Order Number 8914769

Knowledge, consensus and communication

Krasucki, Paul J., Ph.D.

City University of New York, 1988

U·M·I

300 N. Zeeb Rd.
Ann Arbor, MI 48106

KNOWLEDGE, CONSENSUS AND COMMUNICATION

by

PAUL J. KRASUCKI

A dissertation submitted to the Graduate Faculty in Computer Science in partial fulfillment of the requirements for the degree of Doctor of Philosophy, The City University of New York.

1988

This manuscript has been read and accepted for the Graduate Faculty in Computer Science in satisfaction of the dissertation requirement for the degree of Doctor of Philosophy.

June 30, 1988
Date

Rohit Parikh
Chair of Examining Committee
Prof. Rohit Parikh

June 30, 1988
Date

Frank S. Beckman
Executive Officer
Prof. Frank Beckman

Prof. Stathis Zachos

Prof. Kenneth McAloon

Prof. Melvin Fitting
Supervisory Committee

The City University of New York

Abstract

KNOWLEDGE, CONSENSUS and COMMUNICATION

by

Paul J. Krasucki

Adviser: Professor Rohit Parikh

We investigate what levels of knowledge of a formula are attainable in a distributed system depending on available types of communication. Systems with ability for synchronous broadcasts are investigated as well as purely asynchronous systems. Levels of knowledge are defined in a language with knowledge operators and also in a language with common knowledge operators. Characterization of attainable levels of knowledge is given extending the results of [PK1].

We also analyze systems where private information is not communicated directly. Instead, processes communicate values of a certain function. Here we give some sufficient and necessary conditions for reaching consensus between processes on the value of this function. Special consideration is given to the case when values of one-bit functions (the decision functions) are communicated. The results of [PK2] are considerably extended.

Contents

Abstract iv

Table of Contents v

List of Figures viii

1 Introduction 1

1.1 The Notion of Knowledge 1

1.2 Syntax and Semantics 4

1.2.1 Common Knowledge 5

1.3 What is Communicated 7

1.4 Wise Men Puzzle 9

1.5 A note about the next two chapters 13

2 Levels of Knowledge 16

2.1 Model of a Distributed System 16

2.2 Definitions 20
2.2.1 Models 20
2.2.2 Language and Semantics 27
2.3 Levels of Knowledge 29
2.4 Embeddability 32
2.5 Downward Closures 35
2.6 Characterization of levels of knowledge 37
2.6.1 Levels of knowledge in L_K 38
2.6.2 Levels of knowledge in L_C 42
2.7 Realizing Levels 43
2.7.1 Asynchronous Case 44
2.7.2 Synchronous Case 51
2.7.3 Both Kinds of Communication Available 55
2.7.4 Limited n-casts 57
2.8 Examples 58
2.9 Applications 60

3 Consensus **62**

3.1 Introduction 62
3.2 Basic Notions 69
3.3 Sure-Thing Principle 76

3.4 Weakly Convex Functions 80

3.5 Strongly Convex Functions 85

3.6 Domination on Worlds 89

3.6.1 Cycles and the lack of consensus 89

3.7 Decision functions 95

3.7.1 Voting Schemes 100

3.7.2 Other two-valued functions 103

3.8 Summary 104

4 Open Problems **106**

4.1 Problem of logical omniscience 106

4.1.1 Resource-bounded reasoning 108

4.1.2 Probabilistic Knowledge 110

4.1.3 Restricted Views 111

4.2 Lack of Knowledge 113

4.3 Other directions for further research 113

Bibliography **115**

List of Figures

1.1 Wise Men Puzzle 11

1.2 Wise Men Puzzle 2 14

3.1 Example of Exchanging Conditional Probabilities Among 2 Agents 67

3.2 Example when Different Protocols give Different Consensus Values 73

3.3 More Informative Function Gives Less Knowledge 76

3.4 Three agents. Sure-Thing Principle Satisfied, No Consensus in a Fair Protocol 79

Chapter 1

Introduction

> Yeh Chueh asked Wang I, "Do you know what is common to all things?" "How should I know?" he replied. "Do you know that you don't know?" "How should I know?" he replied again. "Then are all things not knowable?" "How should I know? Still, let me put it this way: How do you know that what I say I know may not really be what I don't know? How do you know that what I say I don't know may not really be what I know?..." [a]

[a] Chuang Tse [Ch], p.40

1.1 The Notion of Knowledge

What knowledge is - this problem has been discussed by philosophers for ages and no universally accepted answer is in sight. In spite of a lack of agreement on what knowledge is - the notion of knowledge is constantly used not only in everyday English, but also in science.

Economists might say that an agent's choice of strategy depends on his *knowledge*. Computer scientists analyzing correctness of algorithms for distributed systems might say that some process will not enter a critical section unless it *knows* that no other process is going to enter it (see [PR]).

What is an appropriate formal model for knowledge, that is a much more recent problem. Here also there is no agreement among scientists, although there is a certain formalism - knowledge based on information - which turned out to be useful in both computer science and economics. This notion of knowledge we will adopt in this work.

We know that a certain fact A is true, if according to the information we have about the world it must be true. If our information about the world is false, then it will happen that we won't consider the real world possible, and in such case we will know something that is false. These are the things we want to exclude when we talk about knowledge. Therefore we will assume that information accessible to individuals (processes, agents) is always reliable.

To set up some formal framework let's assume that there is some set of possible worlds W. All the properties that agents are interested in are subsets of W. We furthermore assume that if an agent is in a particular world - and knows that he is in this world (can exclude all other worlds as impossible), then he can answer all the queries he might be interested in (the answers are part of the description of the world, and decoding them is feasible). However, if he cannot narrow down the set of possible worlds to a singleton - there is some uncertainty. Still he knows some fact if it is true in

all the worlds he admits as possible - the real world must be among them. Before we are able to define the meaning of "knows" formally, we need to concentrate on information of agents.

Information is the narrowing down of uncertainty. If I receive some information, then I can exclude all the worlds I was admitting so far as possible, which are incompatible with this new information. In other words I can divide my set of possible worlds into two classes - compatible with received information and incompatible. That is *after* the information is received.

What about the moment before? Suppose that I am waiting for some information. I know what "kind" of information I am waiting for, and therefore I can anticipate possibilities. For example knowing the list of participants in a horse race - bookmakers can put odds on all possible winners; if I know that my friend is arriving to New York City on an afternoon plane from San Francisco - I can consider all possibilities (and make plans for every possibility). Anticipated information partitions our set of possible worlds into disjoint sets. After the information is received - the agent can confine himself to one of these sets - the one that is compatible with the information received.

Information anticipated by an agent defines a partition on the set W, which corresponds to some equivalence relation on W. Two worlds w and w' are equivalent for i, if in both w and w' i would have received the same information. Let's denote the agent i's equivalence relation as $\approx_i$ and the corresponding partition of W as P_i.

We will assume some ground language L, s.t. for every $w \in W$ and for

every formula A of L either $w \models A$ or $w \not\models A$. L can be extended to a larger language L^+ which is the closure of L under knowledge operators K_i and the usual truth functional connectives. $K_i(A)$ means intuitively that i knows A.

1.2 Syntax and Semantics

Technically, the notion $w \models A$ for A in L^+ is defined by recursion on the complexity of A.

If A is in L, then the semantics is given. For the truth functional connectives we have the following:

$$w \models \neg A \text{ iff } w \not\models A$$

$$w \models A \vee B \text{ iff } w \models A \text{ or } w \models B$$

For the knowledge operator we have the following:

$$w \models K_i A \text{ iff } \forall w' \approx_i w,\ w' \models A$$

Note that this is exactly the same as Kripke's semantics of the modal operator $\Box$ (neccessity - see [K]) with imposed properties of modal logic S5. The following system of axioms is sound regardless of L:[1]

(1) all (instances of) tautologies.

(2) $K_i(A) \rightarrow A$

(3) $K_i(A \rightarrow B) \rightarrow (K_i(A) \rightarrow K_i(B))$

[1]The soundness of the system means of course that every theorem of it holds at every model - in every set W with every possible partition.

