A Fine-Grained Modality Classification for Global Predicates

Ajay D. Kshemkalyani, Senior Member, IEEE

Abstract—Specifying and detecting predicates in a distributed execution is an important problem. Distributed execution observation has classically used two modalities—Possibly(φ) and Definitely(φ)—for predicate φ. Based on the temporal interactions of intervals, the author identified a complete, orthogonal set of relationships \( \mathcal{R} \) between pairs of intervals in a distributed execution. This paper shows how to map the rich, orthogonal classification of modalities of pairwise interval interactions, to the classical coarse-grained classification, Possibly(φ) and Definitely(φ), for specifying predicates defined on any number of processes. This increases the power of expressing the temporal modalities under which predicates can be specified, beyond the current Possibly/Definitely classification. This paper also gives timestamp-based tests for the orthogonal modalities in the refined classification.

Index Terms—Predicates, causality, synchronization, distributed execution, distributed system, global state.

1 INTRODUCTION

Predicate specification and detection is an important problem in distributed systems. It is useful in specifying synchronization and coordination conditions in an execution in areas such as sensor networks, distributed debugging, and industrial process control. While stable predicates (once a stable predicate becomes true, it remains true) can be detected by techniques such as global snapshot algorithms, unstable predicates are difficult to deal with because snapshot algorithms cannot capture such predicates. The main difficulties in specifying and detecting unstable predicates in a distributed execution are the following: 1) The asynchronous nature of the process execution speeds and message delivery times leads to many interleavings of the events at different processes, each of which leads to a different sequence of global states corresponding to a different observation of the same execution. 2) A global observation of the execution across different processes is inherently not possible in a distributed system.

A global predicate is defined on variables local to more than one process. Two modalities, Possibly(φ) and Definitely(φ), for the satisfaction of a global predicate φ in a distributed execution were defined by Cooper and Marzullo [3] and Marzullo and Neiger [15]. For a given execution, 1) Possibly(φ) is true if some observation of the execution may pass through a state in which φ is true, and 2) Definitely(φ) is true if every observation of the execution will pass through a state in which φ is true. These two modalities have become widely accepted. The modality classification—Possibly and Definitely—is coarse-grained. The formalism and axiom system given by Kshemkalyani [9] identified a complete orthogonal set \( \mathcal{R} \) of 40 fine-grained temporal interactions between intervals in a distributed execution. Here, any pair of intervals must be related by one and only one of the mutually exclusive interaction types identified in \( \mathcal{R} \). We show that this formalism provides more expressive power than the Possibly and Definitely modalities of [3] and [15] for specifying predicates. Specifically, we show how to map the rich, orthogonal classification of modalities of pairwise interval interactions to the classical coarse-grained classification, Possibly(φ) and Definitely(φ), for specifying predicates defined on any number of processes. This gives flexibility and power to monitor, synchronize, and control distributed executions. We also give timestamp-based tests to detect the modality of predicates defined on variables local to a pair of processes—these tests can then be applied to all pairs of processes to detect global predicates defined on variables local to more than two processes.

The power of our approach stems from the fact that we model intervals as in [9], as opposed to individual events as in [3], [15], in the distributed execution. The intervals at each process are the local time durations of interest. Specifically, in each interval at a process, the local variables using which the global predicate φ is defined, can potentially satisfy φ. Note that the notion of an interval corresponds to an abstract event or a nonatomic event as used by Lamport who argued that “it is useful to assume that primitive elements between which concurrency is modeled are nonatomic” [14]. A nonatomic event naturally models the execution of a code fragment by a process and is an important concept in reasoning about the interaction between multiple code fragments executed by different processes [11].

Section 2 gives the system model and reviews background material. Section 3 shows the refinement of the Possibly(φ) and Definitely(φ) classification in terms of the 40 possible interaction types \( \mathcal{R} \) [9]. Section 4 gives the timestamp-based tests to check for the 40 orthogonal relations. Section 5 discusses the significance and some applications of the fine-grained modality classification. Section 6 gives the conclusions.
2 SYSTEM MODEL AND BACKGROUND

2.1 System Model

In a distributed system, a set of processes $N$ communicate with one another by asynchronous message passing over logical channels. There is no shared memory and no common clock. A poset event structure model $(E, \prec)$, where $\prec$ is an irreflexive partial ordering representing the causality relation on the event set $E$, is used as the model for a distributed system execution. $E$ is partitioned into local executions at each process. $E_i$, the linearly ordered set of events executed by process $P_i$, is an event executed by $P_i$. An event executed by $P_i$ is denoted $e_i$. The causality relation on $E$ is the transitive closure of the local ordering relation on each $E_i$ and the ordering imposed by message send events and message receive events [13]. A subset of set $E$, i.e., if $e \in E$, then a cut is downward-closed with respect to $e$. Thus, a cut is a subset of $E$ that is downward-closed within partitions. A consistent cut $C$ is a downward-closed subset of $E$ in $(E, \prec)$ [17], i.e., if $e \in C$, then $\forall e' \prec e \Rightarrow e' \in C$. Only all downward-closed subsets of $E$ preserve causality and denote observable execution prefixes. For event $e$, there are two special consistent cuts $\downarrow e$ and $\uparrow e$. $\downarrow e$ is the maximal set of events that happen before $e$. $\uparrow e$ is the set of all events up to and including the earliest events at each process for which $e$ happens before the events.

**Definition 1 (Past of an event $e$).** $\downarrow e = \{e' | e' \prec e\}$

(Future of an event $e$). $\uparrow e = \{e' | e' \nprec e\} \cup \{e_i, i = 1, \ldots, |N| \mid e_i \geq e \wedge (\forall e' \prec e_i, e'_i \nprec e)\}.$

The system state after the events in a cut is a global state [2]; if the cut is consistent, the corresponding system state is termed a consistent global state and denotes a meaningful observation of a global state.

