\mathcal{A} \times \mathcal{A}$, where $\mathcal{A}$ is the set of nonatomic poset events of interest to the application.

Definition 1 *An interval A is linear iff* $\forall x, y \in A, x \preceq y \bigvee y \preceq x$.

Definition 2 *The node set of interval A, denoted* $N_A = \{i | E_i \bigcap A \neq \emptyset\}$.

Our results apply to nonlinear, i.e., poset, intervals. The cardinality of the node set of the intervals we consider is greater than one.

All the relations in [12] will be used in Section 2.2 to derive an exhaustive suite of causality relations between nonatomic poset events, denoted as $\mathcal{R}$. As an intermediate step, we propose definitions of certain proxies of a nonatomic event in Section 2.1.

2.1 Proxies of nonatomic poset events in $\mathcal{A}$

In the extensive literature on linear intervals and time durations, for example [3, 4, 6, 7], an interval is always identified by the instants of its beginning and end. The beginning and end instants of a linear interval are atomic events. For a nonatomic poset interval in $\mathcal{A}$, it is natural to identify counterparts for the beginning and end instants. These counterparts will serve as "proxy" events for the poset interval just as the events at the beginning and end of linear intervals such as time durations serve as proxies for the linear interval. The proxies identify the durations on each node, in which the nonatomic event occurs. This is the first motivation to have proxies for nonatomic poset events.

For a pair of non-atomic events $X, Y \in \mathcal{A}$, there are $|X| \times |Y|$ pairs of causality relations between the atomic elements of X and Y. A naive definition of causality would require $|X| \times |Y|$ checks for causality. Clearly, we would like to reduce this complexity. This can be achieved by defining causality between the elements of the proxies of X and Y, respectively, because a proxy is a meaningful, representative subset of the interval. This is the second motivation to define proxies for nonatomic poset events. We show that the evaluation of our causality relations can be reduced to $|N_X| \times |N_Y|$ checks for causality (Definition 2) by choosing appropriate proxy events between which causality is checked.

We now define two proxies corresponding to the beginning and end of a nonatomic interval [11].

Definition 3
- $L_X = \{e_i \in X | \forall e_i' \in X, e_i \preceq e_i'\}$
- $U_X = \{e_i \in X | \forall e_i' \in X, e_i' \preceq e_i\}$

For any poset X, L_X and U_X are the set of the minimal elements in X for each node and the set of the maximal elements in X for each node, respectively. L_X and U_X correspond to the beginning of the poset and the end of the poset, respectively, and can act as a *proxy* for poset X, depending on context and application. As per Definition 3, each of L_X and U_X contains one event from each node in N_X.

An equally valid interpretation of the beginning and end of a poset are the sets of its minimal and maximal elements, respectively, as defined by the irreflexive partial order across the nodes. This leads to the following alternate definition of the proxies L_X and U_X.

Definition 4
- $L_X = \{e \in X | \forall e' \in X, e' \not\prec e\}$
- $U_X = \{e \in X | \forall e' \in X, e \not\prec e'\}$

L_X is the largest anti-chain containing the minimal elements of X. U_X is the largest anti-chain containing the maximal elements of X.

The causality relations between poset intervals will be derived using proxies and will depend on whether proxies are defined by Definition 3 or by Definition 4. We assume that any one of these definitions of proxies is consistently used, depending on context and application.

2.2 Deriving the relations $\mathcal{R}$

We propose that there are two aspects of a relation that can be specified between poset intervals. One aspect deals with the determination of an appropriate *proxy* for each interval. A good choice for the proxy(ies) of the interval are the beginning and end of the interval (Definition 3 or 4), as justified in Section 2.1. Relations between posets are not specified on members $Z \in \mathcal{A}$, but rather on their proxies U_Z and L_Z. The second aspect of specifying relations $r(X, Y)$ specifies how the chosen proxies of X and Y are related. Figure 1 depicts the proxies of X and Y and serves as a visual aid for the following discussion; recall that each poset X and Y represents a grouping of atomic events of interest to the application.

