On the Validity of the Global Time Assumption

Ambuj K. Singh
Department of Computer Science
University of California at Santa Barbara
Santa Barbara, California 93106

Abstract
Concurrent execution in distributed systems is usually modeled by a non-deterministic choice, i.e., a concurrent execution that is a partial order on events is equated with the set of total orders obtained from its interleavings. The validity of this interleaving (or global time) assumption is examined in this paper. A novel construction for atomic registers is presented; this construction has the surprising property that it is correct if the proof is based on partial orders but is incorrect if all possible interleavings are confused with partial orders in the reasoning.

1 Introduction
The majority of specification and reasoning in the area of asynchronous distributed systems is carried out by presupposing a total order (i.e., an interleaving on unordered events) on the events in the system. Because of the inherent asynchrony in these systems, this assumption is fictitious and is introduced primarily to simplify the specification and the reasoning about distributed programs. In this paper we discuss the effects of making this assumption on the specification and the verification of distributed programs.

The question of partial ordering versus total ordering has been examined by a number of authors. In a seminal paper [9], Pratt argues eloquently against the assumption of total orders (global time). The main tour de force of his reasoning is that in the presence of indistinguishable events a partial order cannot be equated with all of its linearizations. In another paper [5], Lamport argues against the total order assumption on grounds of atomicty. He introduces two relations on events, precedes (a partial order) and affects and shows that they are appropriate for reasoning about a computation under varying levels of granularity. One of the main contributions of the paper was the presentation of these two relations independent of any notion of global time. In a subsequent paper [1], Ben-David presents a modal semantics based on global time for Lamport's axiomatic theory; this semantics essentially equates a system execution (defined in terms of Lamport's precedes and affects) to a collection of global time interpretations. Based on the results proved in the paper, he advocates the global-time based semantics as a sound basis for specifying and reasoning about asynchronous systems.

In this paper we show the inadequacy of proofs based on total orders by examining the atomic register construction problem [2, 3, 6, 7, 8, 10, 11]. We assume a partial order based specification of atomic registers as presented by Lamport in [5]. We present an implementation for atomic registers that is correct with respect to the above specification. Later, we show that if global time is assumed and each partial order is equated with all its linearizations, then this implementation is no longer correct. The proof is by presenting a linearization of an underlying partial order for which the specification is violated. This example illustrates that specifying and reasoning about distributed systems based on global time has its pitfalls and naively equating partial orders with global time may lead to unforeseen consequences.

In spite of its naturalness and nice algebraic properties, very few algorithms are proved correct using partial orders. A possible reason for the prejudice against partial orders may be that the proofs based on global time can be carried out in the familiar confines of global states. Reasoning with partial orders, on the other hand, cannot be done in terms of global states; it has to be done in terms of local states and events. In this paper, we show that even complicated algorithms can be proved without any additional effort using partial orders.

The rest of the paper is organized as follows. In the next section the specification for 2-reader atomic registers is presented. This is based on Lamport's original specification and is independent of the notion of a global time. In Section 3, a construction for a 2-reader atomic register is defined. This construction is then proved to be correct. In Section 4, this construction is shown to be
incorrect under the assumption of global time. In Section 5, the issue of reasoning in terms of partial orders versus their linearizations is examined. An alternative specification of atomic registers based on global time is also presented. Finally, Section 6 contains concluding remarks.

2 Specification of a 2-reader Atomic Register

A 2-reader atomic register consists of a writer program \( W \) and two reader programs \( R \) and \( S \). The interface between these programs consists of a collection of single-reader atomic registers. Furthermore, each of the programs is "wait-free," i.e., synchronization primitives and busy-wait loops are not used. Next, we define an execution, and an operation.

Definition: An execution of an atomic register construction is an irreflexive partial order precedes (denoted \( \rightarrow \)) on events. Each event corresponds to an execution of an atomic statement from one of the three programs \( W, R, \) or \( S \). Event \( e \) precedes event \( f \) in an execution, i.e., \( e \rightarrow f \), iff at least one of the following three conditions holds:

- \( e \) and \( f \) access (i.e., read or write) the same variable and \( e \) occurs before \( f \), \hspace{1cm} (A0)
- \( e \) and \( f \) are events of the same program and \( e \) occurs before \( f \), or \hspace{1cm} (A1)
- there exists another event \( g \) such that \( e \rightarrow g \) and \( g \rightarrow f \). \hspace{1cm} (A2)

Definition: An operation represents a single execution of one of the programs and consists of the events belonging to that particular execution. Operation \( P \) precedes operation \( Q \) in an execution if each event of \( P \) precedes each event of \( Q \).

