RONALD FAGIN AND JOSEPH Y. HALPERN

I'M OK IF YOU'RE OK: ON THE NOTION OF TRUSTING COMMUNICATION*

ABSTRACT. We consider the issue of what an agent or a processor needs to know in order to know that its messages are true. This may be viewed as a first step to a general theory of cooperative communication in distributed systems. An *honest* message is one that is known to be true when it is sent (or said). If every message that is sent is honest, then of course every message that is sent is true. Various weaker considerations than honesty are investigated with the property that provided every message sent satisfies the condition, then every message sent is true.

1. INTRODUCTION

In an analysis of communication between people or machines, it is frequently assumed (often implicitly) that all messages sent are truthful. Indeed, Lewis even takes truthfulness to be a convention in language [Lew]. Actually, an even stronger assumption is usually made. The speaker must *know* (or at least strongly believe) that his messages are truthful. Messages that are true only by accident don't count.

Of course, knowing what is and isn't true in a given situation can be subtle. Consider the following informal example, taken from [HF]. For simplicity, let us assume that communication proceeds in synchronous rounds. In the first round, Alice says to Bob, "I will eventually tell you whether I love you", and Bob says the same thing to Alice. There seems to be no problem with these statements, assuming Alice and Bob do indeed know their own feelings. Suppose that in the second round (after the first-round messages are received) Alice says to Bob, "I will tell you whether I love you one round after you tell me whether you love me." This still seems reasonable. After all, Alice knows that Bob will eventually tell her his feelings towards her, and then she can fulfill the pledge that she made one round after Bob's pledge is fulfilled. However, by similar reasoning, in the second round Bob can also send Alice the message "I will tell you whether I love you one round after you tell me whether you love me." But now they are deadlocked! Neither of them can fulfill the pledges that they made

Journal of Philosophical Logic **17** (1988) 329–354.

to each other in the first and second rounds. So in some sense these messages are not truthful. Yet exactly where did the lack of truthfulness arise?[1]

Suppose we even assume, as we do in the rest of this paper, that we can tell whether a message sent (or statement uttered) in a given situation is true and known (by its sender) to be true. We would claim that it is still unreasonable to expect most messages in a given situation to be *honest*, that is, known to be true. In ordinary communication between people, what usually happens is that the information received via messages is combined with other information, and the resulting conclusion is passed on. For example, suppose that Alice knows that Debbie is either in Boston or San Francisco and then Charlie tells her that Debbie is not in Boston. Alice then turns around and tells Bob that Debbie is in San Francisco.

Alice's message is not honest in the technical sense we have defined above. Alice does not *know* that Debbie is in Philadelphia. After all, Charlie might have been lying when he said that she was not in Boston. By producing a more guarded statement (such as "Charlie told me that Debbie is not in Boston, so I concluded that she was in San Francisco"), Alice could have produced an honest message. If Alice believes quite strongly that Charlie is truthful, then it would be quite odd of her to make such a guarded statement: in fact, in practice, such a statement might convey implicitly that Alice has good reason to believe that Charlie is lying! Furthermore, in complex, real-life situations, there are many underlying assumptions, some of which the speaker may not even be aware of. So, in order to guarantee that her messages are honest, it would be necessary for Alice to produce extremely convoluted and unnatural messages. The theme of this paper is to consider less restrictive conditions than honesty, which still guarantee that provided every message satisfies the condition, then every message sent is true.

Our goal is to get a general theory of cooperative communication in distributed systems. Such a theory would lead to more efficient communication in systems of cooperating agents (where the "agents" might well be processors in a distributed system or communicating robots). We expect that the issues that we are considering here in the restricted setting of communication in a distributed system to

be relevant also to a general theory of natural language understanding.

The idea we are trying to capture is essentially that of "I know that my message is true provided everyone else has been telling the truth." Somewhat more formally, we might say that a message m sent by p at time t is *conditionally honest* if p knows (at time t) that if every message sent up to time t is true, then m is true.[2] For example, Alice's message to Bob above that Debbie is in San Francisco is conditionally honest (although, as we observed, it is not honest).

We are really looking for an even less restrictive notion than conditional honesty that, roughly speaking, corresponds to the idea "I'm OK if you're OK". That is, we want a notion *trusting* with the property that the message m sent by p is trusting if p knows that if all previous messages are trusting, then m is true.[3] To understand the difference between conditionally honest and trusting messages, consider the following two situations. First suppose Charlie tells Alice "Richard loves Susan". If Alice now tells Bob "Richard loves Susan", than this would be a conditionally honest message. It is true provided that Charlie's message is true. Suppose instead Alice tells Bob "Charlie knows that Richard loves Susan". Now this is no longer conditionally honest. It is possible that Charlie's message is true without his knowing it (he might not know that Richard loves Susan and make a lucky guess). Thus Alice's message would be false although all previous messages are true. On the other hand, suppose that communication proceeds in rounds and Charlie's message was sent on the first round. Then for Charlie's message to be trusting, Charlie would have to know it were true. Thus Alice's message would be trusting although, as we have observed, it is not conditionally honest.

Since trusting is defined in terms of itself, it is not clear that it is well-defined. Indeed, we shall show that in general trusting is *not* a well-defined notion. More formally, trusting can be viewed as a fixed point of a certain equation. We say that there is a *complete consistent notion of trusting* in a system if there is a way of marking as "trusting" some of the messages that are sent so as to satisfy this equation. A fixed point of this equation might not exist (and thus there are systems in which there is no complete consistent notion of trusting), and even if one exists, it may not be unique. However, in systems

where messages fulfill a certain well-foundedness property (such as systems with a global clock where messages proceed in rounds), we show that there is a unique complete consistent notion of trusting. Moreover, we can show that whenever there is a complete consistent notion of trusting, the messages labelled trusting are a (not necessarily strict) superset of the conditionally honest messages.

The rest of the paper is organized as follows. In the next section we briefly review ideas from [HM, HF] and other papers, describing how to ascribe knowledge to processors in a distributed system. In Section 3 we formally define the notions of honest and conditionally honest, and give the intuition behind the notion of trusting. We show that, in a precise sense, there is no notion with the properties we want trusting to have in a general system, but in well-founded systems, there is. In Section 4 we introduce the fixed point equation defining a complete consistent notion of trusting and examine its solutions in more detail. In Section 5 we give our conclusions.

2. THE MODEL

Starting with [HM], there have been a number of papers recently that have argued that knowledge provides a useful tool for analyzing systems of communicating processors or agents (see [Ha] for an overview and survey). Of particular interest has been the issue of how the knowledge of the agents in a system changes with over time, as a result of communication.

