Relating Computation and Time

Mathai Joseph, Asis Goswami*
University of Warwick

Abstract

In this paper, we examine the relation between the actions of a computation and the nature of time associated with the execution. Using a method suggested by Whitrow, we first define an *instant* of time in terms of computational actions and then show that the set of such instants is isomorphic with the reals provided it satisfies certain well-known properties. Some of these properties can be derived from a straightforward definition of actions and their ordering, but we also provide necessary and sufficient conditions for proving the *denseness* of the set. We then show how different assumptions about computations relate to the requirements for denseness. From this, it can be seen that the apparent choice between representing computational time by a discrete domain, an arbitrary dense domain and the real domain, for example, is really a choice between different computational models.

Keywords: time, denseness, computation

1 Introduction

The relation between time and computation has attracted some recent attention, largely through work on defining semantic models for real-time programs. However, the nature of this relation is of wider interest because of its effect on computational models in general.

Consider the execution of a distributed program in which each process can read a 'local' value of time, represented as a real number. Let the local time of each process be measured in the same units, start with value 0 when the process begins execution, and have the usual properties of increasing monotonically and uniformly.

*Address for correspondence: Department of Computer Science, University of Warwick, Coventry CV4 7AL. This work was supported by research grant GR/D 73881 from the Science and Engineering Research Council.

In such a program, $Clock$ is an alarm-clock-like process which loops waiting for its local time to equal the successive values taken by its variable $Tick$. Assume that the time taken to execute a command is suitably small when compared to the waiting time of this process.

$$Clock :: Tick := 1; *[when\ Tick = Localtime_1\ do\ Tick := Tick + 1]$$

X is a similar process, but the waiting time of its loop is halved for each successive iteration.

$$\begin{aligned} X ::\ & Halftime := 1; Inc := Halftime/2; \\ & *[when\ Halftime = Localtime_2\ do\ Halftime, Inc := Halftime + Inc, Inc/2] \end{aligned}$$

In the parallel composition $Clock \| X$, what is the value of $Halftime$ when $Tick = 2$? Will the value of $Tick$ *ever* become 2 ?

These two questions (and others of this nature) illustrate some of the hazards of using the mathematical properties of a continuous time domain to hide the natural limitations in arbitrarily dividing "real" time. If we make the common assumption that independent actions in different processes in a distributed program can occur arbitrarily close together in time, then although computations are 'discrete', the time of their execution must be represented as some dense domain. But it must also be assumed that there is a lower bound to the execution time of any instruction, for if this is not the case, in the program above $Halftime$ will approach but never become 2 (rather like a computational equivalent of Zeno's paradox of Achilles and the tortoise). On the other hand, if there *is* a fixed lower bound to the execution time of any command, it can be assumed without much loss of precision that the execution times of all commands are multiples of this (or some other) amount and the time can be expressed in natural numbers; in this case, the value of Inc will eventually become smaller than the lower bound. How then is a choice made between representing time by a dense or a discrete domain ?

Quite independently of such considerations of time, Hewitt and Baker [4] discussed this problem and Best [3] pointed out that without some restriction to ensure finiteness, a 'Zeno machine' can be used to solve the Halting problem. With a slightly different objective, the temporal logic model proposed by Barringer, Kuiper and Pnueli [2] contains an axiom of Finite Variability by which only finitely many computations can take place in an finite interval (this axiom enables them to distinguish between finite and infinite stuttering). The spirit of this axiom has been followed in some timed semantic models (e.g. Reed and Roscoe [8]) which represent time using the reals and, while placing no lower bound on the closeness in time of independent actions in different processes, limit a process to performing a finite number of operations in a finite interval. However, the choice of a dense order for time is by no means universal and Milner [7], for example, has argued that by appropriately choosing the level of granularity, a discrete domain can be used for time. And even where a dense domain has been chosen, there is no consensus about whether this should be the reals, the rationals or an arbitrary dense domain.

In the world represented by the laws of physics, time is continuous and can thus be represented by the reals. The question then is how to relate discrete computational actions, which by their nature have granularity, with well-established physical theories of continuous time. However, this problem is not unique to computation and Whitrow [11] describes how from the work of Russell [9] and Walker [10] it is possible to obtain a continuous time domain from an appropriate definition of instants of time. A crucial aspect of this work is that time is defined *in terms of* actions, not conversely; this contrasts with the view so far taken in computational models, where actions are laid out in some order over a pre-ordained domain of time.

