JOURNAL OF COMPUTER AND SYSTEM SCIENCES **52**, 287–298 (1996)
ARTICLE NO. 0022

AJAY D. KSHEMKALYANI*

Networking Systems Architecture, IBM Corporation, P.O. Box 12195, Research Triangle Park, North Carolina 27709

Received October 19, 1994; revised April 7, 1995

In a space–time view of a distributed system, each point in space has its own time axis. This paper studies the interaction of intervals at different spatial points in a distributed system. We formalize the notion of what it means for one such interval to affect another such interval. Thus, this paper contributes to the quintessential field of the study of time. The results extend the 1972 work by Hamblin that demonstrated the interaction of time intervals which shared the same linear global time axis. The results also shed light on the nature of interprocess communication in distributed systems, an area which was pioneered by Lamport in 1986. The diverse suite of temporal relations between intervals included in our results provides much greater flexibility than do Lamport's relations to model interprocess interaction in a distributed system. The results are also useful in specifying global predicates and distributed synchronization conditions. © 1996 Academic Press, Inc.

CONTENTS

1. INTRODUCTION

This paper presents a theory of the interactions of intervals in distributed systems without assuming the existence of global time. This is a fundamental problem whose understanding can simplify the design and analysis of any system that involves concurrent actions. We show separate results for the interactions between dense linear intervals and between nondense linear intervals that occur at different points in space. Although time is dense and linear at a point in space [2], in practice, clocks use nondense scales to measure time. The interactions between nondense linear intervals are also useful because actions in a distributed system are modeled as sequences of discrete events on each node. Hence, both results are useful and contribute to the quintessential study of time and intervals [1, 2]. This is the first contribution of the results.

In an early seminal work that advocated a theory of intervals over a theory of instants as a means to model time, Hamblin studied the relations between intervals and gave a suite of ten axioms in first-order predicate calculus to describe the logic of intervals [7]. A temporal interval is a time duration, which implicitly identifies its start and finish instants. Hamblin identified 13 distinct ways in which two temporal intervals may be related to each other in terms of overlapping or concurrent existence. Specifically, two intervals may be related by the following relations: *before*, *meets*, *overlaps during*$^{-1}$, *starts*, *finishes*$^{-1}$, *equals*. The first six relations above have their corresponding inverses. For example, for intervals X and Y, if X *before* Y, then Y *before*$^{-1}$ X. The seventh relation above is its own inverse. Van Benthem proved in Theorem I.3.1.4 [2] that these 13 possibilities cover all the ways by which two intervals may be related.

Hamblin's work [7] on intervals assumed a universal or global time axis that was instantaneously accessible. The two intervals between which the interaction was studied occurred at a single point in space rather than at different points in Euclidian three-dimensional space. This was reflected by Axiom 2 in [7]. The real world is distributed and the absence of a common global clock as well as the impossibility of perfectly synchronizing local clocks invalidates the existence of a global time axis. This work extends Hamblin's work by giving the interactions between intervals that occur at two different points in space and that do not share a common time axis. This is the second contribution of our results. While Hamblin's result is applicable to a study of interactions of intervals in a uniprocessor system, it is inadequate for modeling the interactions of intervals in distributed systems.

The partial order of events in the universe viewed as Minkowski's four-dimensional space–time [15] has been studied by many logicians, physicists, mathematicians, and philosophers throughout this century, e.g., [1, 2, 4, 5]. The causality or "happens before" relationship between points in space–time was given in [4]. Lamport demonstrated

* E-mail: ajayk@vnet.ibm.com.

0022-0000/96 $18.00

the utility of the causality relation in reasoning about distributed systems [9]. Thus, until the mid-eighties, only the interaction between two points in a distributed system was studied. There were three possibilities for the interaction between two points x and y that were specified using the causality relation $\longrightarrow$: (i) $(x \longrightarrow y) \wedge (y \not\longrightarrow x)$, (ii) $(y \longrightarrow x) \wedge (x \not\longrightarrow y)$, and (iii) $(y \not\longrightarrow x) \wedge (x \not\longrightarrow y)$.

In a landmark paper on interprocess communication, Lamport argued that "it is useful to assume that primitive elements between which concurrency is modeled are non-atomic" [10]. Lamport defined system executions using two relations $\longrightarrow$ and $\dashrightarrow$ between these primitive non-atomic elements ("operation executions") and axioms satisfied by $\longrightarrow$ and $\dashrightarrow$ [10, 12]. Lamport further says "the relations $\longrightarrow$ and $\dashrightarrow$ capture the essential temporal properties of system executions, ..., and A1–A5 (the corresponding axioms) provide the necessary tools for reasoning about these relations" [10, p. 80]. We show that the two relations defined by Lamport are not sufficient to capture the essential temporal properties of system executions in distributed systems. The set of relations we propose can capture the essential temporal properties, model a wide range of interactions, and specify synchronization conditions between various loci of control in distributed systems and in parallel systems. Thus, the third contribution of this work is that it models concurrency more clearly than before.

We proceed by specifying and examining three families of system executions for a semantic model that represents the space–time view and its causality relation. For each family of system executions, we define relations, based on the causality relation, that specify how one interval may be related to another. We give the significance of and an evaluation condition for each relation. We then derive a set of axioms for these relations. For each family of system executions, we enumerate and define all valid combinations of values of the defined relations, to give derived meta-level relations called *interaction types*. Interaction types are orthogonal to the other. We graphically depict each of the derived orthogonal interaction types.

The first family of system executions deals with the interaction of a linear interval and a point interval. The second family deals with the interaction of a pair of dense linear intervals. This family extends Hamblin's result to a distributed system. This family permits 29 interaction types between a pair of intervals. Of these, there are 13 pairs of inverses, while three are inverses of themselves. The third family deals with nondense linear intervals. This family permits eleven interaction types between a pair of intervals in addition to those permitted by the second family. Of these, there are five pairs of inverses, while one is its own inverse. The family of system executions that deals with the interaction between a pair of poset intervals is given in [8].

The paper then identifies two applications of the results. One application is the specification of global predicates [6] which are useful in debugging, industrial process control, and detecting specific states in the distributed system. Another application is the modeling of synchronization conditions between two intervals for distributed applications such as multimedia [13, 16].

The results of this paper are included in [8]. This paper is organized as follows. Section 2 presents the framework and system model. Section 3 presents the results by specifying and examining some families of system executions. Section 4 outlines some applications and examines why Lamport's framework to study interprocess communication is inadequate for distributed systems. Section 5 gives the conclusions.

2. FRAMEWORK AND SYSTEM MODEL

The framework is similar to the one used by Lamport in [10]. Consider a poset $(E, <)$, where $<$ is an irreflexive partial ordering. Let $\mathscr{E}$ denote the power set of E. Let $\mathscr{A}(\neq \emptyset) \subseteq (\mathscr{E} - \emptyset)$. Thus, there is an implicit one–many mapping from $\mathscr{A}$ to E. Let this mapping be denoted by μ. Each element A of $\mathscr{A}$ is a nonempty subset of E, and is termed an *interval*. $(E, <)$ represents space–time coordinates related by the causality relation of special relativity. Each point in $(E, <)$ represents the most primitive atomic entity in space–time. Each set $A \in \mathscr{A}$ is a higher level grouping of these coordinates that is of interest to the particular application.