The formal model for ascribing acknowledge to processors in a distributed system is by now fairly standard. We briefly review the details here. We phrase our definitions in terms of processors in a distributed system; we leave it to the reader to check that these definitions apply perfectly well to any group of communicating agents.

We assume that processors communicate by sending messages from some fixed set $\mathcal{M}$. A *run* r of a distributed system is a description of the execution of the system over time. We assume here for definiteness that "time" ranges over the natural numbers or the nonnegative reals here, although we could perfectly well have assumed that it ranged over any linearly ordered, unbounded set. The only facts about the run which will be relevant to us are the current state of a

processor and the messages that it sends. Thus we take a run to be a function from time to *global states*, which are tuples of the form $\langle e, s_1, \ldots, s_n \rangle$, where s_i describes processor p_i's current *local state*, and e, the environment, describes which messages are currently being sent by each of the processors.[4] Thus, if r is a run and t is a time, then $r(t)$ is a global state. We also assume that only finitely many messages are sent up to any time t in a run r. As has been done in all the papers cited above, we will identify a system with a set R of runs.

We define a *point* in a system R to be a pair (r, t) consisting of a run $r \in R$ and time t. We say that two points (r, t) and (r', t') in R are *indistinguishable* to processor p_i, and write $(r, t) \sim_i (r', t')$, if processor p_i has the same local state in $r(t)$ and $r'(t')$. That is, if $r(t) = \langle e, s_1, \ldots, s_n \rangle$ and $r'(t') = \langle e', s_1', \ldots, s_n' \rangle$, then $s_i = s_i'$.

There are times when we want to restrict our attention to systems with certain properties. Of particular interest will be *synchronous phase systems*. These are systems where communication proceeds in rounds and there is a global clock, so that, intuitively, every processor knows what round it is. More formally, R is a synchronous phase system if time ranges over the natural numbers and for every two points (r, n) and (r', n') in R, we have $(r, n) \sim_i (r', n')$ only if $n = n'$. We do not necessarily assume that a message in a synchronous phase system (or another other system, for that matter) arrives the round after which it is sent, although this will be the case in many of our examples.

In order to capture notions such as honesty, we need to be able to assign truth values to messages (which we are thinking of as formulas in some language). Given a set R of runs, a *semantic function* μ for R associates with each message m a set $\mu(m)$ of points in R. Intuitively, $\mu(m)$ is the set of points in R where m is true. In this paper we are viewing messages as syntactically unstructured, so there are no constraints on the function μ. In practice, there will be some fixed syntax for formulas (and thus for messages). We might expect, for example, that if m and m' are messages, then there is a message $m \wedge m'$ and that $\mu(m \wedge m') = \mu(m) \cap \mu(m')$. We could easily extend μ to allow us to deal with conjunctions and negations of formulas as well as facts such as "all messages sent prior to this time are true". Although we do not do this formally here, we will informally assume that such more complicated statements also have truth values.

We call a pair (R, μ), consisting of a system R and a semantic function μ on R, an *interpreted system*. Finally we say that m is *true at* (or *holds at*) the point (r, t) in the interpreted system $\mathscr{S} = (R, \mu)$ if $(r, t) \in \mu(m)$.

We want to extend the notion of truth at a point to assertions involving knowledge. This will allow us to talk about a processor *knowing* that a message is true, rather than the message just being true. Again, we follow the lines of all the papers previously cited. The intuition, which goes back to Hintikka [Hi], is that a processor knows a fact if it is true in all the worlds it considers possible. The worlds in an interpreted system are simply the points, and the points that processor p_i considers possible at the point (r, t) are exactly those it cannot distinguish from (r, t), i.e. those worlds (r', t') such that $(r, t) \sim_i (r', t')$. Thus, for a message m, we say that $K_i m$ is *true at* or *holds at the point* (r, t) *in the interpreted system* $\mathscr{S} = (R, \mu)$ if $\{(r', t') | (r, t) \sim_i (r', t')\} \subseteq \mu(m)$. A formula such as $K_i m$ is read "processor p_i knowns m".

As has been remarked before (e.g., in [HM, HF, Ha]), this is an *external* notion of knowledge. We ascribe knowledge to a processor based on its local state. A processor cannot necessarily answer questions based on its knowledge, nor do we imagine it doing any computation in order to acquire its knowledge. It may seem somewhat surprising that such a non-computational notion of knowledge is used in computer science. This usage is, however, quite well motivated. This notion of knowledge has been shown to be quite useful for analyzing distributed protocols (see, for example, [CM, DM, Ha, HM, MT]). Moreover, it does capture one way the word "know" has been used informally. For example, when someone says "processor 2 does not know that processor 3 is faulty at the end of round 5 in this run", what is often meant is that if we denote "this run" by r, then there is a point indistinguishable to processor 2 from the point $(r, 5)$ where processor 3 is not faulty. Note that this interpretation of knowledge gives us a concrete model for the S5 notion of knowledge originally discussed by Hintikka. We have chosen to work with this interpretation of knowledge because the underlying model is well understood and has numerous applications. Recently, variants of this model have been introduced that try to take computational issues

into account (see [FH, HMT, Lev, Mo]). It would clearly be interesting to develop a theory of cooperative communication along the lines we suggest here for these (and perhaps other) more computational interpretations of knowledge.

3. HONEST, CONDITIONALLY HONEST, AND TRUSTING MESSAGES

Following [HF], we say that a message is *honest* if it is known to be true. Formally, the message m sent by p_i at the point (r, t) in an interpreted system $\mathcal{S}$ is honest if $K_i m$ holds at (r, t). Note that it is not really a message that is honest or dishonest, it is a message sent by a particular processor at a particular point. The same message m may be honest at one point and not at another, or may be honest if sent by one processor but not by another. It will often be convenient for us to associate with the message m sent by p_i at the point (r, t) the tuple (m, i, r, t). We call such a tuple a *labelled message*. We say a labelled message (m, i, r, t) is true (resp. honest) if m is true (resp. honest) at the point (r, t).

As we mentioned in the introduction, we want to extend the notion of honesty to allow deductions based on previous messages under the assumption that the previous messages are true. As a first step towrds capturing this intuition, we say that the labelled message (m, i, r, t) is *conditionally honest* if p_i knows at the point (r, t) that if all previous messages are true, then m is true. More formally, the labelled message (m, i, r, t) in the interpreted system $\mathcal{S}$ is conditionally honest if for all (r', t') such that $(r', t') \sim_i (r, t)$, either m is true at (r', t') (i.e. $(r', t') \in \mu(m)$) or for some $t'' < t'$, a message m' sent at time t'' in r' was not true when it was sent (i.e. $(r', t'') \notin \mu(m')$). It is easy to see that in the scenario described in the first example in the introduction, Alice's message is not honest, but it is conditionally honest.