Since the relation \( \rightarrow \) on events is an irreflexive partial order, the relation \( \rightarrow \) on operations is also an irreflexive partial order. We denote the \( i \)th operation of the writer \( W \) by \( W^i \). Similarly, the \( i \)th operation of a readers \( R \) and \( S \) are denoted by \( R^i \) and \( S^i \) respectively.

For an atomic register construction to be non-trivial, the following assertion should hold for any two (read or write) operations \( P \) and \( Q \):

\[ \exists e, f : e \in P \land f \in Q : e \rightarrow f \lor f \rightarrow e. \]

Henceforth, we concentrate on non-trivial atomic register constructions. Following [5] and [10], we define the correctness condition for a 2-reader atomic register construction as follows.

Definition: Let \( h \) be any execution of a construction. \( h \) is said to be atomic iff there exists a function \( \phi \) that maps every read operation in \( h \) to some natural number \( i \), where \( W^i \) is a write operation in \( h \), such that the following three conditions hold.

- For each read operation \( P \) in \( h \), the value returned by \( P \) is the same as the value written by \( W^i(P) \). \hspace{1cm} (Integrity)
- For each read operation \( P \) in \( h \), \( \neg(P \rightarrow W^i(P)) \) and \( \neg(W^i(P) \rightarrow P) \). \hspace{1cm} (Safety)
- For any two read operations \( P \) and \( Q \) in \( h \), \( P \rightarrow Q \Rightarrow \phi(P) \leq \phi(Q) \). \hspace{1cm} (Precedence)

Definition: A register construction is correct iff all its executions are atomic.

3 An Implementation of a 2-Reader Atomic Register

The programs for the writer \( W \) and readers \( R \) and \( S \) are given below. The architecture of the construction is shown in Figure 1. The interface between the writer \( W \) and reader \( R \) consists of a register \( WR \) that is written by the writer and read by reader \( R \). Similarly, the interface between the writer \( W \) and reader \( S \) consists of a register \( WS \) that is written by the writer and read by reader \( S \). The interface between the readers consists of a single register \( SR \) that is written by reader \( S \) and read by reader \( R \).

The programs for the writer and the readers are shown in Figure 2. During each write operation, the writer first chooses a new sequence number and writes the tuple consisting of the previous value, the new value, sequence number, and a boolean (set to false) to \( WR \). Next, it writes the new value and the sequence number (also read from the writer) to reader \( R \) through register \( SR \). During each read operation, reader \( R \) first reads the tuple of values from register \( WR \) and the sequence number from register \( SR \). Then, it checks the boolean \( c \) (from the contents of \( WR \)) and compares the sequence numbers. Based on this comparison, it sets local variable \( b \) to true iff \( c \) is true, or if it reads the same sequence number from \( WR \) and \( SR \), or if the sequence number it reads from \( WR \) equals either the sequence number read from register \( SR \) by the previous read operation (this is stored in \( olsseq' \)) or the sequence number read from

283
register \(WR\) by the most recent read operation that returned a new value (this is stored in \(oldseq\)). Reader \(R\) returns the new value (and stores the sequence number read from \(WR\) in \(oldseq\)). Reader \(R\) returns the new value (and stores the sequence number read from \(WR\) in \(oldseq\)) if boolean \(b\) is true; otherwise, the old value is returned. In either case, the sequence number read from \(SR\) is stored in \(oldseq'\).

![Architecture of the Construction](image)

Figure 1: Architecture of the Construction

We prove that our construction is correct by defining a function \(\phi\), and by showing that the defined \(\phi\) meets the three conditions of integrity, safety, and precedence defined in the previous section. The following notations and definitions are used in the proof.

Notation: Let \(P\) be an operation, and \(x\) be any local variable of \(P\). Then, \(P!x\) denotes the final value of variable \(x\) as assigned by operation \(P\).

Let \(P\) be an operation of program \(X\) and \(i\) be a label of a statement in \(X\). Then \(P;i\) denotes the event corresponding to the execution of statement \(i\) in operation \(P\).

Definition: Let \(R^m\) be any read operation of \(R\). Assume that this read operation reads the contents of register \(WR\) from write operation \(W^i\), i.e., \(W^i:1 \rightarrow R^m:0 \rightarrow W^{i+1}:1\). Then, \(\phi(R^m)\) is defined as follows.

\[
\phi(R^m) = \begin{cases} 
   i & \text{if } R^m!b \\
   i - 1 & \text{otherwise}
\end{cases}
\]

Similarly, let \(S^n\) be any read operation of \(S\). Assume that this read operation reads the contents of register \(WR\) from write operation \(W^i\), i.e., \(W^i:2 \rightarrow S^n:0 \rightarrow W^{i+1}:2\). Then, \(\phi(S^n)\) is defined to be \(i\).

