MASSACHUSETTS INSTITUTE OF TECHNOLOGY
ARTIFICIAL INTELLIGENCE LABORATORY

Working Paper 189 June 1979

GLOBAL TIME IN ACTOR COMPUTATIONS

Will Clinger

This research was supported by a National Science Foundation Graduate Fellowship in mathematics.

GLOBAL TIME IN ACTOR COMPUTATIONS

Will Clinger

SECTION 0
Preview

The actor model of parallel computation rejects the notions of global time and global states, using instead local times and local states. Still, some notion of global time is essential to any model of parallel computation. The actor model avoids explicit use of global time by stipulating ordering laws, which are axioms supposed to hold true for all physically realizable computations; these laws were introduced by Hewitt and Baker [HeBa1,2], though they had been used implicitly by Greif [Gr]. Deliberately introducing global time into the model makes clear the intuitive justification for these ordering laws.

As a bonus, this approach reveals the logical relationships between the ordering laws. There are three major ordering laws, which together are equivalent to a single axiom of realizability in global time. All other ordering laws follow from these three. These three laws are strictly stronger than the corollaries stated in [HeBa1,2].

For example, Hewitt and Baker originally conjectured that their most powerful ordering law followed from their other ordering laws together with their laws of locality (which are abstract counterparts of scoping rules). Counterexamples were found by the author and by Berzins [Be]. Both counterexamples were Zeno machines which did an infinite amount of computation in a finite amount of time. The counterexamples suggested the importance of global time even in the actor model.

Section 1 briefly introduces the actors model of parallel computation as message-passing. Section 2 introduces the notion of a global time interpretation for an actors computation, and derives informally a strong and a weak form of the global time realizability axiom. Section 3 uses these axioms to give a formal definition, sufficient for developing the ordering laws, of an actor computation. Section 4 presents the ordering laws as consequences, proving the equivalences between the realizability axioms and the ordering laws, and stating the independence of many laws. Section 5 mentions the use of these results in giving a formal semantics for actor-based languages.

This research has eliminated one of the ordering laws, the Law of Finite Immediate Successors (LFIS), which did not seem as intuitive as the others. This law was used in proving the main theorem of [HeBa1]. The appendix shows how LFIS may be eliminated from that proof without damaging the theorem.

SECTION 1
Introduction to Actors

The actor model is perhaps best motivated by the prospect of highly parallel computing machines consisting of dozens or even hundreds of independent monoprocessors, each with its own local memory and communications processor, communicating via a packet-switched network in a system similar to some of today's distributed computer networks [Ha], [He2].

The model uses partial orders of computation events to represent concurrency [Gr], [HeBa2]. Each such order is composed of a treelike activation order and a set of total arrival orderings, one for each actor. This decomposition distinguishes the Actor model from other models using partial orders.

Each event represents the arrival of a message sent to a target actor, which may be thought of as a program running on one of the monoprocessors. In the simplest case, each processor holds only one actor, but in general a number of actors will be inside a single processor. Each actor has its own arrival ordering, which is total; this means that actors receive messages one at a time. Thus the communications processors must arbitrate among incoming messages, storing messages in a local queue until their host is ready to accept them. This arbitration is assumed to be fair.

All communication between actors is through these messages. This holds even for actors on the same monoprocessor, since parameter passing can be viewed in this light [He1].

If the arrival of a message at an actor causes that actor to send out a message itself, the first event is said to activate the arrival of the second message at its target. Chains of such activations define the arrival ordering. Not every event has an activator (else there could be no First Cause); those events which do not are called external events, because their activator is external to the system being modelled.

There may be many external events in a single computation, even infinitely many if the computation is itself infinite. The graph of the activation ordering will have as many components as there are external events, and each component will be a tree: although no event has more than one activator, an event can activate more than one event. This is because the target actor can transmit several messages as a result of it before processing any more messages itself. In fact, an event can cause its target to go into an infinite sending loop.

In summary, an actor computation has three sets of primitive objects, namely events, actors, and messages. Write the set of events as $\mathbf{E}$, the set of actors as $\mathbf{A}$, and the set of messages as $\mathbf{M}$; then the primitive material of the model is $<\mathbf{E}, \mathbf{A}, \mathbf{M}>$. Associated with each event is both a target actor and a message, so there are functions T: $\mathbf{E} \dashrightarrow \mathbf{A}$ giving the target and M: $\mathbf{E} \dashrightarrow \mathbf{M}$ giving the message of an event. (The sender need not be specified because that information is available from the activation ordering unless the event is external.) Each actor has its associated arrival ordering, so let $\mathbf{Arr}$ be a collection of strict total orderings $\text{-arr}_A\text{->}$ defined on $T^{-1}(A)$, for $A \in \mathbf{A}$. (A partial order $<$ is strict if $x < y$ implies $x \neq y$.) There is also the activation ordering -act->, a strict partial order on $\mathbf{E}$ such that no event has more than one immediate predecessor.