Lamport defined system executions $\langle \mathscr{A}, \longrightarrow, \dashrightarrow \rangle$ and provided a set of axioms on the relations $\longrightarrow$ and $\dashrightarrow$ [10]. To aid in the understanding of a system execution, he provided an independent semantic model which he defined as a triple $E, <, \mu$. $<$ was an irreflexive, partial ordering on set E and μ mapped elements of $\mathscr{A}$ to subsets of E such that for $X, Y \in \mathscr{A}$,

$$\longrightarrow \quad \equiv \quad X \longrightarrow Y \qquad \text{iff } \forall x \in \mu(X)\ \forall y \in \mu(Y), x < y$$
$$\dashrightarrow \quad \equiv \quad X \dashrightarrow Y \qquad \text{iff } \exists \in \mu(X)\ \exists y \in \mu(Y), x < y.$$

A premise of this paper is that the above two relations are not sufficient to capture all possible interactions between two intervals. Additional relations will be defined in the subsequent sections, depending on the assumptions made about elements of $\mathscr{A}$. Specifically, like $\longrightarrow$ and $\dashrightarrow$,[1] these relations are defined over $\mathscr{A} \times \mathscr{A}$ and map to {*true*, *false*} using first-order predicate logic.

The framework and system model differ from those of Lamport [10] in two ways: First, we define a "family of system executions" instead of system executions. Second, the definition of a family of system executions is formulated

[1] We will rename relations $\longrightarrow$ and $\dashrightarrow$ as R1 and R4, respectively, in Section 3.2 to use consistent terminology with the new relations we introduce in that section.

using the semantic model provided by the poset $(E, <)$ and the implicit mapping μ.

DEFINITION 1. A family of system executions is a quadruple $\mathscr{F} = \langle \mathscr{A}, \Psi, \mathscr{R}, \mathscr{X} \rangle$, where $\mathscr{A}$ is a set of intervals, Ψ is an expression specifying constraints satisfied by each element in $\mathscr{A}$, and $\mathscr{R}$ is a set of precedence relations on $\mathscr{A}$, determined by Ψ and satisfying a set of axioms given in $\mathscr{X}$.

DEFINITION 2. The semantic model of any family of system executions $\langle \mathscr{A}, \Psi, \mathscr{R}, \mathscr{X} \rangle$ is the poset $(E, <)$ and a one-many mapping μ: $\mathscr{A}$ to E such that

1. μ correlates the semantics of Ψ specified on $\mathscr{A}$ to the semantics of $\mu(A)$ in $(E, <)$, where $A \in \mathscr{A}$.

2. μ correlates the semantics of $r(A, A')$, where $A, A' \in \mathscr{A}$, $r \in \mathscr{R}$, to the semantics of $<(\mu(A) \times \mu(A'))$.

Each family of system executions is specified in the following steps, each of which uses the semantic model as a reasoning tool: (i) specify Ψ on $\mathscr{A}$; (ii) determine an appropriate $\mathscr{R}$ so that there is sufficient expressive power to reason; and (iii) formulate $\mathscr{X}$ for $\mathscr{R}$. Then define *interaction types* which are orthogonal meta-level relations based on $\mathscr{R}$, using $\mathscr{R}$ and $\mathscr{X}$, to capture all interactions between intervals specified by Ψ on $\mathscr{A}$. As a trivial example of the above method, we describe the well-understood interaction between points in space–time using the family of system executions $\mathscr{F}_{ptpt}$. Let $\mathscr{F}_{ptpt} = \langle \mathscr{A}_{ptpt}, \Psi_{ptpt}, \mathscr{R}_{ptpt}, \mathscr{X}_{ptpt} \rangle$, where (i) $\mathscr{A}_{ptpt}$ is a set of intervals, (ii) $\Psi_{ptpt}: : \forall A \in \mathscr{A}_{ptpt}, |A| = 1$, (iii) $\mathscr{R}_{ptpt} = \{\longrightarrow\}$ and (iv) $\mathscr{X}_{ptpt}$ is the set of axioms derivable from the fact that $\longrightarrow$ is an irreflexive partial ordering on $\mathscr{A}_{ptpt}$. In the semantic model, μ is a 1–1 mapping and there are three orthogonal interaction types. For $X, Y \in \mathscr{A}_{ptpt}$, (a) $(X \longrightarrow Y) \wedge (Y \not\longrightarrow X)$; (b) $(Y \longrightarrow X) \wedge (X \not\longrightarrow Y)$; and (c) $(Y \not\longrightarrow X) \wedge (X \not\longrightarrow Y)$. (a) and (b) are inverses of each other, whereas (c) is its own inverse.

In a distributed system or in a space–time view of the universe, elements of E are partitioned into local computations at a point in space (in a space–time view) or at a process (in a view of processes at discrete nodes). Each local computation is a linearly ordered set of events. Let P be the set of all partitions. An event e in partition i is denoted e_i.

A cut $C \subseteq E$ such that if $e_i \in C$ then $\forall e'_i$: $e'_i < e_i$: : $e'_i \in C$. Thus, a cut is downward closed within partitions. A cut that preserves causality is a consistent cut and denotes a computation [3, 14]. Only all downward-closed subsets of E preserve causality.

DEFINITION 3. A consistent cut is a downward-closed subset of E in $(E, <)$.

For event e, there are two special consistent cuts $\downarrow e$ and $e\uparrow$.

DEFINITION 4. $\downarrow e = \{e' \mid e' \leqslant e\}$.

DEFINITION 5.

$$e\uparrow = \{e' \mid e' \ngeqslant e\} \cup \{e_i, i = 1, ..., |P| \mid e_i \geqslant e \wedge (\forall e'_i < e_i, e'_i \ngeqslant e)\}.$$

$\downarrow e$ is the maximal set of events that happen before or equal e. $e\uparrow$ is the set of all events up to and including the earliest events on each process or point in space, for which e happens before or equals the events.

Given a cut C and set $X \subseteq E$, let C_X be the maximal subset of C that contains elements in those partitions that also have elements in X. Thus, C is projected over the partitions that have elements in X.

Finally, some words about the notation used. Given a boolean variable x, $\bar{x}$ denotes the complement of x. The mapping μ will be kept implicit henceforth to simplify the notation.

3. THEORY OF LINEAR INTERVALS

AXIOM 1. *An internal A is linear iff $\forall x, y \in A$, $x \leqslant y \vee y \leqslant x$.*

In this section, we assume all intervals are linear. Linear intervals adequately model read and write operations at a memory location in a shared-memory multiprocessor system. Linear intervals also naturally model a sequence of events at any process as well as the progression of time at a point in space. We define and examine two families of system executions $\mathscr{F}_{LL}$ and $\mathscr{F}_{L'L'}$ which deal with dense linear and nondense linear intervals, respectively. As an intermediate step to examining $\mathscr{F}_{LL}$ and $\mathscr{F}_{L'L'}$, we consider the interaction between a point interval and a linear interval by defining a family of system executions $\mathscr{F}_{Lpt}$.

3.1. Interaction between a Linear Interval and a Point Interval

Consider the interaction of two elements of $\mathscr{A}$ where one element, say Y, contains only one element of E, and the other element of $\mathscr{A}$, say X, contains one or more elements of E. Also assume without loss of generality that the intervals X and Y are disjoint. The interaction between X and Y is specified by assigning boolean values to $r(X, Y)$ and to $r(Y, X)$, where r is instantiated by $\longrightarrow$ and $\dashrightarrow$ [10, 12]. The axioms on these two relations are A1–A4 given in [10, 12] and discussed in Section 4.1 of this paper.[2] An additional axiom, viz, A6:$A \longrightarrow B \Rightarrow B \not\dashrightarrow A$, for intervals A and B, holds.

[2] As mentioned in Section 2, we will rename relations $\longrightarrow$ and $\dashrightarrow$ as R1 and R4, respectively, in Section 3.2 to use consistent terminology with three new relations we introduce in that section.