It is straightforward to show that the above definition meets the conditions of safety and integrity. We concentrate on proving precedence. For this, we have to show that for any two read operations \(P\) and \(Q\), if \(P \rightarrow Q\) then \(\phi(P) \leq \phi(Q)\). Because there are two readers, we consider four cases. The proof obligations for these four cases are stated below.

1. \(\phi(R^m) \leq \phi(R^{m+1})\),
2. \(\phi(S^n) \leq \phi(S^{n+1})\),
3. If \(R^m \rightarrow S^n\) then \(\phi(R^m) \leq \phi(S^n)\), and
4. If \(S^n \rightarrow R^m\) then \(\phi(S^n) \leq \phi(R^m)\).

These four proof obligations are met next.

Proof of Case 1:
Assume that read operations \(R^m\) and \(R^{m+1}\) read the contents of register \(WR\) from write operations \(W^i\) and \(W^j\) respectively, i.e.,

\[
W^i:1 \rightarrow R^m:0 \rightarrow W^{i+1}:1, \text{ and}
W^j:1 \rightarrow R^{m+1}:0 \rightarrow W^{j+1}:1.
\]

Now, observe the following.

\[
W^i:1 \rightarrow R^m:0
\]
respectively, i.e.,
\[ W_i:1 \rightarrow R_{m+1}:0 \]
\[ W_i:1 \rightarrow R_{m+1}:0 \]
\[ \{ R_{m+1}:0 \rightarrow W_{j+1}:1 \} \]
\[ W_i:1 \rightarrow W_{j+1}:1 \]
\[ \{ \rightarrow \text{ is acyclic} \}
\[ i \leq j \]
\[ \{ \text{arithmetic} \}
\[ i \leq j - 1 \lor i = j \]
\[ \{ \text{by definition, } \phi(R_{m}) \leq i \text{ and } j - 1 \leq \phi(R_{m+1}) } \]
\[ \phi(R_{m}) \leq \phi(R_{m+1}) \lor (i = j \land \phi(R_{m}) = i) \]

So, the remaining proof obligation is to show that
\[ (i = j \land \phi(R_{m}) = i) \Rightarrow i \leq \phi(R_{m+1}). \]

For this observe the following proof.

\[ \phi(R_{m}) = i \]
\[ \Rightarrow \{ \text{definition of } \phi \}
\[ R_{m+1}^b \]
\[ \Rightarrow \{ \text{program of reader } R \}
\[ R_{m+1}^b \text{oldseq} = R_{m+1} \text{seq} \]
\[ \Rightarrow \{ W_i:1 \rightarrow R_{m}:0 \rightarrow W_{j+1}:1 \} \]
\[ R_{m+1}^b \text{oldseq} = W_{j+1} \text{seq} \]
\[ \Rightarrow \{ W_j:1 \rightarrow R_{m+1}:0 \rightarrow W_{j+1}:1 \text{ and } i = j \}
\[ R_{m+1}^b \text{oldseq} = R_{m+1}^b \text{seq} \]
\[ \Rightarrow \{ \text{program of reader } R \}
\[ R_{m+1}^b \]
\[ \Rightarrow \{ \text{definition of } \phi \}
\[ \phi(R_{m+1}) = i \]

(End of Case 1)

Proof of Case 2:

Assume that read operations \( S^m \) and \( S^{n+1} \) read the contents of register \( WR \) from write operations \( W^i \) and \( W^j \) respectively, i.e.,
\[ W_i:2 \rightarrow S^n:0 \rightarrow W_{j+1}:2 \text{ and } W_i:2 \rightarrow S^{n+1}:0 \rightarrow W_{j+1}:2. \]

Now, observe the following.

\[ W_i:2 \rightarrow S^n:0 \]
\[ \Rightarrow \{ S^n:0 \rightarrow S^{n+1}:0 \} \]
\[ W_i:2 \rightarrow S^{n+1}:0 \]
\[ \Rightarrow \{ S^{n+1}:0 \rightarrow W_{j+1}:2 \} \]
\[ W_i:2 \rightarrow W_{j+1}:2 \]
\[ \{ \rightarrow \text{ is acyclic} \}
\[ i \leq j \]
\[ \{ \text{by definition, } \phi(S^n) = i \text{ and } \phi(S^{n+1}) = j \}
\[ \phi(S^n) \leq \phi(S^{n+1}) \]

(End of Case 2)

Proof of Case 3:

Assume that read operations \( R^m \) and \( S^n \) read the contents of register \( WR \) from write operations \( W^i \) and \( W^j \) respectively, i.e.,
\[ W^i:1 \rightarrow R^m:0 \rightarrow W_{j+1}:1, \text{ and } W^i:2 \rightarrow S^n:0 \rightarrow W_{j+1}:2. \]