We assume the availability of vector clocks $\text{Clk}$ to maintain logical time [5], [17]. Vector clocks have the property that $e \prec f \Leftrightarrow \text{Clk}(e) < \text{Clk}(f)$. A vector clock has size $n = |N|$, with the $<$ operator defined as follows: 1) $\text{Clk} \leq \text{Clk}'$ if and only if $\forall k \in N \text{Clk}[k] \leq \text{Clk}'[k]$, and 2) $\text{Clk} < \text{Clk}'$ if and only if $\text{Clk} \leq \text{Clk}'$ and $\text{Clk} \neq \text{Clk}'$. The following rules update the clock at each process:

**V1.** (Internal event) Before process $P_i$ executes the event, $\text{Clk}_i[\hat{e}] = \text{Clk}_i[\hat{e}] + 1$.

**V2.** (Send event) Before process $P_i$ executes the event, $\text{Clk}_i[\hat{e}] = \text{Clk}_i[\hat{e}] + 1$. Send message timestamped with $\text{Clk}_i$.

**V3.** (Receive event) When process $P_j$ receives a message timestamped $T$ from process $P_i$, $(\forall k \in N) \text{Clk}_j[k] = \max(\text{Clk}_j[k], T[k])$, $\text{Clk}_j[\hat{e}] = \text{Clk}_j[\hat{e}] + 1$; deliver the message.

The timestamp $T$ of an event $e_i$ is the local clock value $\text{Clk}(e_i)$ that results after the execution of the event. Unless otherwise indicated, we do not assume any restriction on the predicates defined on the execution. Given a literal/boolean $b$, $\overline{b}$ denotes its complement. A variable $x$ local to process $P_i$ is denoted as $x_i$. To simplify notation, the process set $N$ is also represented as the set of integer process indices. For example, we use "$(\forall i \in N)$" instead of "$(\forall P_i \in N)$." Such usage should be clear from the context.

2.2 Prior Work

For abstract events $X$ and $Y$, Lamport defined the following two relations [14]:

1. $\longrightarrow : X \rightarrow Y$ iff $\forall x \in X, \forall y \in Y, x \prec y$.  
2. $\longrightarrow : X \rightarrow Y$ iff $\exists x \in X, \exists y \in Y, x \prec y$.

Four additional relations besides $\longrightarrow$ and $\rightarrow$ were defined in [9] to capture the interaction between a pair of intervals or abstract events. Henceforth, we will refer to relations $\rightarrow$ and $\longrightarrow$ by R1 and R4, respectively. Relations R1 (strong precedence), R2 (partially strong precedence), R3 (partial weak precedence), and R4 (weak precedence) define causality conditions, whereas S1 (event with no coupling) and S2 (round-trip coupling) define coupling conditions. The relations R1-R4 and S1-S2 are expressed in Table 1.

Axiom 1 gives the density axiom for time in terms of the events/time instants in an interval. It states that any interval $X$ is defined over an infinite dense set $E$, and each $X$ that is not a single-member set contains $\infty$ number of elements.

**Axiom 1 (Density Axiom).** Each interval $X$ is dense, i.e., $\forall x_1, x_2 \in X, x_1 \prec x_2 \Rightarrow \exists x_3 \in X \mid x_1 \prec x_3 \prec x_2$. 

<table>
<thead>
<tr>
<th>Relation $r$</th>
<th>Expression for $r(X, Y)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>R1 (strong precedence)</td>
<td>$\forall x \in X \forall y \in Y, x \prec y$</td>
</tr>
<tr>
<td>R2 (partially strong precedence)</td>
<td>$\forall x \in X \exists y \in Y, x \prec y$</td>
</tr>
<tr>
<td>R3 (partially weak precedence)</td>
<td>$\exists x \in X \forall y \in Y, x \prec y$</td>
</tr>
<tr>
<td>R4 (weak precedence)</td>
<td>$\exists x \in X \exists y \in Y, x \prec y$</td>
</tr>
<tr>
<td>S1 (event with no coupling)</td>
<td>$\exists x_1, x_2 \in X \exists y \in Y, x_1 \prec y \prec x_2$</td>
</tr>
<tr>
<td>S2 (round-trip coupling)</td>
<td>$\exists x_1, x_2 \in X \exists y \in Y, x_1 \prec y \prec x_2$</td>
</tr>
</tbody>
</table>
Assuming that time is dense, it was shown in [9] that there are 29 possible interaction types between a pair of intervals, as given in the upper part of Table 2. The significance of the second part of the table, as well as of the use of the italic type, will be explained later. Of the 29 interactions, there are 13 pairs of inverses, while three are inverses of themselves. The 29 interaction types are specified using Boolean vectors. For intervals $X$ and $Y$, the six relations $R_1-R_4$ and $S_1-S_2$ form a Boolean vector of length 12, (six bits for $r_{X,Y}$ and six bits for $r_{Y,X}$).