In this note, we show how a real domain of time can be derived from a definition of computational actions. We make use of the account given by Whitrow [11] and define *instants* of time in terms of computational actions. The set T of such instants can be shown to be isomorphic with the reals provided it satisfies certain well-known properties. Some properties can be derived from a straightforward definition of actions and their ordering and we also provide necessary and sufficient conditions for proving the *denseness* of T. We then show how different assumptions about computations relate to the requirements for denseness. From this, it can be seen that the apparent choice between discrete time, dense time and real time, for example, is really a choice between different computational models.

2 A Model of Computation

Assume that an execution of a program consists of a set of *actions*; an action can be considered as the execution of a terminating command in a program and so some actions may be performed sequentially and some in parallel. Let ACT be the set of all actions.

An action a *precedes* another action b ($a \prec b$) if b does not start before a terminates (but the beginning of b may coincide with the termination of a). Two actions a and b *overlap* if a does not precede b and b does not precede a. The relation '$\prec$' on ACT is *irreflexive* and *antisymmetric*. It is also assumed to satisfy the following condition, called *Walker's postulate* ([10, 11]):

$$\forall a, b, c, d \in ACT : \ (a \prec b \ \wedge \ b \sim c \ \wedge \ c \prec d) \ \Rightarrow (a \prec d)$$

where '$\sim$' (*overlaps*) is a relation on ACT defined by

$$\forall a, b \in ACT: \; a \sim b \;\Leftrightarrow\; \neg(a \prec b) \;\wedge\; \neg(b \prec a)$$

Note that '$\sim$' is symmetric and reflexive, so Walker's postulate leads to the transitivity of '$\prec$' and thus '$\prec$' is a strict partial order on ACT.

It will often be necessary to deal with *sets* of actions and relations similar to the precedence and overlapping of single actions can be defined for sets of actions. Let A and B be subsets of ACT. Then A *precedes* B ($A \lhd B$) if

$$\forall a \in A : \forall b \in B : \; a \prec b$$

and A *overlaps* B ($A \oslash B$) if

$$\forall a \in A : \exists b \in B : \; a \sim b$$

Note that '$\oslash$' is not symmetric. If $A \lhd B$ then

$$(X \subseteq A \;\Rightarrow\; X \lhd B) \;\wedge\; (Y \subseteq B \;\Rightarrow\; A \lhd Y)$$

For any action x, let the set $After(x)$ consist of all actions which terminate after x has terminated. Similarly, let $Before(x)$ be the set of all actions that have started before x. Assume that the relations representing these functions are irreflexive, antisymmetric, and transitive. For $A \subseteq ACT$, define

$$After(A) \;=\; \begin{cases} ACT & \text{if } A = \emptyset \\ \bigcap_{a \in A} After(a) & \text{otherwise} \end{cases}$$
$$Before(A) \;=\; \begin{cases} ACT & \text{if } A = \emptyset \\ \bigcap_{a \in A} Before(a) & \text{otherwise} \end{cases}$$

Let '$\dashv$' and '$\vdash$' denote the relations 'right-aligned' and 'left-aligned' respectively. Action a is right-aligned with action b (i.e. $a \dashv b$) if a terminates no later than b and b terminates no later than a. Formally,

$$a \dashv b \;\Leftrightarrow\; a \notin After(b) \;\wedge\; b \notin After(a)$$

Similarly,

$$a \vdash b \;\Leftrightarrow\; a \notin Before(b) \;\wedge\; b \notin Before(a)$$

The relations '$\dashv$' and '$\vdash$' are symmetric and transitive, and have the following properties.

a) $\forall a, b, c \in ACT: \; [(a \dashv b) \;\wedge\; (c \in After(a))] \;\Rightarrow\; [c \in After(b)]$
b) $\forall a, b, c \in ACT: \; [(a \vdash b) \;\wedge\; (c \in Before(a))] \;\Rightarrow\; [c \in Before(b)]$

3 Instants

The precedence relation over actions, and the property of overlapping, suggests that each action has some 'duration'. Informally, two actions can be said to overlap if their durations overlap. But it is not possible to go further than that and talk about the extent of the overlap without introducing a metric over these durations, i.e. to introduce a notion of quantified time. A start can be made in that direction by introducing the idea of an instant, defined entirely in terms of sets of actions.

An *instant* of time can be considered as a durationless point in the time domain; it is defined as a set-theoretic 'cut' in ACT. Formally, $\{A, B\}$ is a $\prec$-cut if

$$A \neq \emptyset \ \wedge \ B \neq \emptyset \ \wedge \ A \lhd B$$

$$\wedge \ \forall c \in ACT : \ (\forall b \in B \ : \ c \prec b) \ \Rightarrow \ c \in A$$

$$\wedge \ \forall c \in ACT : \ (\forall a \in A \ : \ a \prec c) \ \Rightarrow \ c \in B$$

and (A, B) is then an *instant*.