Now, observe the following.

\[ W^i:1 \rightarrow R^m:0 \]
\[ \Rightarrow \{ W^i:1 \rightarrow R^m:0 \text{ or } R^m:0 \rightarrow W^i:3 \} \]
\[ W^i:3 \rightarrow R^m:0 \lor W^i:1 \rightarrow R^m:0 \rightarrow W^i:3 \]
\[ W^i:3 \rightarrow R^m:0 \lor (W^i:1 \rightarrow R^m:0 \rightarrow W^i:3 \land R^m:0 \rightarrow S^n:0) \]
\[ W^i:3 \rightarrow W^i:1 \lor (S^n:0 \rightarrow W_{j+1}:2) \]
\[ W^i:3 \rightarrow W^i:1 \lor (W^i:1 \rightarrow R^m:0 \rightarrow W^i:3 \land R^m:0 \rightarrow W_{j+1}:2) \]
\[ \Rightarrow \{ W^i:2 \rightarrow W^i:3 \text{ and } W_{j+1}:2 \rightarrow W_{j+2}:1 \} \]
\[ W^i:3 \rightarrow W_{j+1}:2 \lor (W^i:1 \rightarrow R^m:0 \rightarrow W^i:3 \land W^i:1 \rightarrow W_{j+2}:1) \]
\[ \Rightarrow \{ \rightarrow \text{ is acyclic} \}
\[ i \leq j \lor (W^i:1 \rightarrow R^m:0 \rightarrow W^i:3 \land i \leq j + 1) \]
\[ \Rightarrow \{ \text{predicate calculation} \}
\[ i \leq j \lor (i = j + 1 \land W^i:1 \rightarrow R^m:0 \rightarrow W^i:3) \]
\[ \Rightarrow \{ \text{by definition, } \phi(R^m) \leq i \text{ and } \phi(S^n) = j \}
\[ \phi(R^m) \leq \phi(S^n) \lor (i = j + 1 \land W^i:1 \rightarrow R^m:0 \rightarrow W^i:3) \]

So, the remaining proof obligation is to show that
\[ (i = j + 1 \land W^i:1 \rightarrow R^m:0 \rightarrow W^i:3) \Rightarrow \phi(R_{m}) \leq \phi(S^n). \]

Because \( \phi(S^n) = j \) and \( \neg\phi(R_{m}) \lor \phi(R_{m}) = i - 1, \) this will follow if we show that
\[ (i = j + 1 \land W^i:1 \rightarrow R^m:0 \rightarrow W^i:3) \Rightarrow \neg\phi(R_{m}). \]

Define predicate \( P(m) \) as follows:
\[ P(m) \equiv R^m \rightarrow S^n \land W^i:1 \rightarrow R^m:0 \rightarrow W_{j+1}:1 \land W_{j+1}:2 \rightarrow S^n:0 \rightarrow W_{j+2}:2. \]

Thus, our proof obligation is to show that
\[ P(m) \Rightarrow \neg\phi(R_{m}) \lor \phi(R_{m}). \]

We prove below that
\[ P(m) \Rightarrow \neg\phi(R_{m}) \lor \exists k < m :: R_k^b \land P(k). \]

The desired proof then follows by induction on \( m \) in the above property.

Boolean \( b \) is assigned by the following statement in the program of reader \( R \):
\[ b := c \lor (seq = seq') \lor (seq = oldseq') \lor (seq = oldseq). \]

We prove property \( I \) by considering the four disjuncts in the right hand side separately.

Because \( W^i:1 \rightarrow R^m:0 \rightarrow W^i:3 \) and the writer disjuncts \( c \) to false in \( W^i:1 \), \( R_{m}^b \) is not set to true on account of the first disjunct.

For the second disjunct, assume that reader \( R_{m} \) reads register \( SR \) from read operation \( S^k \), i.e.,
\[ S^k:1 \rightarrow R^m:1 \rightarrow S^{k+1}:1. \]

Also, assume that read operation \( S^k \) reads register \( WS \) from write operation \( W^i \), i.e.,
\[ W^i:2 \rightarrow S^k:0 \rightarrow W^{i+1}:2. \]

Then observe the following.
\[ W^i:2 \rightarrow S^k:0. \]
Therefore, \( R^m b \) is not set to true on account of the second disjunct.