A proxy for X and Y can be chosen in $C_1^2 \times C_1^2$ ways; it can be the set of maximal elements or minimal elements for each of X and Y. This is the first aspect of specifying relations between posets, and corresponds to the relations in $\{R1, R2, R3, R4\}$. From Table 2, it follows that these four relations form a lattice.

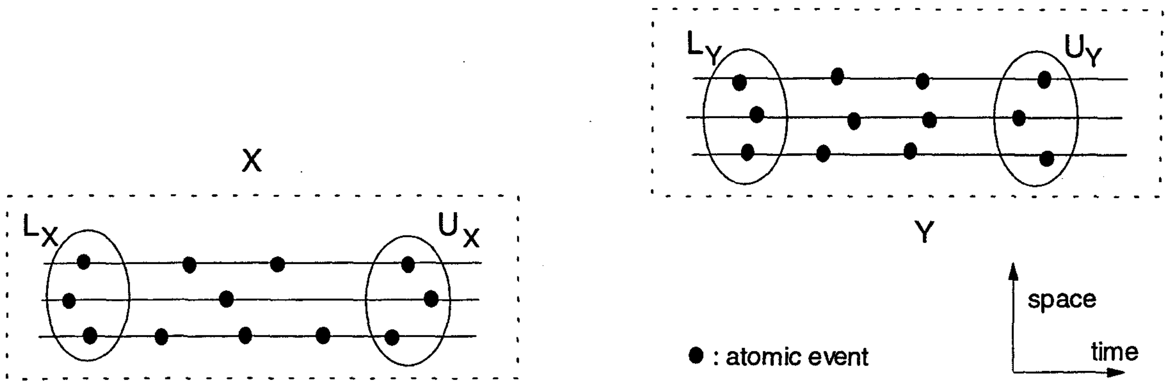

Figure 1. Poset events X and Y and their proxies.

Observe that the relations $R2'$ and $R3'$ defined in Table 1 are different from relations $R2$ and $R3$, respectively, when applied to posets. However, for a linear interval, they are the same as $R2$ and $R3$, respectively. The relations $R1'$ and $R4'$ are the same as the relations $R1$ and $R4$, respectively.

R2' : $R2'(X,Y)$ iff $\exists y \in Y, \forall x \in X, x \prec y$

$R2'(X,Y)$ iff some event in interval Y is preceded by every event in interval X. $R2'(X,Y)$ signifies that Y completes after some one common event in Y knows the full result of X.

R3' : $R3'(X,Y)$ iff $\forall y \in Y, \exists x \in X, x \prec y$

$R3'(X,Y)$ iff every element in interval Y is preceded by some event in interval X. $R3'(X,Y)$ signifies that all events in Y are controlled by some (not necessarily the same) input from X.

The second aspect of specifying the causality relations between posets deals with how the atomic elements of the chosen proxies of X and Y are related by causality. There are $C_1^2 \times C_1^2$ combinations of distinct quantifiers $\exists$ and $\forall$ over the proxies of X and Y to express $r(X,Y)$, and for each combination, there are P_1^2 permutations of the proxies of X and Y. The eight relations so formed correspond to $R1$, $R1'$, $R2$, $R2'$, $R3$, $R3'$, $R4$, $R4'$ of Table 1 and are renamed a, a', b, b', c, c', d, d', respectively, to avoid confusion with their original names used for the first aspect of specifying the relations between poset intervals. Table 3 gives the hierarchy and inclusion relationship among relations a, a', b, b', c, c', d, d'. Each cell in the table indicates the relation of the row header to the column header. The notation used is the same as that used for Table 2. Note that a and a', as well as d and d' are equivalent. The inclusion hierarchy among the six distinct relations forms a lattice ordered by $\sqsubseteq$.

The set of relations in the second column of Table 4 is the set $\mathcal{R}$ [11]. Each relation is formed by combining the two aspects of deriving causality relations as described above, and is labeled in the first column as follows. The relations $R1$, $R2$, $R3$, and $R4$ for linear intervals correspond to the groups of relations $R1*$, $R2*$, $R3*$ and $R4*$, respectively, for poset intervals. The hierarchy among the relations $R1*$, $R2*$, $R3*$ and $R4*$ is isomorphic to the hierarchy among $R1$, $R2$, $R3$, and $R4$.