TABLE 1

Possible Interactions between a Linear Interval X and a Point Interval Y

Interaction type	Relation $r(X, Y)$: $\longrightarrow(X, Y)$	Relation $r(X, Y)$: $\dashrightarrow(X, Y)$	Relation $r(Y, X)$: $\longrightarrow(Y, X)$	Relation $r(Y, X)$: $\dashrightarrow(Y, X)$
I1($I6^{-1}$)	**1**	1	0	0
I2($I5^{-1}$)	**0**	**1**	0	**0**
I3($I3^{-1}$)	**0**	**1**	0	**1**
I4($I4^{-1}$)	**0**	**0**	0	**0**
I5($I2^{-1}$)	0	**0**	**0**	**1**
I6($I1^{-1}$)	0	0	**1**	1
Evaluation conditions if X is a linear interval and Y is a point interval	$y > \max(X)$	$y > \min(X)$	$\min(X) > y$	$\max(X) > y$

PROPOSITION 1. *$\mathcal{F}_{Lpt} = \langle \mathcal{A}_{Lpt}, \Psi_{Lpt}, \mathcal{R}_{Lpt}, \mathcal{X}_{Lpt} \rangle$, where* (i) *$\mathcal{A}_{Lpt}$ is a set of intervals*; (ii) *Ψ_{Lpt}: : each $A \in \mathcal{A}_{Lpt}$ is linear*; (iii) *$\mathcal{R}_{Lpt} = \{\longrightarrow, \dashrightarrow\}$ and each r in $\mathcal{R}_{Lpt}$ is defined on $\mathcal{A}_{Lpt} \times \mathcal{A}'_{Lpt}$ and $\mathcal{A}'_{Lpt} \subseteq \mathcal{A}_{Lpt}$ such that for each $A \in \mathcal{A}'_{Lpt}$, $|A| = 1$*; (iv) *$\mathcal{X}_{Lpt}$ is the set of axioms A1–A4* [10, 12], *and Axiom* A6.

THEOREM 1. *Any two intervals in the family of system executions $\mathcal{F}_{Lpt}$ may interact in one of six ways* (*given in Table 1*).

Proof. Table 1 describes the (valid) interactions using combinations of the values of $\longrightarrow (X, Y)$, $\dashrightarrow (X, Y)$, $\dashrightarrow (Y, X)$, and $\longrightarrow (Y, X)$; all other combinations of the four relations are invalid because they violate $\mathcal{X}_{Lpt}$. The six interaction types are labeled I1–I6. The dominating relations for any interaction type are indicated by displaying their boolean values in boldface. For example, for interaction type I2, $\longrightarrow(X, Y) = false$, $\dashrightarrow(X, Y) = true$, and $\dashrightarrow(Y, X) = false$ uniquely define the interaction type. Observe that relations $\longrightarrow$ and $\dashrightarrow$ suffice to define all interaction types if X is a linear set and $|Y| = 1$. Interaction types I1 and I6 are inverses of each other. Interaction types I2 and I5 are inverses of each other. I3 and I4 are their own inverses. The six possibilities for the interaction between X and Y are pictorally depicted in Fig. 1. Interval X is shown in a fixed position using a rectangular box. Interval Y, shown as a shaded circle to depict a point, is in different positions relative to X. Each position of Y is labeled by an interaction type, I1–I6. The various interaction types are identified by the various positions of Y relative to X. Observe from Fig. 1 that the two intervals can interact in one of only these six ways. ▮

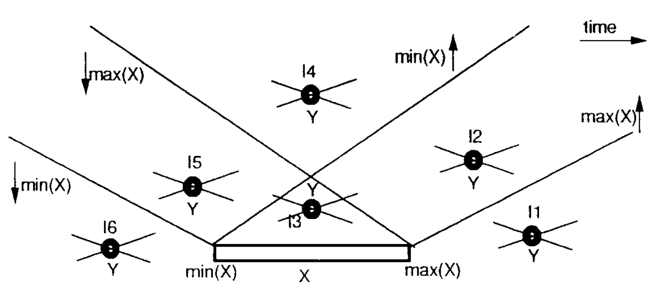

FIG. 1. Illustration of linear interval–point interval interactions.

The interpretation of the six interaction types is:

I1. Interval X entirely happened before interval Y.

I2. Interval Y is affected by part of interval X and no event in interval Y affects any event in interval X.

I3. Events in interval X affect events in Y, and those same events in Y affect some events in interval X.

I4. Events in interval X can neither affect nor are affected by events in interval Y.

I5. Events in interval X cannot affect events in interval Y but events in interval Y affect some events in interval X.

I6. Interval Y entirely precedes interval X.

The interaction types of $\mathcal{F}_{Lpt}$ will be used in defining the interaction types between a pair of linear intervals.

PROPOSITION 2. *$\mathcal{F}_{Lpt'} = \langle \mathcal{A}_{Lpt'}, \Psi_{Lpt'}, \mathcal{R}_{Lpt'}, \mathcal{X}_{Lpt'} \rangle$, where* (i) *$\mathcal{A}_{Lpt'}$ is a set of intervals*; (ii) *$\Psi_{Lpt'}$: : no constraints on each $A \in \mathcal{A}_{Lpt'}$*; (iii) *$\mathcal{R}_{Lpt'} = \{\longrightarrow, \dashrightarrow\}$ and each r in $\mathcal{R}_{Lpt'}$, is defined on $\mathcal{A}_{Lpt'} \times \mathcal{A}_{Lpt'}$*; (iv) *$\mathcal{X}_{Lpt'}$ is the set of axioms A1–A4* [10, 12], *and Axiom A6.*

Proposition 2 gives a family of system executions that corresponds to Lamport's definition of a system execution [10].

COROLLARY 1. *Any two intervals in the family of system executions $\mathcal{F}_{Lpt'}$ may interact in one of same six ways* (*given in Table 1*) *in which any two intervals in $\mathcal{F}_{Lpt}$ may interact.*

Proof. Follows from the proof of Theorem 1 which did not assume that the interaction types given in Table 1 between X and Y required X and Y to be a linear interval and a point interval, respectively. Only the evaluation conditions given in Table 1 for the relations in $\mathcal{R}_{Lpt}$ would not hold. ▮

Corollary 1 will be used in Section 4.1[3] to show that the framework used by Lamport in [10, 12] is not sufficient to capture interprocess communication in a distributed system.

3.2. Interaction between a Pair of Linear Intervals

Four additional relations besides $\longrightarrow$ and $\dashrightarrow$ need to be defined to capture the interaction between a pair of linear intervals where Ψ_{LL}:: $A \in \mathcal{A}_{LL}$ is linear. Henceforth, we will refer to Lamport's relations $\longrightarrow$ and $\dashrightarrow$ by R1 and R4,

[3] As mentioned in Section 2, we will rename relations $\longrightarrow$ and $\dashrightarrow$ as R1 and R4, respectively, in Section 3.2 to use consistent terminology with the new relations we introduce in that section.

TABLE 2

Relations used to define Interval Interactions

Relation r	Expression for $r(X, Y)$	Evaluation condition for $r(X, Y)$ if X, Y are linear intervals
R1 R1′	$\forall x \in X\, \forall y \in Y,\ x < y$ $= \forall y \in Y\, \forall x \in X,\ x < y$	$\min(Y) > \max(X)$
R2 R2′	$\forall x \in X\, \exists y \in Y,\ x < y$ $= \exists y \in Y\, \forall x \in X,\ x < y$	$\max(Y) > \max(X)$
R3 R3′	$\exists x \in X\, \forall y \in Y,\ x < y$ $= \forall y \in Y\, \exists x \in X,\ x < y$	$\min(Y) > \min(X)$
R4 R4′	$\exists x \in X\, \exists y \in Y,\ x < y$ $= \exists y \in Y\, \exists x \in X,\ x < y$	$\max(Y) > \min(X)$
S1	$\exists x \in X\, \forall y \in Y,\ x \nleq y \wedge y \nleq x$	**if** $\min(Y) \nless \min(X) \wedge \max(Y) \ngtr \max(X)$ **then** $\exists x \in X$: $\max((\min(Y) \uparrow)_X) \nleq x \nleq \max((\downarrow \max(Y))_X)$ **else** false
S2	$\exists x_1, x_2 \in X\, \exists y \in Y,\ x_1 < y < x_2$	**if** $\max(Y) > \min(X) \wedge \min(Y) < \max(X)$ **then** $\max((\downarrow \max(X))_Y) \nless \max((\min(X) \uparrow)_Y)$ **else** false

respectively. Relations R1, R2, R3, R4 define *causality conditions*, whereas S1 and S2 define *coupling conditions*. The relations R1–R4 and S1–S2 are expressed in terms of the quantifiers over X and Y in Table 2 along with evaluation conditions for the relations. Although the relations apply to nonlinear intervals, the evaluation conditions in the third column apply only to linear intervals.