This defines a structure <**E**, **A**, **M**, T, M, -act->, **Arr**>. Hewitt and Baker introduced laws which hold for all actor computations and are based on commonly held notions about what counts as a digital computation [HeBa1,2]. The ordering laws are those general facts readily expressible in terms of the substructure <**E**, **A**, T, -act->, **Arr**>. These are the only laws treated in this paper. (The full structure is used to define actor semantics. The appendix uses messages, for example.) In addition, locality laws have been stated for actors [HeBa1,2].

To state the ordering laws it is necessary to define the combined ordering --> as the transitive closure of the activation and arrival orderings. Naturally any statement mentioning the combined ordering, or using the sign -->, is talking about the interaction between the arrival orderings and the activation ordering.

SECTION 2
Intuitions about Time, Causality, and Computation; The Fundamental Axioms on Actor Orderings

The combined ordering is the natural ordering for encoding time. If one event E_1 precedes another event E_2 in the combined ordering (i.e. E_1 --> E_2), then E_1 happens before E_2 in time. Why is that? The definition of --> says that there exists a path of causation and local time from E_1 to E_2, but if that is so then E_1 must occur before E_2 in global time. The combined ordering is not just a global time ordering, though, because there are incomparable events; what that means is that, although presumably one of them happened first in global time, the Actors model of the computation doesn't say which one. The combined ordering codes those aspects of time sequence which are important, but suppresses many accidental details of time order which can have no effect on the outcome of the computation.

Commonly held notions about time and computation constrain the structures possible for the combined ordering. Although these notions are standard, and can be found in some form in any model of computation, their incorporation in the actor model is interesting because the model eschews explicit mention of global time or global states.

The mathematical notion of global time appropriate for event-structured models of computation is of a function from the computation events into the real numbers (or integers, as it will turn out). For Actors, then, a global time is a mapping GT: **E** --> **R**, where **R** denotes the real numbers.

One constraint on this mapping is that cause precedes effect. Thus

[1] GT preserves the strict ordering -act->.

That is, if E_1 -act-> E_2, then $GT(E_1) < GT(E_2)$.

Another constraint is that global time be consistent with all local times. Thus

[2] GT preserves all the strict orderings -arr$_A$->, for $A \in \mathbf{A}$.

Consequently

[3] GT preserves the combined ordering -->.

and

[4] --> is strict.

[3] and [4] are provably equivalent to [1] and [2]. [4] is a fundamental law first stated in [HeBa1], the

Law of Strict Causality: The combined ordering --> is a strict partial ordering.

"Zeno machines" are paradoxical machines which can do infinitely many things in a finite amount of time. An example is Huffman's Lamp, which when switched on lights for only 30 seconds before turning itself off for 15 seconds, and then comes back on for 7 1/2 seconds before turning off for 3 3/4 seconds, and so on. After one minute, is it on or off? Zeno machines such as the counterexample in [Be] are ruled out by requiring that

[5] the range of GT has no accumulation points.

Equivalently, no bounded interval in $\mathbb{R}$ contains infinitely many range points of GT. Equivalently, a GT can be found that is integer valued and one-one (because --> is strict).

The last intuition, that computations begin, has a slightly different status than the others. Many interesting theorems, such as the theorem quoted at the end of the appendix, can be proved without it, and so knowing whether or not it is assumed provides useful information. For example, many properties of a computer network which has been operating continuously for years will in no way depend upon there having been a time before the system was brought up, and so any proof which made use of that fact would be suspect. On the other hand, if the assumption really is necessary to the proof, then that tells something about the property being proved, namely that it depends upon the existence of some initial state.

Together with [5] above, the following implies that there is a "first" event, and thus that the computation has a definite beginning:

[6] the range of GT is a subset of the nonnegative real numbers.

Putting the above constraints together yields a fundamental axiom on actor orderings, which comes in both a strong and a weak form corresponding to whether or not the computation is assumed to have a beginning. That is, combining [3], [4] and [5] above yields the

Weak Axiom of Realizability: There exists a one-to-one mapping GT from the events $\mathbf{E}$ into the real numbers $\mathbf{R}$ which preserves the combined ordering --> and such that $GT^{-1}(I)$ is finite for every bounded interval I of $\mathbf{R}$.
Equivalently there exists a one-to-one mapping GT: $\mathbf{E}$ --> $\mathbf{Z}$ which preserves -->, where $\mathbf{Z}$ is the set of integers.

while adding [6] yields the

(Strong) Axiom of Realizability: There exists a one-to-one mapping GT from the events $\mathbf{E}$ into the nonnegative reals which preserves the combined ordering --> and such that $GT^{-1}(I)$ is finite for every bounded interval I of $\mathbf{R}$.
Equivalently there exists a one-to-one mapping GT: $\mathbf{E}$ --> $\mathbf{N}$ which preserves -->, where $\mathbf{N}$ is the set of natural numbers.

As shown in Section 4, the Actor ordering laws follow from the definitions in Section 1 together with one of the versions of the realizability axiom. (Not all the ordering laws given in [HeBa1,2] so follow, however; see the end of section 4 and the appendix for an argument that those two which do not should not be considered completely general.)