For the third disjunct, assume that \( R^{m-1} \) reads register \( SR \) from read operation \( S^k \), i.e.,
\[
S^k : 1 \rightarrow R^{m-1} : 1 \rightarrow S^k+1 : 1.
\]

Also, assume that read operation \( S^k \) reads register \( WS \) from write operation \( W^i \), i.e.,
\[
W^i : 2 \rightarrow S^k : 0 \rightarrow W^i+1 : 2.
\]

Then observe the following.
\[
W^i : 2 \rightarrow S^k : 0 \rightarrow S^k+1 : 1
\]
\[
W^i : 2 \rightarrow S^k+1 : 1
\]
\[
W^i : 2 \rightarrow R^{m-1} : 1
\]
\[
W^i : 2 \rightarrow S^m : 0
\]
\[
W^i : 2 \rightarrow W^i+1 : 2
\]
\[
\{ \text{is acyclic}\}
\]
\[
l < i
\]
\[
\{ W^i : 2 \rightarrow S^k : 0 \rightarrow W^i+1 : 2 \}
\]
\[
l < i \land W^i seq = S^k seq
\]
\[
\{ S^k : 1 \rightarrow R^{m-1} : 1 \rightarrow S^k+1 : 1 \}
\]
\[
l < i \land W^i seq = R^m \text{ seq}
\]
\[
\{ \text{writer chooses unique sequence numbers} \}
\]
\[
W^i \text{ seq} \neq R^m \text{ seq}
\]
\[
\{ W^i : 1 \rightarrow R^m : 0 \rightarrow W^i+1 : 1 \}
\]
\[
R^m \text{ seq} \neq R^m \text{ seq}
\]

Therefore, \( R^m b \) is not set to true on account of the third disjunct.

For the fourth disjunct, assume that \( R^m \) reads value \( \text{oldseq} \) from read operation \( R^k \) where \( k < m \). Furthermore, assume that \( R^k \) reads register \( WR \) from write operation \( W^i \), i.e.,
\[
W^i : 1 \rightarrow R^k : 0 \rightarrow W^i+1 : 1
\]
Then observe the following.
\[
R^k : 0 \rightarrow R^m : 0
\]
\[
\{ W^i : 1 \rightarrow R^k : 0 \text{ and } R^m : 0 \rightarrow W^i : 3 \}
\]
\[
W^i : 1 \rightarrow R^k : 0 \rightarrow W^i+1 : 3
\]
\[
\{ W^i : 3 \rightarrow W^i+1 : 1 \}
\]
\[
W^i : 1 \rightarrow W^i+1 : 1 \land R^k : 0 \rightarrow W^i+1 : 3
\]
\[
\{ \text{is acyclic}\}
\]
\[
l \leq i \land R^k : 0 \rightarrow W^i+1 : 3
\]
\[
\{ \text{predicate calculus} \}
\]
\[
l < i \lor (l = i \land R^k : 0 \rightarrow W^i+1 : 3)
\]
\[
\{ W^i : 1 \rightarrow R^k : 0 \rightarrow W^i+1 : 3 \}
\]
\[
(l < i \land R^k \text{ seq} = W^i \text{ seq}) \lor
\]
\[
(W^i : 1 \rightarrow R^k : 0 \rightarrow W^i+1 : 3)
\]
\[
\{ \text{writer chooses distinct sequence numbers} \}
\]
\[
(W^i \text{ seq} \neq W^i \text{ seq} \land R^k \text{ seq} = W^i \text{ seq}) \lor
\]
\[
(W^i : 1 \rightarrow R^k : 0 \rightarrow W^i+1 : 3)
\]
\[
\{ k < m \land R^m = S^m \}
\]
\[
R^k \text{ seq} \neq W^i \text{ seq} \lor
\]
\[
(W^i : 1 \rightarrow R^k : 0 \rightarrow W^i+1 : 3) \land R^k : 0 \rightarrow S^m
\]
\[
\{ W^i : 1 \rightarrow R^m : 0 \rightarrow W^i+1 : 1 \}
\]
\[
R^m \text{ seq} \neq R^m \text{ seq} \lor
\]
\[
(W^i : 1 \rightarrow R^k : 0 \rightarrow W^i+1 : 3 \land R^k \rightarrow S^m)
\]
\[
\{ R^m \text{ reads } \text{oldseq} \text{ from } R^k \}
\]
\[
\neg (R^m b) \lor (W^i : 1 \rightarrow R^k : 0 \rightarrow W^i+1 : 3 \land R^k \rightarrow S^m)
\]
\[
\{ \text{by assumption, } R^k b \}
\]
\[
\neg (R^m b) \lor (R^k b \land P(k))
\]

The desired proof of property \( I \) follows from noting that \( k < m \) in the above proof. This completes the proof of the third case.