The following is an interpretation of the relations:

R1. Same as $\longrightarrow$ [10].

R2. $R2(X, Y)$ iff every event in interval X causally happens before some event in interval Y. R2 signifies that interval Y completes after it fully knows the result of interval X. Thus, Y can take actions based on the complete result of X and in this sense, Y is later than X.

R3. $R3(X, Y)$ iff some event in interval X causally happens before every event in interval Y. R3 signifies that interval Y can be fully controlled by some input from X. However, the complete input from X may not be received by all of Y or even some of Y.

R4. Same as $\dashrightarrow$ [10].

S1. $S1(X, Y)$ iff some event in interval X has not affected any event in interval Y and has not been affected by any event in Y.

S1 is useful when programs communicate asynchronously and use the nonblocking mode for operations. $S1(X, Y)$ when X sends a nonblocking request to Y, performs local operations before receiving a reply from Y, and this is the only communication between X and Y.

S2. $S2(X, Y)$ iff interval X completes after having a round-trip interaction with interval Y.

S2 is useful for modeling interactions between programs/processes or groups of statements. For example,

TABLE 3

Reflexivity, Symmetry, and Transitivity of Relations in Table 2

Relation	Reflexive?	Symmetric?	Transitive?
R1 [10]	No	No	Yes
R2	No	No	Yes
R3	No	No	Yes
R4 [10]	No	No	No
S1	No	No	No
S2	No	No	No

TABLE 4

Hierarchy of Causality Relations in Table 2

Quantifiers in row/col. headers are for $x < y$	$R1(X, Y)$ $\forall x\, \forall y$	$R2(X, Y)$ $\forall x\, \exists y$	$R3(X, Y)$ $\exists x\, \forall y$	$R4(X, Y)$ $\exists x\, \exists y$
$R1(X, Y)$: $\forall x\, \forall y$	=	>	>	>
$R2(X, Y)$: $\forall x\, \exists y$	<	=	$\Vert$	>
$R3(X, Y)$: $\exists x\, \forall y$	<	$\Vert$	=	>
$R4(X, Y)$: $\exists x\, \exists y$	<	<	<	=

TABLE 5

Axioms for Causality Relations in Table 2

Axiom label	$r_1(X, Y) \wedge r_2(Y, Z) \Rightarrow r(X, Z)$
AL1	$R1(X, Y) \wedge R2(Y, Z) \Rightarrow R2(X, Z)$
AL2	$R1(X, Y) \wedge R3(Y, Z) \Rightarrow R1(X, Z)$
AL3	$R1(X, Y) \wedge R4(Y, Z) \Rightarrow R2(X, Z)$
AL4	$R2(X, Y) \wedge R1(Y, Z) \Rightarrow R1(X, Z)$
AL5	$R3(X, Y) \wedge R1(Y, Z) \Rightarrow R3(X, Z)$
AL6	$R4(X, Y) \wedge R1(Y, Z) \Rightarrow R3(X, Z)$
AL7	$R2(X, Y) \wedge R3(Y, Z) \Rightarrow$ true
AL8	$R2(X, Y) \wedge R4(Y, Z) \Rightarrow$ true
AL9	$R3(X, Y) \wedge R2(Y, Z) \Rightarrow R4(X, Z)$
AL10	$R4(X, Y) \wedge R2(Y, Z) \Rightarrow R4(X, Z)$
AL11	$R3(X, Y) \wedge R4(Y, Z) \Rightarrow R4(X, Z)$
AL12	$R4(X, Y) \wedge R3(Y, Z) \Rightarrow$ true

$S2(X, Y)$ if program X invokes a remote procedure Y which then completes and X gets back the reply.

$S2(X, Y)$ evaluates to a boolean value 0 or 1. (In Table 6 we refine S2 to be a tri-valued type, where the third value n indicates that the number of round-trips that X can have with Y is greater than 1. Whenever S2 appears in a predicate, value of n is considered as value 1.)

R1′, R2′, R3′, and R4′ are the same as R1, R2, R3, and R4, respectively, when defined on linear intervals. A detailed treatment of relations between nonlinear intervals, based on the above relations is given in [8].

Table 3 describes whether the relations defined in Table 2 are reflexive, symmetric, or transitive. Note that all the above are not independent relations. Table 4 gives the hierarchy and inclusion relationship of the causality relations R1–R4 defined above. Each cell in the grid indicates the relationship of the row header to the column header. The relationship can be one of three possible ones: $<$, $>$, and $\|$, where $\|$ indicates that the two relations being compared are incomparable, $>$ indicates "implies," and $<$ indicates "is implied by." Relations S1 and S2 are measures of the degree of coupling between the two intervals. Table 5 gives axioms AL1–AL12 on the causality relations. Axioms AL7, AL8, and AL12 simply state that no relation between X and Z is implied. The following axioms AL13–AL18 give all the relations that are implied by each of the causality and coupling relations:

AL13: $R1(X, Y) \Rightarrow \overline{S1}(X, Y) \wedge \overline{S2}(X, Y) \wedge \overline{R4}(Y, X) \wedge \overline{S1}(Y, X) \wedge \overline{S2}(Y, X)$

AL14: $R2(X, Y) \Rightarrow \overline{S1}(X, Y) \wedge \overline{R2}(Y, X)$

AL15: $R3(X, Y) \Rightarrow \overline{R3}(Y, X) \wedge \overline{S1}(Y, X)$

AL16: $R4(X, Y) \Rightarrow \overline{R1}(Y, X)$

AL17: $S1(X, Y) \Rightarrow \overline{R2}(X, Y) \wedge \overline{R3}(Y, X) \wedge \overline{S2}(Y, X)$

AL18: $S2(X, Y) \Rightarrow \overline{R1}(X, Y) \wedge R4(X, Y) \wedge \overline{R1}(Y, X) \wedge R4(Y, X) \wedge \overline{S1}(Y, X)$.

Tables 3, 4, 5 and the above axioms, AL13–AL18, collectively form an adequate set of axioms to reason because:

- Table 4 and axioms AL13–AL18 give all enumerations of all relations $r(X, Y)$ and $r(Y, X)$ implied by $R(X, Y)$, $\forall r \forall R \in \{R1, R2, R3, R4, S1, S2\}$.
- Tables 3 and 5 enumerate all relations implied by $r_1(X, Y) \wedge r_2(Y, Z)$, $\forall r_1 \forall r_2 \in \{R1, R2, R3, R4\}$. We do not specify any relation implied between X and Z by $r_1(X, Y) \wedge r_2(Y, Z)$, if either r_1 or r_2 belongs to $\{S1, S2\}$.
- This set of axioms can be used to derive all possible implications, from any given valid predicates on these relations.

Before we examine the interaction types between a pair of linear intervals, we consider an axiom on the nature of linear intervals.

Axiom 2. *Each interval A is dense, i.e.,* $\forall x, y \in A$, $x < y \Rightarrow \exists z \in A \mid x < z < y$.

Axiom 2 states that A is defined over an infinite dense set E and each A that is not a single-member set contains ∞ number of elements. This axiom is Axiom 8 of Hamblin [7].