(4) $K_i(A) \rightarrow K_i(K_i(A))$

(5) $\neg K_i(A) \rightarrow K_i(\neg K_i(A))$

(6) If A is an axiom under (1)-(6) then so is K_iA

The only rule of inference is modus ponens:

(R1) Modus Ponens: From A and $A \rightarrow B$ infer B

i in axioms (2)–(6) is arbitrary. From now on, $\vdash A$ will mean that A has a proof in this system. Similarly $\Gamma \vdash A$ where Γ is a set of additional axioms.

If our language is purely propositional then this set of axioms is complete for the class of all models in which the accessibility relation $\approx_i$ used in the semantics of K_i operators is an equivalence (see [P2]).

Even in the propositional case the problem of deciding the satisfiability of a formula is computationally hard. Ladner [Lad] proved that in the case of one knower it is NP-complete, while Halpern and Moses showed that in the case of n knowers, where $n \geq 2$ the same problem becomes PSPACE-complete [HM2].

Of course in case of a richer language the situation gets even worse.

1.2.1 Common Knowledge

When two persons i and j are both present when a certain fact A is announced, then the person i knows A and the person j knows A. But since they were both present, each of them is aware of the fact that the other one knows A (K_iK_jA and K_jK_iA). Also, i knows that j knows that i knows A

and similarly j knows that i knows that j knows A. Carrying that further we can justify that in the described situation for every string x of alternating operators K_i and K_j, xA will be true.

We will call this *common knowledge* of A among i and j.

We can extend the notion of common knowledge to any group of processes U, and we can add to the language common knowledge operators C_U for all groups of processes $U \subseteq \{1, ..., n\}$. Formally the semantics of the common knowledge operator is given by the following definition:

$$w \models C_U A \text{ iff } w \models xA \text{ forall } x \in (\bigcup_{s \in U} K_s)^*$$

Equivalently, we can define common knowledge as the smallest fixed point of an operator Ψ, where $\Psi(A) = A \wedge \bigwedge_i K_i(A)$[2].

So for every A we have:

$$\forall i \in U \; K_i C_U A \text{ iff } C_U A$$

Another characterization of common knowledge follows directly from the semantics of knowledge operators. If we define $\approx_U$ as a meet of all $\approx_i$ for $i \in U$, we will have the following:

$$w \models C_U A \text{ iff } \forall w' \approx_U w, \quad w' \models A$$

It is easy to check that a logic obtained from L by taking the closure of L under all common knowledge operators and all logical connectives (L_C) will have the properties of the modal logic S5, so the axiom system from the

[2]If X is some other fixed point, then $X \rightarrow C_U A$

previous section will remain sound (where in all axiom schemas we replace K_i by an arbitrary C_U). We can add one more axiom:

$$V \subseteq U \quad \rightarrow \quad (C_U \rightarrow C_V)$$

Note that if $C_U A$ is true for a group of processes U, then everybody in U knows exactly the same about the other members of U knowledge of A. No more knowledge of A can be obtained among the processes in U. No communication within U can add any knowledge of A. The fact that U is a group doesn't change anything, the group U behaves like a single process with respect to A.

Common knowledge has been investigated by many people. It is a notion of paramount importance in distributed systems. As it turns out common knowledge is a necessary condition for synchronization of processes in a distributed system. Common knowledge has this property, that it is always achieved simultaneously by all processes involved (see [HM]).

1.3 What is Communicated

Our knowledge increases when we receive some information. This in turn corresponds to a decrease in the level of uncertainty of how the world looks (in which of the possible worlds do we live). This level of uncertainty can decrease in two ways: some event happens (we were not able to predict the result beforehand - e.g. tossing tossed a coin) or we receive some information from outside. This latter case is the case we are going to deal with.

Information is useful only if it is *relevant* for solving some problem a person is interested in. Moreover this relevance must be apparent to the person involved. Suppose that I am interested in the weather forecast for New York City but instead I hear the forecast for Chicago. Most likely (even if there is some connection between the forecast for the New York City and the forecast for Chicago) I will discard the information about the weather in Chicago as totally useless. On the other hand if I am going to Chicago - and it is Chicago weather I am interested in - then the received information will be all that I needed. That's why before we analyze how some given information affects our knowledge we always have to ask ourselves - knowledge of what?

In our work we will talk about the *knowledge of the truth value of a formula.* Moreover, it will always be fixed beforehand *what* formula we have in mind.

Suppose that the formula we are interested in is A. Information received from outside - from somebody else - may be of two kinds. It can be either direct - we can get a message saying "A is true" (or "A is false"), or we may get indirect information. We can learn something that is relevant, but will not give us immediate knowledge about the truth value of A. For example we can learn that B is true. Now if we know that $C \wedge B \rightarrow A$ then if we know C, we will learn A. If we don't know anything about C, the knowledge of B is not going to help us in determining the truth value of A (although in some cases it may happen that it will increase the *likelihood* of A for us, or our *degree of belief* in A, still it will not change our knowledge).

In the following two chapters I will discuss both situations: when A is

communicated directly and when a function is communicated instead.

Let me illustrate both cases using a well known puzzle of men wearing hats - wise men puzzle (the same puzzle is also known as the puzzle of unfaithful wives, and the puzzle of dirty children see [HM], [MDH], [P2]). Incidentally puzzles played a very important role in the research of knowledge - and this particular puzzle is the most celebrated one...

1.4 Wise Men Puzzle

There are three men in a room. Each of them is wearing a hat. A hat may be either blue or red. Every man can see the colors of the other two hats, but cannot see the color of his own hat. Every man is supposed to find out the color of his hat.

Let's look at a model of that situation. In general there is 8 possible states of the world - eight triples of the form $(a_1a_2a_3)$ where each of a_i can be either b or r (to be interpreted as "a_i is b if man number i is wearing a blue hat", similarly r stands for a red hat).

The men do not directly communicate the colors of the hats they see. Instead they say "yes" if they know the colors of their own hats, "no" if they don't. For every man, the information given is of the form "the two other men have the following colors of their hats". Therefore, in every world for every man there is another possible world, equivalent to the real one. In this other world he sees the same thing, the only difference is in what he cannot see, the color of the hat on his own head. E.g. if 1 sees a blue hat on 2's

head and a red hat on 3's head, the two worlds rbr and bbr are possible for him.

In the puzzle it is announced in the presence of all 3 men that at least one of them is wearing a blue hat. This public announcement creates common knowledge among the 3 men of the fact that there is only 7 possible states of the world. The triple rrr can be excluded.

The reasoning of the agents in case they all are wearing blue hats is given in figure 1.1. (It can be shown that the whole puzzle generalizes to the case of n people for any n).

The key information which enables the third man to find out the color of his hat is that he knows that man 2 knows that man 1 knows that rrr cannot occur. Formally - $K_3K_2K_1A$, where A is "the real world cannot be rrr ". This is true since fact A was announced publicly when all 3 men were present. If instead of a public announcement three letters were sent - one to each man saying: " the real world cannot be rrr" then the third man couldn't legitimately carry out the same reasoning. It is now conceivable for him that he is the only one who received such a letter, so others don't know that the world rrr is impossible. His situation is not much better for him if we change the letters so they will read: " the real world cannot be rrr, moreover the same message is sent to all three men". If our mailing system is not perfect (messages may be lost or delayed) then there will be some doubt in the third man whether the others received relevant information on time or not, and he will not be able to figure out the color of his hat.

Generally if I receive a message "A" then I know A (I will always assume

1st man's partition of the set of possible worlds	bbb,rbb	bbr,rbr	brb,rrb	brr	Class bbb,rbb corresponds to the situation where 1 sees two blue hats. His hat may be either blue or red.
1st man's announcement	no	no	no	yes	First man knows the color of his hat only if he sees two red hats.
2nd man's partition of the set of possible worlds	bbb,brb	rbb,rrb	bbr,brr	rbr	
2nd man's announcement	no	no	yes	yes	Second man knows the color of his hat if he sees two red hats or if the first man would have made different announcements in the worlds in his equivalence class.
3rd man's partition of the set of possible worlds	bbb,bbr	rbb,rbr	brb,brr	rrb	
3rd man's announcement	yes	yes	yes	yes	

Let's suppose that the real world is bbb (all three men are wearing blue hats). Then, when the first person says "I don't know" (no), then the second person doesn't learn anything. If the situation had been brb also the first person wouldn't have known. But if the real world were bbr then the second person would have known that his hat is blue - after hearing "no" from the first person he would exclude the world brr (in that world the first person would have said "yes"). So bbr cannot be the case, therefore the third person learns that the real world is bbb.