An example of how to interpret Table 2 follows. Consider interaction $IB$ and its inverse $IR$. We have $IB(X,Y) = 1R(Y,X)$. The vector components for interaction $IB(X,Y)$ are

$$\begin{align*}
\overline{R}_1(X,Y), \overline{R}_2(X,Y), \overline{R}_3(X,Y), \overline{R}_4(X,Y), \overline{S}_1(Y,X), \\
\overline{S}_2(X,Y), \overline{S}_1(Y,X), \overline{S}_2(Y,X), \overline{S}_1(Y,X), \\
\overline{S}_2(Y,X), \overline{S}_1(Y,X), \overline{S}_2(Y,X).
\end{align*}$$

Analogously, the vector components for $IR(X,Y)$ are

$$\begin{align*}
\overline{R}_1(X,Y), \overline{R}_2(X,Y), \overline{R}_3(X,Y), \overline{R}_4(X,Y), \overline{S}_1(Y,X), \\
\overline{S}_2(X,Y), \overline{S}_1(Y,X), \overline{S}_2(Y,X), \overline{S}_1(Y,X), \\
\overline{S}_2(Y,X), \overline{S}_1(Y,X), \overline{S}_2(Y,X).
\end{align*}$$

Note that with the dense time assumption, $R_1$ to $R_4$, and either $S_1$ or $S_2$ are sufficient to determine the interaction type.

Figs. 1 and 2 (reproduced from [9]) illustrate the complete set of 29 orthogonal relations for dense time. In Fig. 1, 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 $X$. $\min(X)$ and $\max(X)$ denote the least and greatest element of linear interval $X$, respectively. $\downarrow\min(X)$, $\downarrow\max(X)$, $\uparrow\min(X)$, and $\uparrow\max(X)$ denote consistent cuts. The different types of interactions are identified by the various positions of $Y$ relative to $X$. Each position of $Y$ is labeled by an interaction type, $IA$ through $IX$ (and their modifiers). Five positions of $Y$ have two labels each. For each of these five positions of $Y$, the distinction between the two interaction types, represented by the two labels, is

![Fig. 1. Illustration of interaction types between intervals under the dense time model [9].](image-url)
illustrated using Fig. 2. In this figure, the positions of \( Y \) are indicated using a thick horizontal line.

When Axiom 1 does not hold, the intervals may be nondense. This model is significant because clocks which measure dense linear time use a nondense linear scale in practice. Additionally, in some environments, local predicates may meaningfully be defined to hold only when a discrete event occurs. This model is also significant because actions at each node in a distributed system form a linear sequence of discrete events. This model permits 11 interaction actions at each node in a distributed system to form a linear discrete event occurs. This model is also significant because clocks which indicates may meaningfully be defined to hold only when a practice. Additionally, in some environments, local predicates may be defined to hold only when a discrete event occurs.

3 Refining the Possibly(\( \phi \)) / Definitely(\( \phi \)) Classification

3.1 Background and Approach

Specifying predicates on the system state provides an important handle to specify and detect the expected behavior of the system. Unstable predicates pose two challenges for detection in a system execution, as observed in [16]. 1) The predicate may no longer be true at the time the predicate is evaluated by a monitor. 2) Even if a predicate is found to be true, it still may not have ever held during the actual run. In particular, snapshot algorithms are not useful to detect unstable predicates because even if a predicate is detected, it may never have held, and the predicate may have held even if it is not detected.

Evaluating unstable predicates over global states constructed from observations of runs has the same drawbacks. To overcome these drawbacks, Cooper and Marzullo [3]. Marzullo and Neiger [15] proposed to extend predicates to apply to the entire distributed execution rather than to individual observations or global states of it. They defined:

- **Possibly(\( \phi \))**: There exists a consistent observation of the execution such that \( \phi \) holds in a global state of the observation.

Consider the example in Fig. 3. The execution is run at processes \( P_1 \) and \( P_2 \). Event \( e_k^i \) denotes the \( k \)th event at process \( P_i \). Variable \( a \) is local to \( P_1 \) and variable \( b \) is local to \( P_2 \). Observe that Definitely(\( a + b = 10 \)) holds by the following reasoning: When \( b \) is assigned 7 at event \( e_2^1 \), process \( P_1 \)'s execution may be in any state from the initial state up to the state preceding event \( e_3^2 \), in which \( a = 3 \). However, before the value of \( b \) changes from 7 to 5 at event \( e_2^2 \) and, in fact, before \( P_2 \) executes event \( e_2^2 \), \( P_1 \) must have executed event \( e_1^1 \) at which time \( a = 3 \). This is true for all equivalent executions. Hence, Definitely(\( a + b = 10 \)) holds. Although predicate \( (a + b = 5) \) is never true in the shown execution, observe that Possibly(\( a + b = 5 \)) holds by the following reasoning: The predicate \( a + b = 5 \) can be true only if: 1) \( a = 3 \land b = 2 \), 2) \( a = 0 \land b = 5 \), or 3) \( a = 8 \land b = -3 \), simultaneously in some equivalent execution. State (1) is possible in physical time after the occurrence of event \( e_3^2 \) and before the occurrence of \( e_2^1 \). In the execution shown, \( e_3^2 \) occurs after \( e_2^1 \). However, in an equivalent execution, event \( e_1^1 \) may be delayed to occur after event \( e_2^2 \), in which case \( b \) changes to a value other than 2 after \( a \) becomes 8. Hence, the predicate is true in this equivalent execution. It so happens that a similar argument also holds for (2) and (3).

Clearly, Definitely(\( \phi \)) implies Possibly(\( \phi \)), but the implication does not hold the other way unless \( \phi \) is stable. We observe that for any predicate \( \phi \), three orthogonal modal possibilities hold: 1) Definitely(\( \phi \)); 2) \( \neg \) Definitely(\( \phi \)) \land Possibly(\( \phi \)); and 3) \( \neg \) Definitely(\( \phi \)). Based on the formalism in [9], we now show that these three orthogonal modal possibilities can be refined into the exhaustive set of 29 (or 40) possibilities between each pair of intervals, based on whether time is assumed to be dense (or nondense), respectively.