Recall that we are trying to capture the idea "I'm OK if you're OK." The notion of conditional honesty intuitively captures the idea "I'm telling the truth provided all the statements you made were true provided all the statements I made were true provided . . ." In order to capture the former notion, we need a notion "trusting" with the property that (m, i, r, t) is trusting if p_i knows at the point (r, t) that if all previous

messages are trusting, then m is true. Given the circularity inherent in this notion, it is not clear that it is well-defined. Indeed, we shall show that in general it is not. We first provide a few formal definitions.

Suppose we have an interpreted system $\mathcal{S} = (R, \mu)$. A *consistent notion of trusting* in $\mathcal{S}$ is simply a function (called a *marking function*) that marks some labelled messages in $\mathcal{S}$ "trusting" in such a way that if a labelled message (m, i, r, t) is marked "trusting" then for all points (r', t') such that $(r', t') \sim_i (r, t)$, either m is true at (r', t') or some labelled message sent previously (before time t') in r' is not marked "trusting". A *complete* consistent notion of trusting is a consistent notion of trusting where a message is marked "trusting" *if and only if* for all points (r', t') such that $(r', t') \sim_i (r, t)$, either m is true at (r', t') or some labelled message sent previously (before time t') in r' is not marked "trusting".[5] Note that here we have been somewhat lax and have talked about a "message" being marked "trusting" rather than a labelled message. We are similarly lax throughout the paper; we hope that what is meant is clear from context.

We remark that a complete consistent notion of trusting can be characterized as a solution to a fixed point equation. We do this in the next section. We first must check that the notion of trusting has the properties that originally motivated its definition. Note that it is immediate from the definitions that a consistent notion of trusting (resp. complete consistent notion of trusting) has the property that a labelled message is marked "trusting" only if (resp. iff) the sender knows that if all previous labelled messages are marked "trusting", then l is true. It follows that a consistent notion of trusting captures the intuition "I'm OK if you're OK." If every message in a run is OK (i.e. marked trusting), then they are all true. The following proposition makes this precise.

PROPOSITION 3.1. *Let $\mathcal{S}$ be an interpreted system and (r, t) a point in $\mathcal{S}$. For each consistent notion of trusting, if every message that is sent in run r up to time t is marked trusting, then every message sent in run r at time t is true.*

Proof. Suppose we have a consistent notion of trusting with a marking function f such that every message in run r up to time t is marked trusting by f. Further suppose that p sends message m in run

r at time t. It follows from the definition of a consistent notion of trusting that p knows that if every previous message is marked "trusting", then m is true. Since every previous message is indeed marked "trusting", m is true. ■

We now examine the relationship between trusting and the other notions we have defined. Clearly we can always get a trivial consistent notion of trusting by simply not marking any messages "trusting". It is also easy to see that we get a consistent notion of trusting by marking the honest messages "trusting", and we get another one by marking the conditionally honest messages "trusting". Thus a consistent notion of trusting can be viewed as generalizing the notions of honest and conditionally honest. However, it is clearly possible to have consistent notions of trusting where some conditionally honest messages are not marked trusting (for example, the consistent notion of trusting where no messages are marked "trusting"). As the following result shows, this is not the case for a complete consistent notion of trusting.

PROPOSITION 3.2. *Every conditionally honest message is marked "trusting" by every complete consistent notion of trusting.*

Proof. Suppose we have a complete consistent notion of trusting. Let $l = (m, i, r, t)$ be a conditionally honest labelled message. So p_i knows when it sends m at the point (r, t) that if every previous message is marked trusting, then (by Proposition 3.1) every previous message is true, so l is true (since l is conditionally honest). So l must be marked "trusting". ■

It is not hard to show that there are interpreted systems where the messages marked "trusting" according to a complete consistent notion of trusting form a strict superset of the conditionally honest messages. (We gave an informal example in the introduction which can be easily formalized; another example is given at the end of this section.) Of course we still haven't shown that every interpreted system has a complete consistent notion of trusting. Indeed, as we hinted above, this is not the case. We now prove this by example.

Given an interpreted system $\mathscr{S}$, let us say that the labelled messages (m, i, r, t) and (m, i, r', t') of $\mathscr{S}$ are *indistinguishable by the sender* (in

this case p_i) iff $(r, t) \sim_i (r', t')$. Note that in a complete consistent notion of trusting, if two labelled messages are indistinguishable by the sender, then either both are marked "trusting" or neither is marked "trusting". This is as it should be, since we would expect that if two messages are indistinguishable by the sender, then they should be marked the same way.

THEOREM 3.3. *There exists an interpreted system with no complete consistent notion of trusting.*

Proof. Consider an interpreted system $\mathscr{S} = (R, \mu)$ where there are three processors p_1, p_2, and p_3, and where R consists of exactly three runs, say r_1, r_2, and r_3. The only message that is ever sent in any of these runs is the message m, which we can think of as saying "This is the first message in the run". In run r_1, processor p_1 sends m at time 1, and processor p_2 sends m at time 2. In run r_2, processor p_2 sends m at time 1, and processor p_3 sends m at time 2. In run r_3, processor p_3 sends m at time 1, and processor p_1 sends m at time 2. These are the only messages sent. We assume that each processor's local states are such that the processor is in the same state at the two points where it sends a message, and in a different state at the other points. Note that in particular this means that $\mathscr{S}$ is not a synchronous phase system. Finally, we take μ to be such that m is true at time 1 in each of the runs, but false at time 2. The situation is described in Figure 1. In the figure, we label a point T or F to indicate whether or not the message m is true at that point. Whenever an edge labeled i connects two points, the two points are indistinguishable by processor p_i. (We do not include in the figure information about which processor sends m at each point.)

Suppose there were a complete consistent notion of trusting in $\mathscr{S}$. Assume first that some message sent by p_1 is marked "trusting". Since p_1 cannot distinguish the two points where it sent the message, each message sent by p_1 must be marked "trusting". But then our definition forces us not to mark either message sent by p_2 "trusting"' (since, for example, $(m, 2, r_1, 2)$ is false and is preceded only by a message marked "trusting"). We now leave it to the reader to check that this forces us to mark each message sent by p_3 "trusting", which in turn forces us not to mark the message sent by p_1 "trusting". We have

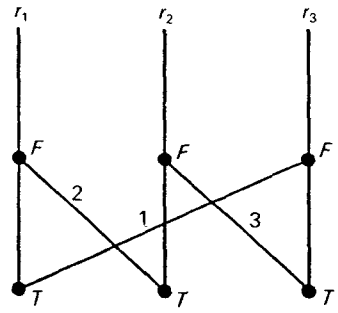

Fig. 1.