Figure 1.1: Wise Men Puzzle

that the messages are true, moreover their truth value once true - will remain true, formulae are persistent). But if there is some uncertainty about the delivery of a message, then the sender will not know that the message was delivered (if i is a sender and j is a receiver then K_iA, K_jA will both be true but K_iK_jA - meaning that i knows that j knows that A is true - will not be true) until he receives an acknowledgment from i. We will investigate in chapter 2 how the knowledge of one process of the knowledge the other process... of the knowledge of still some other process of a formula A changes when processes communicate A (and what do they know about it) to each other.

It is apparent that there is a certain notion of a *level of knowledge* implicit here. This notion will be defined formally in chapter 2. We will investigate, extending slightly the results of [PK], how the attainability of the levels of knowledge depends on the means of communication available in the system.

To anticipate what we will do in chapter 3 let me go back to the Hats Puzzle. As you can see in figure 1.1, no matter what the real situation is, the third person will know it: in every case, he will be able to exclude one of the possibilities, so his equivalence relation becomes equality. Notice that we assumed here that everybody listens to what everybody else is saying - we have an "auction" set-up. What if the first man tells secretly to the second man (so the third man doesn't hear) whether he knows or not. Notice that in such a case also the third man will know the color of his hat! The fact whether all the communications were public or private (one to one) - turned out to be irrelevant! But this is not always the case.

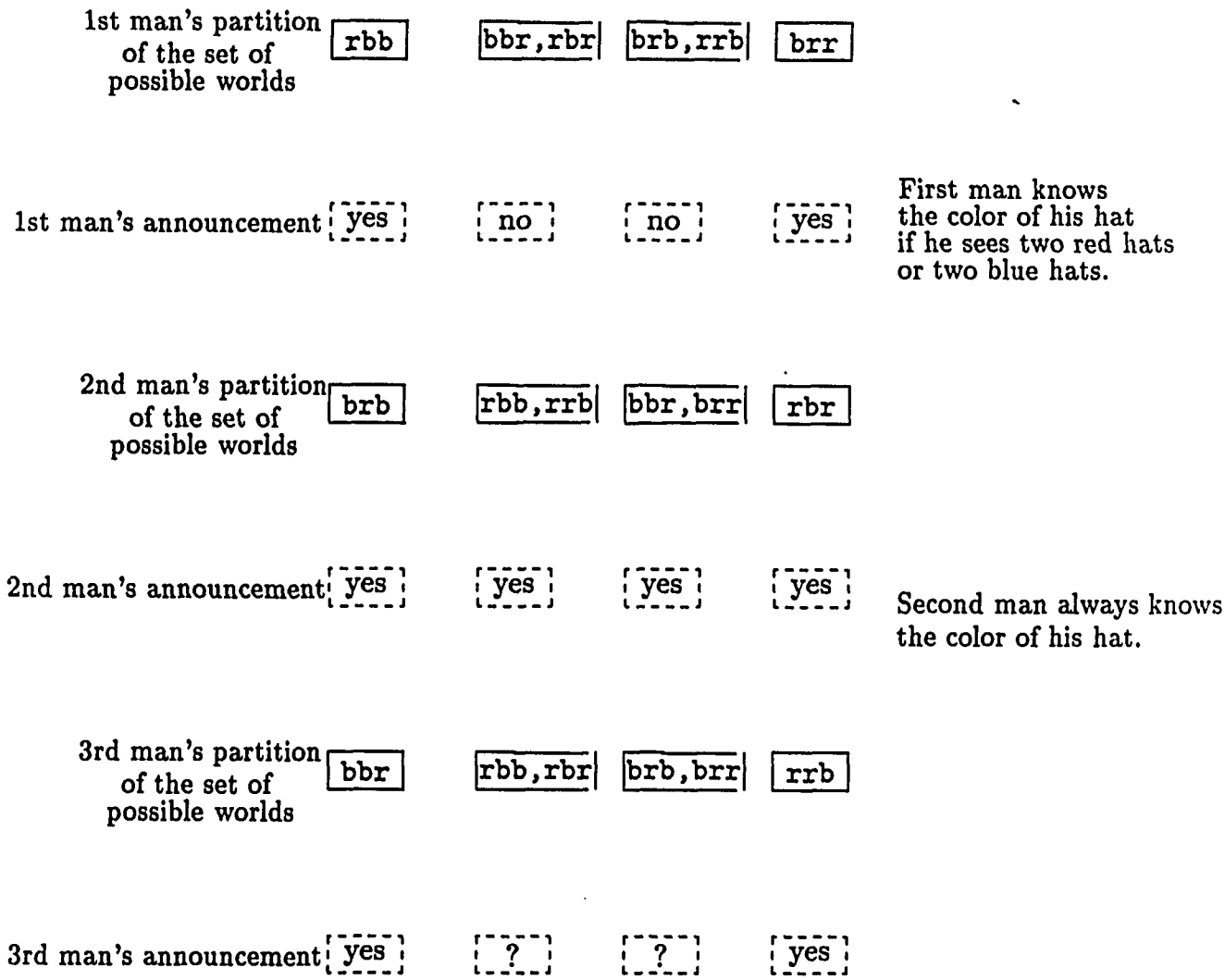
Let's try to modify our example. Let's assume that the initial announcement is "not all three hats are of the same color". In such a situation two worlds are excluded: bbb and rrr. We can see the new situation in figure 1.2.

In this puzzle everybody communicates a different function (for the same set of possible worlds it may happen that one person knows what the color of his hat is, and the other person doesn't). We will deal with simpler cases when all persons communicate values of the *same* function. In this set up an example of such a function would be - person says "yes" if he knows that there is at least one person who knows what the color of his hat is. It is easy to see that for such a function after somebody says "yes". everybody would say "yes". In such a case *consensus* would have been reached, though the individual knowledge would remain different. That is the kind of a problem we will deal with in the chapter 3.

1.5 A note about the next two chapters

The main results of this thesis are contained in chapters 2 and 3. Much of the work done here originated from two papers ([PK1] and [PK2]) written jointly with professor Rohit Parikh.

In chapter 2 there is an extension of the research started with [PK1]. The notion of the level of knowledge originally defined in [PK1] is generalized to a language with common knowledge operators. The model of a distributed system introduced here is a general formal model for the systems where all the information that processes can gain is via communication.



Here "?" indicates that the answer depends on whether the third person has heard what the first person said. In the case when the communication was public - the third person has heard what the first person said - then "?" should be replaced by "yes". In case of private communication - "?" should be replaced by "no". Notice also that in case of public communication, the third person will always say "yes", which in turn gives no information to the first person. In case of the "one to one" communications the third person might not know what the real world is, but if he communicates back whether he knows or not to the first person - the first person will learn about the real world! The two cases are entirely different.

Figure 1.2: Wise Men Puzzle 2

In chapter 3 the model used is exactly the model from [PK2]. The results from [PK2] are followed by new results which give some necessary and some sufficient conditions for reaching consensus where processors communicate values of a function defined on the set of subsets of possible worlds. Special consideration is given to the least informative non-trivial functions, 0-1 functions. These functions in a computer science set-up would correspond to the situation where messages are 1 bit long.

Since there are some results included in this thesis which were obtained jointly with prof. Parikh, these results are marked with (j).

Chapter 2

Levels of Knowledge

2.1 Model of a Distributed System

In order to deal with knowledge in distributed systems, we need first to specify exactly what we mean by a distributed system, describe our model. First, we give an informal description of the model we are going to use. Precise definitions will follow in the next section.

We assume there are a finite number of processes, $1, ..., n$, which compute and communicate with each other by either asynchronous messages or by broadcasts. Our network is fully connected (there is a channel from every process to every other process).

Asynchronous communication consists of two phases: send and receive. All messages sent are ultimately delivered (and they are delivered in the order in which they were sent) but the delay (transmission time) may be arbitrarily long.