The definitions of the modalities on predicates use system states. However, observe that the definitions essentially demarcate intervals at each process such that in each interval, the values of local variables using which \( \phi \) is defined, may satisfy \( \phi \). In general, the semantics of the intervals depends on predicate \( \phi \). For a conjunctive predicate \( \phi = \land \phi_i \), where \( \phi_i \) is a conjunct (predicate) local to process \( P_i \), each interval at \( P_i \) can be locally determined to be the maximum local duration in which local predicate \( \phi_i \) is true. For a nonconjunctive predicate, the interval at a
process needs to be identified carefully, possibly after reexpressing \( \phi \), depending on the structure of \( \phi \), and examining the state lattice for the execution. Whatever semantics is used to identify the intervals, the fine-grained modalities apply.

### 3.2 Predicate \( \phi \) Defined over Two Processes

Let us first consider the case when predicate \( \phi \) is defined on variables that are local to two processes. For a large class of applications, such as those using a pair of interacting agents or a client-server interaction, predicates are defined on variables local to two processes.

For the dense time model, we can formulate an alternate definition of \( \text{Possibly} \) and \( \text{Definitely} \), based on linear extensions of partial orders. For the poset \((E, \vdash)\), its linear extension is a total order that preserves the partial order.

**Definition 2.** Assume the dense time model. Let intervals \( X \) and \( Y \) at two processes be such that during these intervals, the local predicates are true.

- **Possibly(\( \phi \))**: (For some such intervals \( X \) and \( Y \),) there exists a linear extension of \((E, \vdash)\) such that the string \([\text{min}(X), \text{max}(X)]\) overlaps with the string \([\text{min}(Y), \text{max}(Y)]\).

- **Definitely(\( \phi \))**: (For some such intervals \( X \) and \( Y \),) for every linear extension of \((E, \vdash)\), the string \([\text{min}(X), \text{max}(X)]\) overlaps with the string \([\text{min}(Y), \text{max}(Y)]\).

Observe that intervals are qualified before the linear extensions to express the modalities in terms of one interval at each process.

**Theorem 1.** For the dense time model, the mappings of the 29 interaction types to the three classes 1) \( \text{Definitely}(\phi) \), 2) \( \neg\text{Definitely}(\phi) \land \text{Possibly}(\phi) \), and 3) \( \neg\text{Possibly}(\phi) \), are given in the top part of Table 3.

**Proof.** Definition 2 gives an alternate formulation of the \( \text{Possibly} \) and \( \text{Definitely} \) modalities using linear extensions of the partial order \((E, \vdash)\). If the strings \([\text{min}(X), \text{max}(X)]\) and \([\text{min}(Y), \text{max}(Y)]\) do not overlap in any linear extension, then \( R1(X, Y) \lor R1(Y, X) \) and even \( \text{Possibly}(\phi) \) cannot be true. The overlap of the strings in every linear extension of the partial order implies \( \text{Definitely}(\phi) \), and this overlapping is equivalently characterized as \( R4(X, Y) \lor R4(Y, X) \). Otherwise, the strings overlap in some linear extensions and not in some others, which implies \( \text{Possibly}(\phi) \land \neg \text{Definitely}(\phi) \). This is expressed in Fig. 4 in terms of the dependent relations \( R1 \) and \( R4 \). Mapping these conditions of Fig. 4 to the 29 orthogonal relations, we get the mapping shown in the upper part of Table 3.

### Table 3: Refinement Mapping

<table>
<thead>
<tr>
<th>Dense time</th>
<th>( \text{Definitely}(\phi) )</th>
<th>( \text{Possibly}(\phi) \land \neg\text{Definitely}(\phi) )</th>
<th>( \neg\text{Possibly}(\phi) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID and IX</td>
<td>IB and IR</td>
<td>IA and IQ</td>
<td></td>
</tr>
<tr>
<td>ID' and IU</td>
<td>IC and IV</td>
<td>IA and IQ</td>
<td></td>
</tr>
<tr>
<td>IE and IW</td>
<td>IG</td>
<td>IA and IQ</td>
<td></td>
</tr>
<tr>
<td>IE' and IT</td>
<td>IH and IK</td>
<td>IA and IQ</td>
<td></td>
</tr>
<tr>
<td>IF and IS</td>
<td>II and LJ</td>
<td>IA and IQ</td>
<td></td>
</tr>
<tr>
<td>IO and IL</td>
<td>IM</td>
<td>IA and IQ</td>
<td></td>
</tr>
<tr>
<td>IP and IL'</td>
<td>IM' and IN</td>
<td>IA and IQ</td>
<td></td>
</tr>
<tr>
<td>IM</td>
<td>IM' and IN</td>
<td>IA and IQ</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Nondense time (Interpretation 1)</th>
<th>Nondense time (Interpretation 2)</th>
<th>Nondense time (Interpretation 3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>all 11 new interactions</td>
<td>all except IA and IQ</td>
<td>IA and IQ</td>
</tr>
<tr>
<td>for nondense time</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The upper part shows the 29 mappings when the dense time model is assumed. For the nondense time model, the mappings under three possible interpretations of \( \text{Possibly} \) and \( \text{Definitely} \) are shown in the lower part.
Interpretation 1. When events are discrete and predicates can meaningfully take on values only at the events, overlapping strings \([\text{min}(X), \text{max}(X)]\) and \([\text{min}(Y), \text{max}(Y)]\) do not imply the existence of a single instant in global time at which the global predicate is necessarily true. Hence, \(\text{Definitely}(\phi)\) can never hold.

An example application where this interpretation is useful is in particle physics studies. A particle impinging on a surface denotes a discrete event, and a local predicate, although defined on other local variables at the surface, may be meaningfully true only at the time of occurrence of the discrete event.

