Distrib. Comput. (1998) 11: 169–189

DISTRIBUTED COMPUTING

Causality and atomicity in distributed computations

Ajay D. Kshemkalyani

Department of Electrical Engineering and Computer Science (MC 154), University of Illinois at Chicago, 851 South Morgan Street, Chicago, IL 60607-7053, USA (e-mail: ajayk@eecs.uic.edu)

Received: July 1997 / Accepted: May 1998

Summary. In a distributed system, high-level actions can be modeled by nonatomic events. This paper proposes causality relations between distributed nonatomic events and provides efficient testing conditions for the relations. The relations provide a fine-grained granularity to specify causality relations between distributed nonatomic events. The set of relations between nonatomic events is complete in first-order predicate logic, using only the causality relation between atomic events. For a pair of distributed nonatomic events X and Y, the evaluation of any of the causality relations requires $|N_X| \times |N_Y|$ integer comparisons, where $|N_X|$ and $|N_Y|$, respectively, are the number of nodes on which the two nonatomic events X and Y occur. In this paper, we show that this polynomial complexity of evaluation can by simplified to a linear complexity using properties of partial orders. Specifically, we show that most relations can be evaluated in $\min(|N_X|, |N_Y|)$ integer comparisons, some in $|N_X|$ integer comparisons, and the others in $|N_Y|$ integer comparisons. During the derivation of the efficient testing conditions, we also define special system execution prefixes associated with distributed nonatomic events and examine their knowledge-theoretic significance.

Key words: Distributed computation – Distributed system – Atomicity – Time – Causality – Synchronization – Global predicates – Concurrency

1 Introduction

Motivation and objectives

The causality relation represents the partial order of events in a system execution [6, 8, 11, 14–16, 18, 26, 29–31, 36, 37, 39]. Thus far, the causality relation has been studied primarily between single events in space-time. The notion of grouping elementary events in a system execution into higher level nonatomic events is useful for event abstraction [9, 19, 22, 27, 28, 34]; 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. Specifically, nonatomic events are modeled in distributed applications such as distributed multimedia, distributed debugging, coordination in mobile systems, industrial process control, planning, navigation, and virtual reality, and in distributed agent-based programs. These applications deal with nonatomic nonlinear events, also known as nonatomic poset events, where at least some of the component atomic events of each nonatomic nonlinear event occur concurrently [21, 27]. The event abstraction inherent in nonatomic events, however, results in a loss of power to express and reason with various degrees of causality, i.e., the traditional causality relation [6, 8, 11, 14–16, 18, 26, 29–31, 36, 37, 39] defined between individual points in space-time cannot be used to capture or specify the synchronization conditions between two nonatomic events at a fine level of granularity using various degrees of causality, as required for accurately modeling the interactions between the nonatomic events. A broad spectrum of causality relations is needed to allow the expression of various degrees of synchronization between nonatomic events. Applications can choose relations from this fine-grained spectrum of relations, while still retaining and using the benefits of event abstraction. We propose causality relations between nonatomic poset events in a distributed system without assuming a global time axis, and examine their uses.

Clearly, there is a need to evaluate the causality relations between nonatomic poset events efficiently. We devise simplified evaluation conditions for the causality relations between nonatomic poset events. The proposed evaluation conditions provide the following savings. Let a nonatomic event be a set consisting of atomic events. For a pair of nonatomic events X and Y, there are $|X| \times |Y|$ pairs of causality relations between the atomic elements in terms of which the two nonatomic events are defined. A naive definition of causality would require $|X| \times |Y|$ checks for causality. However, for the causality relations that we define, we first show that the evaluation of the causality relations can be reduced to $|N_X| \times |N_Y|$ integer comparisons, where $|N_X|$ and $|N_Y|$, respectively, are the number of nodes on which the two nonatomic events X and Y occur. Then we show that the evaluation can by further simplified using properties of partial orders. Specifically, we show that most relations

can be evaluated in $\min(|N_X|, |N_Y|)$ integer comparisons, some in $|N_X|$ integer comparisons, and the others in $|N_Y|$ integer comparisons. Thus, the simplified evaluation conditions we derive have only a linear computational complexity of testing, whereas evaluation of the relations as per their definitions has a polynomial computational complexity of testing.

This paper directly addresses the concluding thesis of Schwarz-Mattern [37], namely that "... None of the presented schemes is sufficiently mature to serve as a *general-purpose mechanism* for the analysis of causality. ... The problem of anticipating the relevant behavior, assigning meaningful semantics to general global predicates, and finding *correct and efficient algorithms for their detection* remains to be a challenge. ... Anyhow, the holy grail of causality analysis has not been found yet."

Model

A distributed system execution is modeled by the space-time model. This model is a poset event structure model as in [11, 15, 21, 22, 26–31, 34, 37]. Consider a poset $(E, \prec)$ where $\prec$ is an irreflexive partial ordering. Let $\mathscr{E}$ denote the power set of E and let $\mathscr{A}$ $(\neq \emptyset) \subseteq (\mathscr{E} - \emptyset)$. There is an implicit one-many mapping from $\mathscr{A}$ to E. Each element A of $\mathscr{A}$ is a non-empty subset of E, and is termed an *interval* or a *nonatomic event*. $(E, \prec)$ 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 dimensions correspond to the set of processes (also termed *nodes*), and E_i is the set of events executed by process i. Causality between events at different nodes is imposed by asynchronous message passing. In such a distributed computer system, E represents the set of events and is discrete. Moreover, we assume there are a finite number of nodes i, and each E_i has a dummy initial event $(\perp_i)$ and a dummy final event $(\top_i)$. It follows that if $A \bigcap E_i \neq \emptyset$, then $(A \bigcap E_i)$ has a least and a greatest event. Let $E^\perp$ and $E^\top$ denote the sets of initial events and final events, respectively. We assume that $E^\perp$ and $E^\top$ are antichains and that $\forall \perp_i \forall \top_j \forall e \in (E \setminus E^\perp \setminus E^\top)$, $\perp_i \prec e \prec \top_j$. We restrict any event A in $\mathscr{A}$ that is of interest to the application to not contain any dummy events.

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 nonatomic event causally precede some events in the other nonatomic event [37]. Relations between time durations and between instants have been extensively studied in the literature on time and interval algebras. Most previous work assumed that the nonatomic events were linearly ordered and had unique start and finish instants, e.g., [2, 5, 8, 14, 16, 18, 35] – and confined the study of causality to relations between such time durations or linear intervals. The literature above also assumed that the 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 as argued in [26, 29, 37]. [8] includes a comprehensive review of literature in this area.

Table 1. Relations in [21] are given in the first two columns. The third column gives the evaluation conditions derived in this paper

Relation r	Expression for $r(X, Y)$	Evaluation condition using relation $\ll$ between cuts (see Theorem 4)
$R1$	$\forall x \in X \forall y \in Y, x \prec y$	$\bigwedge_{x \in X} [\cap_{\Downarrow} Y \ll x\uparrow]$
$R1'$	$\forall y \in Y \forall x \in X, x \prec y$	$= \bigwedge_{y \in Y} [\downarrow y \ll \cup_{\Uparrow} X]$
$R2$	$\forall x \in X \exists y \in Y, x \prec y$	$\bigwedge_{x \in X} [\cup_{\Downarrow} Y \ll x\uparrow]$
$R2'$	$\exists y \in Y \forall x \in X, x \prec y$	$\cup_{\Downarrow} Y \ll \cup_{\Uparrow} X$
$R3$	$\exists x \in X \forall y \in Y, x \prec y$	$\cap_{\Downarrow} Y \ll \cap_{\Uparrow} X$
$R3'$	$\forall y \in Y \exists x \in X, x \prec y$	$\bigwedge_{y \in Y} [\downarrow y \ll \cap_{\Uparrow} X]$
$R4$	$\exists x \in X \exists y \in Y, x \prec y$	$\cup_{\Downarrow} Y \ll \cap_{\Uparrow} X$
$R4'$	$\exists y \in Y \exists x \in X, x \prec y$	

In addition to dealing only with linear intervals, [2, 14, 18] defined their relations using combinations of the $\prec$ and $=$ relations between the start and finish instants, and such relations were further studied in [5, 8, 16, 35]. However, the $=$ relation between atomic events has not been accorded an important role in the recent literature on causality in distributed computing [6, 11, 15, 26–30, 37]. A possible explanation for this is that in a linear system, as two atomic events slide with respect to each other, the $=$ relation is critical in the transition from the $\prec$ to its inverse $\succ$ relation between the points. However, this is not the case in a distributed system where issues of concurrency are of greater research interest, and results based only on $\prec$ and $\succ$ can be extended in a straightforward manner to include the $=$ relation.

Linear intervals or durations in distributed systems have been used explicitly in specifying and detecting global predicates, an area which was initiated by [13, 38]. The following literature deals with causality between nonatomic poset events in a distributed system execution and does not assume a global time axis. Lamport defined system executions using two relations $\longrightarrow$ and $--\rightarrow$ between nonatomic elements and provided axioms A1 - A5 on these relations [27, 28]. Informally, these relations are as follows. For two nonatomic events X and Y in $\mathscr{A}$, $X \longrightarrow Y$ 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. The model and axioms in [27] were further examined in [1, 4, 5].

Action refinement of posets is studied [17, 19, 33, 34] and surveyed [34] along with a survey of related work in Petri nets [32]. In the literature on action refinement of posets, there is no definition of causality between nonatomic poset events other than the generic definition in [34] that it is "the composition of the causality relation between individual atomic events in unspecified subsets of the two nonatomic poset events".

A study of the temporal interactions of intervals has shown that the two causality relations defined by Lamport are not sufficient to capture the essential temporal properties of system executions and specify synchronization and

Table 2. Inclusion relationships between relations [21]

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

causality relations between nonatomic events in distributed systems [21]. The derivation of the 29 (respectively, 40) possible temporal interactions between two nonatomic linear intervals in a distributed system based on the dense (respectively, nondense) model of time in [21] used a set of new causality relations between nonatomic events [21]. This set of causality relations did not assume a global time axis. Relations $R1$–$R4$ and $R1'$–$R4'$ which form a part of this set of relations [21] are expressed in terms of the quantifiers over X and Y in the first two columns of Table 1. Table 2 gives the hierarchy and inclusion relationship of the causality relations $R1$–$R4$ of Table 1. 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 '$\sqsubseteq$' and '$\sqsupseteq$' 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 $\sqsubseteq$. Relations $R2'$ and $R3'$ are different from $R2$ and $R3$, respectively, when applied to posets but are the same as $R2$ and $R3$, respectively, when applied to linear intervals. $R1'$ and $R4'$ are the same as $R1$ and $R4$, respectively. The complete hierarchy among the relations of Table 1 is shown in Table 3; the motivation for also using alternate names for the relations in Table 3 will be given later. The causality relations between nonatomic poset events will be derived using the relations in Table 1 and the hierarchy among them.

Significance and relation to previous work

The set of relations proposed in [21] formed a comprehensive set of causality relations to derive and express all possible temporal interactions between a pair of linear intervals using only the $\prec$ relation between atomic events, and extended the partial hierarchy of relations of [27, 28]. However, when the relations of [21] are applied to a pair of poset intervals, the hierarchy they form is incomplete. The fine-grained causality relations between a pair of nonatomic poset intervals proposed here extend the results [21] to nonatomic poset events [23, 24]. They form a "comprehensive" set of causality relations between nonatomic poset events using first-order predicate logic and only the $\prec$ relation between atomic events, and fill in the existing partial hierarchy of causality relations between nonatomic poset events, formed by relations in [21, 27, 28]. A relational algebra for the relations in [23, 24] is given in [20, 25]. Given any relation(s) between X and Y, the relational algebra allows the derivation of conjunctions, disjunctions, and negations of all other relations that are also valid between X and Y, as well as between Y and X. As we propose a suite of causality relations between nonatomic poset events, there should be efficient ways to test whether such relations hold between any pair of such events. We derive efficient linear-time evaluation conditions for the proposed relations between nonatomic poset events. Thus, this paper provides a framework not only for specifying causality and global predicates between nonatomic poset events, but also for analyzing and detecting such causality at a low linear-time cost. The results contribute to the fundamental area of causality and atomicity in distributed computing [27, 28, 37].

Organization. Section 2 derives the fine-grained hierarchy of causality relations to fill in the existing partial hierarchy of causality relations. Section 3 derives simplified evaluation conditions for the relations in Sect. 2. The simplified evaluation conditions we derive have only a linear computational complexity of testing, whereas evaluation of the relations as per the definitions of the relations has a polynomial computational complexity. Section 4 gives concluding remarks. The results of this paper are included in [20].

2 Relations between poset events

Recall that each member A of $\mathscr{A}$ represents a higher level grouping of the events of E and is an interval or a nonatomic event.

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

Definition 2 *N_A, the node set of interval A, is* $\{i \mid E_i \bigcap A \not\subseteq \{\perp_i, \top_i\}\}$.

The above definition is preferred over the simpler definition $N_A = \{i \mid E_i \bigcap A \neq \emptyset\}$ to make it applicable not only to nonatomic events of interest to the application but also to auxiliary nonatomic events containing $\perp_i$ and $\top_i$, defined later. Causality relations specific to linear intervals are given in [20]. These are used to derive the fine-grained causality relations between nonatomic poset events, whose node set has a size larger than one [23, 24].

Previous work on linear intervals and time durations, e.g., [2,5,6,8,14–16,18], identified an interval by the instants of its beginning and end. The beginning and end instants of a linear interval are points in space-time which are atomic events in E. For a nonatomic poset interval, it is natural to identify counterparts for the beginning and end instants. These counterparts 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. For a nonatomic interval X, let L_X and U_X correspond to the beginning of X and the end of X, respectively. L_X and U_X can act as a *proxy* for poset X. Two possible definitions of proxies are as follows [20].

Definition 3 • $L_X = \{e_i \in X \mid \forall e_i' \in X, e_i \preceq e_i'\}$
• $U_X = \{e_i \in X \mid \forall e_i' \in X, e_i \succeq e_i'\}$

Definition 4 • $L_X = \{e \in X \mid \forall e' \in X, e \not\succ e'\}$
• $U_X = \{e \in X \mid \forall e' \in X, e \not\prec e'\}$

Table 3. Full hierarchy of relations of Table 1 [21]. Relations are defined between X and Y. Relations $R1$, $R1'$, $R2$, $R2'$, $R3$, $R3'$, $R4$, $R4'$ of Table 1 are also referred to as a, a', b, b', c, c', d, d', respectively

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'): $\exists x \exists y$ (= $\exists y \exists x$)	$\sqsupseteq$	$\sqsupseteq$	$\sqsupseteq$	$\sqsupseteq$	$\sqsupseteq$	=

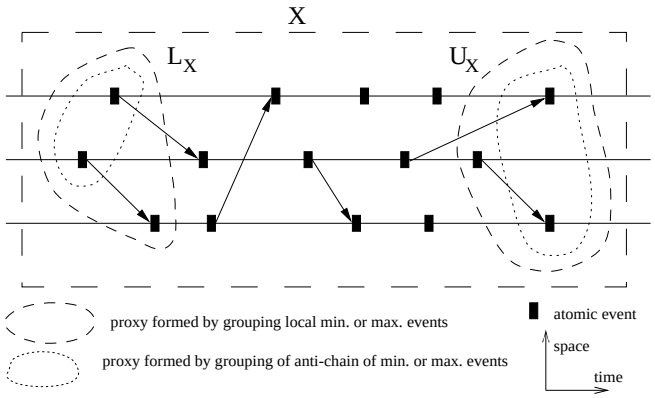

Fig. 1. Two definitions of proxies of a poset event

By Definition 3, L_X and U_X are the sets of the least and greatest events at each node in N_X, respectively. By Definition 4, L_X and U_X are the largest antichains containing the minimal and maximal events of X, respectively. We assume that any one definition is consistently used, depending on context and application. We will denote any proxy of X as $\hat{X}$. From the system model, it follows that for a distributed computer system, each nonatomic event has a well-defined maximal and minimal event at each node in its node set, and the node set is finite. Hence, the proxies of all nonatomic events are finite and well-defined. Figure 1 depicts the proxies of X.

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, and a good choice of a proxy is the beginning or end of an interval. A proxy for X and Y can be chosen in 2×2 ways, corresponding to the relations in $\{R1, R2, R3, R4\}$. From Table 2, it follows that these four relations form a lattice ordered by $\sqsubseteq$. The second aspect deals with how the atomic elements of the chosen proxies of X and Y are related by causality. The chosen proxies can be related by the eight relations $R1$, $R1'$, $R2$, $R2'$, $R3$, $R3'$, $R4$, $R4'$ of Table 1, which are also referred to as a, a', b, b', c, c', d, d', respectively, in Table 3 to avoid confusion with their original names used for the first aspect of specifying the relations between poset intervals. The inclusion hierarchy among the six distinct relations forms a lattice ordered by $\sqsubseteq$, as shown in Table 3.