Broadcasts are fully reliable, synchronous communications[1] where all processes involved simulteneously receive the message sent by one of them.

We consider three kinds of systems - systems where only synchronous communication is available, systems where only asynchronous communication is available and systems where both kinds of communication are available. For simplicity I will assume that these are the only means of communication.

There is assumed to be a global clock in the background which orders all events, but this clock is not available to the processes. Time is discrete and the clock is set to 0 at the beginning of every computation and grows in increments of one.

We want to make sure that for every processor in our system there are facts known initially only to this processor. To accomplish that we assume that every process starts its computation with some initial value, a string (finite) of 0's and 1's. This initial value may correspond to some local nondeterministic input, e.g. the result of a sequence of coin tosses by a processor.

At each moment of global time, zero or more events may take place, at most one at each process. This finite set of *local events* constitutes a *global* event and a *global history* is simply a possible sequence of global events. What global histories are possible is determined by the programs of the various individual processes as well as by the properties of the means of

[1]The two kinds of communications can be looked at as two kinds of communication media e.g. mailing system (asynchronous) and telephone lines (synchronous). Since we allow for synchronous communication between more than two processes at a time, our telephone system must have "conference call" capability.

communication. The *protocol* **P** is just the set of all possible global histories[2] closed under the prefix operation. **P** corresponds to the set W of possible worlds in the general definition. Worlds $w \in W$ are histories $H \in \mathbf{P}$.

Our model is similar to that used by Parikh & Ramanujam ([PR]). The main difference is that we assume that there is no global clock in the system accessible to the processes ("common clock" in [PR]). They do not put this restriction. In [PR] terminology, our local histories are always individual views of "time-free" global histories.

It is also very similar to the model used by Chandy and Misra, but we allow different events to occur locally in different sites at the same instant of time. This can be interpreted in two ways: either we can treat time in our system as "less refined" then in [CM], or we can interpret our system as synchronous, events in all sites are governed by the same global clock.

Even assuming that our system runs in synchronous rounds, we ensure that processes are not able to record it. That is because if a process is inactive in a certain round (his action is "null" event), it cannot perceive it. Consequently he cannot draw any conclusions about the others after observing his local clock (counting his local events).

We allow for broadcasts which are immediate communications in which

[2]The way we use the word "protocol" is the same as in [PR]. Let me quote:

> There may be some finite mechanism for ensuring that no history outside the protocol is actually ever possible and usually it is this mechanism that is called the protocol, but we shall prefer to work abstractly and hope that the reader will forgive this transgression. This decision is really analogous to the decision in mathematics to study sets rather than properties.

a group of processes is involved. It is similar to CSP communication, only it allows more then one recipient at a time.

Since immediate communications may seem unrealistic let me give a possible implementation:

Let's assume that computation is done in synchronous rounds. There are *tokens* in the system which enable processes to broadcast. Assigning of tokens is done by some token manager which makes sure that in every round it is impossible for a processor to participate in 2 broadcasts. In every round the system executes the following sequence of actions:

1. Some processors wake up.
2. Processors with tokens may do a broadcast. If they do, messages in the buffers connecting broadcasting processor with the recipients of the broadcast are reordered so that the broadcast becomes first in all the buffers.
3. Processors who are recipients of a broadcast wake up and read the message.
4. Processors which are not recipients of a broadcast may either read a message from a buffer or send a message or perform some local computatons or finally may sleep.
5. Tokens are reassigned.

A global event in our model is a description of all the local actions taken by all the processors in a single round.

2.2 Definitions

2.2.1 Models

Here we formally specify our class of models. Let $N = \{1, ..., n\}$ be the set of all processors. Every processor i has infinitely many possible initial states v. Every initial state is a string of 0's and 1's ($v \in \{0,1\}^*$). The set of initial states for i we denote by V_i. The set of global initial states is $\mathcal{V} = \times_{i=1}^{n} V_i$.

From now on we will use lower case letters to denote everything pertaining to a single process. Capitals will be used where all the processes are involved (e.g. v_i is an initial state of a processor i, while V is an initial configuration of the whole system: $V = (v_1, ..., v_n)$).

Events: E_i denotes the set of all events in which processor i can participate (events *local* to i). There are the following types of events (or actions):

1. L_i: Local computation steps,
2. $s(i,j,m)$: Sending a message m to a processor j, $j \in N$.
3. $r(j,i,m)$: Receiving a message m from a processor j, $j \in N$.
4. $bc(i,U,m)$: Sending a broadcast m to a group of processors U, $i \in U \subseteq N$. The same event is receiving a broadcast m by a group of processes U.

$$E_i = L_i \cup \{s(i,j,m) | m \in M,\ j \in N\} \cup \{r(j,i,m) | m \in M,\ j \in N\}$$
$$\cup \{bc(j,U,m) | m \in M,\ i,j \in U \subseteq N\} \cup \{bc(i,U,m) | m \in M,\ i \in U \subseteq N\}$$

We define the set of *global events* $\mathbf{G}$ in our system. $\mathbf{G} \subseteq \times_{i=1}^{n}(E_i \cup \{null\})$ (a cartesian product) s.t. if $(e_1, ..., e_i, ..., e_n) \in \mathbf{G}$ for some i and $e_i = bc(j, U, m)$ then for all $i' \in U$, $e_{i'} = bc(j, U, m)$. If $e_i = null$ for some i, it means that there is no local event at i at this point. Note that *null* is *not* local to any process. We use the notation $(G)_i$ to denote the ith coordinate of G, so $(e_1, ..., e_i, ..., e_n)_i = e_i$.

Histories: A history (a run) is an input value followed by a sequence of events. Let's call the set of all possible histories of the system - protocol $\mathbf{P}$. So $\mathbf{P} \subseteq V; \mathbf{G}^*$. Protocols are always closed under taking an initial segment of a history: $H \in \mathbf{P}$ implies that every H' which is an initial segment of H is in $\mathbf{P}$.

We will require that for every receive in every history in every protocol there is exactly one corresponding send and it occurs before receive (this condition we will call *time-consistency*).

We say that two histories H and H' are *compatible* iff they start with the same input values.

We can define the *concatenation* of compatible histories:
If $H_1 = V; G_1; \ldots; G_k$, and $H_2 = V; G'_1; \ldots; G'_l$, then H is the concatenation of H_1 and H_2 iff $H = V; G_1; \ldots; G_k; G'_1; \ldots; G'_l$,

Local histories are the projections of global histories onto the sets of local events of the processors. They are "time-forgetting".

We assume that a global event–the ticking of the clock–takes place even if no local events take place at a particular moment (this corresponds to events

of the form $(null, ..., null)$, and so the length of a global history is just the amount of time elapsed. However, what each process sees at any moment of time is its local event, if any, and its local history is simply the sequence of local events. Given i, and the global history H, the local history h_i is uniquely defined and we let Φ_i be the map which takes us from H to h_i.

We can inductively define Φ_i's as follows:

$$\Phi_i(H;G) = \begin{cases} \Phi_i(H); e_i & \text{iff } (G)_i = e_i \in E_i \\ \Phi_i(H) & \text{iff } (G)_i = null \end{cases}$$

where $\Phi_i((v_1, \ldots, v_n)) = (v_1, \ldots, v_n)_i = v_i$.

So Φ_i is like $(\cdot)_i$ (when we extend projection to histories) only it erases *null* events.

Local history is everything the processor sees, so all the global histories which correspond to the same local history h_i look the same to the processor i. Note that the length of $\Phi_i(H)$ is less than or equal to the length of H. In fact $length(\Phi_i(H)) = length(H)$ iff there are no *null* events on i in H.

For every i we can define an equivalence relation on the set of global histories:

$$H \approx_i H' \text{ iff } \Phi_i(H) = \Phi_i(H')$$

We use capital letters to denote global histories, events etc, lower case letters do denote local histories, events etc.

Time: *Global time* of an event G in a global history H, $Time(G, H)$, can be defined as the length of the initial segment of H up to (and including) G (this is the time when the event G has occurred in the history H. Of

course if the same event occurred more then once we have to specify which occurrence we are interested in).