3.2.1. *Dense Linear Intervals*

We define a family of system executions assuming dense linear intervals. This family is useful because each interval can model the passage of time at a point in space. Time at

TABLE 6

Interaction Types between Dense Linear Intervals

Interaction	Relation $r(X, Y)$						Relation $r(Y, X)$					
type	R1	R2	R3	R4	S1	S2	R1	R2	R3	R4	S1	S2
IA ($=IQ^{-1}$)	1	1	1	1	0	0	0	0	0	0	0	0
IB ($=IR^{-1}$)	0	1	1	1	0	0	0	0	0	0	0	0
IC ($=IV^{-1}$)	0	0	1	1	1	0	0	0	0	0	0	0
ID ($=IX^{-1}$)	0	0	1	1	1	1	0	1	0	1	0	0
ID′ ($=IU^{-1}$)	0	0	1	1	0	n	0	1	0	1	0	n
IE ($=IW^{-1}$)	0	0	1	1	1	1	0	0	0	1	0	0
IE′ ($=IT^{-1}$)	0	0	1	1	0	n	0	0	0	1	0	n
IF ($=IS^{-1}$)	0	1	1	1	0	n	0	0	0	1	0	n
IG ($=IG^{-1}$)	0	0	0	0	1	0	0	0	0	0	1	0
IH ($=IK^{-1}$)	0	0	0	1	1	0	0	0	0	0	1	0
II ($=IJ^{-1}$)	0	1	0	1	0	0	0	0	0	0	1	0
IL ($=IO^{-1}$)	0	0	0	1	1	1	0	1	0	1	0	0
IL′ ($=IP^{-1}$)	0	0	0	1	0	n	0	1	0	1	0	n
IM ($=IM^{-1}$)	0	0	0	1	1	0	0	0	0	1	1	0
IN ($=IM'^{-1}$)	0	0	0	1	1	1	0	0	0	1	0	0
IN′ ($=IN'^{-1}$)	0	0	0	1	0	n	0	0	0	1	0	n

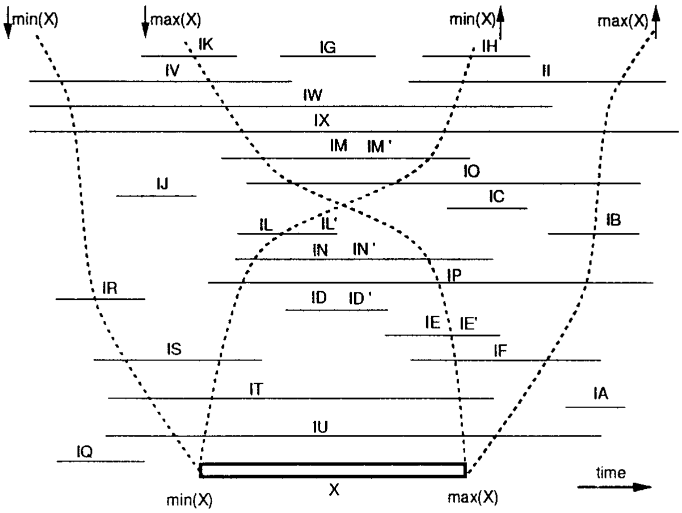

FIG. 2. Illustration of interaction types between dense linear intervals.

a point in space is dense and linear. This family extends Hamblin's result to a distributed system. We show that this family permits 29 interaction types between a pair of intervals. Of these, there are 13 pairs of inverses, while three are inverses of themselves.

PROPOSITION 3. $\mathscr{F}_{LL} = \langle \mathscr{A}_{LL}, \Psi_{LL}, \mathscr{R}_{LL}, \mathscr{X}_{LL} \rangle$, *where* (i) $\mathscr{A}_{LL}$ *is a set of intervals*; (ii) Ψ_{LL}: : *each* $A \in \mathscr{A}_{LL}$ *is linear and dense* (*Axiom* 2); (iii) $\mathscr{R}_{LL} = \{\text{R1, R2, R3, R4, S1, S2}\}$; (iv) $\mathscr{X}_{LL}$ *is the set of axioms in and derivable from Tables 3, 4, and 5, and axioms* AL13–AL18.

THEOREM 2. *Any two intervals in the family of system executions* $\mathscr{F}_{LL}$ *may interact in one of* 29 *ways* (*given in Table 4*).

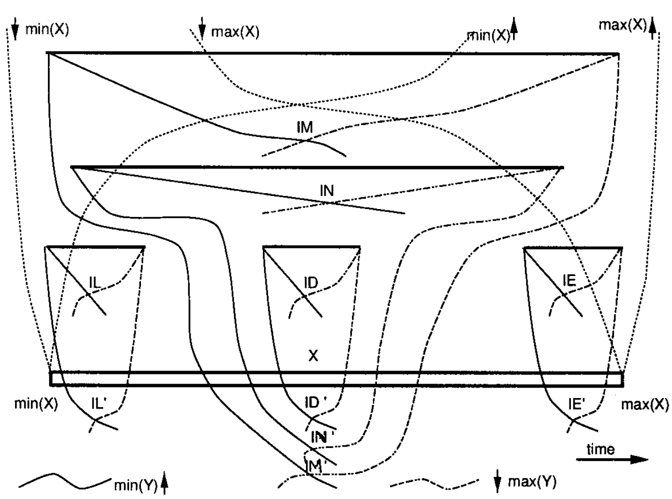

FIG. 3. Illustration of interaction types between dense linear intervals (continued).

Proof. Any pair of linear time intervals lie in (or can be projected onto) a two-dimensional plane where one dimension is the time dimension. Consider Figs. 2 and 3 that enumerate every possible interaction type between a pair of dense linear intervals. In Fig. 2, interval X is shown in a fixed position using a rectangular box whereas interval Y, indicated using horizontal lines, is in different positions relative to X. Each position of Y is labeled by an interaction type, IA through IX (and their modifiers). The different types of interactions are identified by the various positions of Y relative to X. Some positions of Y have two labels each. Specifically, five positions of Y are labeled ID and ID', IE and IE' , IL and IL', IM and IM', and IN and IN', respectively. For each of these five positions of Y, the distinction between the two interaction types, represented by the two labels, is illustrated using Fig. 3. In this figure, the positions of Y are indicated using a thick horizontal line. Observe from Figs. 2 and 3 that X and Y can interact in one of only 29 ways.

Each of the 29 interaction types between two linear intervals in $\mathscr{A}$ satisfying Ψ_{LL} is formally specified using boolean vectors. The six relations R1–R4 and S1–S2[4] in $\mathscr{R}_{LL}$ form a boolean vector of length 12 (six bits for $r(X, Y)$ and six bits for $r(Y, X)$). The 29 possibe interaction types are defined in Table 6. Each combination of values in the 29 boolean vectors can be seen to satisfy $\mathscr{X}_{LL}$. In the 29 interaction types, there are 13 pairs of inverses and three interaction types are inverses of themselves.

An example of how to interpret Table 6 follows: consider interaction IC and its inverse IV. We have $IC(X, Y) = IV(Y, X)$. The vector components for interaction IC (and for IV in brackets) are $\overline{R1}(X, Y)[\overline{R1}(Y, X)]$, $\overline{R2}(X, Y)[\overline{R2}(Y, X)]$, $R3(X, Y)[R3(Y, X)]$, $R4(X, Y)[R4(Y, X)]$, $S1(X, Y)[S1(Y, X)]$, $\overline{S2}(X, Y)[\overline{S2}(Y, X)]$, $\overline{R1}(Y, X)[\overline{R1}(X, Y)]$, $\overline{R2}(Y, X)[\overline{R2}(X, Y)]$, $\overline{R3}(Y, X)[\overline{R3}(X, Y)]$, $\overline{R4}(Y, X)[\overline{R4}(X, Y)]$, $\overline{S1}(Y, X)[\overline{S1}(X, Y)]$, $\overline{S2}(Y, X)[\overline{S2}(X, Y)]$. ∎

3.2.2. *Nondense Linear Intervals*

When Axiom 2 does not hold, the intervals may be nondense. This defines a different family of system executions $\mathscr{F}_{L'L'}$. This family is significant because clocks which measure dense linear time use a nondense linear scale in practice and are, nonetheless, very useful. This family is also significant because actions at each node in a distributed system are often modeled as a linear sequence of discrete events. This family permits eleven interaction types between a pair of intervals in addition to those permitted by $\mathscr{F}_{LL}$. Of

[4] In Table 6 we refine the boolean S2 to be a tri-valued type, where the third value n indicates that the number of round-trips that X can have with Y is greater than 1. Whenever S2 appears in a predicate, value of n is considered as value 1.

these, there are five pairs of inverses, while one is its own inverse.