SECTION 3
Formal Definition of an Abstract Computation

This section consists of a formal definition of an abstract Actors computation, sufficient for developing the ordering laws. (That is, this definition omits messages.)

An actor computation (or message-passing pattern) is a structure

- <E, A, T, -act->, Arr>

such that

E is a set (the set of events)

A is a set (the set of actors)

T is a function: E --> A (the target function)

-act-> (the activation ordering)is a strict partial ordering on E such that no event has more than one immediate predecessor (this means -act-> defines a set of trees). Formally (where quantification is over E)

$$\begin{aligned}&\forall E\ \neg\ E \text{ -act-> } E\\ &\quad \wedge\ \forall E_1, E_2\ (E_1 \text{ -act-> } E \wedge E_2 \text{ -act-> } E\\ &\qquad \wedge\ \neg\exists E'\ (E_1 \text{ -act-> } E' \text{ -act-> } E \vee E_2 \text{ -act-> } E' \text{ -act-> } E))\\ &\quad \supset E_1 = E_2.\end{aligned}$$

Arr is a set of strict total orderings -arr_A-> defined on $T^{-1}(A)$, for $A \in$ A (the arrival orderings)

and the following statement holds, where the combined ordering --> is the transitive closure of ∪({-act->} ∪ Arr):

(Strong) Axiom of Realizability: There exists a one-to-one mapping GT from the events E into the nonnegative reals which preserves the combined ordering --> and such that $GT^{-1}(I)$ is finite for every bounded interval I of R.
Equivalently there exists a one-to-one mapping from the events E into the natural numbers which preserves -->.

(END OF definition.)

When computations with infinite histories are allowed, the (Strong) Axiom of Global Time Realizability is replaced by the

Weak Axiom of Realizability: There exists a one-to-one mapping GT from the events $\mathbf{E}$ into the real numbers $\mathbf{R}$ which preserves the combined ordering --> and such that $GT^{-1}(I)$ is finite for every bounded interval I of $\mathbf{R}$.
Equivalently there exists a one-to-one mapping from the events into the integers which preserves -->.

Notice that the global time GT is not a part of the structure; only its mathematical existence is asserted. Thus, although a particular abstract Actor computation must be realizable in time, no time sequencing is associated with it except the combined ordering. Furthermore, as shown by the two main theorems of the next section, the realizability axioms are equivalent to certain simple ordering laws, so that the set of actor computations may be defined without explicitly mentioning global time at all.

It appears that in practice the global time itself is seldom needed. The mere possibility of one is quite constraining, implying as it does the laws given in the next section, and those laws are usually more convenient for proofs. Once they and the equivalence results have been derived, then, the realizability axioms will have fulfilled their main purpose. Nonetheless, the realizability axioms show how Actors may be related to a global time should the need ever arise, and occasionally that is a useful proof technique.

SECTION 4
Laws on Actor Orderings

This section is a unified treatment of the actor ordering laws, as introduced by Hewitt and Baker in [HeBa1,2]. These laws state restrictions placed on computations by the requirement that they be physically realizable. All the ordering laws in this section follow from the definitions and an axiom from Section 3; in their turn, the definitions were motivated by the actor model in section 1, and the axioms were argued on an intuitive basis in section 2.

The first law follows from either the weak or the strong realizability axiom. It is the

Law of Strict Causality (LSC): For no $E \in \mathbf{E}$ does E --> E.

An extremely important law, provable as well in the systems of [HeBa1,2], is the

Law of Countability (LC): There are at most countably many events. That is, $\mathbf{E}$ is countable (where a finite set is considered countable).

When the Strong Axiom of Realizability is assumed, the intuition that events are only finitely removed from the beginning of computation comes back out as the

Law of Finite Predecessors in the Combined Ordering (LFP): For all events E_1 the set $\{E \mid E \dashrightarrow E_1\}$ is finite.

These three laws are in fact equivalent to the strong Axiom of Realizability; as a result actor computations may be formalized without ever mentioning global time.

Theorem: The Axiom of Strong Realizability of Actor Orderings is equivalent to the conjunction of the Law of Strict Causality, the Law of Countability, and the Law of Finite Predecessors in the Combined Ordering.

Proof: The realizability axiom is easily seen to imply all three (LSC+LC+LFP).

Let $\{E_1, E_2, E_3, \ldots\}$ be the set of events. Define GT inductively as follows.

Let $GT(E_1) = 1$.