**(End of Case 3)**

**Proof of Case 4:**
We first prove the following proposition.

**Proposition:** For any two read operations \( R^m \) and \( S^m \),
\[
S^m \rightarrow R^m \equiv S^m+1 : 1 \rightarrow R^{m-1} : 1 \lor
\]
\[
(3e : S^m : 2 \rightarrow W^k : 2 \land W^k : 3 \rightarrow R^m : 0)
\]

**Proof:**
\[
S^m \rightarrow R^m
\]
\[
\{ \text{definition of } \rightarrow \}
\]
\[
S^m : 2 \rightarrow R^m : 0
\]
\[
\{ S^m : 2 \text{ and } R^m : 0 \text{ do not satisfy axioms } A_0, A_1 \}
\]
\[
(3e :: S^m : 2 \rightarrow e \rightarrow R^m : 0)
\]
\[
= \{ \text{there does not exist an event } e \text{ such that} \}
\]
\[
S^m : 2 \rightarrow e \rightarrow R^m : 0 \text{ holds from axioms } A_0, A_1 \}
\]
\[
(3e, f :: S^m : 2 \rightarrow e \rightarrow f \rightarrow R^m : 0)
\]
\[
= \{ \text{considering all possible choices of } e \text{ and } f \}
Therefore, assume that read operation \( R^m \) reads register \( WR \) from read operation \( W' \) and \( W^j \) respectively, i.e.,
\[
W'_i:2 \rightarrow R^m:0 \rightarrow W'^{i+1}:1, \quad \text{and} \quad W^j:2 \rightarrow S^0:0 \rightarrow W^j+1:2.
\]

Now, observe the following.
\[
W^j:2 \rightarrow S^m:0
\]
\[
\Rightarrow \{ \text{since } S^m \rightarrow R^m, S^0:0 \rightarrow R^m:0 \}
\]
\[
W^j:2 \rightarrow R^m:0
\]
\[
\Rightarrow \{ R^m:0 \rightarrow W^{i+1}:1 \}
\]
\[
W^j:2 \rightarrow W^{i+1}:1
\]
\[
\Rightarrow \{ W^{i+1}:1 \rightarrow W^{i+1}:2 \}
\]
\[
W^j:2 \rightarrow W^{i+1}:2
\]
\[
\Rightarrow \{ \text{acyclic} \}
\]
\[
j \leq i
\]
\[
\Rightarrow \{ \text{arithmetic} \}
\]
\[
j \leq i - 1 \lor i = j
\]
\[
\Rightarrow \{ \text{by definition, } \phi(S^m) = j \text{ and } i - 1 \leq \phi(R^m) \}
\]
\[
\phi(S^m) \leq \phi(R^m) \lor i = j
\]

So, the remaining proof obligation is to show that
\[
i = j \Rightarrow \phi(S^m) \leq \phi(R^m).
\]

Therefore, assume that \( i = j \) in the remainder of the proof. By the proposition proved earlier,
\[
S^m \rightarrow R^m \equiv S^{m+1}:1 \rightarrow R^{m-1}:1 \lor
\]
\[
(\exists k :: S^k:2 \rightarrow W^k:2 \land W^k:3 \rightarrow R^m:0)
\]

Next, we show that the second disjunct cannot be true if \( i = j \).
\[
(3k :: S^k:2 \rightarrow W^k:2 \land W^k:3 \rightarrow R^m:0)
\]
\[
\Rightarrow \{ S^k:0 \rightarrow S^m:2 \text{ and } R^m:0 \rightarrow W^{i+1}:1 \}
\]
\[
(3k :: S^k:0 \rightarrow W^k:2 \land W^k:3 \rightarrow W^{i+1}:1)
\]
\[
\Rightarrow \{ W^j:2 \rightarrow S^0:0 \text{ and } W^{i+1}:1 \rightarrow W^{i+1}:3 \}
\]
\[
(3k :: W^j:2 \rightarrow W^k:2 \land W^k:3 \rightarrow W^{i+1}:3)
\]
\[
\Rightarrow \{ \text{acyclic} \}
\]
\[
(3k :: j < k \land k \leq i)
\]
\[
\Rightarrow \{ \text{arithmetic} \}
\]
\[
j < i
\]
\[
\Rightarrow \{ i = j \text{ by assumption} \}
\]
\[
\text{false}
\]

Therefore, we consider only the first disjunct from the above equivalence and show that
\[
(i = j \land S^{m+1}:1 \rightarrow R^{m-1}:1) \Rightarrow \phi(S^m) \leq \phi(R^m).
\]

Because \( \phi(S^m) = j \), this will follow if we show that
\[
(i = j \land S^{m+1}:1 \rightarrow R^{m-1}:1) \Rightarrow \phi(R^m) = i.
\]