The causality relations are formed by combining the two aspects of deriving the relations, described above. The lattice of relations { $R1$*, $R2$*, $R3$*, $R4$* } between proxies for X and Y, and the lattice of relations { a, a', b, b', c, c', d, d' } between the elements of the proxies, when multiplied give a lattice of 32 relations over $\mathscr{A} \times \mathscr{A}$ to express $r(X, Y)$. The resulting set of poset relations is denoted $\mathscr{R}$ and given in the second column of Table 4; the efficient evaluation conditions that we derive for these relations in Sect. 3 are given in the third column. The relations in $\mathscr{R}$ form a lattice of 24 unique elements as shown in Fig. 2; the strongest relation is $R1a$ and the weakest is $R4d$. Relation $R?\#(X, Y)$ means that the proxies of X and Y are chosen as per ?, and events in the proxies are related as per #. $\mathscr{R}$ is comprehensive using first-order predicate logic and only the $\prec$ relation between atomic events. Specifically, $\mathscr{R}$ defined between nonatomic poset events is richer than the specific causality relations in the literature. The suite of two relations in [27], viz., $\longrightarrow$ and $--\rightarrow$, correspond to $R1a$ and $R4d$, respectively. The

Table 4. Causality relations $r(X, Y) \in \mathscr{R}$ for interactions between nonatomic poset events. The third column gives the evaluation conditions

Relation $r(X, Y)$	Relation definition specified by quantifiers for $x \prec y$, where $x \in X$, $y \in Y$	Evaluation condition using relation $\ll$ between cuts (see Theorem 4)
$R1a$	$\forall x \in U_X \; \forall y \in L_Y$	$\bigwedge_{x \in U_X} [\cap_{\Downarrow} L_Y \ll x\uparrow]$
$R1a'$ (= $R1a$)	$\forall y \in L_Y \; \forall x \in U_X$	$\bigwedge_{y \in L_Y} [\downarrow y \ll \cup_{\Uparrow} U_X]$
$R1b$	$\forall x \in U_X \; \exists y \in L_Y$	$\bigwedge_{x \in U_X} [\cup_{\Downarrow} L_Y \ll x\uparrow]$
$R1b'$	$\exists y \in L_Y \; \forall x \in U_X$	$\cup_{\Downarrow} L_Y \ll \cup_{\Uparrow} U_X$
$R1c$	$\exists x \in U_X \; \forall y \in L_Y$	$\cap_{\Downarrow} L_Y \ll \cap_{\Uparrow} U_X$
$R1c'$	$\forall y \in L_Y \; \exists x \in U_X$	$\bigwedge_{y \in L_Y} [\downarrow y \ll \cap_{\Uparrow} U_X]$
$R1d$	$\exists x \in U_X \; \exists y \in L_Y$	$\cup_{\Downarrow} L_Y \ll \cap_{\Uparrow} U_X$
$R1d'$ (= $R1d$)	$\exists y \in L_Y \; \exists x \in U_X$	
$R2a$	$\forall x \in U_X \; \forall y \in U_Y$	$\bigwedge_{x \in U_X} [\cap_{\Downarrow} U_Y \ll x\uparrow]$
$R2a'$ (= $R2a$)	$\forall y \in U_Y \; \forall x \in U_X$	$\bigwedge_{y \in U_Y} [\downarrow y \ll \cup_{\Uparrow} U_X]$
$R2b$	$\forall x \in U_X \; \exists y \in U_Y$	$\bigwedge_{x \in U_X} [\cup_{\Downarrow} U_Y \ll x\uparrow]$
$R2b'$	$\exists y \in U_Y \; \forall x \in U_X$	$\cup_{\Downarrow} U_Y \ll \cup_{\Uparrow} U_X$
$R2c$	$\exists x \in U_X \; \forall y \in U_Y$	$\cap_{\Downarrow} U_Y \ll \cap_{\Uparrow} U_X$
$R2c'$	$\forall y \in U_Y \; \exists x \in U_X$	$\bigwedge_{y \in U_Y} [\downarrow y \ll \cap_{\Uparrow} U_X]$
$R2d$	$\exists x \in U_X \; \exists y \in U_Y$	$\cup_{\Downarrow} U_Y \ll \cap_{\Uparrow} U_X$
$R2d'$ (= $R2d$)	$\exists y \in U_Y \; \exists x \in U_X$	
$R3a$	$\forall x \in L_X \; \forall y \in L_Y$	$\bigwedge_{x \in L_X} [\cap_{\Downarrow} L_Y \ll x\uparrow]$
$R3a'$ (= $R3a$)	$\forall y \in L_Y \; \forall x \in L_X$	$\bigwedge_{y \in L_Y} [\downarrow y \ll \cup_{\Uparrow} L_X]$
$R3b$	$\forall x \in L_X \; \exists y \in L_Y$	$\bigwedge_{x \in L_X} [\cup_{\Downarrow} L_Y \ll x\uparrow]$
$R3b'$	$\exists y \in L_Y \; \forall x \in L_X$	$\cup_{\Downarrow} L_Y \ll \cup_{\Uparrow} L_X$
$R3c$	$\exists x \in L_X \; \forall y \in L_Y$	$\cap_{\Downarrow} L_Y \ll \cap_{\Uparrow} L_X$
$R3c'$	$\forall y \in L_Y \; \exists x \in L_X$	$\bigwedge_{y \in L_Y} [\downarrow y \ll \cap_{\Uparrow} L_X]$
$R3d$	$\exists x \in L_X \; \exists y \in L_Y$	$\cup_{\Downarrow} L_Y \ll \cap_{\Uparrow} L_X$
$R3d'$ (= $R3d$)	$\exists y \in L_Y \; \exists x \in L_X$	
$R4a$	$\forall x \in L_X \; \forall y \in U_Y$	$\bigwedge_{x \in L_X} [\cap_{\Downarrow} U_Y \ll x\uparrow]$
$R4a'$ (= $R4a$)	$\forall y \in U_Y \; \forall x \in L_X$	$\bigwedge_{y \in U_Y} [\downarrow y \ll \cup_{\Uparrow} L_X]$
$R4b$	$\forall x \in L_X \; \exists y \in U_Y$	$\bigwedge_{x \in L_X} [\cup_{\Downarrow} U_Y \ll x\uparrow]$
$R4b'$	$\exists y \in U_Y \; \forall x \in L_X$	$\cup_{\Downarrow} U_Y \ll \cup_{\Uparrow} L_X$
$R4c$	$\exists x \in L_X \; \forall y \in U_Y$	$\cap_{\Downarrow} U_Y \ll \cap_{\Uparrow} L_X$
$R4c'$	$\forall y \in U_Y \; \exists x \in L_X$	$\bigwedge_{y \in U_Y} [\downarrow y \ll \cap_{\Uparrow} L_X]$
$R4d$	$\exists x \in L_X \; \exists y \in U_Y$	$\cup_{\Downarrow} U_Y \ll \cap_{\Uparrow} L_X$
$R4d'$ (= $R4d$)	$\exists y \in U_Y \; \exists x \in L_X$	

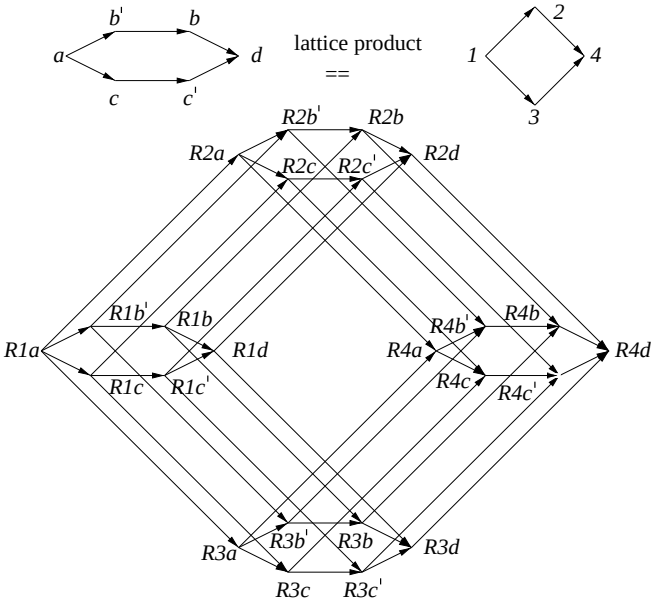

Fig. 2. Hierarchy of relations between poset events

suite of relations in [21] 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.

Note that by construction, $(\mathscr{R}, \sqsubseteq)$ is a lattice as illustrated in Fig. 2. For a given pair of posets X and Y, it may be the case that a combination of the relations in $\mathscr{R}$ 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 $\mathscr{R}$, $R'(X,Y)$ holds. In the partial order $(\mathscr{R}, \sqsubseteq)$, all upward-closed subsets of $\mathscr{R}$ correspond exactly to the combinations of relations in $\mathscr{R}$ that can hold concurrently for a given pair of nonatomic poset events. It follows from the result in [3], 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 antichains in the partial order. Therefore, an enumeration of the antichains in $(\mathscr{R}, \sqsubseteq)$ gives an enumeration of the upward-closed subsets of $(\mathscr{R}, \sqsubseteq)$, which corresponds to all the combinations of the relations in $\mathscr{R}$ that can hold for a pair of nonatomic poset events.

The proposed hierarchy of causality relations in first-order predicate logic is useful for applications that use nonatomic poset events for event abstraction and also need a fine level of granularity of causality relations to specify synchronization relations and their composite global predicates between the nonatomic poset events. The hierarchy gives an insight into the existing possibilities and can be used by an application to select a number of primitive relations with good properties and clear intuitions, depending on the application. There are two broad ways in which the proposed relations will be used.

– The relations and their composite (global) predicates provide a precise handle to *express* a naturally occurring or *enforce* a desired fine-grained level of causality or synchronization in the computation. Processes in a distributed computation can synchronize as per one of the relations in $\mathscr{R}$ whenever needed – each synchronization performed causes a (global) predicate on some relation(s) to become true. As the synchronization is achieved by message-passing, it enables the detection of global functions over the states of the nodes participating in the synchronization.
– One needs to *detect* the occurrence of a specific or any or all relations that hold among nonatomic events in a computation. Achieving this efficiently is the topic of Sect. 3.

A specific meaning and expression for each relation and a brief discussion of how the relation can be used is given in Appendix A. Some broad classes of uses of the relations include modeling various forms of synchronization for group mutual exclusion, initiation of nested computations, termination of nested computations, and monitoring the start or end of a computation.

3 Efficient evaluation of causality relations

Objectives

The proposed relations are useful to distributed applications that need a fine level of discrimination in specifying synchronization and causality conditions, and for subsequent reasoning. Complex conditions can be expressed as a predicate over these relations. Given a trace of a distributed execution, the application identifies pertinent nonatomic events and needs to know what relations are satisfied between pairs of such events. Implicit in the use of these relations by applications is the need to detect whether some specific relation holds between pairs of nonatomic events. We formalize this requirement as follows.

Problem 1 Given a recorded trace of a distributed computation $(E, \prec)$ and a set of nonatomic events $\mathscr{A}$, then for every pair of nonatomic poset events X and Y, where $X, Y \in \mathscr{A}$, determine if a specific relation $r(X,Y)$ holds, for $r \in \mathscr{R}$.

An extension of this problem is the problem that requires the detection of all possible relations that hold between pairs of nonatomic events.

Problem 2 Given a recorded trace of a distributed computation $(E, \prec)$ and a set of nonatomic events $\mathscr{A}$, then for every pair of nonatomic poset events X and Y, where $X, Y \in \mathscr{A}$, determine all the relations $r(X,Y)$ that hold, for $r \in \mathscr{R}$.

Problems 1 and 2 can be answered by testing for the appropriate causality relation(s) in Table 4. Observe from the second column of Table 4 that each relation between nonatomic events X and Y can be evaluated[1] with $|N_X| \times |N_Y|$ checks for causality. This is significantly better than $|X| \times |Y|$ checks for causality that would be needed without the use of proxies in the definitions of causality. However, this evaluation has a polynomial computational complexity ($|N_X| \times |N_Y|$ checks for causality). Our objective is to simplify the tests for the relations, which we achieve by using

[1] We use the terms $|N_X|$ and $|N_Y|$ which are upper bounds on $|N_{\hat{X}}|$ and $|N_{\hat{Y}}|$, respectively.

properties of partial orders. Specifically, we show using Theorem 5 that 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. The simplified evaluation conditions we derive have only a linear computational complexity, whereas evaluation as per the definitions of the relations has a polynomial computational complexity.

Observe that each of the 32 relations $r(X, Y)$, for $r \in \mathscr{R}$, was derived by choosing a proxy $\hat{X}$ of the two possible proxies of X and a proxy $\hat{Y}$ of the two possible proxies of Y, thus making one of four choices, and applying one of the eight relations R in Table 1 to the chosen proxies. Thus, for nonatomic poset events X and Y, there is a 1–1 equivalence between any $r(X, Y)$, for $r \in \mathscr{R}$, and $R(\hat{X}, \hat{Y})$, for some R in Table 1 and some $\hat{X}$ and some $\hat{Y}$. But $\hat{X}$ and $\hat{Y}$ are themselves nonatomic poset events like X and Y – the only difference is that for any node i, $|\hat{X}_i| \leq 1$ and $|\hat{Y}_i| \leq 1$, whereas $|X_i|$ and $|Y_i|$ are bounded only by $|E_i|$. However, we show that the evaluation methodology and complexity for $R(X, Y)$ is independent of the size of $|X_i|$ and $|Y_i|$. Hence, we derive the evaluation methodology for $R(X, Y)$, where R belongs to Table 1. Then, using a suitable quantification of X and Y in these results to represent the various proxies $\hat{X}$ and $\hat{Y}$, we obtain the evaluation methodology for each relation in $\mathscr{R}$ (Table 4).

Notation. We use the notation $\hat{X}$ when we specifically need to distinguish a subset of E that acts as a proxy for another subset X of E. Otherwise, when the distinction is not important, the notation X refers to any subset of E, which can also be a proxy of another set.

Strategy

We derive efficient linear-time tests for the relations in the second column of Table 1 by proceeding by the following steps to evaluate causality between two nonatomic events X and Y.

1. Define prefixes of an execution, termed cuts, and a relation $\ll$ on these cuts. (See Sect. 3.1.)
2. Define certain cuts identified by nonatomic poset events. (See Sect. 3.2.) These cuts represent useful causality information about the poset event, i.e., information about the past and the future of the execution associated with the poset event, in a compact form; moreover, each cut has a different significance. Once identified, the cuts which represent the causality information in a compact form can be reused in evaluating causality relations with other nonatomic poset events. This is *Key Idea 1* that enables the efficient evaluation of the relations in Table 1.
3. Define timestamps for individual atomic events, for cuts, and for nonatomic events. (See Sect. 3.3.) Timestamps are introduced because they provide a practical handle to test for causality. The timestamps of the cuts that represent causality information of the nonatomic events in a compact form, also capture such causality information in a compact form.
4. Derive simplified evaluation conditions for the relation $\ll$ between cuts identified by X and Y, using timestamps and properties of partial orders. (See Sect. 3.4 and specifically Theorem 3.) These simplified evaluation conditions are possible because the cuts identified by X and Y are structured based on the membership of X and Y and are not arbitrary cuts. This is *Key Idea 2* that enables the efficient evaluation of the relations in Table 1. The computational complexity of evaluating each relation is min($|N_X|, |N_Y|$).
5. Show that each of the causality relations between nonatomic posets X and Y in Table 1 holds iff the $\ll$ relation holds between specific cuts identified by X, Y, and the causality relation being considered. (See Sect. 3.5 and specifically Theorem 4.)
   Using step 4, it follows that the computational complexity of evaluating the causality relations is the same as the computational complexity of the evaluation of the relation $\ll$ between the appropriately chosen cuts identified by X, Y, and the causality relation being considered. (See Theorem 5).
6. Examine the overall cost of using the proposed method. The cost includes a one-time cost of setting up the timestamp structure, and the cost of evaluating the relation $\ll$ between appropriately identified cuts based on X, Y, and the relation being evaluated.

3.1 Cuts of an execution

Let P be the set of all process/node partitions. An execution prefix or a *cut* is the union of downward-closed nonempty subsets of each E_i, one for every node $i \in P$.

Definition 5 *A cut C is the union of a downward-closed nonempty subset of each E_i in $(E, \prec)$, where $E = \bigcup_{\forall i \in P} E_i$.*

$$C \equiv C \subseteq E \bigwedge E^{\perp} \subseteq C \bigwedge$$
$$e_i \in C \Longrightarrow (\forall e_i', e_i' \prec e_i \Longrightarrow e_i' \in C)$$

A cut is a nonatomic event and hence it has a well-defined upper bound and lower bound at each node in its node set. We define $S(C)$ to be the set of latest events at each node in cut C. $S(C)$ denotes the "surface" of cut C and is the same as the proxy U_C if U_C is defined by Definition 3. However, we choose to use notation $S(C)$ to allow the application the choice of definition of the proxy as per Definition 3 or 4.

Definition 6 • $S(C) = \{e_i \in C \mid \forall e_i' \in C, e_i \succeq e_i'\}$

Given a cut C, C_i (or $[S(C)]_i$) is a subset of C (or $S(C)$) that contains elements in partition i.

3.1.1 Comparison of cuts

It is a known result from lattice theory that the set of all cuts, denoted $\mathscr{C}$, forms a lattice ordered by the subset relation "$\subset$".