shown that if some message sent by p_1 is marked "trusting", then every message sent by p_1 should not be marked "trusting", a contradiction. On the other hand, if no message sent by p_1 is marked "trusting" then arguing as before, we see that this forces us to mark each message sent by p_2 "trusting", which forces us not to mark any message sent by p_3 "trusting", which forces us to mark every message sent by p_1 "trusting", a contradiction. ■

Intuitively, the problem here is a certain lack of well-foundedness. We define a partial order $\prec$ on labelled messages in an interpreted system $\mathscr{S}$ by taking $\prec$ to be the least transitive relation such that (a) $(m, i, r, t) \prec (m', i', r, t')$ if $t < t'$ and (b) if l_1, l_1' (resp. l_2, l_2') are labelled messages that are indistinguishable by the sender and $l_1 \prec l_2$, then $l_1' \prec l_2'$. We say that $\mathscr{S}$ is *well-founded* if $\prec$ is a well-founded partial order on labelled messages in $\mathscr{S}$; i.e., if there is no infinite descending chain $\cdots \prec l_3 \prec l_2 \prec l_1$ of labelled messages.

It is easy to see that the interpreted system described in Theorem 3.3 is not well-founded. On the other hand, synchronous phase systems are always well-founded. Indeed, any system with a global clock where messages can be sent only at a well-founded set of times is well-founded. Well-foundedness seems to be a reasonable idealization to make about human interactions, especially in certain constrained settings (for example, face-to-face discussions, or correspondence by dated memos). As we now show, well-foundedness is a sufficient condition for a system to have a *unique* complete

consistent notion of trusting. Intuitively, this is because in a well-founded system, we can define the notion of trusting by induction.

THEOREM 3.4. *In each well-founded interpreted system, there is a unique complete consistent notion of trusting.*

Proof. Let $\prec$ be as defined above, and assume that the interpreted system $\mathscr{S}$ is well-founded. By transfinite induction on $\prec$, we can associate a *rank* with each labelled message l by letting $rank(l)$ be the least ordinal that is greater than every $rank(l')$ where $l' \prec l$. We now define, by transfinite induction, a marking function f.

If $rank(l) = 0$, we mark l "trusting" iff l is honest. Suppose $rank(l) > 0$ and we have already decided which labelled messages of rank less than $rank(l)$ are marked "trusting". If $l = (m, i, r, t)$, we mark l "trusting" iff for all points (r', t') such that $(r', t') \sim_i (r, t)$, either m is true at (r', t') or some labelled message l' sent previously in r' is not marked "trusting". This completes the definition of the marking function. A straightforward induction on rank can now be used to show that a labelled message $l = (m, i, r, t)$ is marked "trusting" iff for all points (r', t') such that $(r', t') \sim_i (r, t)$, either m is true at (r', t') or some labelled message sent previously (before time t') in r' is not marked "trusting". Thus f defines a complete consistent notion of trusting in $\mathscr{S}$. If f' is another complete consistent notion of trusting in $\mathscr{S}$ we can show by induction on rank that l is marked "trusting" by f iff it is marked "trusting" by f'. Thus f defines the unique complete consistent notion of trusting in $\mathscr{S}$. ■

It follows that in a well-founded system we can not only speak of a trusting message, but can do so without ambiguity. The following result says that in general we cannot. Even in a system where there is a complete consistent notion of trusting, it might not be unique.

THEOREM 3.5. *There are interpreted systems with more than one complete consistent notion of trusting.*

Proof. We modify the system described in the proof of Theorem 3.3 so that there are now only two processors rather than three. Let $\mathscr{S} = (R, \mu)$ be an interpreted system with two processors, p_1 and p_2, where R consists of exactly two runs, say r_1 and r_2. Again there is

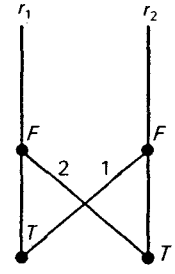

Fig. 2.

only one message m. In run r_1, processor p_1 sends m at time 1, and processor p_2 sends m at time 2. In run r_2, processor p_2 sends m at time 1, and processor p_1 sends m at time 2. These are the only messages sent. We take the processors' local states to be such that each processor is in the same state at the two points where it sends a message, and in a different state at the other points. Finally, we take μ to be such that m is true at time 1 in both runs, but false at time 2. The situation is described in Figure 2.

Thus, as in Theorem 3.3, we can think of m as saying "This is the first message in the run". It is straightforward to verify that we get a complete consistent notion of trusting either (a) by marking only p_1's messages "trusting", or (b) by marking only p_2's messages "trusting". ■

Theorem 3.3, Theorem 3.4, and Theorem 3.5 together show that in an interpreted system there may be no complete consistent notion of trusting, exactly one such notion, or more than one. However, Theorem 3.4 and our observations preceding it together suggest that for many naturally occurring situations there will be a *unique* complete consistent notion of trusting.

Note that in the interpreted system constructed in the proof of Theorem 3.5, none of the messages are conditionally honest. Thus we have an example of a complete consistent notion of trusting where the messages marked "trusting" form a strict superset of the conditionally honest messages. We can also formalize the example given in the introduction to get a well-founded system with a unique complete

consistent notion of trusting where the messages marked "trusting" form a strict superset of the conditionally honest messages.

4. VIEWING TRUSTING AS A FIXED POINT

We have seen that "trusting" is not guaranteed to be a well-defined notion if the interpreted system is not well-founded. We can get a better understanding of the situation by viewing "trusting" as a solution of a certain fixed point equation, and then constructing approximations to it.

Let $\mathscr{S}$ be an interpreted system. If X is a set of labelled messages of $\mathscr{S}$, define $F_1(X)$ to be the set of labelled messages (m, i, r, t) of $\mathscr{S}$ such that p_i knows at (r, t) that either m is true or some previous message is in X (i.e. $(m, i, r, t) \in F_1(X)$ if whenever $(r', t') \sim_i (r, t)$, either $(r', t') \in \mu(m)$, or some message sent before time t' in r' is in X). Similarly, define $F_2(X)$ to be the set of labelled messages (m, i, r, t) of $\mathscr{S}$ such that p_i considers it possible at (r, t) that all previous messages are in X and m is false (i.e. $(m, i, r, t) \in F_2(X)$ if there exists a point (r', t') such that (a) $(r', t') \sim_i (r, t)$, (b) $(r', t') \notin \mu(m)$, and (c) all messages sent before time t' in r' are in X). It is clear that F_1 and F_2 are monotone, in the sense that if $X \subseteq Y$, then $F_1(X) \subseteq F_1(Y)$ and $F_2(X) \subseteq F_2(Y)$.

The relation between these functions and the notion of trusting is brought out in the following lemma (where X^c denotes the complement of X).