PROPOSITION 4. *$\mathscr{F}_{L'L'} = \langle \mathscr{A}_{L'L'}, \Psi_{L'L'}, \mathscr{R}_{L'L'}, \mathscr{X}_{L'L'} \rangle$, where* (i) *$\mathscr{A}_{L'L'}$ is a set of intervals*; (ii) *$\Psi_{L'L'}$: : each $A \in \mathscr{A}_{L'L'}$ is linear*; (iii) *$\mathscr{R}_{L'L'} = \{R1, R2, R3, R4, S1, S2\}$*; (iv) *$\mathscr{X}_{L'L'}$ is the set of axioms in and derivable from Tables 3, 4 and 5 and axioms* AL13-AL18.

$\mathscr{F}_{L'L'}$ differs from $\mathscr{F}_{LL}$ in the specification of Ψ. $\Psi_{L'L'}$ permits $\mathscr{R}$ to be applied to a larger set of intervals than does Ψ_{LL}. Therefore it is plausible that there are more interaction types possible between two intervals in $\mathscr{F}_{L'L'}$ than in $\mathscr{F}_{LL}$. We will identify all possible interaction types for $F_{L'L'}$.

THEOREM 3. *Any two intervals in the family of system executions $\mathscr{F}_{L'L'}$ may interact in one of forty ways* (*given in Tables 6 and 8*).

Proof. To identify all interaction types between a pair of nondense linear intervals in $\mathscr{A}_{L'L'}$ satisfying $\Psi_{L'L'}$, we first identify all combinations of interaction types that various events in a nondense linear interval Y can have with the nondense linear interval X. For each such combination, we will either identify an interaction type already defined in Table 6 for dense intervals or define a new interaction type.

The individual events in Y will interact with nondense interval X in one of the six ways identified by interactions I1–I6 in Table 1. This is because the formulation of interaction types I1–I6 was done using only min(X) and max(X) and did not assume that X was dense. Observe that if some event in Y has interaction type I3 (I4) with X, then no other event in Y can have interaction type I4 (I3) with X. Define an increasing sequence S_{Lpt} as $\langle I1, I2, I3, I5, I6 \rangle$ ($\langle I1, I2, I4, I5, I6 \rangle$) if some event in Y has (does not have) an interaction type I3 with X. Also define $I_{\min}$ and $I_{\max}$ as the interaction types interval X has with min(Y) and max(Y), respectively. For the purpose of classifying interactions, interval Y may be considered to be dense (with respect to interval X) if for each I in S_{Lpt} satisfying $I_{\min} \geqslant I \geqslant I_{\max}$, $\exists y \in Y$, such that y has interaction type I with X.

The number of combinations of interaction types that various events in some Y can have with X is $\sum_{i=1}^{5} C_i^5 + \sum_{i=0}^{4} C_i^4 = (2^5 - 1) + 2^4$. The first (second) term is the number of combinations such that no event (at least one event) in Y has interaction type I4 with X. The above combinations include all cases, where Y is dense and Y is nondense. As noted earlier, X is not assumed to be dense. Therefore, these are all the combinations of interac tion types between individual events in Y and a not necessarily dense X. These combinations are enumerated in the first and fifth columns of Table 7. For each combination, we will

TABLE 7

Enumeration of Interactions between Nondense Linear Intervals

Interaction between X and $y \in Y$	Interaction types from Table 6 if Y is dense	New interaction types if Y is dense	Interaction types if Y is nondense	Interaction between X and $y \in Y$	Interaction types from Table 6 if Y is dense	New Interaction types if Y is dense	Interaction types if Y is nondense
I1	IA			I3, I5, I6	IS		
I2	IC			I1, I2, I3, I5	IP		
I3	ID, ID′	ID″		I1, I2, I3, I6			IU
I5	IJ			I1, I2, I5, I6			IUX
I6	IQ			I1, I3, I5, I6			IU
I1, I2	IB			I2, I3, I5, I6	IT		
I1, I3			IF	I1, I2, I3, I5, I6	IU		
I1, I5			IOP	I4	IG		
I1, I6			IUX	I4, I1			II
I2, I3	IE, IE′	IE″		I4, I2	IH		
I2, I5			IMN, IMN′, IMN″	I4, I5	IK		
I2, I6			ITW	I4, I6			IV
I3, I5	IL, IL′	IL″		I4, I1, I2	II		
I3, I6			IS	I4, I1, I5			IO
I5, I6	IR			I4, I1, I6			IX
I1, I2, I3	IF			I4, I2, I5	IM, IM′	IM″	
I1, I2, I5			IOP	I4, I2, I6			IW
I1, I2, I6			IUX	I4, I5, I6	IV		
I1, I3, I5			IP	I4, I1, I2, I5	IO		
I1, I3, I6			IU	I4, I1, I2, I6			IX
I1, I5, I6			IUX	I4, I2, I5, I6	IW		
I2, I3, I5	IN, IN′	IN″		I4, I1, I5, I6			IX
I2, I3, I6			IT	I4, I1, I2, I5, I6	IX		
I2, I5, I6			ITW				

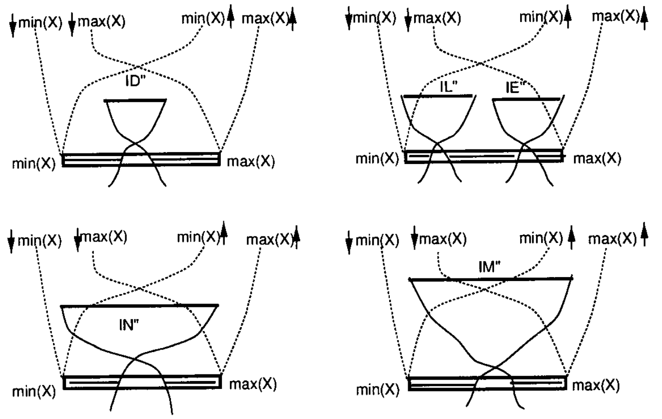

FIG. 4. Illustration of some additional interaction types for nondense linear intervals.

identify an already defined interaction type from Table 6 or define a new interaction type.

Observe that in Table 6 only the interactions ID, ID', IE, IE', IL, IL', IM, IM', IN, and IN' assume X was dense. The second and sixth columns of Table 7 give the interaction types from Table 6. These assume Y is dense; X is not assumed to be dense, except for interactions ID, ID', IE, IE', IL, IL', IM, IM', IN, and IN'. The third and seventh columns give the new interaction types required assuming Y is dense and X is nondense. The interaction types are ID'', IE'', IL'', IM'', and IN''. These are defined in Table 8 and illustrated in Fig. 4. In Fig. 4, X is shown in a fixed position, whereas the positions of Y vary relative to X. The linear interval X is shown as a rectangle. A line running along the length of the rectangle shows a dense region of the interval. The breaks in the line inside the rectangle indicate the absence of events. The linear interval Y is shown as a thick horizontal line.