We introduce a new relation $\ll$ over the set of cuts. Loosely stated, $\ll(C, C')$ signifies that cut C is a proper subset of cut C' and moreover, C_i is a proper subset of C_i'. This relation is useful to derive simplified evaluation conditions for the relations between nonatomic poset events given in Table 1.

Definition 7 *We express the relation $\ll(C, C')$ in different forms, each of which will be used subsequently.*

1. $\ll (C, C')$ *iff* $(\forall z \in (S(C) \setminus E^{\perp}), z \notin S(C') \bigwedge z \in C') \bigwedge C' \neq E^{\perp}$.
2. $\not\ll (C, C')$ *iff* $(\exists z \in (S(C) \setminus E^{\perp}), z \in S(C') \bigvee z \notin C') \bigvee C' = E^{\perp}$.
3. $\ll (C, C')$ *iff* $(\forall z \in (S(C') \setminus E^{\perp}), z \notin C) \bigwedge C' \neq E^{\perp} \bigwedge N_C \subseteq N_{C'}$.
4. $\not\ll (C, C')$ *iff* $(\exists z \in (S(C') \setminus E^{\perp}), z \in C) \bigvee C' = E^{\perp} \bigvee N_C \not\subseteq N_{C'}$.

All the four forms of the definition are equivalent. The terms $C' \neq E^{\perp}$ and $C' = E^{\perp}$ are required to make the definitions robust for certain cases where $C' = E^{\perp}$. The forms in Definition 7.2 and Definition 7.4 express the condition for $\not\ll(C, C')$ which we will use subsequently as follows. The significance of $\not\ll$ is that if $\not\ll(C, C')$, then some event in $S(C)$ (equals or) happens causally after some event in $S(C')$. If we can choose C' and C to correspond to appropriate cuts of X and Y, respectively, for any $R(X, Y)$, where $R \in$ Table 1, then we have a reexpression for the relation R. Then the evaluation of $R(X, Y)$ reduces to the evaluation of $\not\ll(C, C')$ which takes $|P|$ evaluations in the general case. But C' and C are not arbitrary cuts; rather, they are the cuts identified by X and Y and are structured based on the membership of X and Y. We will show in Sect. 3.4 that because of their structure, the evaluation of $\not\ll(C, C')$ can be simplified.

3.2 Past and future cuts of a poset event

For atomic event e, there are two special cuts $\downarrow e$ and $e\uparrow$. $\downarrow e$ is the maximal set of events that happen before or equal e. $\downarrow e$ denotes the causal past (CP) of e. $e\uparrow$ is the union of downward-closed sets of events at each node, such that the set of events at any i is the downward-closed set of events at i upto and including the earliest event at i for which e happens before or equals the event. $e\uparrow$ is the complement of the causal future (CCF) of e and denotes the execution prefix upto and including the beginning of the causal future of e at each node.

Definition 8 *[CP:]* $\downarrow e \equiv \{e' \mid e' \preceq e\}$

Definition 9 *[CCF:]* $e\uparrow \equiv \{e' \mid e' \not\succeq e\} \bigcup \{e_i, i \in P \mid e_i \succeq e \bigwedge (\forall e'_i, e'_i \prec e_i \Longrightarrow e'_i \not\succeq e)\}$

The cuts $\downarrow e$ and $e\uparrow$ have the property that cut $\downarrow e$ has a unique maximal event and cut $e\uparrow$ has a unique minimal event. Also, $\downarrow e$ is downward-closed in $(E, \prec)$ whereas $e\uparrow$ is not.

Given a poset event, we define certain cuts that represent the past and the future of the execution associated with the poset event; each cut has a different significance. Each of the causality relations between nonatomic posets X and Y in Table 1 will be shown to be equivalent to the $\ll$ relation between specific cuts identified by X, Y and the causality relation being considered.

Definition 10 *The second column of Table 5 defines certain sets associated with poset X.*

Lemma 1 *The sets defined in Definition 10 are cuts.*

Proof. The sets defined in Definition 10 are formed by the union and intersection of cuts of the form $\downarrow x$ and $x\uparrow$, and the set of all cuts of E is a lattice $(\mathscr{C}, \subset)$ that is closed under $\bigcup$ and $\bigcap$. □

Figure 3 illustrates the cuts C1–C4 defined in Table 5 for a poset X containing 11 atomic events which are represented by large circles. For each maximal or minimal event of X at a node, the corresponding dotted (dashed) line indicates the surface of the past (future) of that event. Each such event, and events on the surfaces of both its past and its future, are marked using a unique marker (such as small shaded circle, small circle, small shaded rectangle, and small rectangle) for further clarity. The surface of each cut C1–C4 is marked by a thick line and labeled.

The cuts $\cap_{\Downarrow} X$ and $\cup_{\Downarrow} X$ which are determined by the set $\{\downarrow x \mid x \in X\}$ condense the causality information in each cut in the set, i.e., information about the past of the execution associated with events in X. The cuts $\cap_{\Uparrow} X$ and $\cup_{\Uparrow} X$ which are determined by the set $\{x\uparrow \mid x \in X\}$ condense the causality information in each cut in the set, i.e., information about the future of the execution associated with events in X.

Observe that $\cap_{\Downarrow} X$ and $\cup_{\Downarrow} X$ are downward-closed subsets of $(E, \prec)$ whereas $\cap_{\Uparrow} X$ and $\cup_{\Uparrow} X$ are not. We will use this observation about $\cap_{\Downarrow} X$ and $\cup_{\Downarrow} X$ in the proof of Lemma 6.

Lemma 2 *The members of a poset are related to the cuts associated with the poset, defined in Definition 10, as follows.*

2.1 $\forall e' \in S(\cap_{\Downarrow} X) \forall x \in X,\ e' \preceq x$
2.2 $\forall e' \in S(\cup_{\Downarrow} X) \exists x \in X,\ e' \preceq x$
2.3 $\forall e' \in S(\cap_{\Uparrow} X) \exists x \in X,\ x \preceq e'$
2.4 $\forall e' \in S(\cup_{\Uparrow} X) \forall x \in X,\ x \preceq e'$

Proof. From Definitions 8,9, and 10, and the definition of $\bigcup$ and $\bigcap$ operations, we observe the following.

2.1 For any e'_j in $S(\cap_{\Downarrow} X)$, e'_j is $\min(\{[S(\downarrow x)]_j \mid x \in X\})$. Therefore, $\forall x \in X,\ e'_j \in \downarrow x$.
2.2 For any e'_j in $S(\cup_{\Downarrow} X)$, e'_j is $\max(\{[S(\downarrow x)]_j \mid x \in X\})$. Therefore, $\exists x \in X,\ e'_j \in S(\downarrow x)$.
2.3 For any e'_j in $S(\cap_{\Uparrow} X)$, e'_j is $\min(\{[S(x\uparrow)]_j \mid x \in X\})$. Therefore, $\exists x \in X,\ e'_j \in S(x\uparrow)$.
2.4 For any e'_j in $S(\cup_{\Uparrow} X)$, e'_j is $\max(\{[S(x\uparrow)]_j \mid x \in X\})$. Therefore, $\forall x \in X,\ e'_j \succeq e''_j \in S(x\uparrow)$.

The lemma follows from the above observations and by noting that $[S(\downarrow x)]_j \preceq x$ and $x \preceq [S(x\uparrow)]_j$. □

The cuts of a nonatomic poset event defined in Definition 10 represent various execution prefixes associated with the nonatomic event. The cuts C1–C4 will be used in deriving simplified evaluation conditions for the causality relations defined in Table 1. Cuts C1(X) and C2(X) are about the past of the nonatomic event and cuts C3(X) and C4(X) are about the future of the nonatomic event. The significance of these cuts is discussed and expressed in knowledge-theoretic terminology next [10,12]. We will use notation Φ_X and Φ_{cut} to represent knowledge about nonatomic event X and cut

Table 5. Definitions of special sets of poset X. These sets are shown to be cuts. Timestamps of the cuts are given in the third column

Label	Definition	Timestamp, derived from Defn. 14 and Lemma 3
$C1(X)$ or $\cap_{\Downarrow} X$	$\bigcap_{\forall x \in X} \{\downarrow x\}$	$T(\cap_{\Downarrow} X) \equiv \forall i \in P, T(\cap_{\Downarrow} X)[i] = \min_{\forall x \in X}(T(\downarrow x)[i])$
$C2(X)$ or $\cup_{\Downarrow} X$	$\bigcup_{\forall x \in X} \{\downarrow x\}$	$T(\cup_{\Downarrow} X) \equiv \forall i \in P, T(\cup_{\Downarrow} X)[i] = \max_{\forall x \in X}(T(\downarrow x)[i])$
$C3(X)$ or $\cap_{\Uparrow} X$	$\bigcap_{\forall x \in X} \{x\uparrow\}$	$T(\cap_{\Uparrow} X) \equiv \forall i \in P, T(\cap_{\Uparrow} X)[i] = \min_{\forall x \in X}(T(x\uparrow)[i])$
$C4(X)$ or $\cup_{\Uparrow} X$	$\bigcup_{\forall x \in X} \{x\uparrow\}$	$T(\cup_{\Uparrow} X) \equiv \forall i \in P, T(\cup_{\Uparrow} X)[i] = \max_{\forall x \in X}(T(x\uparrow)[i])$

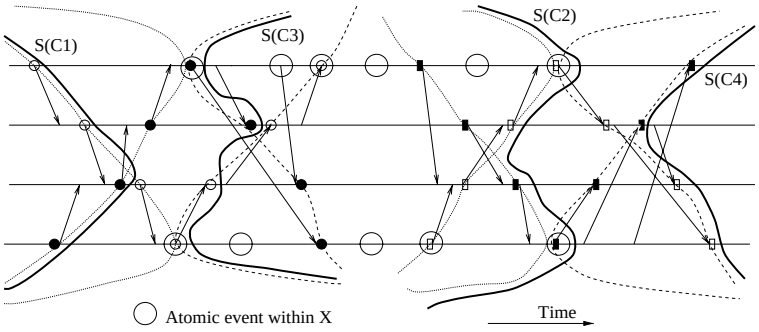

Fig. 3. Cuts C1, C2, C3, and C4 of poset X

cut, respectively. $K_x(\Phi)$ is a predicate that is true if event x has knowledge of Φ. Ψ^x represents the knowledge available at event x.

1. $\cap_{\Downarrow} X$ is the maximum set of events that causally precede every $x \in X$. It represents the maximum execution prefix about which all events in X have knowledge.
   In knowledge-theoretic terms, $\forall x \in X$, $K_x(\Phi_{\cap_{\Downarrow} X}) = true$. Also, $\forall x \in X$, $\Phi_{\cap_{\Downarrow} X} \subseteq \Psi^x$.
2. $\cup_{\Downarrow} X$ is the maximum set of events such that each event causally precedes some $x \in X$. It represents the maximum execution prefix about which only all the events in X collectively have knowledge, but no one event in X may have complete knowledge.
   In knowledge-theoretic terms, $\bigcup_{\forall x \in X}(\Psi^x) = \Phi_{\cup_{\Downarrow} X}$. Also, $\forall e_i \in S(\cup_{\Downarrow} X) \exists x \in X$, $\Psi^x \supseteq \Psi^{e_i}$.
3. $\cap_{\Uparrow} X$ is a cut such that $S(\cap_{\Uparrow} X)$ is the set of earliest events on each node that are causally preceded by some $x \in X$. It represents the minimum execution prefix such that all the maximum events of this prefix are preceded by at least one event in X.
   In knowledge-theoretic terms, $\forall e_i \in S(\cap_{\Uparrow} X) \exists x \in X$, $K_{e_i}(\Phi_x) = true$. Also, $\forall e_i \in S(\cap_{\Uparrow} X) \exists x \in X$, $\Psi^x \subseteq \Psi^{e_i}$.
4. $\cup_{\Uparrow} X$ is a cut such that $S(\cup_{\Uparrow} X)$ is the set of earliest events on each node that are causally preceded by every $x \in X$. It represents the minimum execution prefix such that all the maximum events of this prefix are causally preceded by all the events in X.
   In knowledge-theoretic terms, $\forall e_i \in S(\cup_{\Uparrow} X)$, $K_{e_i}(\Phi_X) = true$. Also, $\forall e_i \in S(\cup_{\Uparrow} X) \forall x \in X$, $\Psi^x \subseteq \Psi^{e_i}$.

Key Idea 1. The cuts $\cap_{\Downarrow} X$, $\cup_{\Downarrow} X$, $\cap_{\Uparrow} X$, and $\cup_{\Uparrow} X$ aggregate the causality information about all x in a nonatomic event X in a condensed form, as described above. Once identified at a one-time cost, these cuts can be reused at a low cost to evaluate causality relations with respect to all other nonatomic events. We will use these condensed forms of causality information to derive efficient tests for the causality relations in Table 1.

Notation. We use $\downarrow X$ and $X\uparrow$, respectively, to denote cuts about the past and future associated with any nonempty subset X of E.

Definition 11 *For any nonatomic poset event X,*
- $\downarrow X$ *denotes either* $C1(X)$ *or* $C2(X)$.
- $X\uparrow$ *denotes either* $C3(X)$ *or* $C4(X)$.

We now informally show the relations $R(X, Y)$, for R in Table 1, are implied by the relation $\not\ll$ on appropriately identified cuts $C1$, $C2$, $C3$, and $C4$ associated with X and Y using Lemma 2 and the knowledge-theoretic analysis of the cuts $C1$, $C2$, $C3$, and $C4$. Formal proofs of the equivalence between $R(X, Y)$, for R in Table 1, and the relation $\not\ll$ on appropriately identified cuts associated with X and Y are given subsequently in Theorem 4, Sect. 3.5. Note that if $\not\ll(C, C')$, then some event in $S(C')$ happens before (or equals) some event in $S(C)$. However, in the following discussion, we assume that if $\not\ll(C, C')$, then some event in $S(C')$ happens before some event in $S(C)$; subsequently, in Sect. 3.5, we justify this assumption.

$R^*a(X, Y)$: This relation holds if $\forall x \in X$, $\not\ll(\cap_{\Downarrow} Y, x\uparrow)$, i.e., $\forall x \in X$, some event in $S(\cap_{\Downarrow} Y)$ happens causally after some event in $S(x\uparrow)$, implying by the use of a transitive argument and Lemma 2.1 that for all events x in X, all events in Y happen causally after x.

$R^*a'(X, Y)$: This relation holds iff $\forall y \in Y$, $\not\ll(\downarrow y, \cup_{\Uparrow} X)$, i.e., $\forall y \in Y$, some event in $S(\downarrow y)$ happens causally after some event in $S(\cup_{\Uparrow} X)$, implying by the use of a transitive argument and Lemma 2.4 that for all events y in Y, y happens causally after all events in X.

$R^*b(X, Y)$: This relation holds iff $\forall x \in X$, $\not\ll(\cup_{\Downarrow} Y, x\uparrow)$, i.e., $\forall x \in X$, some event in $S(\cup_{\Downarrow} Y)$ happens causally after some event in $S(x\uparrow)$, implying by the use of a transitive argument and Lemma 2.2 that for all events x in X, some event in Y happens causally after x.

$R^*b'(X, Y)$: This relation holds iff $\not\ll(\cup_{\Downarrow} Y, \cup_{\Uparrow} X)$, i.e., some event in $S(\cup_{\Downarrow} Y)$ happens causally after some event in $S(\cup_{\Uparrow} X)$, implying by the use of a transitive argument and Lemmas 2.2 and 2.4 that some event in Y happens causally after all the events in X.

$R^*c(X,Y)$: This relation holds iff $\not\ll (\cap_{\Downarrow} Y, \cap_{\Uparrow} X)$, i.e., some event in $S(\cap_{\Downarrow} Y)$ happens causally after some event in $S(\cap_{\Uparrow} X)$, implying by the use of a transitive argument and Lemmas 2.1 and 2.3 that all the events in Y happen causally after the same event in X.

$R^*c'(X,Y)$: This relation holds iff $\forall y \in Y$, $\not\ll(\downarrow y, \cap_{\Uparrow} X)$, i.e., $\forall y \in Y$, some event in $S(\downarrow y)$ happens causally after some event in $S(\cap_{\Uparrow} X)$, implying by the use of a transitive argument and Lemma 2.3 that for all events y in Y, y happens causally after some event in X.

$R^*d(X,Y)$, $R^*d'(X,Y)$: This relation holds iff $\not\ll (\cup_{\Downarrow} Y, \cap_{\Uparrow} X)$, i.e., some event in $S(\cup_{\Downarrow} Y)$ happens causally after some event in $S(\cap_{\Uparrow} X)$, implying by the use of a transitive argument and Lemmas 2.2 and 2.3 that some event in Y happens causally after some event in X.

3.3 Timestamps

3.3.1 Timestamps of atomic events

Conceptually, the causality relation between two events can be determined by examining their space-time coordinates. In practice, logical clocks are used to maintain time at each process/node and track causality. Each atomic event is assigned a timestamp which is the clock value when the event occurs. Dummy events are treated like regular events when assigning or computing timestamps. We assume a clock system such that the timestamps assigned to events have the following property [15, 29] which is essential to determine causality between any two events.