This proof obligation is met next. Assume that read operation \( R^{m-1} \) reads register \( SR \) from read operation \( S^{n+k} \), i.e.,
\[
S^{n+k}:1 \rightarrow R^{m-1}:1 \rightarrow S^{n+k+1}:1.
\]

Because \( S^{n+1}:1 \rightarrow R^{m-1}:1, k \geq 1 \). Furthermore, assume that read operation \( S^{n+k} \) reads register \( WS \) from write operation \( W' \), i.e.,
\[
W' : 2 \rightarrow S^{n+k}:0 \rightarrow W'^{i+1}:2.
\]

Now, observe the following.
\[
W'^i:2 \rightarrow S^{n+k}:0 \land W'^{i+1}:2 \rightarrow S^{n+k+1}:0
\]
\[
W'^i:2 \rightarrow S^{n+k}:0 \land W'^{i+1}:2 \rightarrow S^{n+k+1}:1
\]
\[
W'^i:2 \rightarrow W'^{i+1}:2 \land W'^{i+1}:1 \rightarrow W'^{i+2}:1
\]
\[
W'^i:2 \rightarrow W'^{i+1}:2 \land W'^{i+2}:1 \rightarrow W'^{i+3}:1
\]
\[
W'^i:2 \rightarrow W'^{i+2}:1 \land W'^{i+3}:1 \rightarrow W'^{i+4}:1
\]
\[
W'^i:2 \rightarrow W'^{i+2}:1 \land W'^{i+3}:1 \rightarrow W'^{i+4}:1
\]
\[
\Rightarrow \{ \text{acyclic} \}
\]
\[
j \leq i \land j \leq i
\]
\[
\Rightarrow \{ i = j \}
\]
\[
\text{false}
\]

This completes the proof of all the four required cases and hence the proof of correctness of our construction.

4 A Counterexample

Let us define a history to be any linearization of an execution and denote the ordering of events in a history by the symbol \(<\). We show that if we confuse executions with histories in the specification of atomic registers presented in Section 2 then the construction presented in Section 3 is no longer correct. The proof proceeds by exhibiting a counterexample.

Consider the following history \( h \) consisting of operations \( W^0, W^1, W^2, S^0, S^1, \) and \( R^0 \).

- Initially all the local sequence variables are 0.
Write operation \( W^0 \) chooses sequence number 1 and writes a new value 5 into the atomic register and completes; it precedes all other operations.

Write operations \( W^1 \) and \( W^2 \) choose sequence numbers 2 and 3 and write new values 10 and 15 respectively into the atomic register; they interleave with the read operations \( S^0, S^1 \), and \( R^0 \) as follows:

\[
\begin{align*}
W^1:0 &< W^1:1 < W^1:2 < S^0:0 < S^0:1 < \\
S^0:2 &< R^0:0 < W^1:3 < W^1:4 < W^2:0 < \\
W^2:1 &< W^2:2 < S^1:0 < S^1:1 < R^0:1 < \\
R^0:2 &< R^0:3 < R^0:4.
\end{align*}
\]

In history \( h \), read operation \( S^0 \) reads a value 10 and a sequence number 2 from \( WS \). Consequently, it writes a sequence number 2 into the register \( SR \) and returns 10 as the value read. Read operation \( S^1 \) reads a value 15 and a sequence number 3 from \( WS \). Consequently, it writes a sequence number 3 into the register \( SR \) and returns 15 as the value read. Read operation \( R^0 \) reads the tuple \((5, 10, 2, false)\) from \( WR \), sequence number 3 from \( SR \). Because \( oldseq^1 \) and \( oldseq \) are both 0 initially, \( R^0 \) evaluates \( h \) to be false and returns 5 as the value read. Observe from the constructed history that \( S^0 < R^0 \) and \( S^0 \) returns a value more recent than the \( R^0 \). This violates the precedence condition in specification of atomic registers if we confuse \( \rightarrow \) with \( < \) in the specification presented in Section 2.

Notice that this counterexample is based on an ordering \( S^0 < R^0 \) that arises only because of the assumption of global time. If the precedence of events is determined solely by the three axioms \( A0, A1, A2 \) presented in Section 2, operation \( S^0 \) does not precede operation \( R^0 \) (as the ordering \( S^0:2 < R^0:0 \) is introduced only because of the linearization) and the presented construction works correctly.

5 Relating \( \rightarrow \) and \( < \)

In this section we explore the relationship between \( \rightarrow \) and \( < \). Consider a system execution consisting of a set of events \( E \) and a partial order \( \rightarrow \) on \( E \). Let \( e, e', f, f' \) range over the events and \( h, h' \) range over the set of histories (i.e., linearizations) of the system execution. Let \( <^h \) denote the total order on events implied by history \( h \). Then, it is easily seen that

\( e \rightarrow f \equiv (\forall h :: e <^h f) \). \hspace{1cm} (P6)

The following two properties follow as a result of the above assertion.