Suppose that GT has been defined on $\{E_1, \ldots, E_{n-1}\}$ in such a way that it preserves the combined ordering --> on the events on which it is defined. That is, $GT(E_i) < GT(E_j)$ whenever $E_i \dashrightarrow E_j$, for $i, j < n$. The strategy for defining $GT(E_n)$ will be to place it as far to the right as possible. Precisely, if there exists a $j < n$ such that $E_n \dashrightarrow E_j$, then let k be such that $GT(E_k) = \min \{GT(E_j) \mid E_n \dashrightarrow E_j, j < n\}$. Define

$$GT(E_n) = 1/2\ (GT(E_k) + \max (\{GT(E_j) \mid GT(E_j) < GT(E_k), j < n\} \cup \{0\})),$$

so that $GT(E_k)$ is the first point on the right of $GT(E_n)$. The claim is that GT is now defined on $\{E_1, \ldots, E_{n-1}, E_n\}$ is such a way as to preserve the combined ordering. If not, then, by the induction hypothesis and the fact that $GT(E_n) < GT(E_j)$ whenever $E_n \dashrightarrow E_j$, $j < n$, we must have some particular $i < n$ such that $E_i \dashrightarrow E_n$ but $GT(E_n) < GT(E_i)$. This implies also that $GT(E_k) < GT(E_i)$. Now since $E_n \dashrightarrow E_k$, the transitivity of the combined ordering gives $E_i \dashrightarrow E_k$, which by LSC contradicts the fact that GT preserves --> on $\{E_1, \ldots, E_{n-1}\}$. Thus no such i can exist, and we have succeeded in extending GT to $\{E_1, \ldots, E_{n-1}, E_n\}$ while still preserving the combined ordering.

If there is no j such that $E_n \dashrightarrow E_j$, then just put $GT(E_n)$ out to the right of all other points defined so far, say $GT(E_n) = 1 + \max \{GT(E_j) \mid j < n\}$. As before, the combined ordering is preserved.

By induction the combined ordering is preserved at all stages. Any non-preservation of that ordering in the whole function GT would already have arisen at some finite stage, and so we have inductively defined a one-to-one positive-valued function GT which preserves the combined ordering. It only remains to show that its range has no limit points; this is equivalent to showing that the left-open unit intervals with integral endpoints, that is intervals of the form $(m, m + 1]$ for m a natural number, each contain only finitely many points of the range.

If $(m, m + 1]$ contains any range points at all, then, by the way GT is defined, $m + 1 = GT(E_n)$ for some n, and the interval $(m, m + 1]$ contains none of the points $GT(E_1), \ldots, GT(E_{n-1})$; that is, the interval was empty when $GT(E_n)$ was defined. I claim that the pre-images of all range points placed in that interval after $GT(E_n)$ precede E_n in the combined ordering. Whenever GT(E) is defined to be a non-integer, E precedes the pre-image

of the range point immediately to its right at the time of definition; thus the pre-image of the first range point placed in (m, m + 1] after GT(E_n) precedes E_n in the combined ordering; the second does also, by transitivity of --> if needed; and so on for all the range points placed in the interval. If GT takes infinitely many values in the interval (m, m + 1], then, there must be infinitely many events which precede E_n in the combined ordering, contradicting the Law of Finite Predecessors in the Combined Ordering.
[]

(By assuming in addition the existence of a single initial event which precedes all other events, and that no event has infinitely many immediate successors in the activation ordering, Hewitt and Baker were able to prove that the Law of Finite Activity (given below) implied a statement equivalent to the Strong Axiom of Realizability [HeBa2]; under their assumptions the Law of Countability and the Law of Finite Predecessors in the Combined Ordering hold, so they had a greatly weakened version of the "if" part of the above theorem.)

The Law of Finite Predecessors in the Combined Ordering has two immediate corollaries concerning the primitive orderings, but taken together they remain weaker than LFP itself.

<u>Law of Finite Predecessors in the Activation Ordering</u>: For all events E_1 the set {E | E -act-> E_1} is finite.

<u>Law of Finite Predecessors in an Arrival Ordering</u>: For all events E_1 and actors A the set {E | E -arr_A-> E_1} is finite. (Of course the set is empty if $T(E_1) \neq A$.)

<u>Theorem</u>: The Axiom of Strong Realizability of Actor Orderings is stronger than the conjunction of

1) All the laws in this section minus the Law of Strict Causality.
2) All the laws in this section minus the Law of Countability.
3) All the laws in this section minus the Law of Finite Predecessors in the Combined Ordering.

Proof: 1) and 2) are easy. For 3), consider an infinite backward chain (with order type that of the negative integers) in the combined ordering consisting of alternating arrival and activation ordering links, where each arrival ordering link is taken from a different arrival ordering.
[]

The following law was called the Law of Finitely Many Events Between Two Events in the Combined Ordering in [HeBa1], where it was mentioned that it implies the existence of time functions. Specifically, when taken together with the Law of Countability it is equivalent to the Axiom of Weak Realizability, as will be proved in a moment. It is the

Law of Finite Activity (LFA): For all events E_1 and E_2, the set $\{E \mid E_1 \text{-->} E \text{-->} E_2\}$ is finite.

As is stated in [HeBa1], it can be proved using Konig's Lemma and the

Law of Finite Chains Between Events in the Combined Ordering: There are no infinite chains of events between two events in the strict partial ordering -->. (A chain is just a totally ordered set.)

Theorem: Assume the Law of Strict Causality. Then the Law of Finite Activity is equivalent to the Law of Finite Chains Between Events in the Combined Ordering.

Proof: Clearly LFA implies the Law of Finite Chains Between Events in the Combined Ordering.