Property of timestamps: $e \prec e'$ iff $T(e) <^2 T(e')$

We assume the following canonical vector clocks [15,29] that have this property. Each primitive atomic event e is assigned a timestamp $T(e)$ that is a vector[2] of size $|P|$, where P is the set of all process/node partitions. This is the minimum size of a clock/timestamp that is required to capture the above property of timestamps [11]. Assuming that the identifier of a process/node i is i itself, $T(e)$ is defined as follows.

Definition 12 $T(e) \equiv \forall i \in P$, $T(e)[i] = |\{e_i \mid e_i \preceq e\}|$, *i.e., $T(e)[i]$ is the number of events on node i that causally precede or equal e.*

Let $\mathscr{T}$ be the set $\{T(e) \mid e \in E\}$. Observe that there is an isomorphism between the event structure $(E, \prec)$ and the timestamp structure $(\mathscr{T}, <)$ [30].

Analogous to the timestamp $T(e)$, the reverse timestamp $T^R(e)$ of an event indicates the number of events in the future that are causally affected by the current event [7]. Observe that once the timestamp structure is established for the entire computation, the "reverse" timestamp $T^R(e)$ can also be established.

Definition 13 $T^R(e) \equiv \forall i \in P$, $T^R(e)[i] = |\{e_i \mid e_i \succeq e\}|$, *i.e., $T^R(e)[i]$ is the number of events on node i that causally happen after or equal e.*

[2] We will use the $<$ relation as the "less than" relation between integers and between vectors of integers. The usage should be clear from context.

Given two distinct atomic events e_j and e'_k, the causality between them can be tested as follows. If $j \neq k$, then $e_j \prec e'_k$ iff $T(e_j)[j] \leq T(e'_k)[j]$. If $j = k$, then $e_j \prec e'_k$ iff $T(e_j)[j] < T(e'_k)[j]$.

3.3.2 Timestamps of cuts and nonatomic events

For a cut C, we define its timestamp $T(C)$ such that the ith component of the timestamp is the maximum of the ith components of the timestamps of all the events in C that occur at node i.

Definition 14 $T(C) \equiv \forall i \in P, T(C)[i] = \max_{\forall x_i \in C}(T(x_i)[i])$

Lemma 3 *The timestamp of a cut composed by the union or intersection of other cuts is as follows.*

- *If $C \equiv \bigcap_{s=1,k} C^s$ then $T(C) \equiv \forall i \in P$, $T(C)[i] = \min_{s=1,k}(T(C^s)[i])$*
- *If $C \equiv \bigcup_{s=1,k} C^s$ then $T(C) \equiv \forall i \in P$, $T(C)[i] = \max_{s=1,k}(T(C^s)[i])$*

Proof. Follows from Definitions 5 and 14, and the lattice structure $(\mathscr{C}, \subset)$. □

Corollary 1 *The timestamps of the cuts of a poset defined in the second column of Table 5 are given in the third column of the table.*

Proof. Follows from Lemma 3. □

Causality between nonatomic poset events X and Y is determined as follows. Compare the timestamp of an appropriately chosen cut associated with X with the timestamp of an appropriately chosen cut associated with Y to test for the $\ll$ relation between the two cuts (Sect. 3.4). Then formally show that this test (possibly multiple such tests) is equivalent to the test for causality (Sect. 3.5).

From Definitions 8 and 12, observe that $T(\downarrow x)$, the timestamp of cut $\downarrow x$ associated with any event x is simply $T(x)$. From Definitions 9 and 13, observe that $T(x\uparrow)$, the timestamp of cut $x\uparrow$ associated with any event x is as follows: $T(x\uparrow)[i] = |E_i| - T^R(x)[i] - 1$. (This calculation accounts for the 2 dummy events in E_i.) Using timestamps of cuts $\downarrow x$ and $x\uparrow$, the overhead of computing timestamps of the cuts given in Table 5 for each X is as follows. The i^{th} component of the timestamp of each of $C1(X)$ and $C2(X)$ is a min and max function, respectively, of $\{T([S(\downarrow x)]_i)[i] \mid x \in X\}$. Similarly, the i^{th} component of the timestamp of each of $C3(X)$ and $C4(X)$ is a min and max function, respectively, of $\{T([S(x\uparrow)]_i)[i] \mid x \in X\}$. Observe that for $C1(X)$ and $C3(X)$, it suffices to consider only the least element in $X \bigcap E_i$, for each $i \in N_X$. Similarly, observe that for $C2(X)$ and $C4(X)$, it suffices to consider only the latest element in $X \bigcap E_i$, for each $i \in N_X$. Consequently, the i^{th} component of the timestamp of each of $C1(X)$, $C2(X)$, $C3(X)$, and $C4(X)$ is a min or max function over the i^{th} components of $|N_X|$ timestamps, which has a $|N_X|$ computational complexity. For $|P|$ components of the timestamp, the computational complexity is $|N_X| \times |P|$. Fortunately, we will show that all the $|P|$ components of the timestamps of the cuts are not required for computing the $\ll$ relation between

the cuts. Rather, for event X, only the $|N_X|$ components for the nodes in N_X are relevant, and hence, only these need to be computed. Therefore, the computational complexity of computing the timestamp of a cut $C1(X)$, $C2(X)$, $C3(X)$, or $C4(X)$ is $|N_X|^2$. Observe that this computation of the timestamps of the above cuts has a one-time cost. Once computed, the timestamp of a cut associated with X can be reused in the evaluation of the relation $\ll$ between this cut and cuts associated with multiple other nonatomic events. This cost of computing the timestamps will be considered further in Sect. 3.6.1 when evaluating the efficacy of the proposed method of testing for the relations in Table 1.

3.4 Efficient evaluation of $\ll$ between past and future cuts of posets

This section derives an efficient test (Theorem 3) for the $\ll$ relation between $\downarrow Y$, a past cut of a poset event Y, and $X\uparrow$, a future cut of poset event X. Observe from Definition 7 that $\ll(C, C')$ can be tested by $|P|$ integer comparisons: whether $T(e_i)[i] < T(e'_i)[i]$, where $e_i \in S(C)$ and $e'_i \in S(C')$, for every $i \in P$. The objective is to minimize the number of comparisons needed for this test, given the added information that C and C' are structured cuts of the form $\downarrow Y$ and $X\uparrow$, respectively. (The cuts $\downarrow Y$ and $X\uparrow$ are determined by the sets of cuts $\{\downarrow y \mid y \in Y\}$ and $\{x\uparrow \mid x \in X\}$, respectively, which have the property that each cut $\downarrow y$ has a unique maximal event and each cut $x\uparrow$ has a unique minimal event.) The efficient test we formulate will be used in Sect. 3.5, where it is shown that this test for relation $\ll$ between appropriately chosen cuts of Y and X is equivalent to a test for the causality relations between X and Y proposed in Table 1.

Lemma 4 states that $\not\ll(\downarrow e_i, e_j\uparrow)$ iff e_j causally precedes e_i.

Lemma 4 $\downarrow e_i \not\ll e_j\uparrow$ *iff* $e_j \preceq e_i$.

Proof. Observe that $e_j\uparrow \neq E^\perp$ and that $N_{\downarrow e_i} \subseteq N_{e_j\uparrow}$.

($\Longrightarrow$): By applying Definition 7.2 to the L.H.S., we infer that $\exists k \mid [S(\downarrow e_i)]_k \succeq [S(e_j\uparrow)]_k$. It follows from the definition of $e_j\uparrow$ and $\downarrow e_i$ that $e_j \preceq [S(e_j\uparrow)]_k \preceq [S(\downarrow e_i)]_k \preceq e_i$, hence $e_j \preceq e_i$.

($\Longleftarrow$): From the R.H.S., we infer that $[S(e_j\uparrow)]_i \preceq e_i$. Also, we have $e_i = [S(\downarrow e_i)]_i$ from the definition of $\downarrow e_i$. Therefore, $[S(e_j\uparrow)]_i \preceq [S(\downarrow e_i)]_i$, and by Definition 7.4, $\downarrow e_i \not\ll e_j\uparrow$. □

As per Lemma 4, the test for $\downarrow e_i \not\ll e_j\uparrow$ is exactly a check for causality between the two atomic events. From the discussion in Sect. 3.3.1, this check requires one integer comparison. We would like to generalize Lemma 4 to a test for $\downarrow Y \ll X\uparrow$ and we use Definition 7.2 of $\not\ll(C, C')$ as this form is more convenient to derive the simplified test.

Assertion 1 $\downarrow Y \not\ll X\uparrow$ *iff* $\exists z \in S(\downarrow Y)$, $z \in S(X\uparrow) \bigvee z \notin X\uparrow$. *(From Definition 7.2 and noting that $X\uparrow \neq E^\perp$)*

The R.H.S. of Assertion 1 is true iff $\exists z_i \in S(\downarrow Y)$ such that $T(z_i)[i] \geq T([S(X\uparrow)]_i)[i]$. Observe that there are $|P|$ elements in $S(\downarrow Y)$. Therefore, to determine $\downarrow Y \not\ll X\uparrow$ as per Assertion 1 requires $|P|$ integer comparisons which is the same computational complexity as that for the evaluation as per Definition 7. However, we now show that this test can be reduced to $\min(|N_X|, |N_Y|)$ integer comparisons.

Lemma 5 is an intermediate result. Lemma 5.1 shows the causality relation between each member of $S(X\uparrow)$ and a member(s) of $S(X\uparrow)$ that occur at a node(s) in N_X. Lemma 5.2 shows the causality relation between each member of $S(\downarrow Y)$ and a member(s) of $S(\downarrow Y)$ that occur at a node(s) in N_Y.

Lemma 5 *The members of $S(X\uparrow)$ and $S(\downarrow Y)$ are related to their events that lie in N_X and N_Y, respectively, as follows:*

5.1 $\forall e' \in S(X\uparrow)\, \exists x_i \in S(X\uparrow), i \in N_X \bigwedge x_i \preceq e'$
5.2 $\forall e' \in S(\downarrow Y)\, \exists y_i \in S(\downarrow Y), i \in N_Y \bigwedge y_i \succeq e'$

Proof. The proof of (5.1) follows from Lemma 2.3 and Lemma 2.4. The proof of (5.2) follows from Lemma 2.1 and Lemma 2.2. □

The discussion following Assertion 1 showed that the R.H.S. of Assertion 1 required $|P|$ integer comparisons. Lemma 6 gives a more efficient test for the R.H.S. of Assertion 1, requiring only $|N_X|$ integer comparisons corresponding to the timestamps' components for nodes in N_X. The simplification is possible because we are not evaluating the $\ll$ relation between two arbitrary cuts but rather between the cuts $\downarrow Y$ and $X\uparrow$ which are determined by the sets of cuts $\{\downarrow y \mid y \in Y\}$ and $\{x\uparrow \mid x \in X\}$, respectively, which have the property that each cut $\downarrow y$ has a unique maximal event and each cut $x\uparrow$ has a unique minimal event. This property suggests that sufficient causal information about X is condensed into the N_X components, and leads to the following idea.

Key Idea 2. If $\ll(\downarrow Y, X\uparrow)$ is violated (as in the L.H.S. of Assertion 1), then some event in $S(\downarrow Y)$ equals or happens causally after some event in $S(X\uparrow)$. This violation must occur at a node in N_X because the events $[S(X\uparrow)]_{N_X}$ are the earliest possible events among events in $S(X\uparrow)$, in terms of causality (see Lemma 6). Using analogous reasoning, this violation must occur at a node in N_Y because the events $[S(\downarrow Y)]_{N_Y}$ are the latest possible events among events in $S(\downarrow Y)$, in terms of causality (see Lemma 7). Therefore, the violation of $\ll(\downarrow Y, X\uparrow)$ can be detected by $|N_X|$ checks for causality between atomic events, by comparing for each i in N_X, $T([S(X\uparrow)]_i)[i]$ and $T([S(\downarrow Y)]_i)[i]$. Analogously, the violation of $\ll(\downarrow Y, X\uparrow)$ can be detected by $|N_Y|$ checks for causality between atomic events, by comparing for each i in N_Y, $T([S(X\uparrow)]_i)[i]$ and $T([S(\downarrow Y)]_i)[i]$. Therefore, the violation of $\ll(\downarrow Y, X\uparrow)$ can be detected in $\min(|N_X|, |N_Y|)$ integer comparisons. Whenever $|N_X|$ or $|N_Y|$ is less than $|P|$, we have a more efficient test for the causality relation between $S(\downarrow Y)$ and $S(X\uparrow)$.

Lemma 6 $(\exists z \in S(\downarrow Y), z \in S(X\uparrow) \bigvee z \notin X\uparrow)$ *iff* $(\exists z'_i \in S(\downarrow Y), i \in N_X \bigwedge [S(X\uparrow)]_i \preceq z'_i)$

Proof. See Appendix B. □

Theorem 1 $\downarrow Y \not\ll X\uparrow$ *iff* $(\exists z'_i \in S(\downarrow Y), i \in N_X \bigwedge [S(X\uparrow)]_i \preceq z'_i)$

Proof. Follows from Assertion 1 and Lemma 6. □

The significance of Theorem 1 is that the test for $\downarrow Y \not\ll X\uparrow$ can be performed by only $|N_X|$ integer comparisons, corresponding to the N_X components of the timestamps of $\downarrow Y$ and $X\uparrow$. Specifically, $\downarrow Y \not\ll X\uparrow$ iff $\exists z_i \in S(\downarrow Y)$, where $i \in N_X$, such that $T(z_i)[i] \geq T([S(X\uparrow)]_i)[i]$.

We now also show that the test for $\downarrow Y \not\ll X\uparrow$ can be performed by only $|N_Y|$ integer comparisons, corresponding to the N_Y components of the timestamps of $\downarrow Y$ and $X\uparrow$. Analogous to Assertion 1, Lemma 6, and Theorem 1 which were stated in terms of $z \in S(\downarrow Y)$, we have Assertion 2, Lemma 7, and Theorem 2, respectively, in terms of $z \in S(X\uparrow)$.

Assertion 2 $\downarrow Y \not\ll X\uparrow$ *iff* $\exists z \in S(X\uparrow), z \in \downarrow Y$. *(From Definition 7.4 and noting that* $X\uparrow \neq E^{\perp}$ *and* $N_{\downarrow Y} \subseteq N_{X\uparrow}$*)*

Lemma 7 $(\exists z \in S(X\uparrow), z \in \downarrow Y)$ *iff* $(\exists z'_i \in S(X\uparrow), i \in N_Y \bigwedge z'_i \preceq [S(\downarrow Y)]_i)$

Proof. See Appendix B. □

Theorem 2 $\downarrow Y \not\ll X\uparrow$ *iff* $(\exists\ z'_i \in S(X\uparrow), i \in N_Y \bigwedge z'_i \preceq [S(\downarrow Y)]_i)$

Proof. Follows from Assertion 2 and Lemma 7. □

The significance of Theorem 2 is that the test for $\downarrow Y \not\ll X\uparrow$ can be performed by only $|N_Y|$ integer comparisons, corresponding to the N_Y components of the timestamps of $\downarrow Y$ and $X\uparrow$. Specifically, $\downarrow Y \not\ll X\uparrow$ iff $\exists\ z_i \in S(X\uparrow)$, where $i \in N_Y$, such that $T(z_i)[i] \leq T([S(\downarrow Y)]_i)[i]$.

Theorem 3 $\downarrow Y \not\ll X\uparrow$ *can be determined in* $\min(|N_X|, |N_Y|)$ *integer comparisons.*

Proof. From Theorems 1 and 2, $\downarrow Y \not\ll X\uparrow$ can be evaluated in $|N_X|$ and $|N_Y|$ integer comparisons, respectively. The result follows. □

Theorem 3 states that the test for $\downarrow Y \not\ll X\uparrow$ can be performed by only $\min(|N_X|, |N_Y|)$ integer comparisons. (What is required is either the N_X or N_Y components of the timestamps of $\downarrow Y$ and $X\uparrow$. From the discussion in Sect. 3.3.2, it follows that the one-time overhead of creating these timestamps is as follows. To create either the N_X components of the timestamp of $\downarrow Y$ or the N_Y components of the timestamp of $X\uparrow$ takes a computational complexity of $O(|N_X| \times |N_Y|)$. To create either the N_X components of the timestamp of $X\uparrow$ or the N_Y components of the timestamp of $\downarrow Y$ takes a computational complexity of $O(|N_X| \times |N_X|)$ and $O(|N_Y| \times |N_Y|)$, respectively.) The significance of Theorem 3 is that it will be used in Sect. 3.5 to provide the upper bound on the number of comparisons required to evaluate the causality relations between nonatomic poset events defined in Table 1.