LEMMA 4.1. *Let $\mathscr{S}$ be an interpreted system and let f be a function that marks the precisely the labelled messages in a set $\mathscr{T}$ "trusting".*

1. *f is a consistent notion of trusting iff $\mathscr{T} \subseteq F_1(\mathscr{T}^c)$.*
2. *f is a complete consistent notion of trusting iff $\mathscr{T} = F_1(\mathscr{T}^c)$ and $\mathscr{T}^c = F_2(\mathscr{T})$.*

Proof. Straightforward. ■

Define $F(X, Y) = (F_1(Y), F_2(X))$. By Lemma 4.1, it follows that f is a complete consistent notion of trusting iff $F(\mathscr{T}, \mathscr{T}^c) = (\mathscr{T}, \mathscr{T}^c)$, where $\mathscr{T}$ is the set of labelled messages marked "trusting" by f. We

know from Theorem 3.3 that we will not always be able to find such a set $\mathscr{T}$. However, this fixed point equation suggests a way of approximating a complete consistent notion of trusting. We define two hierarchies T_α and N_α of (labelled) messages of $\mathscr{S}$, where each hierarchy is indexed by ordinals α.

1. $T_0 = N_0 = \varnothing$.
2. $T_{\alpha+1} = F_1(N_\alpha)$.
3. $N_{\alpha+1} = F_2(T_\alpha)$.
4. If α is a limit ordinal, then $T_\alpha = \bigcup_{\delta<\alpha} T_\delta$ and $N_\alpha = \bigcup_{\delta<\alpha} N_\delta$.

Let $T^* = \bigcup_\alpha T_\alpha$ and let $N^* = \bigcup_\alpha N_\alpha$.[6]

It is easy to see that T_1 consists precisely of the honest messages, so the hierarchy extends the notion of honesty. We discuss how conditionally honest messages relate to the hierarchy at the end of this section.

Intuitively, T^* forms a lower bound on the labelled messages marked "trusting" by any complete consistent notion of trusting, while N^* forms a lower bound on the labelled messages not marked "trusting". More formally we have

PROPOSITION 4.2. *For each complete consistent notion of trusting, every message in T^* is marked "trusting" and every message in N^* is not marked "trusting".*

Proof. If f is a complete consistent notion of trusting in an interpreted system $\mathscr{S}$, we can show by a straightforward induction on α that every message in T_α is marked "trusting" by f and every message in N_α is not marked "trusting". ■

Proposition 4.2 would lead us to expect that we get a unique complete consistent notion of trusting when $T^* = (N^*)^c$. It is clear that if we have an interpreted system where $T^* = (N^*)^c$ and there is a complete consistent notion of trusting in this system, then it must be unique. In order to show that there is a complete consistent notion of trusting in such a system, we need an easy lemma, which shows that

the T_α's and N_α's do indeed form a hierarchy. This observation will have a number of other interesting consequences.

LEMMA 4.3.

1. *If* $\alpha < \beta$, *then* $T_\alpha \subseteq T_\beta$ *and* $N_\alpha \subseteq N_\beta$.

2. T^* *and* N^* *are disjoint.*

Proof. We prove (1) by induction on β. The result is clear if β is a limit ordinal. If β is a successor ordinal, suppose that $\beta = \beta' + 1$. Then

$$T_\beta = F_1(N_{\beta'}) \supseteq \bigcup_{\gamma<\beta'} F_1(N_\gamma) = \bigcup_{\gamma<\beta'} T_{\gamma+1} = T_{\beta'}.$$

Note that the inductive step is used in a number of these steps, as well as the fact that F_1 is monotonic. Finally, since $\alpha < \beta$, we have $\alpha \leqslant \beta'$, so from the inductive hypothesis it follows that $T_\alpha \subseteq T_{\beta'}$. Thus $T_\alpha \subseteq T_\beta$. Identically $N_\alpha \subseteq N_\beta$.

In order to prove (2), we show by induction on α that T_α and N_α are disjoint. There is no difficulty for limit ordinals. If α is a successor ordinal, suppose that $\alpha = \alpha' + 1$ and the labelled message $l = (m, i, r, t)$ is in N_α. By definition, there exists a point (r', t') such that (a) $(r', t') \sim_i (r, t)$, (b) $(r', t') \notin \mu(m)$, and (c) all messages sent before time t' in r' are in $T_{\alpha'}$. By the inductive hypothesis, no messages sent before time t' in r' are in $N_{\alpha'}$, so it is easy to see that $l \notin F_1(N_{\alpha'}) = T_\alpha$. Thus T_α and N_α are disjoint. ■

One easy consequence of Lemma 4.3 is that the T_α and N_α hierarchies must eventually collapse.

THEOREM 4.4. *For each interpreted system, there exists* γ *such that* $T_\gamma = T^*$ *and* $N_\gamma = N^*$.

Proof. Let λ be the cardinality of the number of labelled messages. If the T_α hierarchy does not eventually collapse, then there are arbitrarily large values of β (and, in particular, more than λ distinct values of β) such that the difference $T_{\beta+1} - T_\beta$ is nonempty. But this gives us too many labelled messages. ■

It follows easily from this proof that it is sufficient to take γ in the statement of Theorem 4.4 to be the least cardinal greater than the cardinality of the number of labelled messages. (We require the additional observation that if $T_\alpha = T_{\alpha+1} = T_{\alpha+2}$ then $T_\alpha = T^*$.)

Another consequence of the preceding results is that (T^*, N^*) is a fixed point of F; in fact, it is easy to see that the construction guarantees that it is the least fixed point, although we do not bother to prove this here.

THEOREM 4.5. $F(T^*, N^*) = (T^*, N^*)$.

Proof. We must show that $T^* = F_1(N^*)$ and $N^* = F_2(T^*)$. The argument is quite standard. We show that $T^* = F_1(N^*)$; the proof that $N^* = F_2(T^*)$ is similar and left to the reader.