Some of the important properties of sets of instants follow from the properties of an instant; e.g. since a domain of time composed of such instants cannot have any smaller elements, such a construction of time will be in the real domain *iff* the set of instants satisfies the properties associated with real numbers.

If (A, B) is an instant and $X \subseteq ACT$, then

$$(X \lhd B \ \Rightarrow \ X \subseteq A) \ \wedge \ (A \lhd X \ \Rightarrow \ X \subseteq B)$$

Let T be the set of all instants obtained from ACT. A precedence relation on T can be defined as follows. An instant $t \triangleq (A, B)$ *precedes* another instant $t' \triangleq (A', B')$, written as $t \ < \ t'$, if A is a proper subset of A' or if B' is a proper subset of B. If $A = A'$ then $B = B'$, and the instants t and t' are *equal* ($t = t'$). The notation $t \leq t'$ stands for the formula $t = t' \vee t < t'$.

The set T of instants is isomorphic with the set of real numbers if it satisfies the following axioms ([1, 11]):

A1. $(T, <)$ is a dense total order with no least and greatest elements.

A2. Any nonempty subset of T that is bounded above (i.e., has an upper bound) has a least upper bound (or, alternatively, any nonempty subset of T that is bounded below has a greatest lower bound).

A3. T contains a denumerable subset F such that between any two instants of T there is at least one instant in F.

Whitrow [11] has shown that $(T, <)$ is a total order and that it satisfies Axiom A2. Using a proof similar to that outlined by Whitrow, it can be shown that if $(T, <)$ is dense then it also satisfies Axiom A3.

Let Z be the set of all integers. A *bijection* ϕ from T onto itself is called a *time progression*, or simply a progression, if it satisfies the following conditions in which, for positive integer n, $\phi^{-n} = (\phi^{-1})^n$, ϕ^{-1} is the inverse of ϕ.

T1. $\forall t \in T: \ t < \phi(t)$

T2. $\forall t, t' \in T: \ [t < t' \ \Rightarrow \ \phi(t) < \phi(t')]$

T3. $\forall t, t' \in T: \ \exists k \in Z: \ \phi^{k-1}(t) \leq t' < \phi^k(t)]$

Condition T3 states that no chain in $\{\{\phi^n(t) \mid n \in Z\} \mid t \in T\}$ is bounded above or bounded below. A progression ϕ is said to be *slower* than another progression ϕ' $(\phi < \phi')$ if

$$\forall t \in T: \ \phi(t) < \phi'(t)$$

Consider a totally ordered set $(\Phi, <)$ of progressions for which

$$\forall t_1, t_2 \in T: \ t_1 < t_2 \ \Rightarrow \ [\exists \phi \in \Phi: \ \phi(t_1) < t_2] \tag{1}$$

Since T is dense (by assumption), such a set of progressions exists.

Let I be any (possibly infinite) initial interval of the set of positive integers. Since T is dense, given any $t_0, t_1 \in T$ such that $t_0 < t_1$, we can construct a chain $C \triangleq \{t_r \mid r \in I\}$ of instants where $t_0 < t_{r+1} < t_r$, for $r = 1, 2, \ldots$. Note that C is bounded below (t_0 is a lower bound of C). Then, by Axiom A2 for instants, C has greatest lower bound, say τ.

Define subsets $\Phi_{(r,\tau)}$ of Φ, for $r \in I$, by

$$\Phi_{(r,\tau)} \ \triangleq \ \{\phi \mid \phi(\tau) < t_r\}$$

From property (1) of Φ, the set $\Phi_{(r,\tau)}$ is nonempty.