Note that since $null \notin E_i$ for all i, then processes do not have access to the global clock. They can introduce their own local logical clocks, but these clocks do not have to coincide. *Local time* of e_i in h_i can be defined as $time(e_i, h_i)$ – the length of h_i up to (including) e_i.

The lack of the access to the global clock, together with the closure conditions for the protocol guarantee that the only "possible causality" ordering that can be defined corresponds in case of asynchronous systems to Lamport's [Lam] "happened before" ordering[3].

Messages: We require our processes to be "honest" - they only send (or broadcast) messages which they *know* are true. Messages m are knowledge formulae (formulae of the form xA where x is a string of knowledge operators $C_{U_1} C_{U_2} \ldots C_{U_n}$ and A is some boolean combination of $P_1, \ldots P_n$). Formally, if $\Phi_i(H) = h_i; s(i, j, B)$ or $\Phi_i(H) = h_i; bc(i, U, B)$ and $H \in \mathbf{P}$, then if $\Phi_i(H') = h_i$ then $H' \models K_i B$ (semantics of K_i, C_U is given in the following section).

Closure Conditions for the Protocol: We impose some additional conditions on the protocol **P**. We want to ensure that the initial state of i (v_i) cannot be *known* to any other process j at any run of the system, unless j *learns* about v_i from some communication. We want to exclude the possibility that something is common knowledge "accidentally". To achieve

[3] $e_1 \rightarrow e_2$ iff e_1 is send, e_2 is receive of the same message, or e_1, e_2 are local to the same process and e_1 occured earlier than e_2. e_1 happened before e_2 iff $e_1 \rightarrow^* e_2$ where $\rightarrow^*$ is the reflexive, transitive closure of $\rightarrow$. In a system where we allow broadcast one more condition is needed in the definition of $\rightarrow$. $e_1 \rightarrow e_2$ if e_1 and e_2 are two local projections of the same broadcast.

that we will make sure that all the initial states are possible. Moreover, if v_i is the initial state of i, all other strings v_i' will remain possible for j as initial states of i, unless j gets some message from i to the contrary (directly or via some other processors).

1) All vectors of input values are possible: $\forall V$ s.t. $V = (v_1, \ldots, v_n)$ where every v_i is a sequence of 0's and 1's there is some $H \in \mathbf{P}$ s.t. for some H', $H = V; H'$.

2) No sequence of local events on some group of processes can influence possible actions of some other group of processes unless there are some communications (of course assuming that both groups are disjoint).

For that we need some closure conditions on the set of all protocols. The first condition can find in [CM] as the first of their ***principles of computation extension*** .

We need one definition:
Let $G = (e_1, \ldots e_n)$, G is *on* U if $U = \{i | (G)_i \neq null\}$ (so U is the set of processes which have some local events in G).

Closure conditions:

(i) **Extension Rule:**
Let $\forall i \in U$, $H \approx_i H'$, none of e_i is receive $r(j, i, m)$ for any j not in U, then

$$(H' \in \mathbf{P},\ H; G \in \mathbf{P}) \quad \Rightarrow \quad H'; G \in \mathbf{P}$$

The extension rule guarantees that if we have a protocol **P**, some history H in **P** and some action of a group of processes U is possible in H, then the same action must be possible in every history H' which looks the same

to all processes in U unless it violates time–consistency. In order to explain why e_i cannot be a receive from the processor outside of U let us examine an example:

Let $N = \{1,2,3\}$, $U = \{1,2\}$.
$H = (null, null, s(3,1,m))$, $H' = (null, null, null)$. Clearly $H \approx_1 H'$ and $H \approx_2 H'$. If we take $G = (r(3,1,m), null, null)$ s.t. $H;G \in \mathbf{P}$ then requiring $H';G$ to be in $\mathbf{P}$ would violate time–consistency.

The following conditions assure that no process can get any additional information about the other processes by observing its own local events (no hidden synchronization). These conditions are necessary because (unlike [CM]) we allow local events at different sites at the same instant of time. Condition (*ii*) says that if some local events have occurred in parallel, and the sets of participating processes were disjoint, they could have occurred in sequence. We'll call it the splitting rule.

(ii) **Splitting Rule:**
$G = (e_1, \ldots e_n)$, $G \notin V$, G is on U. Given U_1, U_2 s.t. $U_1 \cup U_2 = U$ and U_1, U_2 disjoint, then we can "split" any G into G_1 and G_2:

$$(H;G \in \mathbf{P}) \;\Rightarrow\; H;G_1;G_2 \in \mathbf{P}$$

where $(G)_i = (G_1)_i$ for $i \in U_1$, $(G)_i = (G_2)_i$ for $i \in U_2$, $(G_1)_j = null = (G_2)_k$ for $j \notin U_1$, $k \notin U_2$ provided that we don't split any broadcasts: $(G)_i = bc(i,V,m) \rightarrow V \subseteq U_1 \vee V \subseteq U_2$.

Condition (*iii*) says that if some local events have occurred in sequence, the sets of participating processes were disjoint, and there was no send receive pair in them, they could have occurred in parallel.

(iii) **Joining Rule:**

Given U_1, U_2 s.t. $U_1 \cup U_2 = U$ and U_1, U_2 disjoint, Let G_1 be on U_1, G_2 on U_2, and if $(G_1)_i = s(i,j,m)$ then $(G_2)_j \neq r(i,j,m)$.

$$(H; G_1; G_2 \in \mathbf{P} \quad \Rightarrow \quad H; G \in \mathbf{P})$$

where $(G)_i = (G_1)_i$ for $i \in U_1$, $(G)_i = (G_2)_i$ for $i \in U_2$.

We will show that in fact there is no "accidental" learning of private facts. These facts can be learned by a processor only when it receives a message (either a broadcast or an asynchronous message). We have the following:

Proposition 1: If $H \not\models K_i P_j$, $(G)_i = e$ where e is neither a receive nor a broadcast (where i is not a sender), then $H; G \not\models K_i P_j$.

Proof: If $H \not\models K_i P_j$, then $\exists H' \in \mathbf{P}$ $H \approx_i H'$ $H' \not\models P_j$.
We can split G into G_1 and G_2 so that $(G_1)_i = e$, $(G_2)_i = null$, and for all $j \neq i$, $(G_1)_j = null$, $(G_2)_j = (G)_j$ (splitting rule).
$H; G_1; G_2 \in \mathbf{P}$, $H; G_1; G_2 \approx_i H; G$
$H'; G_1 \in \mathbf{P}$ (extension rule), $H'; G_1 \approx_i H; G_1$, $H'; G_1 \not\models P_j$
$H'; G_1 \approx_i H; G_1 \approx_i H; G_1; G_2 \approx_i H; G$ □

Systems: We consider three kinds of systems. Asynchronous systems are the systems as described above but *without broadcasts.* So in asynchronous systems the only communications are via send and receive. Synchronous systems are the systems in which all the communications are done using broadcasts. Here we *don't have events send and receive.* Finally, we use the name mixed communications systems for the systems with both kinds of communications available.

2.2.2 Language and Semantics

Let L_0 be a language which describes properties of the global histories in a protocol $\mathbf{P}$. So for every sentence A in L_0, and for every history $H \in \mathbf{P}$, A is either true or false in H.

We want to make sure that in every history initially every processor has some "private" information not known to any other processor. To accomplish that we assume, that we have in our language a countable set of propositions $L_1 = \{Q_{i,j}\}_{i,j \in N}$. $Q_{i,j}$ is the proposition that the jth input value of i is 1. All $Q_{i,j}$ are independent. Private information of i in H are $P_{i,j}$ which are $Q_{i,j}$ or its negation depending on whether $Q_{i,j}$ is true in H or not. Note that the private information is not *a truth value* of any formula, but *which* formula we're looking at.

L is the closure of L_0 under truth functional connectives. L can be extended to a larger language L_C which is the closure of L under common knowledge operators C_U (for $U \subseteq N$) and the usual truth functional connectives. $C_U(A)$ means that there is common knowledge of A among processes from U.

Often we talk about the knowledge of a single process, such a case corresponds to $C_{\{i\}}$. We will then use the notation[4] K_i for $C_{\{i\}}$. When we restrict ourselves to a subset of L_C in which all common knowledge operators are in fact the knowledge operators (the sets U in C_U are always singletons) then we use the notation L_K.