Relations $R1*(X,Y)$ relate certain activity of U_X to certain activity of L_Y. Specifically, $R1*(X,Y)$ could be specified by quantifying over all or some elements of U_X, all or some elements of L_Y, and the order of the quantifications of the proxies of X and Y can be permuted. Thus there are eight possibilities for $R1*(X,Y)$, that correspond to relations $\{a, a', b, b', c, c', d, d'\}$. Similarly, $R2*(X,Y)$ relate certain activity of U_X to certain activity of U_Y. $R3*(X,Y)$ relate certain activity of L_X to certain activity of L_Y. $R4*(X,Y)$ relate certain activity of L_X to certain activity of U_Y. For each of $R2*(X,Y)$, $R3*(X,Y)$, and $R4*(X,Y)$, there are eight possible relations like for $R1*(X,Y)$.

The relations $\{ R1*, R2*, R3*, R4* \}$ between proxies for X and Y, and the relations $\{ a, a', b, b', c, c', d, d' \}$ between the elements of the proxies, when multiplied give 32 relations over the domain $\mathcal{A} \times \mathcal{A}$ to express $r(X,Y)$. The resulting set of poset relations, given in the second column of Table 4, is thus a product of the relations represented by the two lattices $\{ R1*, R2*, R3*, R4* \}$ and $\{ a, b, b', c, c', d \}$ of unique elements, as shown in Figure 2. This hierarchy is captured by the following constraints (axioms) XP1-XP6. Let V_1 denote the set $\{1,2,3,4\}$ and let V_2 denote the set $\{a, b, b', c, c', d\}$. The axioms are as follows:

XP1: $R1? \sqsubseteq R2? \sqsubseteq R4?$, where ? is instantiated from V_2

XP2: $R1? \sqsubseteq R3? \sqsubseteq R4?$, where ? is instantiated from V_2

XP3: $R2?||R3\#$, where ? and # are separately instantiated from V_2

XP4: $R?a \sqsubseteq R?b' \sqsubseteq R?b \sqsubseteq R?d$, where ? is instantiated from V_1

Relation names: its quantifiers for $x \prec y$	$R1$, a (=$R1'$, a'): $\forall x \forall y$ (=$\forall y \forall x$)	$R2$, b: $\forall x \exists y$	$R2'$, b': $\exists y \forall x$	$R3$, c: $\exists x \forall y$	$R3'$, c': $\forall y \exists x$	$R4$, d (=$R4'$, d'): $\exists x \exists y$ (=$\exists y \exists x$)
$R1$, a (=$R1'$, a') : $\forall x \forall y (= \forall y \forall x)$	$=$	$\sqsubseteq$	$\sqsubseteq$	$\sqsubseteq$	$\sqsubseteq$	$\sqsubseteq$
$R2$, b: $\forall x \exists y$	$\sqsupseteq$	$=$	$\sqsupseteq$	$\parallel$	$\parallel$	$\sqsubseteq$
$R2'$, b': $\exists y \forall x$	$\sqsupseteq$	$\sqsubseteq$	$=$	$\parallel$	$\parallel$	$\sqsubseteq$
$R3$, c: $\exists x \forall y$	$\sqsupseteq$	$\parallel$	$\parallel$	$=$	$\sqsubseteq$	$\sqsubseteq$
$R3'$, c': $\forall y \exists x$	$\sqsupseteq$	$\parallel$	$\parallel$	$\sqsupseteq$	$=$	$\sqsubseteq$
$R4$, d (=$R4'$,d'): $\exists x \exists y (= \exists y \exists x)$	$\sqsupseteq$	$\sqsupseteq$	$\sqsupseteq$	$\sqsupseteq$	$\sqsupseteq$	$=$

Table 3. Full hierarchy of relations of Table 1 [12]. Relations $R1$, $R1'$, $R2$, $R2'$, $R3$, $R3'$, $R4$, $R4'$ of Table 1 are renamed a, a', b, b', c, c', d, d', respectively. Relations in the row and column headers are defined between X and Y.