To show that T has a denumerable subset for which between any two instants u, w in T there is an instant in that subset, suppose $u < w$. By property (1) of Φ, there is a progression ϕ in Φ such that $\phi(u) < w$. Since $u < \phi(u)$ (property T1 of progressions), it follows that

$$u < \phi(t_1) < w \tag{2}$$

Now consider the chain C. Since $\tau < \phi(\tau)$, $\phi(\tau)$ cannot be a lower bound of C (because τ is the greatest lower bound of C). Hence, there is some $t_n \in C$ so that $t_n < \phi(\tau)$. For some progression ϕ_n from the set $\Phi_{(n,\tau)}$, $\phi_n(\tau) < t_n$. Since $t_n < \phi(\tau)$, this gives $\phi_n(\tau) < \phi(\tau)$ which implies that ϕ cannot be slower than ϕ_n. Φ is totally ordered, so $\phi_n < \phi$, i.e.

$$\forall t \in T: \; \phi_n(t) < \phi(t)$$

Thus, $\phi_n(u) < \phi(u)$, and since $\phi(u) < w$ (result (2)), it follows that

$$\phi_n(u) < w \tag{3}$$

From property T3 of progressions, for some $k \in Z$

$$\phi_n^{k-1}(\tau) \leq u < \phi_n^k(\tau) \tag{4}$$

Since $\phi_n^{k-1}(\tau) \leq u$, we deduce $\phi_n^k(\tau) \leq \phi_n(u)$. This together with the result (3) implies that $\phi_n^k(\tau) < w$. From the result (4) we then obtain

$$u < \phi_n^k(\tau) < w$$

The subsets of instants $\phi_n^k(\tau)$ is denumerable, since p and n are both integers and τ is fixed. Therefore, there exists an instant which lies between any two instants of T and the set of such instants is a denumerable subset of T.

4 Denseness of T

We have so far proved that $(T, <)$ is a total order and that it satisfies axioms A2 and A3 provided that $(T, <)$ is dense. Hence $(T, <)$ is isomorphic with the set of real numbers if it can be shown that it satisfies the remaining part of Axiom A1, i.e. that $(T, <)$ is dense and that it has no least or greatest elements.

Consider an action x. Let A'_x, B'_x be subsets of ACT for which

a) A'_x is the maximal subset of ACT containing x so that $\forall a \in A'_x : \; a \notin After(x)$

b) B'_x is the maximal subset of ACT for which $A'_x \lhd B'_x$

Let C'_x be the maximal subset of $(ACT - (A'_x \cup B'_x))$ for which $C'_x \lhd B'_x$, $A_x = (A'_x \cup C'_x)$ and $B_x = B'_x$. Then, provided that A_x and B_x are nonempty, $\{A_x, B_x\}$ is a $\prec$-cut in ACT and $x_+ \triangleq (A_x, B_x)$ is an instant.

Consider any instant $t \triangleq (A, B)$. T contains another instant t' so that $t < t'$ *iff* there exists an action $x \in After(A)$ and an action b in ACT so that $x < b$ ($t < x_+$ if this condition holds).

Theorem 1 *The set T of instants is dense* iff
$\forall x, y, x', y' \in ACT :$
$[[x \prec y \;\wedge\; x' \prec y' \;\wedge\; \neg(x' \prec y)]$
$\Rightarrow \; [\exists x'', y'' \in ACT : \; x'' \prec y'' \;\wedge\; x'' \in After(x) \;\wedge\; y'' \in Before(y')]]$

The proof is given in the Appendix.

5 Actions and Computations

At one level, the relation between an action and a computation can be simply defined: an action is an execution of a finite (i.e. terminating) program command. If the time associated with computation is to be in the real domain, then the actions derived from computations must be capable of producing a set of instants which is dense.

To examine how program actions can result in a dense set of instants it is necessary to consider denseness in sets of actions. First, define intervals in sets of actions as follows. Let A be a set of actions. For any $a, b \in A$ such that $b \in After(a)$, let the set $A[a, b]$ consist of all actions of A that do not terminate before a or after b. i.e.

$$A[a, b] \triangleq \{c \in A \mid a \notin After(c) \ \wedge \ c \notin After(b)\}$$

The set $A[a, b]$ is an *interval* of A. When no confusion arises, we shall omit the prefix 'A' from the notation $A[a, b]$ for intervals.

A set A of actions is $\sim$-*dense* (or overlap-dense) in $A[a, b]$ *iff* for any action $c \notin A$ which lies in the interval $(A \cup \{c\})[a, b]$, there is some action $d \in A$ which does not terminate after c and c does not terminate after d.

$$\forall c \notin A : \ (c \in (A \cup \{c\})[a, b]) \ \Rightarrow \ (\exists d \in A[a, b] : c \dashv d)$$

Assumption 1: Fixed Command Execution
The execution of a command is modelled by exactly one action which bears a fixed relation to all other actions in the program.

This is the simplest assumption that can be made about the execution of a command and it corresponds to the assumption that each command has a fixed execution time. Since in any program of finite size, there can be only a finite number of commands executing at any instant there can only a be a finite number of actions overlapping) at any instant. From *Assumption 1* it follows that the set of actions in any single program execution is not $\sim$-dense. However, a number of different assumptions can be made about commands and their execution and, as will be seen, some do lead to a computational model in which the set of actions describing all possible executions of a program is $\sim$-dense.