To prove the converse, assume there are no infinite chains between two events in the combined ordering. Then by the totality of the arrival orderings, every event has either no predecessors at all in the arrival ordering of its target, or it has a unique immediate predecessor. Since no event has more than one immediate predecessor in the activation ordering, then, no event has more than two immediate predecessors in the combined ordering.

Now suppose that for some E_1 and E_2 the set $\{E \mid E_1 \text{-->} E \text{-->} E_2\}$ is infinite. We will inductively construct an infinite chain, contrary to hypothesis. Let $e_0 = E_2$.

We have a sequence $e_0, \ldots, e_n$ such that $E_1 \text{-->} e_n \text{-->} e_{n-1} \text{-->} \cdots \text{-->} e_0$ and $\{E \mid E_1 \text{-->} E \text{-->} e_n\}$ is infinite. If e_n is not an initial event, let e denote the unique immediate predecessor of e_n in the activation ordering, and if e_n is not the first event in the arrival ordering for $T(e_n)$, let e' denote the unique immediate predecessor of e_n in the arrival ordering for $T(e_n)$. If e exists and $\{E \mid E_1 \text{-->} E \text{-->} e\}$ is infinite, then define $e_{n+1} = e$. Otherwise e' exists and $\{E \mid E_1 \text{-->} E \text{-->} e'\}$ is infinite, so define $e_{n+1} = e'$.
[]

This proof is essentially the proof of Konig's Lemma for ordered trees, and does not assume an axiom of choice [Sm]. Thus the two laws may be interchanged freely. Usually the Law of Finite Chains in the Combined Ordering will be easier to prove, and the Law of Finite Activity will seem stronger in use.

Here is the second major equivalence theorem, as promised. Even when computations are not assumed to have beginnings it is still possible to formalize actor computations without explicitly mentioning global time:

Theorem: The Weak Axiom of Realizability is equivalent to the conjunction of the Law of Strict Causality, the Law of Countability, and the Law of Finite Activity. (Equivalently, to the first two and the Law of Finite Chains Between Events in the Combined Ordering.)

Proof: The weak axiom clearly implies LSC+LC+LFA.

Let $E_0, E_1, E_2, \ldots$ be the events. Define a global time GT inductively as follows.

Define $GT(E_0) = 0$.

The induction hypothesis for n, IH(n), is the following: $GT(E_0), \ldots, GT(E_{n-1})$ have been defined so that

1) GT is one-one. I.e. $\forall i, j,\ 0 \le i, j \le n - 1,\ GT(E_i) = GT(E_j) \supset i = j$.
2) GT is integer valued.
3) the combined ordering is preserved. I.e. $\forall i, j,\ 0 \le i, j \le n - 1$, $E_i \dashrightarrow E_j \supset GT(E_i) < GT(E_j)$.
4) GT is already defined on all E_k lying between any two of $E_0, \ldots, E_{n-1}$ in the combined ordering. I.e. $\forall i, j, k\ \ 0 \le i, j \le n - 1 \ \wedge E_i \dashrightarrow E_k \dashrightarrow E_j \ \supset\ 0 \le k \le n - 1$.

Clearly 4) will be impossible to arrange without periodically re-ordering the E_i's, and we must be careful in that re-ordering not to upset the main induction.

Assume IH(n). There are two cases, depending on whether or not E_n is related by --> to any of $E_0, \ldots, E_{n-1}$. In the simple case, when E_n is not related, define $GT(E_n) = 1 + \max\ \{GT(E_i) \mid 0 \le i \le n - 1\}$. Clearly 1), 2), and 3) of IH(n + 1) hold. Also 4) holds because --> is transitive and E_n is unrelated to $E_0, \ldots, E_{n-1}$.

Now the hard case, where E_n is related to at least one of $E_0, \ldots, E_{n-1}$. By 4) of IH(n), either E_n precedes all those it is related to, or it follows all those it is related to. Let us say E_n follows all of $E_0, \ldots, E_{n-1}$ that it is related to, since the other possibility is handled in exactly the same fashion. (That is, with arrows reversed, $1 + \max\ \{GT(E_i) \mid 0 \le i \le n - 1\}$ replaced by $\min\ \{GT(E_i) \mid 0 \le i \le n - 1\} - 1$, et cetera.)

If there does not exist a E_k such that $k > n$ and, for some i, $0 \le i \le n - 1$, $E_i \dashrightarrow E_k \dashrightarrow E_n$ is true, then define $GT(E_n) = 1 + \max\ \{GT(E_i) \mid 0 \le i \le n - 1\}$. IH(n + 1) then clearly holds. Otherwise we must re-order $\{E_i \mid i \ge n\}$.