XP5: $R?a \sqsubseteq R?c \sqsubseteq R?c' \sqsubseteq R?d$, where ? is instantiated from V_1

XP6: $R?b \| R?c'$, $R?b' \| R?c'$, $R?b \| R?c$, $R?b' \| R?c$, where ? is instantiated from V_1

The resulting hierarchy of 24 unique relations provides a fine-grained choice of causality relations for specification of concurrency and synchronization conditions. The suite of causality relations we formulated is "complete" under first-order predicate logic.

Note that by construction, $(\mathcal{R}, \sqsubseteq)$ is a partial order. For a given pair of posets X and Y, it may be the case that a combination of the relations in $\mathcal{R}$ may hold. Specifically, if $R(X,Y)$ holds, then $\forall R' \mid R \sqsubseteq R'$, $R'(X,Y)$ holds. Thus, if $R(X,Y)$ holds, then for each R' in the upward-closed subset of $\mathcal{R}$, $R'(X,Y)$ holds. In the partial order $(\mathcal{R}, \sqsubseteq)$, all upward-closed subsets of $\mathcal{R}$ correspond exactly to the combinations of relations in $\mathcal{R}$ that can hold concurrently for a given pair of nonatomic poset events. It follows from the result in [2], page 400, that there is a 1-1 correspondence between the set of all upward-closed subsets of a partial order and the set of anti-chains in the partial order. Therefore, an enumeration of the anti-chains in $(\mathcal{R}, \sqsubseteq)$ gives an enumeration of the upward-closed subsets of $(\mathcal{R}, \sqsubseteq)$, which corresponds to all the combinations of the relations in $\mathcal{R}$ that can hold for a pair of nonatomic poset events.

The problem of counting all the anti-chains in a partial order is a #P-complete problem [19] and is therefore at least as hard as an NP-complete problem [8]. A recursive backtracking algorithm to enumerate the anti-chains of a poset is given in [5].

3 Significance and an example application

3.1 Significance

Section 2 examined the causality relations between two poset events that model nonatomic actions in distributed applications. Specifically, we formulated an exhaustive set of causality relations between nonatomic poset events using first-order predicate logic. These relations form a lattice hierarchy. The strongest relation is $R1a$ and the weakest is $R4d$. The significance of a relation $R?\#(X,Y)$ is determined by examining ? for the choice of proxies of X and Y, and examining # for how these proxies are related. The proposed set of causality relations between nonatomic poset events is richer than the specific causality relations in the literature. The suite of two relations in [16, 17], viz., $\longrightarrow$ and $--\rightarrow$, correspond to $R1a$ and $R4d$, respectively. The suite of relations in [12] and listed in Table 1 correspond to the new relations as follows: $R1$=$R1'$, $R2$, $R2'$, $R3$, $R3'$, $R4$=$R4'$ correspond to $R1a$, $R2b$, $R2b'$, $R3c$, $R3c'$, $R4d$, respectively.

Distributed applications (see Section 1.2) which use nonatomicity in reasoning and modeling need a fine level of discrimination of causality relations. Each application can choose appropriate causality relations from the exhaustive fine-grained hierarchy to specify and capture concurrency and synchronization conditions between its nonatomic poset events at a fine level of discrimination. This allows for a sophisticated modeling of the interactions in the application and system. The exhaustive classification gives an insight into the existing possibilities and can be used to select a number of primitive relations with good properties and clear intuitions, depending on the application. Observe that the results are significant for all distributed applications, not just those which are characterized by real-time constraints.