In any single execution of a program, an action represents one finite execution of a program command. Due to the physical processes involved (e.g. gates with variable delays, buses with variable transmission times, etc.), each such execution is just one of a set of possible executions and it is not possible to determine *a priori* which of these executions will occur. A model of computation must consider *all* possible executions of a program.

Lemma 1 *A set A of actions is $\sim$-dense in $[a, b]$ only if for all x, x' in $[a, b]$ for which $x' \in After(x)$, there is some $x'' \in A$ for which*

$$x'' \in After(x) \ \wedge \ x' \in After(x'')$$

Proof: Follows from the definition of $\sim$-denseness and the property of '$\dashv$'. $\square$

For any command C executed in isolation, i.e. not in composition with any other command, let $ACT(C)$ denote the set of all actions of C. Assume that, in any context of execution, C has an *earliest* action c_{min} and a *latest* action c_{max}, where

$ACT(C) \cap Before(c_{min}) = \emptyset$
$\wedge [\forall b \in ACT(C) : \ c_{min} \notin After(b)]$

and

$ACT(C) \cap After(c_{max}) = \emptyset$
$\wedge [\forall b \in ACT(C) : \ c_{max} \notin Before(b)]$

For any two commands C and D, let c_{min} in $ACT(C)$ be left-aligned with d_{min} in $ACT(D)$.

Assumption 1a: Denseness in Execution of Primitive Commands
For any primitive command C, the actions in $ACT(C)$ are left-aligned and the termination point of these actions is densely dispersed, i.e. for any primitive command C, $ACT(C)$ is $\sim$-dense in the interval $[c_{min}, c_{max}]$.

Although $ACT(C)$ is $\sim$-dense by assumption, in isolation it produces just one instant. More instants can be produced *iff* one or more actions of the command precede some other actions. Such a precedence relation on actions is obtained through the sequential composition of commands.

In the sequential composition $C_1; C_2$ the starting and termination points of any action in $ACT(C_2)$ must be shifted, or delayed, by the same amount. For a command C, let $Shifted(C)$ be the set of all actions generated by all possible ways of shifting the actions in $ACT(C)$.

Assumption 2: Sequential Composition
Let C_1 and C_2 be two commands and C the sequential composition $C_1; C_2$. If $Shifted(C_2, C_1)$ is the maximal subset of $Shifted(C_2)$ in which the starting point of any action coincides with the termination point of some action in $ACT(C_1)$, then

$$ACT(C) = ACT(C_1) \cup Shifted(C_2, C_1)$$

Thus, sequential composition does not introduce any 'holes' between the execution of its constituent commands and one action starts immediately after the termination of the previous action. This provides an important result — if the set of actions of an infinite sequential composition is $\sim$-dense, then the condition of Theorem 1 is satisfied and the set of instants derived from that set of actions is dense. An infinite sequential composition leads to a $\sim$-dense set of actions.

Consider the sequential composition $C_1; C_2$. Based on *Assumption 2* the set A of the actions of this sequential composition is $\sim$-dense in the intervals $[c_{1_{min}}, c_{1_{max}}]$ and $[c_{2_{min}}, c_{2_{max}}]$. However, as the following lemma shows, A is not $\sim$-dense in the interval $[c_{1_{max}}, c_{2_{min}}]$ if $c_{2_{min}} \in After(c_{1_{max}})$.

Lemma 2 *Let a set A of actions be $\sim$-dense in the intervals $[a, b]$ and $[c, d]$ where $c \in After(a)$; if A does not contain any action e for which*

$$e \in After(b) \ \wedge \ c \in After(e)$$

then A is $\sim$-dense in $[a, d]$ iff $c \notin After(b)$.