The fourth and eighth columns of Table 7 give the interaction types between X and nondense Y. X is not assumed to be dense. Some of these interaction types for nondense Y intervals are the same as the interaction types in Table 6 for dense Y, using the expressive power of the defined suite of relations. For example, intervals Y whose individual events have interaction types ($I1$, $I2$, $I3$, $I5$, $I6$) or ($I1$, $I3$, $I5$, $I6$) or ($I1$, $I2$, $I3$, $I6$) have the same interaction type IU with interval X. Similarly, intervals Y whose individual events have interaction types ($I1$, $I3$) or ($I1$, $I2$, $I3$) have the same interaction type IF with X. However, some new interaction types are required for nondense Y intervals, namely, IMN, IMN′, IMN″, ITW, IUX, and IOP. These are defined in Table 8 and described using Figs. 2 and 4. Interaction types IMN, IMN′, IMN″, IOP, IUX, and ITW are like interaction types IN, IN′, IN″, IP, IU, and IT, respectively, except that there is no event $y \in Y$ in region I3 with respect to X, i.e., $\nexists y \in Y \mid \min(X) < y < \max(X)$. Alternately viewed, interaction types IMN, IMN′, IMN″, IOP, IUX, and ITW are like interaction types IM, IM′, IM″, IO, IX, and IW, respectively, except that there is no event $y \in Y$ in region I4 with respect to X, i.e., $\nexists y \in Y \mid y \nless \max(X) \wedge y \ngtr \min(X)$.

Of the above new interaction types in columns 4 and 8, only IMN″ requires that X is nondense. It is observed that the interaction type between X and various Y, that have different combinations of interaction types between X and the events of Y, may be the same using the expressive power of the defined suite of relations. For example, intervals Y whose individual events have interaction types (I1, I2, I5, I6) or (I1, I2, I6) or (I1, I5, I6) have the same interaction type IUX with X.

Thus, an exhaustive enumeration of the combinations of the interaction types between X and individual events in Y yields eleven additional valid interaction types that have not been defined in Table 6. These eleven interaction types are defined in Table 8 using $\mathscr{R}_{L'L'}$ and can be seen to satisfy $\mathscr{X}_{L'L'}$. Of these, there are five pairs of inverses, and one is its own inverse. It follows that there are 40 interaction types for linear (possibly nondense) interval interactions. $\Psi_{L'L'}$ differed from Ψ_{LL} in that it permitted $\mathscr{R}$ to be applied to a larger set of intervals than did ψ_{LL}. We showed above that $\mathscr{F}_{L'L'}$ allowed more interaction types between two intervals than did $\mathscr{F}_{LL}$. ▌

4. APPLICATIONS

The results presented in this paper contribute to the quintessential study of time [1, 2] and give an insight into the

TABLE 8

Additional Interaction Types for Nondense Linear Intervals

Interaction type	Relation $r(X, Y)$						Relation $r(Y, X)$					
	R1	R2	R3	R4	S1	S2	R1	R2	R3	R4	S1	S2
ID″ (= (IUX)$^{-1}$)	0	0	1	1	0	1	0	1	0	1	0	0
IE″ (= (ITW)$^{-1}$	0	0	1	1	0	1	0	0	0	1	0	0
IL″ (= (IOP)$^{-1}$)	0	0	0	1	0	1	0	1	0	1	0	0
IM″ (= (IMN)$^{-1}$)	0	0	0	1	0	0	0	0	0	1	1	0
IN″ (= (IMN′)$^{-1}$)	0	0	0	1	0	1	0	0	0	1	0	0
IMN″ (= (IMN″)$^{-1}$)	0	0	0	1	0	0	0	0	0	1	0	0

concurrency and interprocess communication in a distributed system. In this section, we consider the relation of our results to Lamport's work and present two applications. We anticipate that in the future, as applications and systems get more sophisticated, they will need to resort to the theory presented here.

4.1. Relation to Lamport's Work

We apply our theory to Lamport's relations $\longrightarrow$ and $\dashrightarrow$ [10, 12], which we renamed R1 and R4, respectively, in Section 3.2. We proved in Corollary 1 that for intervals X and Y, of the 2^4 combinations of the values of $R1(X, Y)$, $R4(X, Y)$, $R1(Y, X)$, and $R4(Y, X)$, the only ones that are valid using semantics and axioms defined in [10, 12] (and used in Proposition 2 for $\mathscr{F}_{Lpt'}$) are ones corresponding to the six combinations I1 through I6. By comparing Table 1 and Table 6, it can be deduced that these six combinations are not sufficient to model all possible interactions between a pair of dense linear intervals in a distributed system. Specifically,

1. I1 maps to IA,
2. I2 maps to IB,IC,IH,II,
3. I3 maps to ID, IX, ID′, IU, IE, IW, IE′, IT, IF, IS, IL, IO, IL′, IP, IM, IN, IM′, IN′,
4. I4 maps to IG,
5. I5 maps to IR, IV, IK, IJ, and
6. I6 maps to IQ.

Because of the above ambiguity, Lamport's framework cannot capture all possible interactions between two (linear) intervals in a distributed system and is "incomplete." The relations R1 and R4 defined by Lamport are a part of a larger suite of relations formulated in $\mathscr{R}_{LL}$ and $\mathscr{R}_{L'L'}$, which can capture a wider range of interaction types between linear intervals in a distributed system. Any application that needs to model the interaction of intervals more powerfully than can be done with R1 and R4 will use our results.

Next, we examine how Lamport's axioms A1–A5 [10] relate to the ones presented here. Axiom A1 (R1 is an irreflexive partial order) is included in Table 3. Axiom A2 ($R1(X, Y) \Rightarrow R4(X, Y) \wedge \overline{R1}(Y, X)$) can be inferred from Table 4 and Axiom AL15. Axiom A3 ($R1(X, Y) \wedge R4(Y, Z)$ or $R4(X, Y) \wedge R1(Y, Z) \Rightarrow R4(X, Z)$) can be refined to a finer granularity by axioms AL3 and AL6 presented here. Axiom A4 ($R1(X, Y) \wedge R4(Y, Z) \wedge R1(Z, W) \Rightarrow R1(X, W)$) is really a composite of more elementary axioms and can be derived from the axioms in Table 5. Axiom A5 of Lamport can be used if a system execution with a finite starting instant in time is to be modeled. Our axioms AL1–AL18 are far more inclusive than Lamport's axioms [10].

Relations R1 and R4, and axioms A1–A5 along with Lamport's register axioms [11] were sufficient to study concurrent read and write accesses to shared memory because all interactions between read and write operations were through the common register. The features of shared registers that made Lamport's framework [10, 11] complete were:

1. Intervals X and Y were read and write operations, Rd and W, respectively, that interacted through a common register. This register provided a common time axis, which is implied by the register axiom "$R4(Rd, W)$ or $R4(W, Rd)$" [11].
2. The central assumption about the dynamic atomic registers (as well as regular and safe registers which are more permissive) Lamport considered was the following. The read or write operation on the register can go into effect at any point during the operation interval (… as long as the resulting history was equivalent to a serial execution).

If Rd and W are related by R1, then the two operations are unambigiously serialized. If there is any overlap at all then R4 holds and the ordering between the two operations is necessarily ambigious. The distinction of whether any of $R2(Rd, W)$, $R3(Rd, W)$, $R2(W, Rd)$, $R3(W, Rd)$ hold is not useful, i.e., a further level of granularity beyond whether $R1(Rd, W)$ or $R1(W, Rd)$ holds is not useful. Again, S1 and S2 are not useful for this particular problem because the Rd and W operations interact only through a common register and not directly with one another. Thus, S1 and S2 are not interesting for analysing dynamic atomic registers.

However, if a stronger model than dynamic atomic were assumed, i.e., item 2 above was relaxed, then the distinctions offered by R2, R3, S1, and S2 would be useful. For example, if $R2(X, Y)$ and $R3(X, Y)$, then a certain probability could be associated to "X precedes Y". The results can be applied to defining a spectrum of registers that vary in the degree of concurrency they permit for reads and writes.

4.2. Specification of Global Predicates

Global predicates play an important role in distributed systems for applications such as debugging, industrial process control, and detecting specific states in the system ([6], which also refers to related literature). To adapt to the framework of specifying global predicates, we assume that each process/point partition of the system has a local state and there is a local state transition at every local event.