Since F_1 is monotonic, we clearly have

$$F_1(N^*) \supseteq \bigcup_\alpha F_1(N_\alpha) = \bigcup_\alpha T_{\alpha+1} = T^*.$$

From Theorem 4.4, it follows that there exists γ such that $N^* = N_\gamma$, so that

$$F_1(N^*) = F_1(N_\gamma) \subseteq T_{\gamma+1} \subseteq T^*.$$

Thus $T^* = F_1(N^*)$. ■

THEOREM 4.6. *For all α, the function which marks all the labelled messages in T_α "trusting" is a consistent notion of trusting.*

Proof. Using Lemma 4.3, Theorem 4.5, and the fact that F_1 is monotonic, we get the following chain of containments:

$$T_\alpha \subseteq T^* = F_1(N^*) \subseteq F_1((T^*)^c) \subseteq F_1(T_\alpha^c).$$

Since $T_\alpha \subseteq F_1(T_\alpha^c)$, it follows from Lemma 4.1 that the function which marks all the labelled messages in T_α "trusting" is a consistent notion of trusting. ■

We are now ready to prove:

THEOREM 4.7. *If $T^* = (N^*)^c$, then there is a unique complete consistent notion of trusting f, and the labelled messages marked "trusting" by f are precisely the members of T^*.*

Proof. Let f be the function which marks the members of T^* as "trusting". It follows immediately from Proposition 4.2 that the only possible complete consistent notion of trusting is f. We now show that f is indeed a complete consistent notion of trusting. We must show that a labelled message l is marked "trusting" iff the sender knows that if all previous messages are marked "trusting", then l is true. We shall simply show the "only if" direction, since the "if" direction is very similar. Let γ be so large that $T_\gamma = T_{\gamma+1} = T^*$ and $N_\gamma = N_{\gamma+1} = N^*$. Assume that a labelled message $l = (m, i, r, t)$ is marked "trusting". Hence, $l \in T^*$. Therefore, $l \in T_{\gamma+1}$, and so p_i knows at (r, t) that either m is true or some previous message is in $N_\gamma = N^*$. Therefore, p_i knows that if all previous messages are marked "trusting", then l is true. This was to be shown. ■

We have already shown (Theorem 3.4) that for well-founded interpreted systems there is a unique complete consistent notion of trusting. It will probably not surprise the reader to learn that the proof of this fact easily generalizes to show that in a well-founded interpreted system we also have $T^* = (N^*)^c$.

THEOREM 4.8. *If the interpreted system is well-founded, then* $T^* = (N^*)^c$.

Proof. It is easy to modify the proof of Theorem 3.4 to show that T_α consists precisely of those labelled messages l with $rank(l) = \alpha$ that are marked "trusting" while N_α consists of those labelled messages l with $rank(l) = \alpha$ that are not marked "trusting". The result follows. ■

In the important special case of synchronous phase systems, we have the following result.

COROLLARY 4.9. *For synchronous phase systems,* $T_\omega = T^* = (N^*)^c = (N_\omega)^c$.

Proof. It follows immediately from the proof of Theorem 4.8 that if γ is larger than the rank of every labelled message in a well-founded system $\mathcal{S}$, then $T^* = T_\gamma$ and $N^* = N_\gamma$. It is clear that the rank of a labelled message sent in the kth round of a synchronous phase system

is at most $k - 1$. Therefore, ω is larger than the rank of every labelled message in a synchronous phase system, so $T^* = T_\omega$ and $N^* = N_\omega$. This, together with Theorem 4.8, gives us the result desired. ■

Thus, for synchronous phase systems, we have

$$(*) \qquad \begin{aligned} T_0 \subseteq T_1 \subseteq \ldots \subseteq T_k \subseteq T_{k+1} \subseteq \ldots \subseteq T_\omega &= T^*. \\ N_0 \subseteq N_1 \subseteq \ldots \subseteq N_k \subseteq N_{k+1} \subseteq \ldots \subseteq N_\omega &= N^*. \end{aligned}$$

Does the T_α hierarchy (or the N_α hierarchy) collapse earlier than level ω in this case or are the inclusions in (*) strict? In general, the latter holds:

THEOREM 4.10. *There is a synchronous phase system where all of the inclusions in* (*) *are strict.*

Proof. Consider a synchronous phase system $\mathscr{S} = (R, \mu)$ where there are two processors p_1 and p_2, and where R consists of countably many runs $r_1, r_2, r_3, \ldots$ and $r'_1, r'_2, r'_3, \ldots$. In each run, each processor sends exactly one message. In run r_k, message m_k is sent in round k and message m_{k+1} is sent in round $k + 1$. In run r'_k, message m'_k is sent in round k and message m'_{k+1} is sent in round $k + 1$. Processor p_1 sends the odd-numbered messages $(m_1, m'_1, m_3, m'_3, \ldots)$ while p_2 sends the even-numbered messages. We think of m_1 as being logically true, m'_1 as being logically false, and the messages m_{k+1} (resp. m'_{k+1}) for $k \geq 1$ as saying "if this is the point $(r_k, k + 1)$ or $(r_{k+1}, k + 1)$ (resp. $(r'_k, k + 1)$ or $(r'_{k+1}, k + 1)$), then this is the first message in the run". Thus, we take μ to be such that m_1 is true at all points, m'_1 is false at all points, and m_{k+1} (resp. m'_{k+1}) is false at the point $(r_k, k + 1)$ (resp. $(r'_k, k + 1)$) but true at all other points. Finally, we assume that for all runs r, each of the processors is in distinct local states at every round of r (so that $\mathscr{S}$ really is a synchronous phase system) and that in odd-numbered rounds (resp. even-numbered rounds) processor p_1 (resp. p_2) is in the same state at the two points where it sends the message m_k and in the same state at the two points where it sends the message m'_k. This situation is described in Figure 3.

An easy induction on k now shows that for $k \geq 1$, if k is odd, then (the labelled version of) the message m_k is in T_k but not in T_{k-1}, and

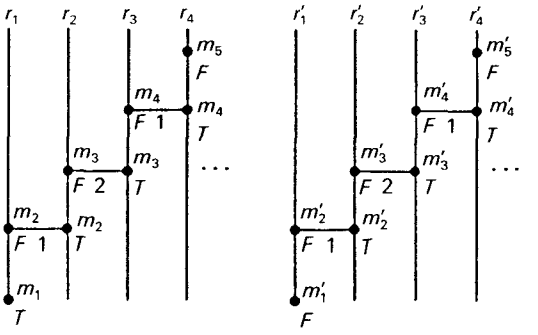

Fig. 3.

if k is even, then (the labelled version of) the message m_k is in N_k but not in N_{k-1}. A very similar argument shows that if k is odd, then (the labelled version of) the message m'_k is in N_k but not in N_{k-1}, and if k is even, then (the labelled version of) the message m'_k is in T_k but not in T_{k-1}. Taken together, or course, these results are sufficient to prove the theorem. We leave details to the reader. ■

We note that similarly, for each γ, we can find a well-founded interpreted system where $T_\alpha \neq T_\beta$ and $N_\alpha \neq N_\beta$ when α and β are distinct ordinals less than γ.

From Theorem 3.3 and Theorem 4.7 it follows that there will be interpreted systems where we do not have $T^* = (N^*)^c$. Techniques similar to those of Theorem 3.3 and Theorem 3.5 can be used to show that in interpreted systems where $T^* \neq (N^*)^c$, anything may happen. We may have no complete consistent notion of trusting, a unique complete consistent notion of trusting, and more than one complete consistent notion of trusting. We omit details here.