[4]Fact that $C_{\{i\}} = K_i$ was noticed earlier, compare e.g. [FI]. It is important that we assume that L_K and L_C are S5 (we need at least S4).

The class of all models we are going to consider is the class of all protocols **P** as described in the previous section. Let's fix **P**. Now we define the notion $H \models A$ for A in L^+ by recursion on the complexity of A.

0) If A is from L_0 then the semantics is given.

1) If A is $Q_{i,j}$ then A is true in H if the jth bit of an input of processor i in H is 1:

$$H \models A \quad \text{iff} \quad H = (v_1, \ldots, v_n); H', \ (v_i)_j = 1$$

2) The case for the truth functional connectives is clear. If A is $\neg A'$ then

$$H \models A \quad \text{iff} \quad H \not\models A'$$

If A is $B \vee C$ then

$$H \models A \quad \text{iff} \quad (\ H \models B \ \text{ or } \ H \models C \ )$$

3) If A is of the form $C_U(B)$, then A is true in H iff A is true in every H' s.t. there is a chain of processors $i_1, ..., i_k$ and a chain of histories $H = H_1, ..., H_{k-1}, H_k = H'$ s.t. for every $1 \leq j < n$ $H_j \approx_{i_j} H_{j+1}$.

Also, we can define a new equivalence relation:

$\approx_U$ as $H \approx_U H'$ iff $\exists i \in U \ H \approx_i H'$.

Then, if $\approx_U^*$ is defined as the reflexive, transitive closure of $\approx_U$[5], then we have the following equivalent semantics for the common knowledge operator:

$$H \models A \quad \text{iff} \quad \forall H' \in \mathbf{P} \quad H \approx_U^* H' \quad \rightarrow \quad H' \models B$$

[5]Note that $\approx_U^*$ is the meet of all $\approx_i$.

If U is a singleton then we have semantics for knowledge operators:

$$H \models K_i A \quad \text{iff} \quad \forall H' \in \mathbf{P} \quad H \approx_i H' \quad \rightarrow \quad H' \models A$$

Also if U is empty, then $C_U A$ iff A.

Notice that $\approx_i$ is an equivalence relation, and we have the same semantics for the knowledge operator as in the general case. Moreover $\approx_U^*$ is also an equivalence relation. So S5 axioms hold.

In our constructions we will often use "private facts" P_i. They have the following important property: $H \models K_i P_j$ iff $H \models P_j$ and

$$\forall V_1, V_2, H_1, H_2((V_1; H_1 \approx_i H \ \wedge \ V_2; H_2 \approx_i H) \rightarrow (V_1)_j = (V_2)_j)$$

The m^{th} private fact $P_{i,m}$ of a processor i is whether $Q_{i,m}$ or its complement is true. In our system every processor may have any number of private facts.

2.3 Levels of Knowledge

Consider a formula A and some global history H. The formula may be true but not known to anyone. In that case we have $H \models A$ but not $H \models C_U(A)$ for any non–empty U. Or perhaps it may be known to some i that A is true. In the latter case, $K_i(A)$ is true. The formula $K_j K_i(A)$ expresses the stronger assertion that not only is A known to i, but also that this fact is known to j, (it is known to j, that it is known to i, that A is true). Note that $K_i(A)$ stands for $C_{\{i\}}(A)$ and $K_j K_i(A)$ stands for $C_{\{j\}} C_{\{i\}}(A)$. Still more is known in the system if among i and j everybody knows, that everybody knows,

... , that everybody knows A. This is common knowledge of A between i and j which is denoted by $C_{\{i,j\}}(A)$. Thus it is clear that a formula that is known, in some sense, may be known at a higher or lower level. The highest possible level of knowledge here is $C_N(A)$, A is *common knowledge for the whole group*, which holds if for all strings x of knowledge operators xA holds. We shall give now the precise definition of the level of knowledge, but first let's look at the set $T(A,H)$ of all strings of knowledge operators x s.t. xA is true in H:

$$T(A,H) = \{x | x \in \Sigma_C^* \text{ and } H \models xA\}$$

where $\Sigma_C = \{C_U\}_{U \subseteq N}$. If we replace Σ_C by $\Sigma_K = \{K_1, K_2, ..., K_n\}$, we use the notation $T_K(A,H)$. If there is some non-empty string x_0 in $T(A,H)$ then $T(A,H)$ is infinite ($|T(A,H)| = \aleph_0$) and $T_K(A,H)$ is also infinite. This is a consequence of the following theorem:

Theorem 1 : Let Σ_C be the alphabet whose symbols are $\{C_U\}_{U \subseteq N}$ For all x, y, in Σ_C^*, and all formulae A, for all H, $V \subseteq U \subseteq N$, $H \models xC_UC_VyA$ iff $H \models xC_VC_UyA$ iff $H \models xC_UyA$.

Proof: Notice that if $V \subseteq U \subseteq N$, then $H' \approx_U^* H'' \approx_V^* H$ iff $H' \approx_V^* H'' \approx_U^* H$ iff $H' \approx_U^* H$. Therefore $\forall H \quad H \models C_UC_VyA$ iff $H \models C_VC_UyA$ iff $H \models C_UyA$. Now by the necessitation rule, for all H, all $a \in \Sigma$, $H \models aC_UC_VyA$ iff $H \models aC_VC_UyA$ iff $H \models aC_UyA$. Using induction on the length of x we get: $H \models xC_UC_VyA$ iff $H \models xC_VC_UyA$ iff $H \models xC_UyA$ for all $x \in \Sigma^*$. □

We can obtain the following as a corollary, but I will prove it directly using a syntactic argument.

Corollary 1 (j): Let Σ_K be the alphabet whose symbols are $\{K_1, \ldots, K_n\}$ For all a in Σ_K, and for all x, y, in Σ_K^*, and all formulae A,

$$\vdash xayA \leftrightarrow xaayA$$

and hence for all H, $H \models xayA$ iff $H \models xaayA$. I.e. repeated occurrences of a are without effect and if $xay \in T_K(A, H)$ then $\forall n\ xa^n y \in T_K(A, H)$.

Proof: By axiom 4 and axiom 2 $\vdash ayA \leftrightarrow aayA$. It is easy to show that because of the axiom scheme (6) the following limited necessitation rule works:

If $\vdash A$ then $\vdash K_iA$ for any i.

Applying this rule, we have $\vdash K_i(ayA \leftrightarrow aayA)$. Using axiom 3 we obtain $\vdash K_iayA \leftrightarrow K_iaayA$, so by induction we get for all x, in Σ^*, $\vdash xayA \leftrightarrow xaayA$.

□

Since occurrences of substrings of the form C_UC_V or C_VC_U don't carry any more information than strings C_U (if $V \subseteq U$), we define levels of knowledge of formulae by excluding all strings containing consecutive occurrences of two knowledge operators indexed by sets of which one is included in another.

Definition : Given a formula A and a history H, the *level* of A at H, $L(A, H)$ is the set of x in Σ_C^* such that $H \models xA$, and x contains no substrings C_UC_V, C_VC_U for any $V \subseteq U \subseteq N$.

If H is clear from the context, or not important, then we shall drop it as a parameter. If we restrict ourselves to the K_i operators, we denote the level of A in H by $L_K(A, H)$.

2.4 Embeddability

Now we will try to characterize levels of knowledge. First we need to introduce the embeddability ordering on strings which turns out to be important here.

Definition : Given two strings x and y, we say that x is *embeddable* in y ($x \leq y$), if all the symbols of x occur in y, in the same order, but not necessarily consecutively. Formally:

1) $x \leq x$, $\epsilon \leq x$ for all x

2) $x \leq y$ if there exist x', x'', y', y'', $(y', y'' \neq \epsilon)$, such that $x = x'x''$, $y = y'y''$, and $x' \leq y'$, $x'' \leq y''$.

and $\leq$ is the smallest relation satisfying (1) and (2).

Thus the string *aba* is embeddable in itself, in *aaba* and in *abca*, but not in *aabb*.