3.5 Evaluation conditions for causality relations

In this section, we formalize the evaluation conditions for the causality relations in Table 1 and show that they have a linear computational complexity. In Theorem 4, we formally show the equivalence of the relation definitions $R(X, Y)$ in the second column of Table 1 to the validity of the $\ll$ relation between one (in some cases, $|N_X|$ or $|N_Y|$) pairs of cuts of the form $X\uparrow$ and $\downarrow Y$, as specified in the third column of Table 1. This equivalence was informally introduced in Sect. 3.2. By Theorem 3, the evaluation of $\downarrow Y \ll X\uparrow$ requires $\min(|N_X|, |N_Y|)$ integer comparisons. In Theorem 5, we use Theorems 3 and 4 to derive the exact number of integer comparisons needed to evaluate each of the causality relations of Table 1. Relations $R1$, $R1'$, $R2'$, $R3$, $R4$, and $R4'$ can be evaluated in $\min(|N_X|, |N_Y|)$ integer comparisons, relation $R2$ in $|N_X|$ integer comparisons, and relation $R3'$ in $|N_Y|$ integer comparisons. These simplified evaluation conditions also give a physical interpretation to the relations in terms of the cuts defined in Table 5 and which can be visualized using Fig. 3.

The causality relations between nonatomic poset events in the second column of Table 1 are defined using the "causes" ($\prec$) relation between the atomic events. However, the evaluation conditions in the third column of Table 1 are based on Theorems 1 and 2, and are therefore true for the "causes or equals" ($\preceq$) relation between atomic events. The $\preceq$ relation between events of X and Y evaluates to the same as the $\prec$ relation iff no two events, one from X and one from Y, in the evaluation have an identical timestamp. Hence, we use the following assertion.

Assertion 3 *When evaluating* $R(X, Y)$, $\forall R$ *in Table 1, no two events in* X *and* Y *have an identical timestamp.*

Assertion 3 can be satisfied even if there are common events in X and Y by using the following trick. Let z_i^x and z_i^y denote the event z_i that is common to X and Y, respectively. Solely for the purpose of satisfying Assertion 3, when computing the timestamp of Y as per Definition 10, assign to $T(z_i^y)[i]$, a value $T(z_i^y)[i] - \delta$, where δ is smaller than the smallest increment to the timestamp component on the occurrence of an event. Then $z_i^x \prec z_i^y$ iff $T(z_i^x) \leq T(z_i^y)$. Thus, by evaluating $T(z_i^x) \leq T(z_i^y)$ for $z_i^x \preceq z_i^y$, we effectively evaluate $z_i^x \prec z_i^y$. Observe that $(\mathcal{T}, <)$ remains isomorphic to $(E, \prec)$ even if this trick is used.

Theorem 4 *The relations defined in the second column of Table 1 are true iff the corresponding evaluation conditions given in the third column of Table 1 are true.*

Proof. In the proof, we assume that an event x occurs at node i and event y occurs at node j.

Part 1 ($\Rightarrow$): We show that if any relation from $\{R1, R2, R2', R3, R3', R4\}$ in the second column of Table 1 holds, then the corresponding evaluation condition in the third column is true.

(I) Relation $R1(X, Y)$, i.e., $\forall x \forall y, x \prec y$: $\forall y \in Y$, $[S(\downarrow y)]_i \succeq x$. Thus, $\min(\{[S(\downarrow y)]_i \mid y \in Y\}) \succeq x$. We also have $\min(\{[S(\downarrow y)]_i \mid y \in Y\}) = [S(\cap_{\Downarrow} Y)]_i$. So $[S(\cap_{\Downarrow} Y)]_i \succeq x = [S(x\uparrow)]_i$. Hence, $\cap_{\Downarrow} Y \not\ll x\uparrow$. But this is true for all $x \in X$. The evaluation condition $\bigwedge_{x \in X}[\cap_{\Downarrow} Y \not\ll x\uparrow]$ follows.

The other evaluation condition corresponding to $R1'$ can be similarly shown.

(II) Relation $R2(X, Y)$, i.e., $\forall x \exists y, x \prec y$: For any $y \in Y$, $[S(\cup_{\Downarrow} Y)]_i \succeq [S(\downarrow y)]_i$ because of construction of $\cup_{\Downarrow} Y$

(see Definition 10). Also, because $R2$ holds, for each $x \in X$, for some $y \in Y$, $y \succeq [S(\downarrow y)]_i \succeq x = [S(x\uparrow)]_i$. Thus, $[S(\cup_{\Downarrow} Y)]_i \succeq [S(x\uparrow)]_i$ and this holds for each $x \in X$. The evaluation condition follows.

(III) Relation $R2'(X, Y)$, i.e., $\exists y \forall x, x \prec y$: From the relation, we can infer that $\exists y \in Y \forall x \in X$, $[S(x\uparrow)]_j \preceq y$. Thus, max($\{[S(x\uparrow)]_j \mid x \in X\}$) $\preceq y$. We also have max($\{[S(x\uparrow)]_j \mid x \in X\}$) = $[S(\cup_{\Uparrow} X)]_j$. So $[S(\cup_{\Uparrow} X)]_j \preceq y$. We also have $y \preceq [S(\cup_{\Downarrow} Y)]_j$. The evaluation condition follows because $[S(\cup_{\Downarrow} Y)]_j \succeq [S(\cup_{\Uparrow} X)]_j$.

(IV) Relation $R3(X, Y)$, i.e., $\exists x \forall y, x \prec y$: From the relation, we can infer that $\exists x \in X \forall y \in Y$, $[S(\downarrow y)]_i \succeq x$. Thus, min($\{[S(\downarrow y)]_i \mid y \in Y\}$) $\succeq x$. We also have min($\{[S(\downarrow y)]_i \mid y \in Y\}$) = $[S(\cap_{\Downarrow} Y)]_i$. So $[S(\cap_{\Downarrow} Y)]_i \succeq x$. We also have $x \succeq [S(\cap_{\Uparrow} X)]_i$. The evaluation condition follows because $[S(\cap_{\Downarrow} Y)]_i \succeq [S(\cap_{\Uparrow} X)]_i$.

(V) Relation $R3'(X, Y)$, i.e., $\forall y \exists x, x \prec y$: For any $x \in X$, $[S(\cap_{\Uparrow} X)]_j \preceq [S(x\uparrow)]_j$ because of the construction of $\cap_{\Uparrow} X$ (see Definition 10). Also, because $R3'$ holds, for each $y \in Y$, for some $x \in X$, $x \preceq [S(x\uparrow)]_j \preceq y = [S(\downarrow y)]_j$. Thus, $[S(\downarrow y)]_j \succeq [S(\cap_{\Uparrow} X)]_j$, and this holds for each $y \in Y$. The evaluation condition follows.

(VI) Relation $R4(X, Y)$, i.e., $\exists x \exists y, x \prec y$: As noted in (II) above, for any $y \in Y$, $[S(\cup_{\Downarrow} Y)]_i \succeq [S(\downarrow y)]_i$. Also, because of the construction of $\cap_{\Uparrow} X$ (see Definition 10), for any $x \in X$, $x \succeq [S(\cap_{\Uparrow} X)]_i$, similar to the reasoning in (V) above. Given that the relation holds, then $y \succeq [S(\downarrow y)]_i \succeq x$, therefore, $[S(\cup_{\Downarrow} Y)]_i \succeq [S(\downarrow y)]_i \succeq x \succeq [S(\cap_{\Uparrow} X)]_i$ by combining the above, and hence $[S(\cup_{\Downarrow} Y)]_i \succeq [S(\cap_{\Uparrow} X)]_i$. The evaluation condition follows.

(end of Part 1).

Part 2 ($\Longleftarrow$): We show that a relation from $\{R1, R2, R2', R3, R3', R4\}$ given in the second column of Table 1 holds if the corresponding evaluation condition in the third column is true. Throughout this proof, we will refer to Assertion 1 and its use of variable z which was defined such that $z \in S(\downarrow Y)$ and $z \in S(X\uparrow) \bigvee z \notin X\uparrow$. Observe that z exists in all the cases considered.

(I) Relation $R1(X, Y)$, i.e., $\forall x \forall y$, $x \prec y$: Consider any $x \in X$. From the definition of $x\uparrow$ (Definition 9) and Assertion 1, it follows that $x \preceq z$. From Lemma 2.1, it follows that $\forall y \in Y$, $z \preceq y$. By transitivity, $\forall y \in Y$, $x \preceq y$.
The above argument is true for all $x \in X$. Hence, $\forall x \in X \forall y \in Y$, $x \preceq y$.
The validity of the other evaluation condition is similarly shown.

(II) Relation $R2(X, Y)$, i.e., $\forall x \exists y$, $x \prec y$: Consider any $x \in X$. From the definition of $x\uparrow$ (Definition 9), and from Assertion 1, $x \preceq z$. From Lemma 2.2, it follows that $\exists y \in Y$, $z \preceq y$. By transitivity, $x \preceq y$.
The above argument is true $\forall x \in X$. Hence, $\forall x \in X \exists y \in Y$, $x \preceq y$.

(III) Relation $R2'(X, Y)$, i.e., $\exists y \forall x$, $x \prec y$: Consider an event z that exists by Assertion 1. From Lemma 2.4, it follows that $\forall x \in X$, $x \preceq z$. From Lemma 2.2, it follows that $\exists y \in Y$, $z \preceq y$. By transitivity, $\exists y \in Y \forall x \in X$, $x \preceq y$.

(IV) Relation $R3(X, Y)$, i.e., $\exists x \forall y$, $x \prec y$: Consider an event z that exists by Assertion 1. From Lemma 2.3, it follows that $\exists x \in X$, $x \preceq z$. From Lemma 2.1, it follows that $\forall y \in Y$, $z \preceq y$. By transitivity, $\exists x \in X \forall y \in Y$, $x \preceq y$.

(V) Relation $R3'(X, Y)$, i.e., $\forall y \exists x$, $x \prec y$: Consider any $y \in Y$. From the definition of $\downarrow y$ (Definition 8) and Assertion 1, $z \preceq y$. From Lemma 2.3, it follows that $\exists x \in X$, $x \preceq z$. By transitivity, $x \preceq y$.
The above argument is true $\forall y \in Y$. Hence, $\forall y \in Y \exists x \in X$, $x \preceq y$.

(VI) Relation $R4(X, Y)$, i.e., $\exists x \exists y$, $x \prec y$: Consider an event z that exists by Assertion 1. From Lemma 2.3, it follows that $\exists x \in X$, $x \preceq z$. From Lemma 2.2, it follows that $\exists y \in Y$ such that $z \preceq y$. By transitivity, $\exists x \in X \exists y \in Y$, $x \preceq y$.

From (I)–(VI), each evaluation condition in the third column of Table 1 implies the corresponding causality relation between nonatomic poset events in the second column, but with the $\prec$ relation between atomic events replaced by $\preceq$ (see Definition 7). Recall that the evaluation conditions were evaluated after assigning modified timestamps to satisfy Assertion 3. The use of such modified timestamps was to trick the evaluation of $\preceq$ using these timestamps into being equivalent to the evaluation of $\prec$, as explained in the discussion following Assertion 3. Therefore, the $\preceq$ relation is equivalent to the $\prec$ relation in the evaluation method. Hence, the relations in the second column of Table 1 are true if the corresponding evaluation conditions in the third column are true. (end of Part 2). □

Theorem 5 *Each relation $R(X, Y)$ in Table 1 can be evaluated with the following computational complexity: relations $R1$, $R1'$, $R2'$, $R3$, $R4$, and $R4'$ can be evaluated in* min($|N_X|, |N_Y|$) *integer comparisons, relations $R2$ in $|N_X|$ integer comparisons, and relations $R3'$ in $|N_Y|$ integer comparisons.*

Proof. The computational complexity of testing the conditions in the third column of Table 1 is the computational complexity of testing the corresponding relations in the second column, by Theorem 4.

Relations $R2'$, $R3$, $R4$, $R4'$: These relations can be evaluated using a single test $\downarrow Y \ll X\uparrow$. By Theorem 3, these relations can be evaluated in min($|N_X|, |N_Y|$) integer comparisons.

Relation $R2$: This relation can be evaluated using $|N_X|$ tests of the form $\downarrow Y \ll X''\uparrow$, where $|N_{X''}| = 1$. By Theorem 3, each test can be evaluated in 1 integer comparison. So the relation can be evaluated in $|N_X|$ integer comparisons.

Relation $R3'$: This relation can be evaluated using $|N_Y|$ tests of the form $\downarrow Y'' \ll X\uparrow$, where $|N_{Y''}| = 1$. By Theorem 3, each test can be evaluated in 1 integer comparison. So the relation can be evaluated in $|N_Y|$ integer comparisons.

Relations $R1$, $R1'$: By reasoning similar to that for $R2$ and $R3'$, these relations can be evaluated in $|N_X|$ and also in

$|N_Y|$ integer comparisons, i.e., $\min(|N_X|, |N_Y|)$ integer comparisons. □

The relations in $\mathscr{R}$ can be replaced by similar relations with the $\preceq$ relation between the individual atomic events instead of the $\prec$ relation. As seen above, the evaluation conditions work for these modified relations with the same linear complexity – the only difference is that Assertion 3 is not relevant.

Recall that each of the 32 relations $r(X, Y)$, for $r \in \mathscr{R}$, is equivalent to a relation $R(\hat{X}, \hat{Y})$, where R belongs to Table 1, by using a suitable quantification of X and Y in Table 1 to represent their various proxies $\hat{X}$ and $\hat{Y}$ instead. Each of the 2 proxies of a nonatomic event has 4 cuts associated with it. Figure 4 illustrates the four cuts associated with the two proxies of the event X of Fig. 3. The surfaces of the cuts are marked as in Fig. 3. These cuts can be used in Theorems 3 and 4 upon which Theorem 5 is based. Therefore, by using a suitable quantification of X and Y to represent their proxies, we have the following.

- Theorem 4 gives the evaluation methodology for each relation $r(X, Y)$, for $r \in \mathscr{R}$ (see the third column of Table 4). This also results from a suitable quantification of X and Y in Table 1.
- Theorem 5 gives the exact computational complexity of evaluating each relation $r(X, Y)$, for $r \in \mathscr{R}$. Specifically, relations R^*a, R^*a', R^*b', R^*c, R^*d, and R^*d' can be evaluated in $\min(|N_{\hat{X}}|, |N_{\hat{Y}}|)$ integer comparisons, relations R^*b in $|N_{\hat{X}}|$ integer comparisons, and relations R^*c' in $|N_{\hat{Y}}|$ integer comparisons. These evaluation conditions have only a linear computational complexity.

3.6 Overhead analysis

Implicit in the analysis of the computational complexity of evaluating the relations in $\mathscr{R}$ (see Theorem 5, the discussion following it, and Table 4), we assumed that the timestamp structure was established. However, that requires some overhead. We now analyze the overall overhead in answering Problems 1 and 2.

3.6.1 Timestamp structure

As the computation progresses, the vector timestamp of each event is locally recorded, and periodically, the vector timestamps are collected by a central process to establish the timestamp structure. (Note that the vector timestamps can get large but as shown in Theorem 3, the evaluation of $r(X, Y)$ requires only those components of the timestamps of X and Y that correspond to nodes in $N_{\hat{X}}$ and/or $N_{\hat{Y}}$.) The cost of maintaining a local vector clock and of piggybacking a vector timestamp on messages is essential to track causality between atomic events in a distributed computation based on message-passing [15, 29]. Therefore, they contribute no overhead to the computation of causality between nonatomic poset events. As the trace of the execution is collected at the central process, the application identifies $\mathscr{A}$, the set of nonatomic events of interest to it. Implicit in the identification of each $A \in \mathscr{A}$ is the identification of the proxies L_A and U_A. These are trivially identified assuming that the execution trace stores atomic events that occurred at each node in sorted order. For each proxy $\hat{X}$, where $\hat{X}$ is L_X and U_X, of each $X \in \mathscr{A}$, the timestamps of $C1(\hat{X})$, $C2(\hat{X})$, $C3(\hat{X})$, $C4(\hat{X})$ are calculated using the timestamps of each $\downarrow x$ and $x\uparrow$, where $x \in \hat{X}$. The ith component of the timestamp of $C1(\hat{X})$ and $C2(\hat{X})$ can be computed in $|N_{\hat{X}}|$ steps as it requires the identification of the min and max, respectively, of a set of $|N_{\hat{X}}|$ integers $\{T([S(\downarrow x)]_i)[i] \mid x \in \hat{X}\}$. Similarly, the ith component of the timestamp of $C3(\hat{X})$ and $C4(\hat{X})$ can be computed in $|N_{\hat{X}}|$ steps as it requires the identification of the min and max, respectively, of a set of $|N_{\hat{X}}|$ integers $\{T([S(x\uparrow)]_i)[i] \mid x \in \hat{X}\}$. Thus, the ith component of the timestamps of the cuts $C1(L_X)$, $C2(L_X)$, $C3(L_X)$, $C4(L_X)$, $C1(U_X)$, $C2(U_X)$, $C3(U_X)$, $C4(U_X)$ are computed in $4 \times |N_{\hat{X}}|$ steps.