Interpretation 2. Despite the nondense time model and events being discrete, if we apply Definition 2, then all the 11 new interaction types for nondense time map to \(\text{Definitely}(\phi)\). The mapping for the other 29 interactions is the same as for dense time.

Interpretation 3. To explicitly capture the notion of the strong coupling of events in \(X\) and \(Y\) for \(\text{Definitely}(\phi)\) to hold, we can require \(S2(X,Y)\) or \(S2(Y,X)\), which explicitly signify a \("round-trip dependence chain of events\)” between the two intervals, to be true. This would be in addition to the requirement in Definition 2 that strings \([\text{min}(X), \text{max}(X)]\) and \([\text{min}(Y), \text{max}(Y)]\) overlap for all linear extensions. The condition for \(\text{Definitely}(\phi)\) to hold can be expressed as \(R4(X,Y) \lor R4(Y,X) \lor (S2(X,Y) \lor S2(Y,X))\).

With this interpretation, observe that interaction type IM maps to \(\text{Possibly}(\phi) \land \neg\text{Definitely}(\phi)\), instead of to \(\text{Definitely}(\phi)\) if the dense time model had been used. The mappings for the other 28 interactions (applicable to the dense time model) are the same as for the dense time model. Of the 11 new interactions for nondense time, IM”, IMN, and IMN” map to \(\text{Possibly}(\phi) \land \neg\text{Definitely}(\phi)\), whereas the others map to \(\text{Definitely}(\phi)\).

Theorem 2. For the nondense time model, the mappings of the 40 interaction types to the three classes 1) \(\text{Definitely}(\phi)\), 2) \(\neg\text{Definitely}(\phi) \land \text{Possibly}(\phi)\), and 3) \(\neg\text{Possibly}(\phi)\), as per the three interpretations above, are given in the lower part of Table 3.

Proof. The proof follows by combining the explanations of the three interpretations, the proof of Theorem 1, Table 2, and Definition 2. The logic for the mappings is expressed in Fig. 5 in terms of the dependent relations \(R1, R4,\) and \(S2\).
In the example of Fig. 3, it can be shown that $IM(X_1(a = 3), Y_2(b = 7))$ (for dense time) and $IM^{\text{NH}}(X_1(a = 3), Y_2(b = 7))$ (for interpretation 2 of nondense time) hold; hence, $\text{Definitely}(a = 3 \land b = 7)$. The following can also be shown: 1) $IB(X_1(a = 3), Y_2(b = 2))$, hence, $\text{Possibly}(a = 3 \land b = 2) \land \neg\text{Definitely}(a = 3 \land b = 2), 2)$ $IG(X_1(a = 8), Y_2(b = 5))$, hence, $\text{Possibly}(a = 8 \land b = 5) \land \neg\text{Definitely}(a = 8 \land b = 5)$, and 3) $IV(X_1(a = 8), Y_2(b = 7))$, hence, $\text{Possibly}(a = 8 \land b = 7) \land \neg\text{Definitely}(a = 8 \land b = 7)$.

3.3 Predicate $\phi$ Defined over More than Two Processes

In general, a predicate is defined on variables local to some subset $V$ of the set of all the processes, $\mathbb{N}$. When a predicate $\phi$ is defined over variables that are local to more than two processes, one can express the three modal possibilities 1) $\text{Definitely}(\phi), 2) \neg\text{Definitely}(\phi) \land \text{Possibly}(\phi)$, and 3) $\neg\text{Possibly}(\phi)$, in terms of the more fine-grained pairwise orthogonal modalities in $\mathbb{O}$, between $C_{V}^{2}$ pairs of processes. Note that not all $40^{2^{2}}$ combinations will be valid—the combinations have to satisfy the axioms given in [9]. When considering the nondense time model in the rest of this section, the proofs implicitly assume Interpretation 2 of Possibly and Definitely. The proofs for the other interpretations follow directly.

**Theorem 3.** For a conjunctive predicate $\phi$ defined on processes in $V \subseteq \mathbb{N}$, let $\phi$ be expressed as $\bigwedge_{i \in V} \phi_i$, where $\phi_i$ is the component of $\phi$ local to process $P_i$. The following results are implicitly qualified over a set of intervals, containing one interval from each process.

1. $\text{Definitely}(\phi)$ holds if and only if $\bigwedge_{(i \in V)(j \in V)} \{\text{Definitely}(\phi_i \land \phi_j)\}$.

2. $\neg\text{Definitely}(\phi) \land \text{Possibly}(\phi)$ holds if and only if

   $\left(\exists i \in V \exists j \in V \neg\text{Definitely}(\phi_i \land \phi_j) \bigwedge \bigwedge_{(i \in V)(j \in V)} \{\text{Possibly}(\phi_i \land \phi_j)\}\right)$.

3. $\neg\text{Possibly}(\phi)$ holds if and only if $\exists i \in V \exists j \in V \neg\text{Possibly}(\phi_i \land \phi_j)$.

**Proof.** Part 1: $\text{Definitely}(\phi)$ implies that every observation of the execution must include a state in which $\phi$ holds. Thus, in every linear extension of the execution, the strings $[\text{min}(X), \text{max}(X)]$ and $[\text{min}(Y), \text{max}(Y)]$ for each pair of intervals $X$ and $Y$ overlap. We also have that in every observation of the execution, there must exist an instant in time when $\phi = \bigwedge_{i \in V} \phi_i$ is true. At this instant, the consequent must hold for the predicate decomposed pairwise, i.e., the execution must pass through this instant at which $\bigwedge_{(i \in V)(j \in V)} \{\text{Definitely}(\phi_i \land \phi_j)\}$ holds.