Let $\{E_{k_1}, \ldots, E_{k_m}\} = \cup_{i=1}^{n-1} \{E_k \mid k > n \text{ and } E_i \dashrightarrow E_k \dashrightarrow E_n\}$; the finiteness of this set is guaranteed by LFA. We may assume $k_1 < k_2 < \cdots k_m$. We re-order the set $\{E_i \mid n \le i \le k_m\}$ by pulling $E_{k_1}, \ldots, E_{k_m}, E_n$ out of it and placing them in front, so that the new order looks like

$$E_{k_1}, E_{k_2}, \ldots, E_{k_m}, E_n, E_{n+1}, \ldots, E_{k_1-1}, E_{k_1+1}, \ldots, E_{k_m-1}$$

and relabel as

$$E'_n, E'_{n+1}, \ldots, E'_{n-1+m}, E'_{n+m}, E'_{n+m+1}, \ldots, E'_{k_1-1+m}, E'_{k_1+1+(m-1)}, \ldots, E'_{k_m}.$$

What has been accomplished by this re-ordering? First of all, nothing has been ruined by it; GT is still defined in the same way on the same events, and IH(n) still holds. Some points are now farther back (at most m events farther back) in the new ordering, but if GT were to be defined on $E_{k_1}, \ldots, E_{k_m}$ and E_n (newly labelled $E'_n, \ldots, E'_{n+m-1}, E'_{n+m}$) without further relabelling of the E'_i, $i \ge n + m$, then every event E'_i would be at least one event closer to being defined than in the original labelling. And I claim it is possible to define GT on $E'_n, \ldots, E'_{n+m-1}, E'_{n+m}$ while maintaining the induction hypothesis and without disturbing E'_i, $i \ge n + m$.

Proof of claim: IH(n) still holds, so try again to define GT on the nth event but this time use the new ordering, i.e. define $GT(E'_n)$. Relabelling may again be necessary, but no E'_i with $i \ge n + m$ will be relabelled: $E_j \dashrightarrow E'_i \dashrightarrow E'_n$ for some j, $0 \le j \le n - 1$, would imply

$E_j \dashrightarrow E'_i \dashrightarrow E_n$ since $E'_n \dashrightarrow E_n$, contradicting $E'_i \notin \{E_{k_1}, \ldots, E_{k_m}\}$. In fact, several relabellings may be necessary before GT becomes defined on an nth event, but these relabellings can only affect the order of $E'_n, \ldots, E'_{n+m-1}$; each relabelling changes the labels on a smaller initial segment of $\{E'_i \mid i \geq n\}$, and so finally E'_n becomes such that no E'_i, $i > n$, lies between it and any of $E_0, \ldots, E_{n-1}$ in the combined ordering. At that point GT becomes defined on its nth event. Furthermore GT will be defined on all of $E'_n, \ldots, E'_{n+m-1}, E'_{n+m}$ before it is necessary to disturb the labelling above n+m, by the same reductio ad absurdum as above. Thus the claim.

For each event E_i, therefore, $GT(E_i)$ is eventually defined. GT is a one-one integer-valued function which preserves -->, since any non-preservation would show up at a finite stage contrary to the induction. Hence AWR is satisfied.
[]

It is not difficult to see the first two parts of the following

Theorem: The Weak Axiom of Realizability is stronger than the conjunction of

1) All the laws in this section minus the Law of Strict Causality.
2) All the laws in this section minus the Law of Countability.
3) All the laws in this section minus the Law of Finite Activity, the Law of Finite Chains Between Events in the Combined Ordering, and the Law of Finite Predecessors.

The third follows from (for example) the counterexamples found by the author and by Berzins [Be].

The following facts about the primitive orderings follow immediately from either of LFA or the Law of Finite Chains Between Events in the Combined Ordering.

Law of Finite Activation Chains Between Two Events: If C is a chain of events in the activation ordering from E_1 to E_2, then C is finite.

Law of Finite Chains Between Events in an Arrival Ordering: For all events E_1 and E_2 such that $T(E_1) = T(E_2) = A$, $\{E \mid E_1 \text{ -arr}_A\text{->} E \text{ -arr}_A\text{->} E_2\}$ is finite.

The counterexample in [Be] shows that the Law of Finite Chains Between Events in the Combined Ordering, and thus the Law of Finite Activity, is stronger than these last two laws taken together, and indeed remains stronger even in the presence of the locality laws of [HeBa1,2].

The Law of Finite Predecessors in the Combined Ordering is stronger than the Law of Finite Activity, and hence than the Law of Finite Chains Between Events in the Combined Ordering. The Law of Finite Predecessors in the Activation Ordering is stronger than the Law of Finite Chains Between Events in the Activation Ordering, and the Law of Finite Predecessors in an Arrival Ordering is stronger than the Law of Finite Chains Between Events in an Arrival Ordering. Once again, the counterexample in [Be] shows that the Law of Finite Predecessors in the Combined Ordering is stronger than the conjunction of the two corresponding laws on the primitive orderings.

Two of the ordering laws stated in [HeBa1] are definitions in the system of this paper. They are the Law of Uniqueness of Immediate Predecessors in the Activation Ordering, and the Arrival Ordering Law which states that two events having a common target are comparable under the arrival ordering for the target.