To evaluate $r(X, Y)$, where $r \in \mathscr{R}$, we evaluate $R(\hat{X}, \hat{Y})$, where $R \in$ Table 1. As per Theorem 4, this requires the comparison of the $N_{\hat{X}}$ components of the timestamps of both $\hat{X}$ and $\hat{Y}$, or the comparison of the $N_{\hat{Y}}$ components of the timestamps of both $\hat{X}$ and $\hat{Y}$. As X and Y can be two arbitrary events in $\mathscr{A}$, in the worst-case, we need to compute the components of the timestamps corresponding to nodes[3] in $\bigcup_{\forall A_i \in \mathscr{A}} N_{A_i}$. Hence, the worst-case complexity of computing the timestamp structure of all the proxies of all the events in $\mathscr{A}$ is $|\mathscr{A}| \times 4|N_A| \times |\bigcup_{\forall A_i \in \mathscr{A}} N_{A_i}|$. (We can do better by defining an ordering on members of $\mathscr{A}$ and when evaluating a relation between X and Y, always comparing the components of the timestamps corresponding to the node set of the lower ordered of X and Y. This reduces the above overhead of setting up timestamps by about half.) Further practical aspects of handling the trace are given in Appendix C.

3.6.2 Problem 1

Problem 1 aims at detecting a specific relation from $\mathscr{R}$ between every ordered pair of nonatomic events X and Y in $\mathscr{A}$.

The computational complexity of evaluating any relation using the naive definition of the relation given in the second column of Table 4 is $|\mathscr{A}|^2 \times |N_A|^2$.

The computational complexity of the proposed method to evaluate any relation in Table 4 is the sum of two components. The first component deals with the overhead of establishing the timestamp structure, computed in Sect. 3.6.1. The second component gives the actual overhead of evaluating the causality relations by the proposed method. From Theorem 5, it follows that the overhead of detecting a specific relation from $\mathscr{R}$ between each pair of nonatomic events in $\mathscr{A}$ by using the proposed method is $|\mathscr{A}|^2 \times |N_A|$.

The combined overhead of establishing the timestamp structure of cuts and evaluating the causality relations by the proposed method is thus $(|\mathscr{A}| \times 4|N_A| \times |\bigcup_{\forall A_i \in \mathscr{A}} N_{A_i}|) + (|\mathscr{A}|^2 \times |N_A|)$.

[3] In the following analysis, we will use N_A and $|N_A|$ instead of $N_{\hat{A}}$ and $|N_{\hat{A}}|$ because $|N_A|$ provides an upper bound on $|N_{\hat{A}}|$.

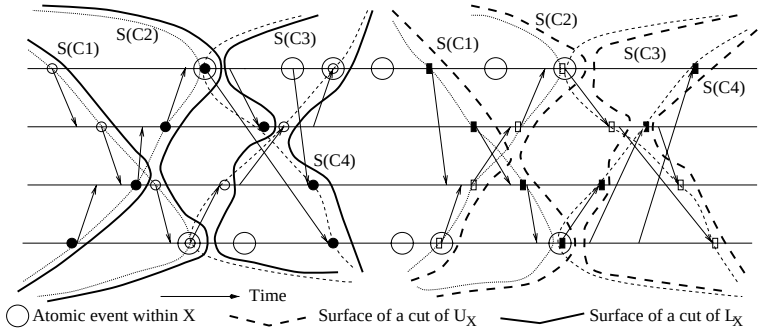

Fig. 4. Cuts of proxies L_X and U_X

The proposed method is better than the naive method in computational complexity for values of $|\mathscr{A}|$ and $|N_A|$ that satisfy the following inequality.

$$\left(|\mathscr{A}| \times 4|N_A| \times |\bigcup_{\forall A_i \in \mathscr{A}} N_{A_i}|\right) + (|\mathscr{A}|^2 \times |N_A|) \leq |\mathscr{A}|^2 \times |N_A|^2$$

The above inequality simplifies to the following.

$$\left(4 \times |\bigcup_{\forall A_i \in \mathscr{A}} N_{A_i}|\right) \div (|N_A| - 1) \leq |\mathscr{A}|, \text{ for } |N_A| \neq 1 \quad (1)$$

The term $|\bigcup_{\forall A_i \in \mathscr{A}} N_{A_i}|$ is the set of all nodes at which events of interest occur in the distributed computation. Clearly, $|\bigcup_{\forall A_i \in \mathscr{A}} N_{A_i}| \leq |P|$. Also $\mathscr{A} \subseteq 2^E$, the power set of all events that occur at all the nodes, (the set P) during the computation. Thus, $|\bigcup_{\forall A_i \in \mathscr{A}} N_{A_i}|$ is typically very much less than $|\mathscr{A}|$ and hence, it is profitable to use the proposed method. For $|N_A|$ = 1, we use the results in [21] that deals with interactions between linear intervals; the proposed theory and evaluation methods are applicable for $|N_A| > 1$. For a wide range of applications such as industrial process control and monitoring distributed activities, events at different sites collectively trigger various events at the same set of sites, which requires modeling of interaction between 2 nonatomic events having the same node set. In this case, the node sets of all the nonatomic events of interest to an application are identical and Inequality 1 simplifies as follows.

$$(4 \times |N_A|) \div (|N_A| - 1) \leq |\mathscr{A}|, \text{ for } |N_A| \neq 1 \quad (2)$$

Thus, the proposed method is profitable to address Problem 1 for all values of $|\mathscr{A}|$ and $|N_A|$ satisfying Equation 2. In particular, for $|N_A|$ = 2, $|\mathscr{A}| \geq 8$; for $|N_A|$ = 3, $|\mathscr{A}| \geq 6$; for $|N_A|$ = 5, $|\mathscr{A}| \geq 5$; and asymptotically as $|N_A| \longrightarrow \infty$, $|\mathscr{A}| \geq 4$. As $|\mathscr{A}|$ is typically much larger than these values, the proposed method of evaluating causality is very efficient.

3.6.3 Problem 2

Problem 2 aims at detecting every possible relation from $\mathscr{R}$ that holds between every ordered pair of nonatomic events X and Y in $\mathscr{A}$.

A set of axioms to reason with the proposed relations was proposed in [25]. For a given pair of nonatomic events, having detected that some relation holds between the events, several other relations can be deduced between the pair of events using the axiom system. Application-specific heuristics should be used to determine the order of evaluating the relations in $\mathscr{R}$ in conjunction with the axiom system because in each application, it is likely that some relations are more likely to hold than other relations. The savings offered by the use of the axiom system and by the choice of a judicious order of evaluation of relations are similar, whether the relations are evaluated using the naive method as per the definitions in the second column of Table 4, or using the method proposed in this paper. Therefore, we disregard the use of the axiom system for purposes of assessing the savings offered by the proposed method over the naive method in evaluating the relations. For a fair comparison of the two methods, we assume that each relation in $\mathscr{R}$ is evaluated.

The overhead of evaluating all the 24 relations from $\mathscr{R}$ between each pair of nonatomic events in $\mathscr{A}$ by using the naive definitions given in the second column of Table 4 is $24 \times |\mathscr{A}|^2 \times |N_A|^2$.

The combined overhead of establishing the timestamp structure of cuts and evaluating the causality relations by the proposed method is $(|\mathscr{A}| \times 4|N_A| \times |\bigcup_{\forall A_i \in \mathscr{A}} N_{A_i}|) + 24 \times (|\mathscr{A}|^2 \times |N_A|)$. The first term is the overhead to establish the timestamp structure for each of the eight cuts associated with the two proxies of each nonatomic event, as discussed in Sect. 3.6.1. The second term gives the overhead of detecting all the 24 relations from $\mathscr{R}$ between each pair of nonatomic events in $\mathscr{A}$ by using the proposed method, and is derived from Theorem 5.

The proposed method is better than the naive method in computational complexity when the following inequality holds.

$$\left(|\mathscr{A}| \times 4|N_A| \times \left|\bigcup_{\forall A_i \in \mathscr{A}} N_{A_i}\right|\right) + 24 \times (|\mathscr{A}|^2 \times |N_A|) \leq 24 \times (|\mathscr{A}|^2 \times |N_A|^2)$$

The above inequality simplifies to the following.

$$\left(\left|\bigcup_{\forall A_i \in \mathscr{A}} N_{A_i}\right|\right) \div (6 \times (|N_A - 1|)) \leq |\mathscr{A}|, \text{ for } |N_A| \neq 1 \quad (3)$$

From the discussion following Inequality 1, we have:
(a) $|\bigcup_{\forall A_i \in \mathscr{A}} N_{A_i}|$ is typically very much less than $|\mathscr{A}|$ and hence, it is profitable to use the proposed method.

(b) $|N_A| \neq 1$, and the results are applicable for $|N_A| > 1$. (c) For applications for which it can be assumed that the node sets of all the nonatomic events of interest to the application are identical, Inequality 3 simplifies to the following.

$$(|N_A|) \div (6 \times (|N_A - 1|)) \leq |\mathscr{A}|, \text{ for } |N_A| \neq 1 \quad (4)$$

From the above analysis, it follows that for all values of $|N_A|$ ($|N_A| \neq 1$), independent of the value of $|\mathscr{A}|$, the proposed method offers savings over the naive method.

3.7 A note on alternate evaluation conditions

Observe from the derivation of Theorem 3 that several of the relations can be evaluated either by projecting $X\uparrow$ on N_Y or by projecting $\downarrow Y$ on N_X. The projection of $X\uparrow$ on N_Y requires cuts of the form $X\uparrow$ which are somewhat harder to determine than cuts of the form $\downarrow X$. This is because it is not possible to determine a nondummy event $[S(X\uparrow)]_k$, for some k, until after that event actually occurs. Alternate evaluation conditions can be formulated, that project $\downarrow Y$ on N_X and deal with cuts of the form $\downarrow X$ rather than of the form $X\uparrow$. Evaluation conditions for the relations in Table 1, using only cuts of the form $\downarrow Z$, are given in the fourth column of Table 6. Next, we determine the complexity of evaluating $\downarrow X \subseteq \downarrow Y$, prove the correctness of the evaluation conditions in the fourth column of Table 6 and determine the complexity of these evaluation conditions.

Theorem 6 $\downarrow X \subseteq \downarrow Y$ *iff* ($\forall z_i \in S(\downarrow Y)$ *where* $i \in N_X$, $[S(\downarrow X)]_i \preceq z_i$)

Proof. ($\Longrightarrow$): $\forall i \in P$, $[S(\downarrow X)]_i \preceq [S(\downarrow Y)]_i$ by definition of the $\subseteq$ relation. Hence, the R.H.S. follows.

($\Longleftarrow$): For each node $i \in N_X$, we have $[S(\downarrow X)]_i \preceq [S(\downarrow Y)]_i$. As $\downarrow X$ is a downward-closed set whose maximum elements are members of X that occur at nodes in N_X, and $\downarrow Y$ is also downward-closed, for any node k, $[S(\downarrow X)]_k \preceq [S(\downarrow Y)]_k$. The L.H.S. follows. □

The significance of Theorem 6 is that the test for $\downarrow X \subseteq \downarrow Y$ can be performed by only $|N_X|$ integer comparisons, corresponding to the N_X components of the timestamps of $\downarrow X$ and $\downarrow Y$. Specifically, $\downarrow X \subseteq \downarrow Y$ iff $\forall z_i \in S(\downarrow Y)$, where $i \in N_X$, $T(z_i)[i] \geq T([S(\downarrow X)]_i)[i]$.

Theorem 7 $\downarrow X \subseteq \downarrow Y$ *can be determined in* $|N_X|$ *integer comparisons.*

Proof. Follows from Theorem 6 and the fact that $[S(\downarrow X)]_i \preceq [S(\downarrow Y)]_i$ can be evaluated in one integer comparison. □

Theorem 8 *The relations defined in the second column of Table 6 are true iff the corresponding evaluation conditions given in the fourth column of Table 6 are true.*

Proof. In the proof, we assume that an event x occurs at node i and event y occurs at node j.

Part 1 ($\Longrightarrow$): We show that if any relation from {$R1$, $R2$, $R2'$, $R3$, $R3'$, $R4$} in the second column of Table 6 holds, then the corresponding evaluation condition in the fourth column is true.

(I) Relation $R1(X, Y)$, i.e., $\forall x \forall y$, $x \prec y$: Consider any $x \in X$. $\forall y \in Y$, $y \succeq [S(\downarrow y)]_i \succeq x$. Thus, min({$[S(\downarrow y)]_i \mid y \in Y$}) $\succeq x$. We also have min({$[S(\downarrow y)]_i \mid y \in Y$}) = $[S(\cap_{\Downarrow} Y)]_i$. So $[S(\cap_{\Downarrow} Y)]_i \succeq x$, for each $x \in X$. As $\cup_{\Downarrow} X$ is downward-closed with the maximal elements being members of X, and $\cap_{\Downarrow} Y$ is also downward-closed, hence for any node k, $[S(\cup_{\Downarrow} X)]_k \preceq [S(\cap_{\Downarrow} Y)]_k$.

(II) Relation $R2(X, Y)$, i.e., $\forall x \exists y$, $x \prec y$: For any $y \in Y$, $[S(\cup_{\Downarrow} Y)]_i \succeq [S(\downarrow y)]_i$ because of construction of $\cup_{\Downarrow} Y$ (see Definition 10). Also, because $R2$ holds, for each $x \in X$, there is some $y \in Y$ such that $y \succeq [S(\downarrow y)]_i \succeq x$. As $\cup_{\Downarrow} X$ is downward-closed with the maximal elements being members of X, and $\cup_{\Downarrow} Y$ is also downward-closed, hence for any node k, $[S(\cup_{\Downarrow} X)]_k \preceq [S(\cup_{\Downarrow} Y)]_k$.

(III) Relation $R2'(X, Y)$, i.e., $\exists y \forall x, x \prec y$: From the relation, we can infer that $\exists y \in Y$ such that given any $x \in X$, $x \preceq [S(\downarrow y)]_i$. We also have that $\cup_{\Downarrow} X$ is downward-closed with the maximal elements being members of X, and $\downarrow y$ is also downward-closed, hence for any node k, $[S(\cup_{\Downarrow} X)]_k \preceq [S(\downarrow y)]_k$.

(IV) Relation $R3(X, Y)$, i.e., $\exists x \forall y$, $x \prec y$: From the relation, we can infer that $\exists x \in X \forall y \in Y$, $[S(\downarrow y)]_i \succeq x$. Thus, min({$[S(\downarrow y)]_i \mid y \in Y$}) $\succeq x$. We also have min({$[S(\downarrow y)]_i \mid y \in Y$}) = $[S(\cap_{\Downarrow} Y)]_i$. So $[S(\cap_{\Downarrow} Y)]_i \succeq x$. As $\downarrow x$ is downward-closed with x as its maximal element, and $\cap_{\Downarrow} Y$ is also downward-closed, hence for any node k, $[S(\downarrow x)]_k \preceq [S(\cap_{\Downarrow} Y)]_k$.

(V) Relation $R3'(X, Y)$, i.e., $\forall y \exists x$, $x \prec y$: The condition of column 4 is a reexpression of column 2.

(VI) Relation $R4(X, Y)$, i.e., $\exists x \exists y$, $x \prec y$: As noted in (II) above, for any $y \in Y$, $[S(\cup_{\Downarrow} Y)]_i \succeq [S(\downarrow y)]_i$. Given that the relation holds, then $\exists x$ $\exists y$ such that $y \succeq [S(\downarrow y)]_i \succeq x$, therefore, $[S(\cup_{\Downarrow} Y)]_i \succeq [S(\downarrow y)]_i \succeq x$ by combining the above. As $\downarrow x$ is downward-closed with x as its maximal element, and $\cup_{\Downarrow} Y$ is also downward-closed, hence for any node k, $[S(\downarrow x)]_k \preceq [S(\cup_{\Downarrow} Y)]_k$.

In each of the above cases, the evaluation condition in the fourth column follows. (end of Part 1).

Part 2 ($\Longleftarrow$): We show that a relation from {$R1$, $R2$, $R2'$, $R3$, $R3'$, $R4$} given in the second column of Table 6 holds if the corresponding evaluation condition in the fourth column is true.

(I) Relation $R1(X, Y)$, i.e., $\forall x \forall y$, $x \prec y$: For any $x \in X$, $x \in \cup_{\Downarrow} X$ and $x \preceq [S(\cap_{\Downarrow} Y)]_i = z_i$. From Lemma 2.1, it follows that $\forall y \in Y$, $z_i \preceq y$. By transitivity, $\forall y \in Y$, $x \preceq y$.
The above argument is true for all $x \in X$. Hence, $\forall x \in X \forall y \in Y$, $x \preceq y$.

(II) Relation $R2(X, Y)$, i.e., $\forall x \exists y$, $x \prec y$: For any $x \in X$, $x \in \cup_{\Downarrow} X$ and $x \preceq [S(\cup_{\Downarrow} Y)]_i = z_i$. From Lemma 2.2, it follows that $\exists y \in Y$, $z_i \preceq y$. By transitivity, $x \preceq y$.
The above argument is true $\forall x \in X$. Hence, $\forall x \in X \exists y \in Y$, $x \preceq y$.

(III) Relation $R2'(X, Y)$, i.e., $\exists y \forall x$, $x \prec y$: Consider any $y \in Y$ that makes the evaluation condition true. For each $x \in X$, $x \in \cup_{\Downarrow} X$ and $x \preceq [S(\downarrow y)]_i = z_i$. From Lemma 2.1, it follows that $z_i \preceq y$. By transitivity, $x \preceq y$. Hence, $\exists y \in Y \forall x \in X$, $x \preceq y$.