We now prove that $\bigwedge_{(i \in V)(j \in V)} \{\text{Definitely}(\phi_i \land \phi_j)\}$ implies $\text{Definitely}(\phi)$. The antecedent implies that in each linear extension, for each pair of intervals $X$ and $Y$, $\text{min}(X)$ precedes $\text{max}(Y)$. Hence, given any linear extension, its $\text{max}(\{\text{min}(X_i) \mid i \in V\})$ precedes its $\text{min}(\{\text{max}(Y_j) \mid j \in V\})$ with $\phi$ being true between the two. Hence, $\text{Definitely}(\phi)$ holds. We prove this alternately using time durations. $\text{Definitely}(\phi_i \land \phi_j)$ implies that for every observation of the execution, there exists a time instant when intervals $I_i$ and $I_j$ overlap. Let $I$ be the set of $|V|$ intervals, one per process, that overlap pairwise thus. Assume that $|I|$ is a power of 2. (If it is not, add $2^{\lceil \log |I| \rceil} - |I|$ duplicates of any elements of $I$ to $I$; $I$ can be viewed as a multiset in this part of the proof.) Assign the elements of $I$ to an array $A$ that represents the leaf nodes of a complete binary tree. We use induction on the distance $h$ of nonleaf nodes from the leaf nodes in the tree to show that the invariant on line 4 of the algorithm in Fig. 6 is true.

For the base case iteration $h = 1$, the invariant is true for all $j$ between 1 and $2^{d_{L} - 1}$ because there are two intervals in the set, and we have that each pair of intervals overlaps in time. We assume that the invariant is true for iteration $h = x, x \geq 1$. We now show that the invariant holds for iteration $h = x + 1$. The $2^{x+1}$ intervals in the set $\{A[2^{x+1}, (j - 1)], \ldots, A[2^{x+1}, (j - 1) + 2^{x+1}]\}$ correspond to an internal node in the tree at height $x + 1$ from the leaves. The two child nodes correspond to intervals in the two sets $G_L = \{A[2^{x+1}, (j - 1)], \ldots, A[2^{x+1}, (j - 1) + 2^{x+1}]\}$ and $G_R = \{A[2^{x+1}, (j - 1) + 2^{x+1}], \ldots, A[2^{x+1}, (j - 1)]\}$ of size $2^x$ each. As the invariant holds for height $x$, intervals in each set overlap at some point in time. Let $d_L$ and $d_R$ denote the overlap durations of the intervals in $G_L$ and $G_R$, respectively. If $d_L$ and $d_R$ do not overlap, we have the scenario in Fig. 7 where time is divided into five zones $Z_1$ to $Z_5$. There must exist some interval $I_{L}^{j}$ occurring in $G_L$ and some interval $I_{R}^{j}$ occurring in $G_R$ that do not extend into zone $Z_3$. Hence, intervals $I_{L}^{j}$ and $I_{R}^{j}$ will not overlap in time, a contradiction because each pair of intervals must overlap in time. Thus, $d_L$ and $d_R$ must overlap and the invariant holds. When $h = \log |I|$, all the intervals in $I$ overlap at a common instant in time. This reasoning is true in all observations. Hence, $\text{Definitely}(\phi)$ holds.

Part 2: If $\text{Possibly}(\phi)$, then some observation of the execution includes a state in which $\phi$ holds. Thus, in some linear extension of the execution, the strings...
\[ [\min(X), \max(X)] \text{ and } [\min(Y), \max(Y)] \text{ for each pair of intervals } X \text{ and } Y \text{ overlap. We also have that in some observation of the execution, there must exist an instant in time when } \phi = \bigwedge_i \phi_i \text{ is true. At this instant, the consequent must hold for the predicate decomposed pairwise, i.e., the execution must pass through this instant at which } \bigwedge_{i \in V} (\text{Possibly}(\phi_i \land \phi_j)). \]

We now show that \( \bigwedge_{i \in V} (\text{Possibly}(\phi_i \land \phi_j)) \) implies \( \text{Possibly}(\phi) \). We have that for each pair of processes, there exists a linear extension in which the corresponding interval strings \([\min(X), \max(X)]\) and \([\min(Y), \max(Y)]\) overlap. Hence, (from the properties of linear extensions), it follows that there must also exist some linear extension (of the same execution) in which \( \min(X) \) can precede \( \max(Y) \) for each pair of intervals \( X \) and \( Y \). Analogous to Part (1) for \( \text{Definitely}(\phi) \), in this linear extension, each pair of intervals overlap at some instant in time. Hence, \( \text{Possibly}(\phi) \) holds.

We now have the lemma: “\( \text{Possibly}(\phi) \) holds if and only if \( \bigwedge_{i \in V} (\text{Possibly}(\phi_i \land \phi_j)) \).” The expression for \( \neg(\text{Definitely}(\phi) \land \text{Possibly}(\phi)) \) follows.

\textbf{Part 3:} The third mutually exclusive possibility \( \neg(\text{Possibly}(\phi)) \) must correspond exactly to the remaining cases \( \forall i \in V \left( \exists j \in V \right) \neg(\text{Possibly}(\phi_i \land \phi_j)) \) when considering the intervals pairwise.

\[ \square \]

By Theorem 3, given a predicate \( \phi \) defined on any number of processes, \( \text{Definitely}(\phi) \) and \( \text{Possibly}(\phi) \) can be expressed in terms of the \( \text{Possibly} \) and \( \text{Definitely} \) modalities on predicates defined over pairs of processes. By Theorems 1 and 2, the \( \text{Possibly} \) and \( \text{Definitely} \) modalities on predicates defined over a pair of processes have been refined into the set \( \mathbb{R} \) of 40 orthogonal modalities. Therefore, \( \text{Definitely}(\phi) \) and \( \text{Possibly}(\phi) \), where \( \phi \) is defined over any number of processes, can be expressed in terms of the fine-grained orthogonal set \( \mathbb{R} \) of modalities over predicates defined over various pairs of processes.