Proof: *only if*: Assume that A is $\sim$-dense in $[a, d]$. Suppose $c \in After(b)$. Since both b and c are in the interval $[a, d]$, from Lemma 1 there exists an action e in A such that

$$e \in After(b) \ \wedge \ c \in After(e)$$

But this contradicts the hypothesis. Hence, $c \notin After(b)$.
if: Suppose the hypothesis of the lemma holds. Consider any $e \notin A$ so that $e \in (A \cup \{e\})[a, d]$. Since $c \notin After(b)$, $e \in (A \cup \{e\})[a, b]$ or $e \in (A \cup \{e\})[c, d]$. Since A is $\sim$-dense in both $[a, b]$ and $[c, d]$, there exists an $f \in A$ such that $e \dashv f$. Hence, A is $\sim$-dense in $[a, d]$. □

To show that the set of actions of an infinite sequential composition is $\sim$-dense everywhere, consider the *termination dispersion* of a terminating command C. This is a measure of the distance between the earliest and the latest termination points of the executions of C. Assuming that intuitive meaning of the *length* or duration of an action, for any primitive command E, let $min(E)$ and $max(E)$ be the lengths of the actions e_{min} and e_{max} respectively. The termination dispersion τ_C of C is

a) $(max(C) - min(C))$ if C is a primitive command, and

b) $\tau_{C_1} + ((max(C_2) - min(C_2))$ if C is the sequential composition $C_1; C_2$ and C_2 is a primitive command.

Let $C^{(i,n)}$ be the sequential composition $C_i; C_{i+1}; \ldots; C_n$, $i \leq n$. The set $ACT(C^{(1,n)})$ is $\sim$-*dense from C_i onwards* if, for any $b \in After(c_{i_{min}})$, $ACT(C^{(1,n)})$ is $\sim$-dense in $[c_{i_{min}}, b]$.

Theorem 2 *Let C be the infinite sequential composition*

$$C_1; C_2; \ldots; C_i; \ldots$$

where each C_i is a terminating primitive command; then there is some finite k such that $ACT(C)$ is $\sim$-dense from C_k onwards.

Proof: Let D' be any terminating primitive command. Then $ACT(D')$ is $\sim$-dense. From Lemma 2, the necessary and sufficient condition for the set $ACT(D; D')$ of actions of $D; D'$ to be $\sim$-dense is $min(D') \leq \tau_D$. The proof of this theorem is based on this result.

$\tau_{C^{(1,i)}}$ increases monotonically with i. Since all the components of C are terminating primitive commands, there is some finite k such that $min(C_{i+1}) \leq \tau_{C^{(1,i)}}$ for all $i \geq k$. Hence, $ACT(C)$ is $\sim$-dense from C_k onwards. $\square$

Theorem 3 *If D is a primitive command such that $\tau_D \geq min(E)$ for any primitive command E then, for any command D', $ACT(D; D')$ is everywhere $\sim$-dense.*

Proof: Similar that of Theorem 2. $\square$

Assumption 3: Parallel Composition
Let C be the parallel composition $C_1 \| C_2$; then

$$ACT(C) = ACT(C_1) \cup ACT(C_2)$$

If a set A of actions is $\sim$-dense in an interval $[a, b]$ then, for any $c \notin A$, the set $A \cup \{c\}$ is $\sim$-dense in the interval $(A \cup \{c\})[a, b]$. So the set of possible executions of $C_1 \| C_2$ is $\sim$-dense wherever the sets of possible executions of C_1 and C_2 are $\sim$-dense. This parallel composition rule models the case where the first commands of all parallel components begins execution simultaneously. Let this be *Assumption 0*. An alternative is to assume that the starting points of the first commands in the parallel components can be delayed arbitrarily within a limit; let this be *Assumption 0a*. The effect of making this assumption is similar to that obtained by adding to the start of each C_i $(i = 1, 2)$ a primitive command E_i for which

$$min(E_i) = 0 \ \wedge \ \tau_{E_i} > 0$$

Let M be the longest of the lengths of the earliest actions of the primitive commands. Let $maxdelay(C)$ be the greater of τ_{E_1} and τ_{E_2}. From Theorem 3, if $maxdelay(C) \geq M$, then $ACT(C)$ is everywhere $\sim$-dense. Thus, a sufficient condition for the set of actions obtained by sequential and parallel composition to be everywhere $\sim$-dense is that either the termination dispersion of a primitive command or the delay in the starting of a parallel component is greater than or equal to M.

Components of a program executed in parallel usually communicate with each other by sending messages. With *synchronous* communication, a message is transferred by executing the sending command and the receiving command simultaneously. Such communication is consistent with Assumption 3. But if the communication is *asynchronous*, there may be an arbitrary delay between the despatch of a message and its receipt. If $C!$ is a sending command and $C?$ its matching receiving command, then with asynchronous communication the possible executions of $C?$ will span an interval $[c_{min}?, c_{max}?]$. Let this be *Assumption 4*. As was shown for sequential composition, it can be shown that after a finite number k of communications, the set of actions is everywhere $\sim$-dense.