We close this section with a comparison of conditionally honest messages and the hierarchy we have defined.

THEOREM 4.11.

1. *No conditionally honest messages is in N^*.*
2. *In general the set of conditionally honest messages is incomparable to T^*.*

Proof. For part 1, we use induction on α to show that no conditionally honest message is in N_α. There is no difficulty if α is a limit ordinal. If $\alpha = \alpha' + 1$, suppose that $l = (m, i, r, t)$ is a conditionally honest message in N_α. By definition of N_α we know that $l \in F_2(T_{\alpha'})$. Thus there must be a point (r', t') such that (a) $(r', t') \sim_i (r, t)$, (b) m is false at (r', t'), and (c) all earlier messages in r' are in $T_{\alpha'}$. By Theorem 4.6, the function which marks all the formulas in $T_{\alpha'}$ "trusting" is a consistent notion of trusting. By Proposition 3.1, it follows that all the messages sent in run r' up to time t' are true. But this contradicts the assumption that l is conditionally honest.

For part 2, we first construct an interpreted system where T^* is a strict subset of the set of conditionally honest messages, then construct one where T^* is a strict superset of the set of conditionally honest messages. Putting these two examples together (by taking the union of the sets of runs), we can get an interpreted system where the set of conditionally honest messages is incomparable to T^*.

Our first construction is a modification of that given in the proof of Theorem 3.5. The situation is described in Figure 4. Again we have two processors p_1, p_2 and two runs r_1, r_2. After the message m is sent by both processors in both runs, processor p_1 sends another message m' in both runs. We take μ to be such that m is true in the first round of both runs and false at all other points, while m' is false at all points. Processor p_1 is in state s at the two points where it sends the message m, in a different state s' at the two points where it sends the message m', and in states other than s and s' at all other points (it is not important whether or not these states are distinct); processor p_2 is in state s at the two points where it sends m and in states other than s at all other points. We leave it to the reader to check that m' is conditionally honest, but $T^* = N^* = \varnothing$ in this interpreted system.

We next construct an interpreted system where three are no conditionally honest messages, but $T^* \neq \varnothing$. Consider the following synchronous system (which actually formalizes the example given in the introduction), described in Figure 5. Again there are two processors p_1, p_2 (which correspond to Alice and Bob) and two runs. In round 1 of both runs p_1 sends the message m_1, while in round 2 of both runs p_2 sends the message m_2. We take μ to be such that m_1 is true at

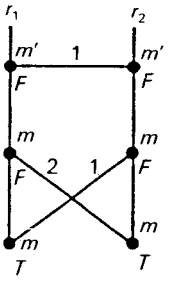

Fig. 4.

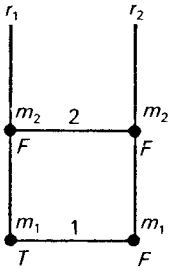

Fig. 5.

$(r_1, 1)$ and false at $(r_2, 1)$, while m_2 is false at both $(r_1, 2)$ and $(r_2, 2)$. (We are thinking of m_1 as the statement "Richard loves Susan" and of m_2 as the statement "Charlie knows that Richard loves Susan".) We take the processors' local states to be such that p_1 is in state s in round 1 of both runs and state t in round 2 of both runs, while p_2 is in state s in round 1 of both runs and in state t in round 2.

It is now easy to check that $N_1 = N^* = \{(m_1, 1, r_1, 1), (m_1, 1, r_2, 1)\}$ and $T_2 = T^* = \{(m_2, 2, r_1, 2), (m_2, 2, r_2, 2)\}$. (Note that we get a complete consistent notion of trusting since this is a synchronous phase system.) It is also easy to check that there are no conditionally honest messages.

We can simply combine these two examples to get an interpreted system with four runs where the set of conditionally honest messages is incomparable to T^*. ■

We remark that we could define another hierarchy starting with T_0' consisting of the conditionally honest messages and with $N_0' = \varnothing$, and again applying the function F_1 and F_2. Again we could construct a fixpoint $((T')^*, (N')^*)$, but this time our construction would guarantee that the conditionally honest messages are a subset of $(T')^*$. We omit further details here.[7]

5. CONCLUSIONS

We investigated the general issue of what an agent or processor needs to know in order to know that its messages are true. This led us to define the notion of conditionally honest messages, where a message is conditionally honest if it is known to be true provided that all previous messages are true. We tried to generalize this notion by defining trusting messages, where intuitively a trusting message is one that is known to be true provided that all previous messages are trusting. We showed that trusting messages are not well-defined in general. However, they are well-defined in many natural systems, such as synchronous phase systems, where communication proceeds in rounds and there is a global clock.

None of the results in this paper is difficult to prove. We believe that the major contribution of the paper lies not in the depth of the results, but in the ideas. We made a number of false starts in pursuing the ideas in this paper, and we now feel that we have the "right" definitions (although, as we indicated in a few places in the paper, there are some quite reasonable variants of the definitions we have chosen that lead to essentially the same results).

While we defined honest, conditionally honest, and trusting messages in the context of the S5 notion of knowledge used in distributed systems, they should make sense for other notions of knowledge as well. It would be interesting to pursue these issues in the context of other notions of knowledge, perhaps ones that better model resource-bounded human reasoners.

ACKNOWLEDGEMENTS

We would like to thank Gordon Plotkin, Rich Thomason, Moshe Vardi, and Ed Wimmers for their comments on a previous draft of

this paper, and Anil Gupta for suggesting some useful references on truthfulness. Moshe also pointed out to us the connection between trusting and the Kripke truth predicate. Rich suggested the term "conditionally honest".

NOTES

* This is an expanded version of a paper that appears in the *Proceedings of the Second IEEE Symposium on Logic in Computer Science*, 1987.

[1] Bob doesn't really have to send his message at the same round as Alice for the deadlock to occur. All that is required is that Bob send his second message after receiving Alice's first message and before receiving her second one, and similarly for Alice. Thus, we can arrange for this type of deadlock in any setting where a reasonable amount of time may pass between when a message is sent and when it is received.

[2] We could have taken a variant of this notion by defining a message m sent by p at time t to be conditionally honest if p knows that m is true provided that all messages previously received by p are true; i.e., instead of considering all messages sent, we could consider only the messages received by p. Everything we say in this paper goes through with this definition as well. The definition we actually use is slightly less restrictive, in that more messages will be considered conditionally honest.

[3] As Rich Thomason has pointed out, trusting is perhaps a bit of a misnomer; I can presuppose what someone has told me for the sake of conversation without believing it at all. We have used this name partly for historical reasons (it was used in an earlier version of this paper, although for the notion we are now calling conditional honesty), and partly because we feel it gets across the idea that conversation often presupposes trust on the part of the communicating parties that only truthful messages are being communicated.