Properties of the embeddability relation $\leq$

Fact 1: Embeddability is a *well partial order*, i.e. it is not only well founded, but every linear order that extends it is a well order (equivalent condition: it is well founded and every set of mutually incomparable elements is finite).

Fact 2: Embeddability can be tested in linear time.

For a proof of fact 1 see [H]. Fact 2 is straightforward.

We also need a stronger relation defined on Σ_C^*, which we call *C-embeddability.*

Definition : Given two strings x and y, we say that x is *C-embeddable*

in y ($x \preceq y$), if

1) If $V \subseteq U$ then $C_V \preceq C_U$

2) $x \preceq y$ if there exist x', x'', y', y'', $(y', y'' \neq \epsilon)$, such that $x = x'x''$, $y = y'y''$, and $x' \preceq y'$, $x'' \preceq y''$.

and $\preceq$ is the smallest relation satisfying (1) and (2).

Fact 3: For any $x, y \in \Sigma^*$, $x \leq y$ iff $x \preceq y$.

Fact 4: C-embeddability is a well partial order.

Fact 3 is easy. It is also easy to check that C-embeddability is a partial order. It is well founded, because regular embeddability is well founded and for given $x \in \Sigma_C^*$ there are only finitely many $y \in \Sigma_C^*$ s.t. $|x| = |y|$ and $y \preceq x$. There are only finitely many incomparable elements in Σ_C^* with respect to $\leq$, and there are more incomparable elements with respect to $\leq$ than with respect to $\preceq$, so $\preceq$ is a well partial order. □

If $\leq$ is a partial order on S, we can define a notion of a *downward closed* subset of S:

Definition : $R \subseteq S$ is downward closed iff $x \in R$ implies $\forall y \leq x \;\; y \in R$.

We will look at downward closed sets with respect to embeddability and C-embeddability.

Theorem 2 : Let Σ_C be the alphabet whose symbols are $\{C_U\}_{U \subseteq N}$. Then for all strings x, y Σ_C^*, if $x \preceq y$ then for all histories H, if $H \models yA$ then $H \models xA$.

Proof: We use induction on the sum of lengths of x and y. If the sum is 0, then the lemma is immediate. Otherwise y must be nonempty and let

y be $C_U y'$. Now either $x \preceq y'$ or x is $C_V x'$ ($V \subseteq U$) and $x' \preceq y'$. In the first case yA implies $y'A$ which (by induction hypothesis) implies xA. In the second case, $y'A$ implies $x'A$ by induction hypothesis, and therefore by necessitation $C_V y'A$ implies $C_V x'A$, so since $C_U y'A$ implies $C_V y'A$, we get yA implies xA. □

Corollary 1 : Every level of knowledge is a downward closed set with respect to $\preceq$. □

Corollary 2 (j)**:** Let Σ_K be the alphabet whose symbols are $\{K_1, ..., K_n\}$. Then for all histories H and all x,y in Σ^* , if $x \leq y$ and $H \models yA$ then $H \models xA$. □

Corollary 3 (j)**:** Every level of knowledge in L_K is a downward closed set with respect to $\leq$. □

Note also the following:

Corollary 4 (j)**:** The complement of every level of knowledge is upward closed with respect to the order $\preceq$ ($\leq$ if we are restricted to L_K). □

So far we have a necessary condition for the set of strings of knowledge operators to be a level of knowledge of some formula in some history. So, for example we know that there is no formula A and no history H, s.t. $H \models K_2K_1A$ and $H \models \neg K_2A$. This is because if $K_2K_1 \in L(A, H)$ then since $L(A, H)$ is downward closed, and K_2 is in the smallest downward closed set of strings containing K_2K_1, therefore it must be in every downward closed set of knowledge strings containing K_2K_1, so it must be in $L(A, H)$. This shows the importance of the notion of the *smallest downward closed* set of

strings of knowledge operators including given set X. We will call this set a *downward closure* of A (we denote downward closure of X by $dc(X)$). Since downward closure is an important notion, we start by investigating some properties of the operation of downward closure.

2.5 Downward Closures

Definition: The downward closure of $X \subseteq \Sigma^*$ is the smallest (with respect to inclusion) downward closed set containing X. Let's denote the downward closure of X by $dc(X)$.

Properties of Downward Closure

In fact 5 we can replace $\preceq$ by $\leq$ (case of L_K) and it remains true. Also in facts 6 and 7, downward closure can be taken with respect to either $\leq$ or $\preceq$. In fact 8 we are talking about $\leq$.

Fact 5: If A, B are downward closed then $A \subseteq B$ iff for every $a \in A$ there is $b \in B$ s.t. $a \preceq b$.

Fact 6: $dc(A \cup B) = dc(A) \cup dc(B)$

Fact 7: $dc(AB) = dc(A)dc(B)$ (the downward closure of the concatenation of sets is equal to the concatenation of downward closures)

Fact 8: $dc(A^*) = U^*$ where $U = \{\sigma | \sigma \in \Sigma, \exists x \in A \ \ \sigma \leq x\}$

Facts 5,6 and 7 are straightforward. To prove fact 8 first notice that for any $U \subseteq \Sigma$, U^* is downward closed, and clearly $dc(A^*) \subseteq U^*$. To show that $U^* \subseteq dc(A^*)$ let's assume that in fact there is some string $x \in \Sigma^*$, s.t.

$x \in U^*$ and $x \notin dc(A^*)$. Say $x = a_1a_2...a_n$, where for every a_i, $a_i \in \Sigma$ and there is $x_i \in A$, s.t. $a_i \leq x_i$. $x_1x_2...x_n \in A^n$, therefore $x_1x_2...x_n \in A^*$. Clearly $dc(\{x_1x_2...x_n\}) \subseteq dc(A^*)$, but by fact 7, $dc(\{x_1x_2...x_n\}) = dc(\{x_1\}...dc(\{x_n\})$. So since $a_i \leq x_i$ for all $i = 1,...,n$ then $a_i \in dc(\{x_i\})$ for all $i = 1,...,n$, so $x = a_1a_2...a_n \in dc(\{x_1x_2...x_n\}) \subseteq dc(A^*)$ and we get a contradiction.

Note that as a consequence of the fact 8 we have the following:

Fact 9: $dc(A^*) = dc(\{C_U(A)\})$ where U is the set of all processes mentioned in A.

Fact 10: If $ax \preceq by$ and $a \not\preceq b$ then $ax \preceq y$.

We'll prove fact 10 by induction on the sum of lengths of x and y.
If both x and y are empty then fact 10 is trivially true. So we will assume that y is not empty.
If $x = \epsilon$ then since $a \preceq by$ there must be (both non-empty) y_1, y_2 s.t. $a \preceq y_1y_2 = by$. y_1 is by' for some y'. If $a \preceq y_2$, then we are done since y_2 is embeddable in y. Otherwise $a \preceq y_1$ and since $a \not\preceq b$, then by I.H. $a \preceq y'$, so $a \preceq y$ (y' is embeddable in y).
So suppose that both x and y are not empty, $ax \preceq by$ and $a \not\preceq b$. So, there must be some x_1, x_2 and y_1, y_2, (non-empty) s.t. $ax = x_1x_2$, $by = y_1y_2$ and $x_1 \preceq y_1$ and $x_2 \preceq y_2$. But then $x_1 = ax' \preceq y_1 = by'$ and $a \not\preceq b$ and x' is no longer then x while y' is shorter then y and by the I.H. $ax' \preceq y'$ and since $x_2 \preceq y_2$ then $ax'x_2 = ax \preceq y'y_2 = y$. □

Finally we have an analog of fact 8 for the full language L.

Fact 11: $dc(\{C_U\}) = \bigcup(C_V)^*_{V \subseteq U}$

The proof of fact 11 is very similar to the proof of fact 8.

2.6 Characterization of levels of knowledge

Now we look at the possibility of characterization of levels of knowledge.

Let $L(A, H)$ be a level of knowledge. Let $\overline{L}(A)$ denote the complement of $L(A)$. Then under each element of $\overline{L}(A)$ there is a minimal element. Moreover the set of minimal elements is a set of mutually incomparable elements and is therefore finite. Let $m(A) = \{x_1, ..., x_k\}$ be the set of minimal elements of $\overline{L}(A)$. Then

$$\overline{L}(A) = \{y | \exists x \in m(A) \;\; x \preceq y\}$$

Thus the level of A is completely characterized by $m(A)$ and we get the next theorem.