3.2 A Distributed synchronization problem

As a generic example, we now show the use of the proposed relations to a classical distributed synchronization problem such as distributed mutual exclusion between groups of processes. Consider two groups of distributed processes G_1 and G_2. Processes in each group are at different sites and communicate by message passing. In the

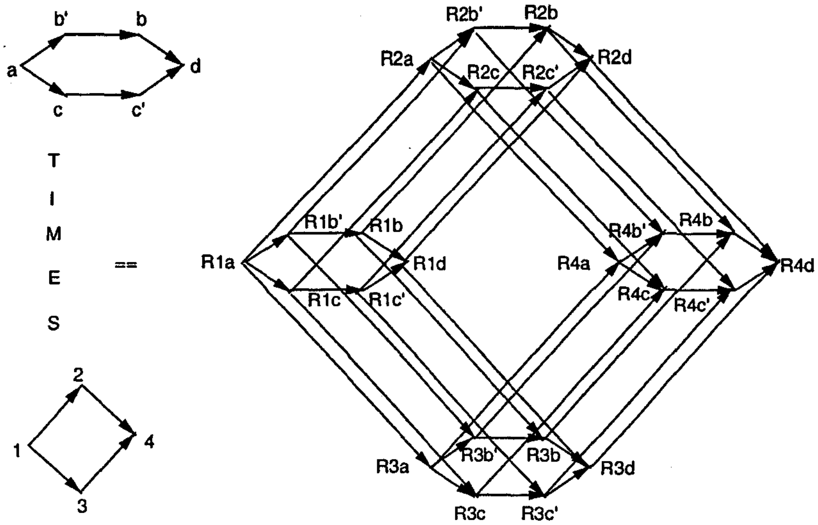

Figure 2. Inclusion hierarchy for poset interval - poset interval relations.

distributed mutual exclusion problem, processes of only one group can access a class of critical resources, such as files, which are also distributed in the system. Assume that processes in G_1 have access to the exclusive resources. Some of the ways in which the transfer of access rights from G_1 to G_2 can be effected are:

1. Each process in G_1 sends a message to each process in G_2 when it is done with its access to the files. The synchronization relation between the corresponding events on the processes in G_1 and G_2 is $R1a$.

   This option requires a priori knowledge of all the processes in G_2 by processes in G_1 as well as a priori knowledge of the number of processes in G_1 by processes in G_2. A total of $|G_1| \times |G_2|$ messages are needed. The delay is exactly that of one message transfer from the time the last process in G_1 completes its file accesses.

2. The processes in G_1 elect a leader to whom they send a message when they are finished with their file accesses. This leader then sends a message to each process in G_2 when it has received a message from each process in G_1. The synchronization relation between the corresponding events at the leader process of G_1 and the processes in G_2 is $R1c$.

   This option requires the knowledge of the leader of G_1 by processes in G_1, and knowledge of the group membership of G_2 by the leader of G_1. A total of $|G_1| + |G_2| - 1$ messages are needed. The delay is exactly that of two message transfers from the time the last process in G_1 completes its file accesses.

3. The processes in G_2 elect a leader to which each process in G_1 sends a message when it is finished with its file accesses. This leader then sends a message to each process in G_2 when it has received a message from each process in G_1. The synchronization relation between the corresponding events at the processes in G_1 and the leader process of G_2 is $R1b'$.

   This option requires the knowledge of the leader of G_2 by each member of G_1, and knowledge of the group membership of G_2 by its leader. A total of $|G_1| + |G_2| - 1$ messages are needed. The delay is exactly that of two message transfers from the time the last process in G_1 completes its file accesses.

4. The processes in G_1 elect a leader to whom they send a message when they are finished with their file accesses. This leader then sends a message to a (leader) process in G_2, which then sends a message to each of the processes in G_2. The synchronization relation between the corresponding events at the leader processes in G_1 and G_2 is $R1d$.

   This option requires the knowledge of the leader in G_1, knowledge of the leader of G_2 by the leader of G_2, and knowledge of the group membership of G_2 by the leader of G_2. A total of $|G_1| + |G_2| - 1$ messages are needed. The delay is exactly that of three message transfers from the time the last process in G_1 completes its file accesses.