There are only two ordering laws of [HeBa1,2] not yet accounted for. One is the existence of only one initial event, which was a simplification assumed only in [HeBa2]. The other is the Law of Finite Immediate Successors in the Activation Ordering (LFIS). This law is not a general law in the same sense as the others, for reasons given in the appendix. In the actor-based language ETHER, for example, it does not hold [Ko].

SECTION 5
Conclusions

To assert the existence of a global time, as the realizability axiom does, is to place necessary constraints on the ordering structure of an actor computation. The global time need not be given explicitly, and in fact the realizability axiom may be replaced by simple statements about the cardinality of certain sets easily definable from the actor orderings. Thus global time need not appear at all in the actor model.

The actor ordering laws justify actor induction [HeBa2]. Induction over partial orders is simpler than induction over sets of total orders obtained by "linearizing" the partial orders in all possible ways (that is, considering all global times). Hence the ordering laws have unusual simplicity and elegance which makes them very convenient for proving properties of programs.

This research has given a precise definition of actor computations and a characterization of the ordering laws. These are needed to develop the mathematical semantics of actor-based programming languages [Cl].

I wish to thank Carl Hewitt for his encouragement, for explaining the actor model to me, and for his willingness to argue over its fine points. Bruce Schatz read an earlier draft of this paper, giving extensive comments which made the current draft much more readable.

APPENDIX
Modifying a Proof by Hewitt and Baker

The following descriptively named law appears in [HeBa1,2], [Ba] but not in the present paper:

Law of Finite Immediate Successors in the Activation Ordering (LFIS): No event has infinitely many immediate successors in the activation ordering.

Some of its effects are to prohibit pattern-directed invocation and to assume that all actors are primitive in the sense of loop-free -- that is, they contain no internal loops; this is explicit in [Ba]. Two or more actors may still form loops among themselves, but the looping behavior will be manifest in their patterns of communication and hence will be embodied in the abstract actor computation. LFIS thus may provide a convenient means for "cutting all loops" for purposes of proving correctness or defining the semantics of a programming language. It is not, however, a general law in the same sense as the laws presented in sections 2, 3, and 4.

Furthermore, it cannot be assumed without ruling out an actor semantics for languages using pattern directed invocation such as ETHER [Ko].

One purpose of this paper has been to remove LFIS from proofs of properties of the ordering laws. This appendix shows how to eliminate it from a proof that actors behaving like mathematical functions compute functions that are least fixed points of certain continuous functionals [HeBa1].

Some notation and definitions from [HeBa1] will be needed to show how to modify the proof. We will say that an event E with target T(E) and message M(E) is of the form

[T(E) <~~ M(E)].

Messages have structure; they are represented in some data language. We will need messages of two sorts, corresponding to the two kinds of events we need, requests and replies. A request is an event of the form

[f <~~ [request: x, reply-to: c]]

which represents passing an argument x to the actor f, with instruction to send any result to a continuation actor c. An event of the form

[c <~~ [reply: y]]

represents the arrival of a result y at the continuation actor c. By convention, replies are responses to previous request events. Formally (adapted from [HeBa1]):

Definition: If an event E_1 is of the form [. . . <~~ [request: . . ., reply-to: c]], and E_2 is of the form [c <~~ [reply: . . .]], and E_1 -act-> E_2, and for no event E of the same form as E_2 is E_1 -act-> E -act-> E_2 true, then E_2 is said to be a reply to E_1.

A request event may have no replies, one reply, nineteen replies, or infinitely many replies. For a request event whose target is an actor which behaves as a procedure (defined in [HeBa1]), however, there is at most one reply, by definition.

For an event E define $E\text{--}\geq \equiv \{E\} \cup \{E' \mid E \text{ --> } E'\}$ and $\text{--}\geq E \equiv \{E\} \cup \{E' \mid E' \text{ --> } E\}$.

Definition: If E is a request event then the activity corresponding to E is $E\text{--}\geq \cap (\cup \{\text{--}\geq E' \mid E'$ is a reply to $E\})$.

Perhaps not all events in the activity corresponding to E actually contribute to answering the request E, but we can be sure that all events which do contribute are in the activity. An activity may not be finite, because a request can have infinitely many replies; if a request has only finitely many replies, though, as is the case if its target is a procedure, then its activity is guaranteed to be finite by the Law of Finite Activity.

Definition: If E and E' are events, E --> E', and there is some activity α such that E, E' $\in \alpha$, then we say E -cont-> E'.

Although -cont-> is called the continuation ordering, it is not in general a true ordering because it may not be transitive. It is transitive when restricted to activities corresponding to requests of a procedure, because by definition the activities of a procedure are properly nested. Note that -cont-> is a subrelation of -->.

An actor which is a procedure and initiates the same activity (in the sense of the same messages with the same targets and the same relationships between events) whenever it is sent the same request is said to behave like a function [HeBa1].

The definition of an immediate f-descendant in the first version of [HeBa1] contained a small but subtle error which was partially corrected in subsequent versions. The idea is that the immediate f-descendants of <x y> $\in$ graph(f) are those <x' y'> $\in$ graph(f) which must be known in order to compute f(x) without recursing. As is often the case, the proof is correct because it depends on what the definition is supposed to be, not its formal specification. The definition below is supposed to be what the definition was supposed to be.