6 Range of Execution Models

There are, of course, many different assumptions that can be made to relate computation with time. The assumptions chosen above are fairly basic and it has been shown how they lead to denseness in the set of possible executions of actions, and thereby to time represented by a dense domain. The table below shows how the assumptions interact to produce different representations of time. The conclusions in columns 3 and 4 have been proved above or can be derived by similar reasoning.

Starting Assumption	*Other Assumptions*	*Set of Possible Actions*	*Time*
0	1 + 2	Discrete	$\aleph$
	1a + 2	Dense after a finite number of commands	$\Re$
	1 + 3	Discrete	$\aleph$
	1a + 3	Dense after a finite number of commands	$\Re$
	4	Dense after a finite number of commands	$\Re$
0a	1 + 2	Partly dense	$\Re$
	1a + 2	Dense after a finite number of commands	$\Re$
	1 + 3	Dense if $maxdelay(C) \geq M$ Partly dense otherwise	$\Re$
	4	Dense after a finite number of commands	$\Re$

When 'translated' into the more familiar association between commands and execution times, one common conclusion from the table is that a real number representation of time is needed when (from Assumption 0a) the relative starting times of different parallel components in a program are comparable to the execution time of commands, or when the execution times of commands have a continuous dispersion.

7 Discussion

The choice between representing time by a dense domain or a discrete domain clearly follows from the assumptions made in the model of computation. However, whether the dense domain should consist of all the reals, the computable reals, the rationals or an arbitrary dense domain is a more difficult question, and one for which the answers depend on further assumptions. If the physical processes, which have been assumed here to be the cause of the dispersion of the execution times, can produce times which are rational or irrational, then computations will take times which are in the real domain; given the nature of the equations governing propogation and delay, and the inherently statistical nature of many of the processes, a rigorous approach may require computational time to be represented by the reals. And in the same way that, for example, the area of a circle is *specified* in a way that may not lead to a computable real quantity, the specification of time may need to assume that it takes values from the whole real domain. However, the limitations of measurement and the lack of effect of minor variations in time may make it possible for simpler domains to be used.

Throughout this paper we have assumed that commands have uniform executions, i.e. that the duration of a command and any dispersion in durations is independent of the context in which the command is executed. This assumption leads to the expected uniform nature of time.

One argument given for the use of time in the real domain in real-time systems is that the physical processes being observed and controlled by the system are often continuous, rather than discrete. However, the observations and control commands of the system can only occur in synchrony with the execution of commands by the system, and therefore they will lie in the same domain of time as the other actions of the system.

Lamport [5] has considered interprocess communication in some detail and, making use of time in which clocks run continuously (i.e. have values which are differentiable), he shows how timestamps can be used in clock synchronization problems. In a later paper [6] he associates a global time model with time in the real domain and shows how a model of a system execution can be defined in terms of a mapping *from time* to the executions of the non-atomic operations of a program. Our task here has been more basic, in that we have sought to determine what properties of time can be justified by different models of computation.

In a comment, Lamport [6] mentions the condition that "the system is not expanding faster than the speed of light", as it could otherwise "have an infinite number of operation executions ... ". Whether on these grounds, or on account of a finite variability assumption, it must be the case that a system performs only a finite number of actions in a finite interval of time: if this does not hold, it is easy to describe a solution to the Halting problem [3]. If we assume that a computer can be constructed either of matter or of energy, using if necessary some faster-than-light means of communication, an interesting speculation is then that according to the theory of computability, such a computer cannot travel faster than light!

8 Acknowledgements

We are grateful to David Wood for introducing us to the philosophy of time, and to Howard Barringer for explaining the origins of the Finite Variability Axiom, Amir Pnueli for his comments on time and temporal logic and Mike Paterson and Jay Misra for their careful reading of the material. Leslie Lamport was kind enough to comment at length on the differences between the nature of time proposed in this paper and his assumptions about time, computations and physical actions [6].

References

[1] A. Abian. *The Theory of Sets and Transfinite Arithmetic.* W.B. Saunders Company, Philadelphia and London, 1965.

[2] H. Barringer, R. Kuiper, and A. Pnueli. A really abstract concurrent model and its temporal logic. In *Proceedings of the 13th ACM Symposium on Principles of Programming Languages*, pages 173–183, Florida, 1986.

[3] E. Best. A theorem on the characteristics of non-sequential processes. *Fundamenta Informaticae III.1*, pages 77–94, 1980.

[4] C. Hewitt and H. Baker. Actors and continuous functionals. In E.J. Neuhold, editor, *Formal Description of Programming Concepts*, pages 367–390. North-Holland, Amsterdam, 1987.