As an example, consider \( \phi: a_1 = 2 \land b_1 = 3 \land c_k = 5 \). Let \( a_2, b_2, \) and \( c_k \) be 2, 3, and 5, respectively, in intervals \( X_2, Y_2, \) and \( Z_{2k} \), respectively, and let \( ID(X_1, Y_1), IV(Y_1, Z_{2k}), \) and \( IN(Z_{2k}, X_2) \) be true. Then, by Theorems 1 and 2, we have 1) \( \text{Definitely}(a_1 = 2 \land b_1 = 3) \), 2) \( \text{Possibly}(b_1 = 3 \land c_k = 5) \) and \( \neg(\text{Definitely}(b_1 = 3 \land c_k = 5)) \), and 3) \( \text{Definitely}(a_1 = 2 \land c_k = 5) \). By Theorem 3, we have the modality \( \text{Possibly}(\phi) \land \neg(\text{Definitely}(\phi)) \). Conversely, if \( \text{Possibly}(\phi) \land \neg(\text{Definitely}(\phi)) \) is known in the classical course-grained classification, the fine-grained classification gives the added information: \( ID(X_1, Y_1), IV(Y_1, Z_{2k}), \) and \( IN(Z_{2k}, X_2) \).

Based on the above, the alternate definition (Definition 2) of \( \text{Possibly} \) and \( \text{Definitely} \) for \( |V| = 2 \) processes is enhanced for \( |V| > 2 \) processes. Let \( I \) be a set of intervals, one from each process, such that during these intervals, the local predicates are true.

\textbf{Possibly}(\phi): (For some such set \( I \) of intervals,) there exists a linear extension of \( (E, \prec) \) such that for each pair of intervals \( X \) and \( Y \) in \( I \), the string \([\min(X) - \max(X)]\) overlaps with the string \([\min(Y), \max(Y)]\).
TABLE 4

<table>
<thead>
<tr>
<th>Relation $r$</th>
<th>Expression for $r(X, Y)$</th>
<th>Test for $r(X, Y)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>R1</td>
<td>$\forall x \in X \forall y \in Y_j, x &lt; y$</td>
<td>$T(Y_j)[i] \geq T(X_i)[i]$</td>
</tr>
<tr>
<td>R2</td>
<td>$\forall x \in X_i \exists y \in Y_j, x &lt; y$</td>
<td>$T(Y_j)[i] \geq T(X_i)[i]$</td>
</tr>
<tr>
<td>R3</td>
<td>$\exists x \in X \forall y \in Y_j, x &lt; y$</td>
<td>$T(Y_j)[i] \geq T(X_i)[i]$</td>
</tr>
<tr>
<td>R4</td>
<td>$\exists x \in X \exists y \in Y_j, x &lt; y$</td>
<td>$T(Y_j)[i] \geq T(X_i)[i]$</td>
</tr>
<tr>
<td>S1</td>
<td>$\exists x \in X \forall y \in Y_j, x \not\in y \wedge y \not\in x$</td>
<td>if $T(Y_j)[j] \leq T(X_i)[j] \wedge T(Y_j)[j] \geq T(X_i)[j]$ then $\exists x' \in X_i; T(Y_j)[j] \leq T(x')[j] \wedge T(x')[j] \leq T(Y_j)[j]$ else false</td>
</tr>
<tr>
<td>S2</td>
<td>$\exists x, x2 \in X_i \exists y \in Y_j, x1 &lt; y &lt; x2$</td>
<td>if $T(Y_j)[j] \geq T(X_i)[j] \wedge T(Y_j)[j] &lt; T(X_i)[j]$ then $\exists y' \in Y_j; T(X_i)[j] \leq T(y')[j] \wedge T(y')[j] \leq T(X_i)[j]$ else false</td>
</tr>
</tbody>
</table>

Intervals $X_i$ and $Y_j$ occur at processes $P_i$ and $P_j$, respectively.

timestamps of the start and end events of the intervals; timestamps of events in the intervals are also needed to detect the events $x_i'$ and $y_j'$, respectively. Also note that in the tests for $S1$ and $S2$, the “if” conditions are redundant; however, when $S1$ or $S2$ do not hold, the redundant condition provides the answer using two comparisons only.

When a predicate is defined over $n$ (>$2$) processes, a naive approach to detect Possibly($\phi$) and Definitely($\phi$), in terms of the above tests for a two fine-grained classifications per pair of processes, will require $O(n^2)$ times the overhead for two processes, because $C_n^2$ pairings of processes may have to be tested for. The axioms given in [9] can be used to reduce this overhead. It is a challenging problem to devise algorithms to detect Possibly($\phi$) and Definitely($\phi$) in terms of the fine-grained modalities per pair of processes, that have the same complexity overhead as [3], [6], [7], [8], [15], [18], [19], which simply detect Possibly($\phi$) and Definitely($\phi$). Such algorithms have been devised for conjunctive predicates [1]. The global predicate detection problem is formulated as the Detection of Orthogonal Relations (DOOR) problem as follows: “Given a relation $r_{i,j}$ from $\mathcal{R}$ for each pair of processes $P_i$ and $P_j$, devise an algorithm to identify the intervals, if they exist, one from each process, such that each relation $r_{i,j}$ is satisfied by the ($P_i, P_j$) process pair.”