\( e \rightarrow f \wedge e' \rightarrow f' \equiv (\forall h :: e <^h f \wedge e' <^h f') \). \hspace{1cm} (P1)

\( e \rightarrow f \vee e' \rightarrow f' \Rightarrow (\forall h :: e <^h f \vee e' <^h f') \). \hspace{1cm} (P2)

However, because universal quantification does not distribute over disjunction, the converse of property \( (P2) \) does not hold.

Let \( P, Q, R, S \) be operations, i.e., sets of events. Then, the following properties follow as a consequence of the above definitions.

\( P \rightarrow Q \equiv (\forall h :: P <^h Q) \). \hspace{1cm} (P3)

\( P \rightarrow Q \wedge R \rightarrow S \equiv (\forall h :: P <^h Q \wedge R <^h S) \). \hspace{1cm} (P4)

\( P \rightarrow Q \vee R \rightarrow S \Rightarrow (\forall h :: P <^h Q \vee R <^h S) \). \hspace{1cm} (P5)

(As before, the converse of this property does not hold.)

\( \neg(P \rightarrow Q) \equiv (\exists h :: \neg(P <^h Q)) \). \hspace{1cm} (P6)

On account of property \( (P6) \), the specification of atomic registers (from Section 2) can be restated in terms of histories as follows.

**Definition:** Let \( h \) be any history of a construction. \( h \) is said to be atomic iff there exists a function \( \phi \) that maps every read operation in \( h \) to some natural number \( i \), where \( W^i \) is a write operation in \( h \), such that the following three conditions hold. (In all the quantifications below, \( k' \) ranges over histories that have the same underlying partial order as \( h \).)

- For each read operation \( P \) in \( h \), the value returned by \( P \) is the same as the value written by \( W^{\phi(P)} \).
- For each read operation \( P \) in \( h \),
  \[ (\exists h' :: (P <^h W^{\phi(P)})) \wedge (\exists h' :: (W^{\phi(P)+1} <^h P)) \]. \hspace{1cm} (Safety)
- For any two read operations \( P \) and \( Q \) in \( h \),
  \[ (\exists h' :: P <^h Q \Rightarrow \phi(P) \leq \phi(Q)) \]. \hspace{1cm} (Precedence)

**Definition:** A register construction is correct iff all its histories are atomic.

At this time it may be worthwhile to examine the counterexample history \( h \) presented in Section 5. The underlying partial order for \( h \) is as follows.

\( W^0 \rightarrow W^1 \rightarrow W^2, \quad S^0 \rightarrow S^1, \quad W^1:1 \rightarrow R^0:0 \rightarrow W^1:3, \quad W^0:0 \rightarrow W^1:2, \quad S^0:1 \rightarrow R^0:3 \rightarrow W^1:3, \quad S^0:2 \rightarrow R^0:3 \rightarrow W^1:3. \)
The following history $h'$ has the same underlying partial order as $h$.

$W^1:0 < W^1:1 < W^1:2 < S^0:0 < S^0:1 < R^0:0 < S^0:2 < W^1:3 < W^1:4 < W^2:0 < W^2:1 < W^2:2 < S^1:0 < S^1:1 < R^0:1 < R^0:2 < R^0:3 < R^0:4$.

In history $h'$, $R^0:0 < S^0:2$ and therefore, the condition $S^0 < R^0$ no longer holds. Consequently, the property of precedence is no longer violated according to the specification of atomic registers presented above.

6 Discussion

The relationship between $\rightarrow$ and $<$ has also been examined by others. Ben-David takes a model theoretic approach and defines two kinds of formulas — distributed and transferable in [1]. He presents a syntactic characterization of the two classes along the lines of properties (P0) — (P4). Katz and Peled have also examined the relationship between $\rightarrow$ and $<$ [4]. They propose the idea of an interleaving set and argue that a protocol should be deemed correct if there exists a history from each interleaving set that meets the specification.

In this paper, we have presented a construction with the interesting property that it is correct if reasoning is carried out with partial orders but is not correct if reasoning is carried out with total orders and $\rightarrow$ is confused with $<$. Thus, though the global time assumption simplifies reasoning, naively equating $\rightarrow$ with $<$ may lead to unexpected results. This suggests that we should be careful about the global time assumption, especially when specifying and reasoning about distributed programs. Finally, it may be worth pointing out that some specifications of atomic registers are based on global time and consequently, the construction presented here would be incorrect (wrongly so!) with respect to these specifications.

References