Table 6. The fourth column extends Table 1 by giving evaluation conditions using cuts of form $\downarrow Z$

Relation r	Expression for $r(X,Y)$	Evaluation condition using relation $\ll$ between cuts (see Theorem 4)	Evaluation condition using relation $\subseteq$ between cuts (see Theorem 8)
$R1$	$\forall x \in X \forall y \in Y,\ x \prec y$	$\bigwedge_{x \in X}[\cap_{\Downarrow} Y \not\ll x\uparrow]$	$\cup_{\Downarrow} X \subseteq \cap_{\Downarrow} Y$
$R1'$	$\forall y \in Y \forall x \in X,\ x \prec y$	$= \bigwedge_{y \in Y}[\downarrow y \not\ll \cup_{\Uparrow} X]$	
$R2$	$\forall x \in X \exists y \in Y,\ x \prec y$	$\bigwedge_{x \in X}[\cup_{\Downarrow} Y \not\ll x\uparrow]$	$\cup_{\Downarrow} X \subseteq \cup_{\Downarrow} Y$
$R2'$	$\exists y \in Y \forall x \in X,\ x \prec y$	$\cup_{\Downarrow} Y \not\ll \cup_{\Uparrow} X$	$\bigvee_{y \in Y}[\cup_{\Downarrow} X \subseteq \downarrow y]$
$R3$	$\exists x \in X \forall y \in Y,\ x \prec y$	$\cap_{\Downarrow} Y \not\ll \cap_{\Uparrow} X$	$\bigvee_{x \in X}[\downarrow x \subseteq \cap_{\Downarrow} Y]$
$R3'$	$\forall y \in Y \exists x \in X,\ x \prec y$	$\bigwedge_{y \in Y}[\downarrow y \not\ll \cap_{\Uparrow} X]$	$\bigwedge_{y \in Y} \bigvee_{x \in X}[\downarrow x \subseteq \downarrow y]$
$R4$	$\exists x \in X \exists y \in Y,\ x \prec y$	$\cup_{\Downarrow} Y \not\ll \cap_{\Uparrow} X$	$\bigvee_{x \in X}[\downarrow x \subseteq \cup_{\Downarrow} Y]$
$R4'$	$\exists y \in Y \exists x \in X,\ x \prec y$		

(IV) Relation $R3(X,Y)$, i.e., $\exists x \forall y,\ x \prec y$: Consider any $x \in X$ that makes the evaluation condition true. $x = [S(\downarrow x)]_i \preceq [S(\cap_{\Downarrow} Y)]_i = z_i$. From Lemma 2.1, it follows that $\forall y \in Y, z_i \preceq y$. By transitivity, $\forall y \in Y,\ x \preceq y$. Hence, $\exists x \in X \forall y \in Y,\ x \preceq y$.

(V) Relation $R3'(X,Y)$, i.e., $\forall y \exists x,\ x \prec y$: The condition in column 4 is a reexpression of column 2.

(VI) Relation $R4(X,Y)$, i.e., $\exists x \exists y,\ x \prec y$: Consider any $x \in X$ that makes the evaluation condition true. $x = [S(\downarrow x)]_i \preceq [S(\cup_{\Downarrow} Y)]_i = z_i$. From Lemma 2.2, it follows that $\exists y \in Y,\ z_i \preceq y$. By transitivity, $x \preceq y$. Hence, $\exists x \in X \exists y \in Y,\ x \preceq y$.

From (I)–(VI), each evaluation condition in the fourth column of Table 6 implies the corresponding causality relation between nonatomic poset events in the second column, but with the $\prec$ relation between atomic events replaced by $\preceq$ (see Definition 7). Using the timestamp modification technique used in the proof of Theorem 4, the relations in the second column of Table 6 are true if the corresponding evaluation conditions in the fourth column are true. (end of Part 2). □

Theorem 9 *Each relation $R(X,Y)$ in Table 6 can be evaluated with the following computational complexity: relations $R1$, $R1'$, $R2$, $R3$, $R4$, and $R4'$ can be evaluated in $|N_X|$ integer comparisons, and relations $R2'$ and $R3'$ in $|N_X| \times |N_Y|$ integer comparisons.*

Proof. The proof is along the lines of the proof of Theorem 5, except that it uses Theorems 7 and 8 instead of Theorems 3 and 4. □

There does not appear to be a more efficient way than $|N_X| \times |N_Y|$ integer comparisons to evaluate $R2'$ and $R3'$ on X and Y using cuts of the form $\downarrow Z$ on X and Y. If relations $R2'$ and $R3'$ are not used, the other relations in Table 6 can be evaluated in $|N_X|$ integer comparisons, instead of $\min(|N_X|, |N_Y|)$ integer comparisons needed for $R1$, $R1'$, $R3$, $R4$, and $R4'$ (Theorem 5) using cuts of the form $X\uparrow$ and $\downarrow Y$. However, the overhead of establishing the timestamp structure is half that determined in Sect. 3.6.

4 Concluding remarks

This paper examined a hierarchy of causality relations between nonatomic poset events in distributed computations. The hierarchy of relations is complete using first-order predicate logic and only the relation $\prec$ between atomic events, and extends the hierarchy of Lamport [27,28] and the hierarchy of [21], for nonatomic poset events. The causality relations are useful for distributed applications and agent-based programs that use nonatomicity for event abstraction, but also need a fine level of granularity of causality relations to specify synchronization relations and their composite global predicates. Each application can choose appropriate causality relations from the hierarchy to specify and capture causality and synchronization conditions between its nonatomic poset events. The 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. We also expect that researchers currently working in specifying distributed predicates using linear intervals [13, 38] will leverage the expressive power and convenience offered by the suite of relations on poset events, which are shown to have a low linear-time evaluation cost.

Complex conditions can be expressed as a predicate over the proposed causality relations. Implicit in the use of these relations by distributed applications are the needs (i) to detect whether some specific relation holds between each pair of nonatomic events in a given set of nonatomic events, and (ii) to determine all the relations that are true between each pair of nonatomic events, in a given set of nonatomic events. We derived efficient evaluation conditions for the proposed causality relations between nonatomic poset events X and Y; most relations can be evaluated in $\min(|N_X|, |N_Y|)$ integer comparisons, some in $|N_X|$ integer comparisons, and the others in $|N_Y|$ integer comparisons, where $|N_X|$ and $|N_Y|$, respectively, are the number of nodes on which the two nonatomic events X and Y occur. Thus, the simplified evaluation conditions we derive for the relations have only a linear computational complexity of testing, whereas a naive evaluation of the relations as per their definitions has a polynomial computational complexity ($|N_X| \times |N_Y|$) of testing. The use of the simplified evaluation conditions incurs a one-time cost of setting up the timestamp structure, which we analyzed. We then examined the conditions under which it is profitable to use the proposed method of evaluating the relations over the naive method based on the definitions.

During the derivation of the efficient testing conditions, we also defined special system execution prefixes associated with nonatomic poset events and examined their knowledge-theoretic significance. We also saw how to capture causality

information associated with a nonatomic event, i.e., information about the past and future execution associated with the nonatomic event, in a condensed and aggregated form via the definition of special execution prefixes associated with the nonatomic event. Furthermore, we provided a mechanism to capture such condensed information about causality of a nonatomic event using a timestamp that has the same size as the timestamp of a single atomic event. As distributed applications become more widespread and sophisticated, the proposed theory will be useful to evaluate causality relations between distributed nonatomic events. The results contribute to the fundamental area of causality and atomicity in distributed computing [27, 28, 37] and attempt to provide an answer in the search for the holy grail of causality analysis [37].

Having defined a suite of causality relations, it is interesting to determine all the orthogonal relations that can exist between two nonatomic poset events, analogous to the results for linear intervals in [21]. (A set of orthogonal relations is such that for any two events, (i) the events must be related by one and only one of these relations, and (ii) no relation in this set can be expressed as the disjunction of other relations in this set.) Let $\mathscr{R}^*$ be the set of all conjunctions of relations in $\mathscr{R}$ that can hold for $r(X, Y)$, for $r \in \mathscr{R}$ – each member of $\mathscr{R}^*$ is a conjunction of the members of an antichain of Fig. 2 and can be identified as discussed in Sect. 2. For each $rel1(X, Y)$, where $rel1 \in \mathscr{R}^*$, determine which $rel2(Y, X)$ can hold, where $rel2 \in \mathscr{R}^*$, using the relational algebra [25] which allows the derivation of all $r'(Y, X)$ from any $r(X, Y)$, where r, $r' \in \mathscr{R}$. Then each conjunction of a relation $rel1(X, Y)$ and a compatible relation $rel2(Y, X)$ is orthogonal from every other such conjunction; denote this set of conjunctions as $\mathscr{R}^{**}$, which then represents all the possible orthogonal relations between two posets, using only the $\prec$ relation between atomic events.

A hierarchical framework for defining views of a computation at higher levels of granularity was given in [22], in which the events in any view partitioned the set of events in any finer-level view. Also, the definition of the ordering relation among the events in each view was flexible but useful definitions were seen to capture some notion of causality. The hierarchical framework can be adapted to include the fine-grained causality relations between overlapping nonatomic events and their evaluation conditions.

Acknowledgments. The comments of the three anonymous referees were very useful in improving the presentation of the paper. Referee 2 detected a loophole in the definition of cuts, which resulted in the addition of $E^{\perp}$ and $E^{\top}$ to the model.

References

1. Abraham U, Ben-David S, Magidor M: On global-time and interprocess communication, In: Kwiatkowska M, Shields M, Thomas R, (eds.), *Semantics for Concurrency*, Workshops in Computing, pp 311–323, London: Springer 1990
2. Allen J: Maintaining knowledge about temporal intervals, CACM 26(11): 832–843 (1983)
3. Aigner M: *Combinatorial Theory*, Berlin Heidelberg New York: Springer 1979
4. Anger F: On Lamport's interprocessor communication model, ACM Trans. Prog. Lang. and Syst. 11(3): 404–417 (1989)
5. Anger F, Rodriguez R: The lattice structure of temporal interval relations, Journal of Applied Intelligence 6(1): 29–38 (1996)
6. de Bakker JW, de Roever WP, Rozenberg G (eds.): *Linear time, branching time, and partial orders in logics and models of concurrency*, LNCS 354, Berlin Heidelberg New York: Springer 1989
7. Basten T: Breakpoints and time in distributed computations, In: Tel G, Vitányi P (eds.), *Proc. 8th Int. Workshop on Distributed Algorithms*, LNCS 857, pp 340–354, Berlin Heidelberg New York: Springer 1994
8. Benthem JV: *The Logic of Time*, Dordrecht: Kluwer 1991
9. Boudol G, Castellani I: Concurrency and atomicity, Theoretical Computer Science 59: 25–84 (1988)
10. Chandy KM, Misra J: How processes learn, Distributed Computing 1: 40–52 (1986)
11. Charron-Bost B: Concerning the size of clocks in distributed systems, Information Processing Letters 39: 11–16 (1991)
12. Charron-Bost B, Delporte-Gallet C, Fauconnier H: Local and temporal predicates in distributed systems, ACM Trans. on Prog. Lang. and Sys. 17(1): 157–179 (1995)
13. Cooper R, Marzullo K: Consistent detection of global predicates, *Proc. ACM/ONR Workshop on Parallel and Distributed Debugging*, pp 163–173, May 1991
14. Dobbs HAC: The dimensions of the sensible present, In: *The Study of Time*, pp 274–292, Berlin Heidelberg New York: Springer 1972
15. Fidge CA: Timestamps in message-passing systems that preserve partial ordering, Australian Computer Science Communications 10(1): 56–66 (1988)
16. Fishburn PC: *Interval Orders and Interval Graphs: A Study of Partially Ordered Sets*, New York: Wiley 1985
17. Goltz U, Rensink A: Finite Petri nets as models for recursive causal behavior, Theoretical Computer Science 124(1): 169–179 (1994)
18. Hamblin CL: Instants and intervals, In: *The Study of Time*, pp 324–332, Berlin Heidelberg New York: Springer 1972
19. Janssen W, Poel M, Zwiers J: Action systems and action refinement in the development of parallel systems, In: Baeten JCM, Groote JF, (eds.) *Concur '91*, LNCS 527, pp 298–316, Berlin Heidelberg New York: Springer 1991
20. Kshemkalyani A: Temporal interactions of intervals in distributed systems, *Technical Report TR-29.1933, IBM*, September 1994
21. Kshemkalyani A: Temporal interactions of intervals in distributed systems, Journal of Computer and System Sciences 52(2): 287–298 (1996) (Contains some parts of [20]).
22. Kshemkalyani A: Framework for viewing atomic events in distributed computations, Theoretical Computer Science 196(1–2): 45–70 (1998) (Abstract appears in *Proc. of Euro-Par'96*, In: Bougè L, Fraigniaud P, Mignotte A, Roberts Y (eds.), LNCS 1123, pp 496–505, Berlin Heidelberg New York: Springer 1996.)
23. Kshemkalyani A: Relative timing constraints between complex events, *Proc. 8th IASTED Conf. on Parallel and Distributed Computing and Systems*, pp 324–326, October 1996
24. Kshemkalyani A: Synchronization for distributed real-time applications, *Proc. 5th Workshop on Parallel and Distributed Real-time Systems*, IEEE Computer Society Press, pp 81–90, April 1997
25. Kshemkalyani A: Causality between nonatomic poset events in distributed computations, *Proc. 5th IEEE Workshop on Future Trends in Distributed Computing Systems*, pp 276–282, 1997
26. Lamport L: Time, clocks, and the ordering of events in a distributed system, CACM 21(7): 558–565 (1978)
27. Lamport L: On interprocess communication, Part I: Basic formalism, Part II: Algorithms, Distributed Computing 1: 77–101 (1986)
28. Lamport L: The mutual exclusion problem, Part I: A theory of interprocess communication, Part II: Statements and solutions, Journal of the ACM 33(2): 313–348 (1986)
29. Mattern F: Virtual time and global states of distributed systems, *Parallel and Distributed Algorithms*, pp 215–226, Amsterdam: North-Holland 1989
30. Mattern F: On the relativistic structure of logical time in distributed systems, In: Datation et Controle des Executions Reparties, Bigre 78: 3–20 (1992)
31. Minkowski H: Space and time, In: *The Principle of Relativity*, pp 73–81, Dover New York: 1905
32. Olderog ER: *Nets, Terms, and Formulas*, Cambridge Tracts in Theo-

retical Computer Science, 1991
33. Reisig W: Temporal logic and causality for concurrent systems, *Concurrency '88*, LNCS 335, pp 121–139, Berlin Heidelberg New York: Springer 1992
34. Rensink A: *Models and Methods for Action Refinement*, Ph.D. thesis, University of Twente, The Netherlands, 1993
35. Rodriguez R, Anger F, Ford K: Temporal reasoning: A relativistic model, Int. Journal of Intelligent Systems 6: 237–254 (1991)
36. Russell B: *Our Knowledge of the External World*, London: Allen & Unwin 1926
37. Schwarz R, Mattern F: Detecting causal relationships in distributed computations: In search of the holy grail, Distributed Computing 7(3): 149–174 (1994)
38. Spezialetti M, Kearns P: A framework for the consistent monitoring of distributed computations, *Proc. IEEE International Conf. on Distributed Computing Systems*, pp 61–68 (1989)
39. Wiener N: A contribution to the theory of relative position, Proc. Cambridge Philosophical Society 17: 441–449 (1914)

Appendix A: Meaning and use of relations in $\mathscr{R}$

Distributed applications and agent-based programs that model nonatomic poset events, e.g., applications for distributed multimedia, distributed debugging, coordination in mobile systems, industrial process control, navigation, planning, and virtual reality will find use for the proposed relations $\mathscr{R}$. And as applications get more sophisticated, they will increasingly use the proposed relations to express and enforce fine-grained causality relations between nonatomic poset events. In the following discussion, the "X computation" and "Y computation" refer to the computation performed by the nonatomic events X and Y, respectively.

We first consider the significance of the groups of relations $R^*a(X,Y)$, $R^*b(X,Y)$, $R^*b'(X,Y)$, $R^*c(X,Y)$, $R^*c'(X,Y)$, and $R^*d(X,Y)$. Each group deals with a particular proxy $\hat{X}$ and $\hat{Y}$.