5 APPLICATIONS

For a large class of applications, such as those involving a pair of interacting agents, a client-server interaction, or industrial process control, predicates are defined on variables local to two processes. The pairwise interaction between processes is a critical mode of communication and information exchange even in many large-scale and peer-to-peer distributed systems. Examples of such systems are sensor networks, ad-hoc mobile networks, mobile agent systems, and online collaborative motion planning and navigation systems. In a sensor network being set up dynamically, the sensors configure themselves by exchanging messages with their neighboring sensors. In the case of ad-hoc mobile networks, pair-wise information exchange with neighbors is used 1) to dynamically compute global functions such as to select routes (AODV), or 2) to dynamically compute time-dependent local functions such as the velocity of the mobile agent during navigation in a collaborative endeavor. In adaptive routing algorithms at the network layer—such as distance vector routing and link state routing—periodic information exchange with neighboring routers is a key feature of computing the global routing function. We note that the paradigm of pair-wise information exchange with each neighbor is common in other contexts also. Dynamic and online computation of local functions are used for centroidal Voronoi tessellations (a Voronoi tessellation whose generating points are the centroids of the corresponding Voronoi diagrams) with applications to problems as diverse as the following [4]—image compression, quadrature, finite difference methods, optimal placement of resources (e.g., mailboxes in a city, or cache servers in the internet), cellular biology, statistics, navigation by animals (e.g., flock of birds or school of fish), and territorial behavior of animals.

Information about the specific interaction type can provide useful information to the application. Consider a specific execution and the interaction types that map to Possibly($\phi$) $\wedge$ Definitely($\phi$) in the dense time model—IB, IC, IG, IH, II, and their inverses (refer to Table 3). Recall that we assume intervals $X$ and $Y$ occur at $P_i$ and $P_j$, respectively, and are denoted by $X_i$ and $Y_j$, respectively. Examples of information useful to the application are the following:

- IB and II imply that even though interval $Y_j$ may possibly have occurred concurrently, $P_i$‘s local state at the end of interval $X_i$ is necessarily visible to the other process $P_j$ before its interval $Y_j$ ends, in all equivalent executions. Therefore, $P_j$ can perform actions in interval $Y_j$ by taking into account $P_i$‘s state at the end of $P_i$‘s interval $X_i$. This is not the case for IC, IG, and IH.
- IG implies that even though the other interval $Y_j$ may have occurred concurrently, in all equivalent executions, neither is $P_i$‘s state at any time during interval $X_i$ visible to the other process $P_j$ at any time during its interval $Y_j$, nor is the other process $P_j$‘s state at any time during its interval $Y_j$ visible to $P_i$ at any time during interval $X_i$. Therefore, there is no mutual knowledge about the other interval. This is not the case for IB, IC, IH, and II.
- IG, IH, and II imply that even though interval $Y_j$ of the other process $P_j$ may have occurred concurrently, in all equivalent executions, the other process
Such information (not just $\text{Possibly}(\phi) \land \neg\text{Definitely}(\phi)$) distinguishes each interaction type from every other, and allows processes to infer about the degree of information transfer that has occurred between the two intervals at different processes. This enables application-specific actions to be taken in and at the end of each interval. A similar analysis can reveal that the fine-grained classification for $\text{Definitely}(\phi)$, as well as for the three coarse-grained categories under the nondense time model under each of the three interpretations, provides useful information to the application. Thus, the fine-grained modality classification provides useful information that is not available under the $\text{Possibly}/\text{Definitely}$ classification.

6 Conclusions

Specifying temporal modalities on predicates in a distributed execution is an important problem. Distributed execution observation has classically used two modalities—$\text{Possibly}(\phi)$ and $\text{Definitely}(\phi)$—for predicate $\phi$. Based on the temporal interactions of intervals, a complete, orthogonal set of relationships $\mathfrak{R}$ between pairs of intervals in a distributed execution had been identified by Kshemkalyani [9]. We leveraged this set of orthogonal interactions between pairs of intervals and mapped this rich, orthogonal classification of modalities of pairwise interval interactions, to the classical coarse-grained classification, $\text{Possibly}(\phi)$ and $\text{Definitely}(\phi)$, for specifying predicates defined on any number of processes. This significantly increases the power of expressing the temporal modalities under which predicates can be specified, beyond the current $\text{Possibly}/\text{Definitely}$ classification. We also gave timestamp-based tests for the orthogonal modalities in the refined classification.

Algorithms to detect the orthogonal relations of the fine-grained classification between pairs of processes, in addition to identifying the $\text{Possibly}$ or $\text{Definitely}$ modality, have recently been devised [1]. Future work would be to explore how the fine-grained orthogonal relations can formalize the information exchange patterns for applications such as those described in Section 5, and to integrate this framework in reasoning about applications.

Acknowledgments

This material is based upon work supported by the US National Science Foundation under Grant No. CCR-9875617.

Figs. 1 and 2 and Tables 1 and 2 (except its italicized prime cover), are reprinted from the Journal of Computer and System Sciences, volume 52, 1996, Kshemkalyani, Temporal Interactions of Intervals in Distributed Systems, pages 287-298, Copyright 1996, with permission from Elsevier Science.

References


Ajay Kshemkalyani received the BTech degree in computer science and engineering from the Indian Institute of Technology, Bombay, in 1987, and he received the PhD degree in computer and information science from The Ohio State University in 1991. His research interests are in computer networks, distributed computing, algorithms, and concurrent systems. Since 2000, he has been as associate professor at the University of Illinois at Chicago. Before that, he spent several years at IBM Research Triangle Park working on various aspects of computer networks. In 1999, he received the US National Science Foundation’s CAREER Award. He is a member of the ACM and a senior member of the IEEE and the IEEE Computer Society.

For more information on this or any other computing topic, please visit our Digital Library at http://computer.org/publications/dlib.