Each of the above four ways which are differentiated by the different relations $R1*$ between the events involves a different amount of knowledge at the participating processes, a different delay, a different message complexity, and dif-

Relation $r(X,Y)$	Relation definition specified by quantifiers for $x \prec y$, where $x \in X$, $y \in Y$
$R1a$	$\forall x \in U_X \forall y \in L_Y$
$R1a'$ (=$R1a$)	$\forall y \in L_Y \forall x \in U_X$
$R1b$	$\forall x \in U_X \exists y \in L_Y$
$R1b'$	$\exists y \in L_Y \forall x \in U_X$
$R1c$	$\exists x \in U_X \forall y \in L_Y$
$R1c'$	$\forall y \in L_Y \exists x \in U_X$
$R1d$	$\exists x \in U_X \exists y \in L_Y$
$R1d'$ (=$R1d$)	$\exists y \in L_Y \exists x \in U_X$
$R2a$	$\forall x \in U_X \forall y \in U_Y$
$R2a'$ (=$R2a$)	$\forall y \in U_Y \forall x \in U_X$
$R2b$	$\forall x \in U_X \exists y \in U_Y$
$R2b'$	$\exists y \in U_Y \forall x \in U_X$
$R2c$	$\exists x \in U_X \forall y \in U_Y$
$R2c'$	$\forall y \in U_Y \exists x \in U_X$
$R2d$	$\exists x \in U_X \exists y \in U_Y$
$R2d'$ (=$R2d$)	$\exists y \in U_Y \exists x \in U_X$
$R3a$	$\forall x \in L_X \forall y \in L_Y$
$R3a'$ (=$R3a$)	$\forall y \in L_Y \forall x \in L_X$
$R3b$	$\forall x \in L_X \exists y \in L_Y$
$R3b'$	$\exists y \in L_Y \forall x \in L_X$
$R3c$	$\exists x \in L_X \forall y \in L_Y$
$R3c'$	$\forall y \in L_Y \exists x \in L_X$
$R3d$	$\exists x \in L_X \exists y \in L_Y$
$R3d'$ (=$R3d$)	$\exists y \in L_Y \exists x \in L_X$
$R4a$	$\forall x \in L_X \forall y \in U_Y$
$R4a'$ (=$R4a$)	$\forall y \in U_Y \forall x \in L_X$
$R4b$	$\forall x \in L_X \exists y \in U_Y$
$R4b'$	$\exists y \in U_Y \forall x \in L_X$
$R4c$	$\exists x \in L_X \forall y \in U_Y$
$R4c'$	$\forall y \in U_Y \exists x \in L_X$
$R4d$	$\exists x \in L_X \exists y \in U_Y$
$R4d'$ (=$R4d$)	$\exists y \in U_Y \exists x \in L_X$

Table 4. Proposed Relations $r(X, Y)$.

ferent fault tolerance implications. Similar examples can be constructed for relations $R2*$, $R3*$, and $R4*$.

Example of Air Defence Control: Consider an air defence control system. Let G_1 be the group of AWAC (Advanced Warning Air Control) radars and/or satellites that detect approaching potential threats and are mobile or located at different sites. AWACs track threats and update the distributed database of approaching threats. A meta-process constructs a consistent view of the approaching threats after the AWACs (G_1 processes) have updated the distributed database. Let G_2 be the group of distributed processes that plan the war strategy dynamically and make decisions on deployments and redistributions of defence resources such as anti-missile shields and missile launches. Processes in G_2 must have mutually exclusive access to the distributed database after processes in G_1 have completed their updates, to operate on a consistent view. Mutually exclusive access to the distributed database alternates in real-time between G_1 and G_2 as the groups update and evaluate the database, respectively, in rapid alternate succession.

The four ways of expressing the distributed mutual exclusion discussed above provide a choice in trade-offs of (i) knowledge of membership of G_1 and/or G_2, particularly with mobile processes, (ii) different delay, critical for rapid exchange of access rights to the database, (iii) bandwidth utilization, in view of the low bandwidth available with the use of crypto techniques, and (iv) fault-tolerance implications for this critical problem of air defence.

3.3 Distributed predicate specification

Detecting specific relations from $\mathcal{R}$ between a pair of nonatomic events on processes in groups G_1 and G_2, and global predicates that are logical expressions of such relations is important for several applications given in Section 1.2.