Definition: Suppose an actor f behaves like a mathematical function and that <x y> $\in$ graph(f) and <x' y'> $\in$ graph(f). Then <x' y'> will be said to be an immediate f-descendant of <x y> if there is some history of f which has events E_1 and E_2 of the form

E_1: [f <~~ [request: x, reply-to: . . .]]

E_2: [f <~~ [request: x', reply-to: . . .]]

such that E_2 belongs to the activity initiated by E_1 (so that E_1 -cont-> E_2) and it is not the case that there is an event E of the form

E: [f <~~ [request: . . ., reply-to: . . .]]

such that E_1 -cont-> E -cont-> E_2.

Definition: Suppose that <x y> $\in$ graph(f). Then

immediate-descendants$_f$(<x y>) $\equiv$

{<x' y'> | <x' y'> is an immediate f-descendant of <x y>}.

As an example, Hewitt and Baker give the following procedure.

```
(fib n) ≡
        (if
            (n = 1) then 1
            (n = 2) then 1
            (n > 2) then ((fib (n - 1) + (fib (n - 2))))
```

immediate-descendants$_{fib}$(<1 1>) = { }
immediate-descendants$_{fib}$(<2 1>) = { }
immediate-descendants$_{fib}$(<3 2>) = {<1 1> <2 1>}
immediate-descendants$_{fib}$(<5 5>) = {<3 2> <4 3>}

Now the only real use Hewitt and Baker make of LFIS is in proving the following lemma.

Lemma: If an actor f behaves like a mathematical function and <x y> $\in$ graph(f) then immediate-descendants$_f$(<x y>) is finite.

Proof using the Law of Finite Activity (instead of the Law of Finite Immediate Successors in the Activation Ordering):

Let E_1 be a request for the procedure f to compute the value f(x); that is E_1 is of the form

E_1: [f <~~ [request: x, reply-to: . . .].

By the way Hewitt and Baker define "function" there can be at most one reply to this request [HeBa1]; there is a reply, since <x y> $\in$ graph(f), so call it E_3. Since E_1 has a unique reply E_3, the activity initiated by E_1 is just $\{E_1, E_3\} \cup \{E_2 \mid E_1 \text{-->} E_2 \text{-->} E_3\}$. This set is finite by the Law of Finite Activity, and so immediate-descendants$_f$(<x y>) is finite by the definition.
[]

As this lemma is the only place in the proof where Hewitt and Baker use the Law of Finite Immediate Successors in the Activation Ordering, the theorem to be stated below no longer depends upon that hypothesis.

Definition: If G is a set of input-output pairs then
$D_f(G) \equiv$ {<x y> | <x y> $\in$ graph(f) and immediate-descendants$_f$(<x y>) $\subset$ G}.

Theorem (Hewitt and Baker): If an actor f behaves like a mathematical function then D_f is a continuous functional in the sense of Scott and graph(f) is the limit of D_f beginning with the empty graph. Also graph(f) is the minimal fixed point of D_f.

REFERENCES

[Ba] Henry Baker. Actor Systems for Real-Time Computation. MIT LCS Technical Report 197, March 1978.

[Be] Valdis Berzins. An Independence Result For Actor Laws. Computation Structures Group Note 34, MIT LCS, December 1977.

[Cl] Will Clinger. A Fair Power Domain for Actor Computations. MIT AI working paper, June 1979.

[Gr] Irene Greif. Semantics of Communicating Parallel Processes. MAC TR-154, MIT LCS, September 1975.

[Ha] Robert Halstead. Multiprocessor Implementations of Message-passing Systems. MIT LCS Technical Report 198, February 1978.

[He1] Carl Hewitt. Viewing Control Structures as Patterns of Passing Messages. MIT AI Memo 410, December 1976. Also in Artificial Intelligence, Vol 8, Nr 3, June 1977, p 323-364.

[He2] Carl Hewitt. Preliminary Design of the APIARY for VLSI Support of Knowledge-Based Systems. MIT AI working paper, April 1979.

[HeBa1] Carl Hewitt and Henry Baker. Actors and Continuous Functionals. MIT AI Memo 436, July 1977. Also in Proceedings of the IFIP Working Conference on the Formal Descriptions of Programming Concepts, August 1977, 16.1-16.21. The AI Memo has been distributed in at least two versions; some references to [HeBa1] in this paper refer only to a particular version.

[HeBa2] Carl Hewitt and Henry Baker. Laws for Communicating Parallel Processes. IFIP 77, p 987-992.

[Ko] Bill Kornfield. ETHER -- A Parallel Problem Solving System. To appear in IJCAI-79.

[Sc] Dana Scott. Outline of a Mathematical Theory of Computation. Proceedings of the Fourth Annual Princeton Conference on Information Sciences and Systems, 1970, 169-176.

[Sm] Raymond Smullyan. First Order Logic. Springer-Verlag, New York, 1968.