[5] L. Lamport. Time, clocks, and the ordering of events in a distributed system. *Communications of the ACM*, 21(7):558–565, 1978.

[6] L. Lamport. On interprocess communication: Part I: Basic formalism. *Distributed Computing*, 1:77–85, 1986.

[7] R. Milner. Calculi for synchrony and asynchrony. *Theoretical Computer Science*, 25:267–310, 1983.

[8] G.M. Reed and A.W. Roscoe. A timed model for Communicating Sequential Processes. In *Lecture Notes in Computer Science 226*, pages 314–323. Springer-Verlag, Heidelberg, 1986.

[9] B. Russell. On order in time. In *Proceedings of the Cambridge Philosophical Society, vol. 32*, pages 216–228, 1936.

[10] A.G. Walker. Durées et instants. *Revue Scientifique*, 85:131–134, 1947.

[11] G.J. Whitrow. *The Natural Philosophy of Time.* Clarendon Press, Oxford, 1980.

9 Appendix

Theorem 1 *The set T of instants is dense* iff

$\forall x, y, x', y' \in ACT :$
$[[x \prec y \ \wedge \ x' \prec y' \ \wedge \ \neg(x' \prec y)]$
$\Rightarrow \ [\exists x'', y'' \in ACT : \ x'' \prec y'' \ \wedge \ x'' \in After(x) \ \wedge \ y'' \in Before(y')]]$

Proof: (*only if*): Suppose T is dense. Let x, y, x', and y' be actions such that

$$x \prec y \ \wedge \ x' \prec y' \ \wedge \ \neg(x' \prec y)$$

Consider the instants $x_+ \triangleq (A_x, B_x)$ and $x'_+ \triangleq (A'_x, B'_x)$. We have $y \in B_x$. Since $x' \notin A_x$ (because x' does not precede y) and $x' \in A'_x$, we have $x_+ < x'_+$ (because T is a total order). Now T is dense, so there is an instant $t \triangleq (A, B)$ such that $x_+ < t < x'_+$. Consider any action x'' in the non-empty set $(A - A_x)$ and the corresponding instant $x''_+ \triangleq (A''_x, B''_x)$. Since $x'' \notin A_x$, $x'' \in After(x)$. We now show that there is an action y'' such that $x'' \prec y'' \ \wedge \ y'' \in Before(y')$.

The instant t cannot precede x''_+, because if it does, then x'' cannot be in A and this contradicts the selection of x''. Since T is a total order, we have $x''_+ \leq t$ and, hence, $x''_+ < x'_+$ (because $t < x'_+$). Since $B'_x \subset B''_x$, the set $(B''_x - B'_x)$ is nonempty. Consider any action y'' in this set. Since $y'' \in B''_x$, we have $x'' \prec y''$. It now remains to be shown that $y'' \in Before(y)$. Suppose $y'' \notin Before(y)$. Then $x' \prec y''$ (because $x' \prec y'$). Consequently, $y'' \in B'_x$ which also contradicts the selection of y''. So $y'' \in Before(y)$.

if: Consider instants $t \triangleq (A, B)$ and $t' \triangleq (A', B')$ such that $t < t'$. We show that

$$\exists x'' \in After(A) : \ \exists y'' \in Before(B') : \ x'' \prec y'' \tag{5}$$

Suppose that the condition does not hold. Then for any x in A and any y in $(B - B')$, since $y \in B$, $x \prec y$. Now consider any x' in $After(A)$. Since $y \in Before(B')$, from the negation of condition 5 we have $\neg(x' \prec y)$. Since, for any $y' \in B'$, $x' \prec y'$, the actions x, y, x', y' satisfy the premise of the condition of the theorem. Hence, there exist $x'', y'' \in ACT$ such that

$$x'' \prec y'' \ \wedge \ x'' \in After(x) \ \wedge \ y'' \in Before(y')$$

But, y' is an arbitrary element of B'. Therefore, since $y'' \in Before(y')$, such a y'' can be selected from $Before(B')$, and condition 5 holds. Thus the negation of condition 5 leads to a contradiction. Therefore, condition 5 holds.

Now consider the instant $x''_+ \triangleq (A''_x, B''_x)$. Since $x'' \in A''_x$ and $x'' \notin A$, we have $t < x''_+$. Also, $y'' \in Before(B')$. Thus, $y'' \notin B'$ and, consequently, $x''_+ < t'$. So an instant x''_+ exists such that $t < x''_+ < t'$. □