Example of Air Defence Control: In the battle scenario, assume that the velocity of defence missiles in flight can be controlled. Let f_o and f_m be functions that compute future trajectories of enemy missiles and defence missiles, respectively. If $\vec{x}$ is the current trajectory of a defence missile and $\vec{y}$ is the current trajectory of an enemy missile, then $f_m(\vec{x}) \prec f_o(\vec{y})$ indicates that the defence missile can, by controlling its velocity, reach certain points in space-time, that are reachable by the enemy missile's trajectory, before the enemy missile can reach those points. Therefore, by controlling its velocity, the defence missile can collide with and destroy the enemy missile.

We identify some useful relations for the example of the air defence control system. Consider the meta-process that uses the distributed database to construct a consistent view of the airspace and battlefield with its approaching threats, and defences being deployed or in reserve. It runs a lookahead simulation to determine exposed targets, and deploy defence resources. The following are some useful relations in this simulation analysis.

- Let $\vec{y}$ denote the latest event in the trajectory of an approaching missile threat. Let $\vec{x}$ denote the latest event in the trajectory of a launched or ready to be launched defence missile or defence shield.
  - R^*b, i.e., $\forall\ \vec{x}\ \exists\ \vec{y},\ f_m(\vec{x}) \prec f_o(\vec{y})$, in this simulation indicates that every launched / deployed missile or defence is "useful" because it is tracking an approaching missile threat.
  - R^*c', i.e., $\forall\ \vec{y}\ \exists\ \vec{x},\ f_m(\vec{x}) \prec f_o(\vec{y})$, in this simulation indicates that every approaching threat will be countered by the defence.
- If $\vec{x}$ represents an earlier recorded trajectory of the enemy missile, (hence, the current position of the missile previously observed at $\vec{x}$ is calculated by $f_o(\vec{x})$), and $\vec{y}$ represents the latest observed trajectory of an approaching missile threat, then $\overline{R * b}$, i.e., $\neg\ (\forall\ \vec{x}\ \exists\ \vec{y},\ f_o(\vec{x}) \prec \vec{y})$, in this simulation indicates that a previously known approaching missile has somehow become "stealth" or is out of range of the AWAC radar.
- If $\vec{x}$ is the latest event in the trajectory of an approaching missile and y is a target like a military installation or a metropolitan area, then in the simulation, R^*d, i.e., $\exists\ \vec{x}\ \exists\ y,\ f_o(\vec{x}) \prec y$, indicates that the target is exposed and will be destroyed if nothing is done. Therefore, the missile is lethal (not astray) and the defence should counter it.

Thus, we have seen that the proposed relations are useful for the specification of various distributed synchronization conditions such as those involved for distributed mutual exclusion, as well as for the specification of distributed predicates. These are very fundamental problems and have a broad range of applications.

4 Conclusions

We presented a hierarchy of 32 causality relations between nonatomic poset events in distributed systems using compositional construction. The hierarchy of relations is complete using first-order predicate logic. The results are useful for distributed applications which have real-time constraints and which need a fine level of discrimination of synchronization and causality relations. We examined the use of the proposed relations to the synchronization problem of distributed mutual exclusion as well as to the problem of specifying global predicates. We showed the example of an air defence control system to illustrate the use of some of the proposed relations.

Observe from Table 4 that each relation between nonatomic events X and Y can be evaluated with $|N_X| \times |N_Y|$ checks for causality (Definition 2 defined N_A). This is significantly better than $|X| \times |Y|$ checks for causality that naive definitions of causality require and is suited for efficient real-time evaluation. More efficient real-time evaluation conditions are derived in the full paper [11]. Specifically, relations R^*a, R^*a', R^*b', R^*c, R^*d, and R^*d' can be evaluated in $min(|N_X|, |N_Y|)$ integer comparisons, relations R^*b in $|N_X|$ integer comparisons, and relations R^*c' in $|N_Y|$ integer comparisons. Thus, it is shown in [11] that the simplified evaluation conditions have only a linear complexity of testing, whereas evaluation for the synchronization relations as per the definitions of the relations have a polynomial complexity of testing.

References

[1] U. Abraham, S. Ben-David, S. Moran, On the limitations of the global time assumption in distributed systems, *Proc. 5th Workshop on Distributed Algorithms*, LNCS 579, Springer-Verlag, 1-8, 1991.