[4] In a more detailed model of a distributed system, the global state would include more information and satisfy extra conditions. For example, we would expect the environment in the global state in run r at time t to describe all the messages that have been sent but not delivered up to that time. Moreover, we would expect a processor's local state to be a function of its *message history*, the sequence of messages it has sent and received in run r up to, but not including, time t, together with the times they were sent and received on the processor's local clock (if there are clocks in the system). However, the results and definitions of this paper are independent of these conditions, so we do not impose them here. We remark that all our impossibility results hold if we restrict attention to systems where a processor's state encodes its complete message history (this is the *total view* or *complete history* interpretation discussed in [HM]).

[5] We could have considered a slight variant of the definition of "consistent notion of trusting" by defining a consistent′ notion of trusting to be a function that marks some (not necessarily all) labelled messages in $\mathscr{S}$ as either "trusting" or "not trusting" (but not both!) in such a way that (a) a labelled message (m, i, r, t) is marked "trusting" only if for all points (r', t') such that $(r', t') \sim_i (r, t)$, either m is true at (r', t') or some labelled message sent previously is labelled "not trusting" and (b) a labelled message (m, i, r, t) is marked "not trusting" only if at some point (r', t') such that $(r', t') \sim_i (r, t)$ we have both that m is false and that all messages sent previously in r'

are marked "trusting". We could then define a *complete* consistent′ notion of trusting to be one where all messages are marked either "trusting" or "not trusting". It is easy to see that a complete consistent′ notion of trusting is simply a complete consistent notion of trusting where all the messages not labelled "trusting" are labelled "not trusting". Thus, complete consistent and complete consistent′ notions of trusting are essentially identical. Moreover, if we consider the messages marked "trusting" by a consistent′ notion of trusting, we get a consistent notion of trusting. The converse, however, is not true in general. Consider an interpreted system $\mathscr{S}$ where a false message m is sent at the point (r, t) by p_1. Suppose that there are no other points in $\mathscr{S}$ where p_1 has the same local state as in (r, t), and further suppose that m is preceeded in (r, t) by one message, which happens to be honest. The function that labels only m "trusting" gives a consistent notion of trusting, but it is easy to see that there is no consistent′ notion of trusting that marks m "trusting". All the theorems we state here hold if we replace consistent by consistent′, with essentially no change in proof. We gave the definitions we did simply because they are somewhat simpler and they make our proofs go through a little more smoothly. Both definitions agree on what we consider to be the most important properties.

[6] The reader familiar with the work of Kripke and other philosophers on defining a truth predicate [Kr, Gu] will immediately see the similarities between our approach and theirs. Kripke also defines a function analogous to our F and constructs its fixed point just as we have constructed T^* and N^*. Kripke is interested in all the fixed points of the function that he constructs. We only focus here on fixpoints of F of the form $(\mathscr{T}, \mathscr{T}^c)$ (which, as shown in Lemma 4.1, correspond to complete consistent notions of trusting) and the fixed point (T^*, N^*). We remark that the labelled messages in $T^* \cup N^*$ are analogous to what Kripke called *grounded* formulas.

[7] More generally, given any consistent′ notion of trusting (as defined in Note 6), we could construct a hierarchy by setting T_0' to consist of all the labelled messages that are marked "trusting" and setting N_0' to consist of all the labelled messages that are marked "not trusting". We could then prove analogues to all the theorems in this section. However, the results do *not* go through in general if we use a consistent (rather than consistent′) notion of trusting to define T_0' (no matter how N_0' is defined).

REFERENCES

[CM] M. Chandy and J. Misra, 'How processes learn', *Distributed Computing* **1**(1), 1986, pp. 40–52.

[DM] C. Dwork and Y. Moses, 'Knowledge and common knowledge in a Byzantine environment I: Crash failures', *Theoretical Aspects of Reasoning about Knowledge: Proc. of the 1986 Conference* (ed. J. Y. Halpern), Morgan Kaufmann, 1986, pp. 149–169.

[FH] R. Fagin and J. Y. Halpern, 'Belief, awareness, and limited reasoning', *Artificial Intelligence* **34**, 1988, pp. 39–76.

[Gu] A. Gupta, 'Truth and paradox', *Journal of Philosophical Logic* **11**, 1982, pp. 1–60. Reprinted in *Recent Essays on Truth and the Liar Paradox* (ed. R. L. Martin), Oxford University Press, 1984, pp. 175–235.

[Ha] J. Y. Halpern, 'Using reasoning about knowledge to analyze distributed systems', *Annual Review of Computer Science, Vol. 2* (ed. J. Traub *et al.*), Annual Reviews Inc., 1987, pp. 37–68.

[HF] J. Y. Halpern and R. Fagin, 'A formal model of knowledge, action, and communication in distributed systems', *Proceedings of the 4th ACM Symposium on Principles of Distributed Computing*, 1985, pp. 224–236.

[HM] J. Y. Halpern and Y. O. Moses, 'Knowledge and common knowledge in a distributed environment', *Proceedings of the 3rd ACM Symposium on Principles of Distributed Computing*, 1984, pp. 50–61; a revised version appeared as IBM Research Report RJ 4421, 1986.

[HMT] J. Y. Halpern, Y. O. Moses, and M. Tuttle, 'A knowledge-based analysis of zero knowledge', *Proceedings of the 20th ACM Symposium on Theory of Computing*, 1988, pp. 132–147.

[Hi] J. Hintikka, *Knowledge and Belief*, Cornell University Press, 1962.

[Kr] S. A. Kripke, 'Outline of a theory of truth', *Journal of Philosophy* **72**, 1975, pp. 640–716. Reprinted in *Recent Essays on Truth and the Liar Paradox* (ed. R. L. Martin), 1984, Oxford University Press.

[Lev] H. J. Levesque, 'A logic of implicit and explicit belief', *Proc. National Conf. on Artificial Intelligence*, 1984, pp. 198–202; a revised and expanded version appears as Fairchild Lab. Technical Report FLAIR #32, 1984.

[Lew] D. Lewis, *Convention, A Philosophical Study*, Harvard University Press, 1969.

[Mo] Y. Moses, 'Resource-bounded knowledge', *Proceedings of the Second Conference on Theoretical Aspects of Reasoning about Knowledge* (ed. M. Y. Vardi), Morgan Kaufmann, 1988, pp. 261–295.

[MT] Y. Moses and M. Tuttle, 'Programming simultaneous actions using common knowledge', *Algorithmica* **3**, 1988, pp. 121–169.

IBM Almaden Research Center,
650 Harry Road,
San Jose, CA 95120-6099,
U.S.A.