An unstable predicate is a predicate whose value may change with time. Specifying unstable predicates is a challenging problem. Unstable predicates have been classified as being either *strong* or *weak*. A strong (weak) predicate holds iff each (some) run of the system execution goes through a state in which the predicate holds. An orthogonal classification is *conjunctive* and *disjunctive* predicates. Global predicates can be expressed as a conjunction (*conjunctive* predicates) or disjunction (*disjunctive* predicates) of local predicates.

We will disregard disjunctive predicates because they are locally detectable. Conjunctive predicates are a subset of the set of possible interaction types identified here.

Let X and Y represent the linear intervals for which the local predicates hold on two nodes. The evaluation condition of a strong conjunctive predicate can be expressed in our framework as $R4(X, Y) \wedge R4(Y, X)$, for every pair of nodes whose local predicates are used to specify the global predicate.

The necessary and sufficient condition for a weak conjunctive predicate to hold was given in [6] as "the existence of an incomparable set of local states in which the local predicates are true." The condition for a weak conjunctive predicate to exist can be expressed using intervals. We specify an interval to be the set of contiguous events within a partition such that (i) all these events must lie between two consecutive message send events in the partition and (ii) the local predicate is true for all such events. For any two such intervals X and Y, it is observed that $\min(X)\uparrow = \max(X)\uparrow$ when projected on partitions other than the one that contains X, and hence, regions I2 and I3 of Fig. 1 disappear. Also, constraint (i) above implies that interval Y has to lie entirely in $\downarrow\min(X)$, in $(\downarrow\max(X)) - (\downarrow\min(X))$ or in $E - (\downarrow\max(X))$. By examining the interaction types between X and Y given in Table 6, it is seen that only for interaction types II, IG, IJ are there local states in X and Y that are concurrent and for which the local predicates are true.

Currently, global predicates under the existing classification can be specified using only R1 and R4; moreover, they do not even use all the valid combinations of R1 and R4 which we derived in Table 1. We can specify a broad range of global predicates using R1–R4, S1, and S2, to denote any of the interaction types in Tables 6 and 8. These interaction types can be used to specify any arbitrary global predicates. It is my conjecture that some day applications will be sophisticated enough to require this range of predicates.

4.3. Synchronization Conditions in a Distributed System

The relations and interaction types for each family of system executions can serve as synchronization conditions on the two intervals between which they are specified. The use of these synchronization conditions was described in Section 3, along with the formulations of the relations. For example, S1 and S2 are useful for modeling remote procedure calls issued in blocking and nonblocking modes, as well as data dependency relations between programs or statement blocks. Interaction types ID and ID′ can model nested transactions.

Distributed multimedia applications require synchronization within a media, among the media at a site and among different sites [13, 16]. The 13 interactions identified by Hamblin [7] have been used in multimedia systems to meaningfully manage and coordinate the data representation and playback [13]. The Son–Agarwal technique [16] provides for playback, where the different media streams satisfy the interactions given by Hamblin [7]. The synchronization conditions that can be specified on different data streams in a distributed environment using the relations and interaction types presented here offer much wider flexibility.

There is a wide class of multimedia applications such as videoconferencing, where it is sufficient to have a part of the coordination among multiple sites based on causality relations. For example, if $S2(X, Y)$ holds, then it may be inferred that the videoconferencing party Y has had a chance to respond to the query or data furnished by party X, and party X has received the response from Y. The relations and interaction types proposed here allow a wider expression of coordination conditions among different sites and multiple data streams, than do any previous results.

5. CONCLUSIONS

This paper studied how two intervals at distinct points in space–time in a distributed system interact with each other. There are three main contributions of this paper. The first contribution is that it identified all possible interaction types between two linear time intervals on different time axes, i.e., at different spatial coordinates, in a distributed system. We showed separate results for interaction types between dense linear intervals and between nondense linear intervals. Although time is dense and linear at a point, in practice, clocks use nondense scales to measure time. Also, the interaction types for nondense linear intervals are useful because actions in a distributed system are modeled as sequences of discrete events on each node. Therefore, both these results are useful and contribute to the study of time [1, 2]. The second contribution is that the results extend the 1972 result of Hamblin which considered interaction of intervals that occurred at the same spatial point, i.e., the intervals used a single global time axis [7]. The third contribution is that it enhances the understanding of concurrency in a distributed system and the nature of interprocess communication. A consequence of this contribution is that Lamport's modeling of interprocess communication is seen to be adequate for access to a shared register but not for distributed systems.

The applications of the results to the specification of global predicates and to synchronization conditions in a distributed system were examined.

The results of this paper have already been extended to provide a theory of the interaction of intervals, each of which spans across multiple time axes signifying a grouping of instants across various spatial points [8]. This grouping of space–time instants into an interval can be arbitrary, allowing an interval to model arbitrarily chosen levels of atomicity across space–time.

A future work direction is to identify more applications of the results to the state-of-the-art computing technology. Another direction for future work is to study hierarchical views [10] for the new relations we have defined.

ACKNOWLEDGMENTS

We thank the anonymous referees for some very useful comments.

REFERENCES

1. J. W. de Bakker, W. P. de Roever, and G. Rozenberg (Eds.), "Linear Time, Branching Time, and Partial Orders in Logics and Models of Concurrency," Lecture Notes in Computer Science, Vol. 354, Springer-Verlag, New York/Berlin, 1988.
2. J. V. Benthem, The Logic of Time, Kluwer Academic Publishers, (1 ed. 1983), 2 ed. 1991.
3. B. Charron-Bost, Combinatorics and geometry of consistent cuts: Application to concurrency theory, *in* "Proceedings of the International Workshop on Distributed Algorithms, 1989," Lecture Notes in Computer Science, Vol. 392, pp. 45–56, Springer-Verlag, New York/Berlin, 1989.
4. H. A. C. Dobbs, The dimensions of the sensible present, *in* "The Study of Time," pp. 274–292, Springer-Verlag, New York/Berlin, 1972.
5. P. C. Fishburn, "Interval Orders and Interval Graphs: A Study of Partially Ordered Sets," Wiley, New York, 1985.
6. V. K. Garg and B. Waldecker, Detection of weak and unstable predicates in distributed programs, *IEEE Trans. Parallel Distrib. Systems* **5**, No. 3 (1994), 299–307.
7. C. L. Hamblin, Instants and intervals, *in* "The Study of Time," pp. 324–332, Springer-Verlag, New York/Berlin, 1972.
8. A. D. Kshemkalyani, "Temporal Interactions of Intervals in Distributed Systems," Technical Report TR29.1933, IBM Research, Triangle Park, North Carolina, Sept. 1994.
9. L. Lamport, Time, Clocks, and the ordering of events in a distributed system, *Comm. ACM* **21**, No. 7 (1978), 558–565.
10. L. Lamport, On interprocess communication. I. Basic formalism, *Distrib. Comput.* **1** (1986), 77-85.
11. L. Lamport, On interprocess communication. II. Algorithms, *Distrib. Comput.* **1** (1986), 86–101.
12. L. Lamport, The mutual exclusion problem. I. A theory of interprocess communication, *J. Assoc. Comput. Mach.* **33**, No. 2 (1986), 313–326.
13. T. D. C. Little and A. Ghafoor, lnterval-based conceptual models for time-dependent multimedia data, *IEEE Trans. Knowledge Data Eng.* **5**, No. 4 (1993), 551–563.
14. F. Mattern, Virtual time and global states of distributed systems, *in* "Parallel and Distributed Algorithms," pp. 215–226, North-Holland, Amsterdam, 1989.
15. H. Minkowski, Space and time, *in* "The Principle of Relativity," pp. 73–91, Dover, New York, 1923.
16. S. Son and N. Agarwal, Synchronization of temporal constructs in distributed multimedia systems with controlled accuracy, *in* "Proceedings, 1st IEEE International Conf. on Multimedia Computing and Systems, May 1994."