[2] M. Aigner, *Combinatorial Theory*, Springer-Verlag, 1979.

[3] J. Allen, Maintaining knowledge about temporal intervals, *CACM*, 26(11):832-843, 1983.

[4] J. W. de Bakker, W. P. de Roever, G. Rozenberg (Eds.), *Linear Time, Branching Time, and Partial Orders in Logics and Models of Concurrency*, LNCS 354, Springer-Verlag, 1989.

[5] M. Ball, J. S. Provan, Calculating bounds on reachability and connectedness in stochastic networks, *Networks*, 13, 253-278, 1983.

[6] J. V. Benthem, *The Logic of Time*, Kluwer Academic Publishers, (1ed. 1983), 2ed. 1991.

[7] P. C. Fishburn, *Interval Orders and Interval Graphs: A Study of Partially Ordered Sets*, Wiley & Sons, 1985.

[8] M. Garey, D. Johnson, *Computers and Intractability: A Guide ot the Theory of NP-Completeness*, W. H. Freeman, 1979.

[9] W. Janssen, M. Poel, J. Zwiers, Action Systems and Action Refinement in the Development of Parallel Systems, In J.C.M. Baeten, J.F. Groote (Eds.) *Concur'91*, LNCS 527, Springer-Verlag, 298-316, 1991.

[10] F. Jahanian, R. Rajkumar, S. Raju, Run-time monitoring of timing constraints in distributed real-time systems, *Real-Time Systems*, 7(3), 247-273, November 1994.

[11] A. Kshemkalyani, Temporal interactions of intervals in distributed systems, *Tech. Report 29.1933, IBM*, Sept. 1994.

[12] A. Kshemkalyani, Temporal interactions of intervals in distributed systems, *Journal of Computer and System Sciences*, 52(2), 287-298, April 1996 (contains some parts of [11]).

[13] A. Kshemkalyani, A framework for viewing atomic events in distributed computations, *Theoretical Computer Science* (to appear). (Extended abstract appears in *EuroPar'96*, LNCS 1123, Springer-Verlag, 496-505, Aug. 1996.)

[14] A. Kshemkalyani, Relative timing constraints between complex events, *Proc. 8th IASTED Conf. on Parallel and Distributed Computing and Systems*, 324-326, Oct. 1996.

[15] L. Lamport, Time, clocks, and the ordering of events in a distributed system, *Communications of the ACM*, 558-565, 21(7), July 1978.

[16] L. Lamport, On interprocess communication, Part I: Basic formalism, Part II: Algorithms, *Distributed Computing*, 1:77-101, 1986.

[17] L. Lamport, The mutual exclusion problem, Part I: A theory of interprocess communication, Part II: Statements and solutions, *J. ACM*, 33(2), 313-348, April 1986.

[18] E.R. Olderog, *Nets, Terms, and Formulas*, Cambridge Tracts in Theoretical Computer Science, 1991.

[19] J. S. Provan, M. Ball, The complexity of counting cuts and of computing the probability that a graph is connected, *SIAM Journal of Computing,* 12(4), 777-788, Nov. 1983.

[20] A. Rensink, *Models and Methods for Action Refinement,* Ph.D. thesis, University of Twente, The Netherlands, Aug. 1993.

[21] S. Rajan, H. M. Vin, Formal basis for structured multimedia collaborations, *Proc. 2nd IEEE Int. Conf. on Multimedia Computing and Systems*, 194-201, 1995.

[22] R. Rajkumar, *Synchronization in Real-Time Systems,* Kluwer Academic Press, 1991.

[23] R. Schwarz, F. Mattern, Detecting causal relationships in distributed computations: In search of the holy grail, *Distributed Computing*, 7:149-174, 1994.

[24] L. Welch, State of the art in real-time computing, *Proc. NATO Advanced Study Institute on Real-Time Computing*, Oct. 1992.

[25] G. Winskel, Modeling concurrency with partial orders, *International Journal of Parallel Programming*, 15(1), 33-71, 1986.