- $R^*a(X,Y)$: All events in $\hat{Y}$ know the results of the X computation (if any,) upto all the events in $\hat{X}$. This is a strong form of synchronization between $\hat{X}$ and $\hat{Y}$.
- $R^*b(X,Y)$: For each event in $\hat{X}$, some event in $\hat{Y}$ knows the results of the X computation (if any,) upto that event in $\hat{X}$. The Y computation may then exchange information about the X computation upto $\hat{X}$, among the nodes participating in the Y computation.
- $R^*b'(X,Y)$: Some event in $\hat{Y}$ knows the results of the X computation (if any,) upto all events in $\hat{X}$. These relations are useful when it is sufficient for one node in $N_{\hat{Y}}$ to detect a global predicate across all nodes in $N_{\hat{X}}$. If the event in $\hat{Y}$ is at a node that behaves as the group leader of $N_{\hat{Y}}$, then it can either inform the other nodes in $N_{\hat{Y}}$ or make decisions on their behalf.
- $R^*c(X,Y)$: All events in $\hat{Y}$ know the results of the X computation (if any,) upto some common event in $\hat{X}$. This group of relations is useful when it is sufficient for one node in $N_{\hat{X}}$ to inform all the nodes in $N_{\hat{Y}}$ of its state, such as when all the nodes in $N_{\hat{X}}$ have a similar state. If the node at which the event in $\hat{X}$ occurs has already collected information about the results/ states of the X computation upto $\hat{X}$ from other nodes in $N_{\hat{X}}$ (thus, that node behaves as the group leader of $\hat{X}$), then the events in $\hat{Y}$ will know the states of the X computation upto $\hat{X}$.
- $R^*c'(X,Y)$: Each event in $\hat{Y}$ knows the results of the X computation (if any,) upto some event in $\hat{X}$. If it is important to the application, then the state at each event in $\hat{X}$ should be communicated to some event in $\hat{Y}$.
- $R^*d(X,Y)$: Some event in $\hat{Y}$ knows the results of the X computation (if any,) upto some event in $\hat{X}$. The nodes under consideration at which the events in $\hat{Y}$ and $\hat{X}$, respectively, occur may be the group leaders of $N_{\hat{Y}}$ and $N_{\hat{X}}$, respectively. This group leader of $N_{\hat{X}}$ may have collected relevant state information from other nodes in $N_{\hat{X}}$, and conveys this information to the group leader of $N_{\hat{Y}}$, which in turn distributes the information to all nodes in $N_{\hat{Y}}$.

The above significance of each group of relations applies to each individual relation of that group. The specific use and meaning of each of the 24 relations in $\mathscr{R}$ is given next. We do not restrict the explanation that follows to any specific application.

$R1^*(X,Y)$: This group of relations deals with U_X and L_Y. Each relation signifies a different degree of transfer of control for synchronization, as in group mutual exclusion (*gmutex*), from the X computation to the Y computation.

- $R1a(X,Y)$: The Y computation at any node in N_Y begins only after that node knows that the X computation at each node in N_X has ended, e.g., a conventional distributed gmutex in which each node in N_Y waits for an indication from each node in N_X that it has relinquished control.
- $R1b(X,Y)$: For every node in N_X, the final value of its X computation is known by (or its mutex token is transferred to) some node in N_Y before that node in N_Y begins its Y computation. Thus, nodes in N_Y collectively (but not individually) know the final value of the X computation by the time the last among them begins its Y computation. This is a weak version of synchronization/gmutex.
- $R1b'(X,Y)$: Before beginning its Y computation, some node in N_Y knows the final value of the X computation at each node in N_X. This is a weak version of synchronization/gmutex (but stronger than $R1b$) with the property that at least one node in N_Y cannot begin its Y computation until the final value of the X computation at each node in N_X is known to it.
- $R1c(X,Y)$: The final value of the X computation at some node in N_X is known to all the nodes in N_Y before they begin their Y computation. This is a weak form of synchronization/gmutex which is useful when it suffices for a particular node in N_X to grant all the nodes in N_Y gmutex permission to proceed with the Y computation; this node in N_X may be the group leader of N_X, or simply all the nodes in N_X have the same final local state of the X computation within this application.
- $R1c'(X,Y)$: Each node in N_Y begins its Y computation only after it knows the final value of the X computation of some node in N_X. This is a weak form of synchronization/gmutex (weaker than $R1c$) which requires each node in N_Y to receive a final value (or gmutex token) from at least one node in N_X before starting its Y computation. This relation is sufficient for some applications

such as those requiring that at most one (additional) process be admitted to join those in the critical section when one process leaves it.
- $R1d(X, Y)$: Some node in N_Y begins its Y computation only after it knows the final value of (or receives a gmutex token from) the X computation at some node in N_X. This is the weakest form of synchronization/gmutex.

$R2^*(X, Y)$: This group of relations deals with U_X and U_Y. The relations can signify various degrees of synchronization between the termination of computations X and Y, where X is nested within Y or X is a subcomputation of Y. Alternately, Y could denote activity at processes that have already spawned X activity in threads, and Y can complete only after X completes.

- $R2a(X, Y)$: The Y computation at any node in N_Y can terminate only after that node knows the final value of (or the termination of) the X computation at each node in N_X. This is a strong synchronization before termination, between X and Y.
- $R2b(X, Y)$: For every node in N_X, the final value of its X computation is known by at least one node in N_Y before that node in N_Y terminates its Y computation. Thus, all the nodes in N_Y collectively (but not individually) know the final values of the X computation before they terminate their Y computation. This is a weak synchronization before termination.
- $R2b'(X, Y)$: Before terminating its Y computation, some node in N_Y knows the final value of the X computation at all nodes in N_X. This is a stronger synchronization before termination than $R2b$ wherein at least one node in N_Y cannot terminate its Y computation without knowing the final state of the X computation at all nodes in N_X. This suffices for all applications in which it is adequate for one node in N_Y to detect the termination of the X computation at each node in N_X before that node terminates its Y computation.
- $R2c(X, Y)$: The final value of the X computation at some node in N_X is known to all the N_Y nodes before they terminate the Y computation. This is a weak form of synchronization. The pertinent node in N_X could represent a critical thread in the X computation, or could be the group leader of N_X that represents the X computation at all nodes in N_X.
- $R2c'(X, Y)$: Each node in N_Y terminates its Y computation only after it knows the final value of the X computation at some node in N_X. This is a weaker form of synchronization before termination than $R2c$, but is adequate when all the nodes in N_X are performing a similar X computation.
- $R2d(X, Y)$: Some node in N_Y terminates its Y computation after it knows the final value of the X computation at some node in N_X. This is a weak form of synchronization; however, if the concerned nodes in N_X and N_Y are the respective group leaders of the X and Y computations and, respectively, collect/distributed information from/to their groups, then a strong form of synchronization can be implicitly enforced because when Y terminates, it is known to each node in N_Y that the X computation has terminated.

$R3^*(X, Y)$: This group of relations deals with L_X and L_Y. The relations can signify various degrees of synchronization between the initiation of computations X and Y, where Y is nested within X or Y is a subcomputation of X. Alternately, X could denote activity at processes that have already spawned Y activity in threads.

- $R3a(X, Y)$: The Y computation at any node in N_Y begins after that node knows the initial values of the X computation at each node in N_X. This is a strong form of synchronization between the beginnings of the X and Y computations.
- $R3b(X, Y)$: For each node in N_X, the initial state of its X computation is known to some node in N_Y before that node in N_Y begins its Y computation. Thus, all the nodes in N_Y collectively (but not individually) know the initial state of the X computation. This synchronization is sufficient when the forked Y computations at each node in N_Y are only loosely coupled and should not know each others' initial states communicated by the X computation; while at the same time ensuring that the initial state of the X computation at each node in N_X is available to at least one node in N_Y before it commences its Y computation.
- $R3b'(X, Y)$: Before beginning its Y computation, some node in N_Y knows the initial state of the X computation at all the nodes in N_X. Thus the Y computation at this node can run a parallel X computation for fault-tolerance, or can be made an entirely deterministic function of the inputs to the X computation. This node in N_Y can coordinate the Y computation of the other nodes in N_Y. This synchronization is weaker than $R3a$ but stronger than $R3b$.
- $R3c(X, Y)$: The initial state of the X computation at some node in N_X is known to all the nodes in N_Y before they begin their Y computation. This is a weak synchronization; however, it is adequate when the subject node in N_X has forked all the threads that will perform Y, and behaves as the group leader of X that initiates the nested computation Y.
- $R3c'(X, Y)$: Each node in N_Y begins its Y computation only after it knows the initial state of the X computation at some node in N_X. Thus each node executing the computation Y has its Y computation forked or spawned by some node in N_X and its Y computation corresponds to a nested branch of X. The nodes in N_Y may not know each others' initial values for the Y computation; the X computations at (some of) the N_X nodes have semi-independently forked the Y computations at the nodes in N_Y. This form of synchronization is weaker than $R3c$.
- $R3d(X, Y)$: Some node in N_Y begins its Y computation only after it knows the initial state of the X computation at some node in N_X. This is a weak form of synchronization in which only one node in N_X and one node in N_Y coordinate their respective initial states of their local X and Y computations. However, if the node in N_X that initiated the X computation forks off the main thread for the Y computation, then this form of synchronization between the initiations of X and Y is adequate to have Y as an entirely nested computation within X.

$R4^*(X,Y)$: This group of relations deals with L_X and U_Y. The relations signify different degrees of synchronization between a monitoring computation Y that knows the initial values with which the X computation begins, and then the monitoring computation Y terminates.

- $R4a(X,Y)$: The Y computation at any node in N_Y terminates only after that node knows the initial values of the X computation at each node in N_X. This is a strong form of synchronization between the start of X and the end of Y.
- $R4b(X,Y)$: For every node in N_X, the initial state of its X computation is known by at least one node in N_Y before that node in N_Y terminates its Y computation. Even if there is no exchange of information in the Y computation about the state of the X computation at individual nodes in N_X, this relation guarantees that when Y completes, the (initial) local states at each of the N_X nodes are collectively (but not individually) known by N_Y.
- $R4b'(X,Y)$: Before terminating its Y computation, some node in N_Y knows the initial state of the X computation at all the nodes in N_X. This node in N_Y can detect if an initial global predicate of the X computation across the nodes in N_X is satisfied, before it terminates its Y computation. If this node in N_Y behaves as a group leader, it can then inform all the other nodes in N_Y to terminate their Y computations.
- $R4c(X,Y)$: The initial state of the X computation at some node in N_X is known to all the nodes in N_Y before they terminate their Y computation. This weak synchronization is adequate for applications where all the N_X nodes start their X computation with similar values. Alternately, if the node in N_X behaves as a group leader, it can first detect the initial global state of the X computation and then inform all the nodes in N_Y.
- $R4c'(X,Y)$: Each node in N_Y terminates its Y computation only after it knows the initial state of the X computation at some node in N_X. This is a weaker form of synchronization than $R4c$ because the states of all nodes in N_X may not be observed before the nodes in N_Y terminate their Y computation. But this will be adequate for applications in which each node in N_X is reporting the same state/value of the X computation, and each node in N_Y simply needs a confirmation from some node in N_X before it terminates its Y computation. For example, a mobile host (an N_Y node) may simply need a confirmation from some base station (an N_X node) before it exits its Y computation.
- $R4d(X,Y)$: Some node in N_Y terminates its Y computation after it knows the initial state of the X computation at some node in N_X. This weak form of synchronization is sufficient when the group leader of X which is responsible for kicking off the rest of X informs some node (or the group leader) of the monitoring distributed program Y that computation X has successfully begun.

Appendix B: Proofs of Lemmas 6 and 7

Lemma 6 ($\exists z \in S(\downarrow Y)$, $z \in S(X\uparrow) \bigvee z \notin X\uparrow$) *iff* ($\exists z'_i \in S(\downarrow Y)$, $i \in N_X \bigwedge [S(X\uparrow)]_i \preceq z'_i$)

Proof. ($\Longrightarrow$): The L.H.S. is an expression over the events in $S(X\uparrow)$ and $S(\downarrow Y)$ that occur at the node at which event z occurs. Let $z = e_j$ and $[S(X\uparrow)]_j = e'_j$. Then from the L.H.S., it follows that $e'_j \preceq e_j = z$. (1)

From Lemma 5.1, we can infer that $\exists i \in N_X$ such that $x_i = [S(X\uparrow)]_i \bigwedge x_i \preceq e'_j$. (2)

As the relation $\preceq$ is transitive, from (1) and (2), we infer that $z \succeq e'_j \succeq x_i$. As $\downarrow z$ has a unique maximal element z and there is a causality path from x_i to z, we have $x_i \preceq [S(\downarrow z)]_i$. (3)

As $z \in S(\downarrow Y)$, and $\downarrow z$ and $\downarrow Y$ are both downward-closed subsets of E, we have $\downarrow z \subseteq \downarrow Y$; hence, $[S(\downarrow z)]_i \preceq [S(\downarrow Y)]_i$. (4)

Let $z'_i = [S(\downarrow Y)]_i$. Then $z'_i = [S(\downarrow Y)]_i \succeq [S(\downarrow z)]_i \succeq x_i = [S(X\uparrow)]_i$ by combining (3) and (4). The RHS follows.

($\Longleftarrow$): z' that exists as per the R.H.S. of the lemma satisfies the condition on z in the L.H.S. of the lemma. The proof follows. □

Lemma 7 ($\exists z \in S(X\uparrow)$, $z \in \downarrow Y$) *iff* ($\exists z'_i \in S(X\uparrow)$, $i \in N_Y \bigwedge z'_i \preceq [S(\downarrow Y)]_i$)

Proof. ($\Longrightarrow$): The L.H.S. is an expression over the events in $S(X\uparrow)$ and $\downarrow Y$ that occur at the node at which event z occurs. Let $z = e_j$ and $[S(\downarrow Y)]_j = e'_j$. Then from the L.H.S., it follows that $z = e_j \preceq e'_j$. (1)

From Lemma 5.2, we can infer that $\exists\ i \in N_Y$ such that $y_i = [S(\downarrow Y)]_i \bigwedge y_i \succeq e'_j$. Using (1), we have $y_i \succeq z$. (2)

Observe that z is either $[S(C3(X))]_j$ or $[S(C4(X))]_j$.

- If $X\uparrow = C3(X)$, we can infer from the definition of $C3(X)$ that z is the earliest event at node j among all $[S(x\uparrow)]_j$, $\forall x \in X$, and therefore $\exists x \in X$, $x \preceq z$. Combining this with (2), we have $\exists x \in X$, $x \preceq z \preceq y_i$, and hence $[S(C3(X))]_i \preceq y_i$.
- If $X\uparrow = C4(X)$, we can infer from the definition of $C4(X)$ that z is the latest event at node j among all $[S(x\uparrow)]_j$, $\forall x \in X$, and therefore $\forall x \in X$, $x \preceq z$. Combining this with (2), we have $\forall x \in X$, $x \preceq z \preceq y_i$, and hence $[S(C4(X))]_i \preceq y_i$.

In either case, we have $[S(X\uparrow)]_i \preceq y_i = [S(\downarrow Y)]_i$. The R.H.S. follows.

($\Longleftarrow$): Event z' that exists as per the R.H.S. of the lemma satisfies the condition on z in the L.H.S. of the lemma. The proof follows. □

Appendix C: Discussion on the evaluation methodology

Issues concerning the evaluation of causality relations between nonatomic poset events are discussed next. Recall that a central monitoring process collects the trace of the distributed execution. The central process need not perform an exhaustive trace because it is prohibitively expensive. Rather, the trace can focus on events that are "relevant" or "potentially relevant" to the application. The central process has to identify nonatomic poset events – this is done using the $\prec$ relation between atomic events using application-specific and domain-specific information. Once the trace of the distributed execution has been collected, an off-line analysis as described in Sect. 3.6.1 is always feasible. An on-line

analysis of the computation as the trace is being collected by the central process is also achievable as follows. The timestamps of individual events and cuts $C1(X)$ and $C2(X)$ are computable/available as soon as X is identified and are independent of the future computation. Timestamps of $C3(X)$ and $C4(X)$ depend on the reverse timestamps of individual events and depend on the future computation. These can be determined once the nondummy events in each $S(x\uparrow)$ have occurred. The central process periodically considers traces of the computation collected thus far and assuming that this is the complete computation, uses reverse timestamps to compute timestamps of $C3(X)$ and $C4(X)$ to compute/evaluate causality relations among nonatomic events contained entirely within this trace. Only for those events X whose timestamps for $C3(X)$ and $C4(X)$ cannot be computed because the nondummy events in each $S(x\uparrow)$ have not yet occurred, the following technique can be used. The reverse timestamps of events and timestamps of cuts $C3$ and $C4$ are incrementally updated the next time the central process runs a trace analysis. Let Q^r and Q^s be the traces at time r and s, respectively, at the central process. Let $r < s$. Then Q^r is a prefix of Q^s. Let $T^{R\_r}(X)$ be a reverse timestamp computed at time r. $T^{R\_s}(X)$, the reverse timestamp of X at time s is computed as follows. If $T^{R\_r}(X)[i] \neq \top_i$, then $T^{R\_s}(X)[i] = T^{R\_r}(X)[i] + (Q^s[i] - Q^r[i])$, otherwise $T^{R\_s}(X)[i]$ has to be explicitly computed.

Ajay Kshemkalyani is an Assistant Professor of Electrical Engineering and Computer Science at the University of Illinois at Chicago since September 1998. From 1991 to 1997, he worked at IBM Research Triangle Park in computer networks and distributed systems and subsequently was an Assistant Professor at the University of Cincinnati from 1997 to 1998. He received a Ph.D. in Computer and Information Science from The Ohio State University in 1991, and a B.Tech. in Computer Science and Engineering from the Indian Institute of Technology, Mumbai, in 1987. His current research interests are in distributed computing, computer networking, and operating systems. He is a member of the ACM and a senior member of the IEEE.