Theorem 3 (j): There are only countably many levels of knowledge and in fact all of them are regular subsets of Σ^* (where Σ is either Σ_K or Σ_C). □

Corollary : The membership problem for a level of knowledge can be solved in linear time.

The fact that $L(A)$ can be characterized by a finite number of minimal elements of $\overline{L}(A)$ is true about the levels in L_C as well as about the levels in the smaller language L_K. The difference between the full language and the restricted language becomes obvious when we try to characterize $L(A)$ using

maximal elements.

Let's first consider the case of the restricted language L_K.

2.6.1 Levels of knowledge in L_K

In this section, by Σ_K we denote $\{K_1, ..., K_n\}$, downward closure is with respect to $\leq$.

For every set $L(A)$ we can look at the set of all maximal elements of $L(A)$ with respect to $\leq$ (may be empty !). Clearly these maximal elements are necessarily mutually incomparable. However, the set $L(A)$ need not be *characterized* by them since if A is common knowledge, then the set of maximal elements is empty but this will also be the case if A is not even true, in which case $L(A)$ does not even contain the empty string.

Since distinct sets $L(A)$ have the same maximal elements, maximal elements cannot characterize $L(A)$. However, in case of finite levels, maximal elements do in fact characterize $L(A)$.

Theorem 4 (j) : If L is a non-empty finite subset of Σ_K^*, then L is downward closed iff for some k,

$$L = \bigcup_{i=1}^{k} dc(\{x_i\})$$

where $x_i \in \Sigma_K^*$. This theorem reiterates the fact that the finite levels are characterized by their maximal elements ($x_1, ..., x_k$ are maximal).

Proof : Clearly if L is a union of downward closed sets, by fact 6 L is downward closed.

Let L be a finite downward closed set, $L = \{x_1, ..., x_m\}$. Consider $L' = \bigcup_{i=1}^{m} dc(\{x_i\})$. Clearly $L \subseteq L'$. Let $x \in L'$. Then there is i s.t. $x \in dc(\{x_i\})$. Since $x_i \in L$ and L is downward closed, then $x \in L$. So $L = L'$ and therefore L' is the required representation of L. □

Corollary: If two *finite* sets $L(A)$, $L(A')$ have the same maximal elements, they must coincide.

In order to analyze infinite levels, first we need to establish some more properties of downward closed sets. The following theorem generalizes the representation from the previous theorem to the case of infinite levels. It will be used in obtaining a "normal form" theorem for the levels of knowledge.

Theorem 5 : If L is a subset of Σ_K^*, then L is downward closed iff for some k,

$$L = \bigcup_{i=1}^{k} dc(\{x_{i_1}\})u_{i_1}^* dc(\{x_{i_2}\})u_{i_2}^* ... u_{i_n}^* dc(\{x_{i_{n+1}}\})$$

where $x_{i_j} \in \Sigma_K^*$, $u_{i_j} \subseteq \Sigma_K$ for $j = 1, ..., k$.

Proof :

$\Longleftarrow$

u_{i_j} are all downward closed, so by facts 6,7,8 L is also downward closed.

$\Longrightarrow$

L is a downward closed subset of Σ_K^*, so L is regular, so L is the language described by some regular expression α. We prove that L can be expressed in the proper form by induction on complexity of α.

if $\alpha \in \Sigma_K$ or $\alpha = \epsilon$ or α empty - theorem is obviously true.

if $\alpha = \alpha_1 \cup \alpha_2$ then by the I.H. $\alpha_1 = \bigcup_{i=1}^{k_1} A_i$, $\alpha_2 = \bigcup_{j=1}^{k_2} B_i$, where A_i, B_j are all of the required form. Then $\alpha = \alpha_1 \cup \alpha_2 = \bigcup_{i=1}^{k_1} A_i \cup \bigcup_{j=1}^{k_2} B_i$. Since union is associative we can obtain the proper representation by simply renaming elements of α_2.

if $\alpha = \alpha_1\alpha_2$ then by distributivity of concatenation with respect to the union: $((x_1 \cup x_2)(y_1 \cup y_2) = x_1y_1 \cup x_1y_2 \cup x_2y_1 \cup x_2y_2)$, we get the right representation (of course there are more elements in the sum, and elements are longer).

if $\alpha = \alpha_1^*$ then $L = \alpha = dc(\alpha_1^*) = u^*$, where $u = \{\sigma | \sigma \in \Sigma_K, \exists x \in \alpha_1\ \sigma < x\}$ by fact 9, so $\alpha = \bigcup_{i=1}^{1} u_i^*$, where $u_i \subseteq \Sigma_K$. □

We found a representation of downward closed sets (therefore levels of knowledge). We prove now that there is a unique minimal representation of such a form.

Theorem 6 : Every set of the form $X = dc(\{x_1\})u_1^*...dc(\{x_n\})u_n^*dc(\{x_{n+1}\})$ is *directed* with respect to the embeddability relation $\leq$. In other words: for all $x', x'' \in X$ there is $x \in X$ s.t. $x' \leq x$ and $x'' \leq x$.

Proof : Let $x' = y_1v_1...y_nv_ny_{n+1}$ and $x'' = z_1w_1...z_nw_nz_{n+1}$ where $y_i, z_i \in dc(\{x_i\})$, $v_i, w_i \in u_i^*$ for $i = 1, ..., n$. Then if we take $x = x_1s_1...x_ns_nx_{n+1}$ where $s_i = v_iw_i$ for $i = 1, ..., n$ then clearly $x' \leq x$, $x'' \leq x$ and of course $x \in X$. □

Theorem 7 : If $A \subseteq \bigcup_{j=1}^{l} B_j$, where A, B_j are all of the form as in theorem 6, then there must be some j s.t. $A \subseteq B_j$.

Proof : Suppose that $A \subseteq \bigcup_{j=1}^{l} B_j$ and for all j $A \not\subseteq B_j$. then there are

$x_j \in A$, $j = 1, ..., l$, s.t. $x_j \notin B_j$. By the theorem 6 applied $l-1$ times we can find $x \in A$ s.t. for all j $x_j \leq x$. Since all B_j are downward closed, then x is not in any of them, so $x \notin \bigcup_{j=1}^{l} B_j$ and we get a contradiction. □

Theorem 8 : If L is downward closed then there is a representation of L of the form $L = \bigcup_{i=1}^{k} A_i$, $i \neq j \rightarrow A_i \not\subseteq A_j$, where for all A_i, $A_i = dc(\{x_{i_1}\})u_{i_1}^* ... u_{i_n}^* dc(\{x_{i_{n+1}}\})$, for some $x_{i_j} \in \Sigma_K^*$, $u_{i_j} \subseteq \Sigma_K$, $j = 1, ..., n$ s.t. for every other representation of L of the same form, if $L = \bigcup_{j=1}^{l} B_i$, then $k \leq l$ and for every A_i there is some B_j, s.t. $A_i = B_j$.

Proof : Clearly we have a minimal representation of L, $L = \bigcup_{i=1}^{k} A_i$ and suppose we have some other representation of L of the same form $L = \bigcup_{j=1}^{l} B_i$, then for every i, $A_i \subseteq \bigcup_{j=1}^{l} B_j$ and all A_i and B_j's are directed sets of the proper form, so by the theorem 7, for every A_i there is B_j s.t. $A_i \subseteq B_j$. Similarly, by applying theorem 7 to B_j we get some m s.t. $B_j \subseteq A_m$, so we have for every i, some j, m s.t. $A_i \subseteq B_j \subseteq A_m$. By minimality of $\{A_i\}_{i=1,...,k}$ representation $i = m$, therefore we have $A_i \subseteq B_j \subseteq A_i$ so $A_i = B_j$. □

Corollary (Normal Form Theorem) : Every level of knowledge L of a formula in a distributed system has a unique representation of the following form:

$$L = \bigcup_{i=1}^{k} A_i$$

where

$$A_i = dc(\{x_{i_1}\})u_{i_1}^* ... dc(\{x_{i_{n+1}}\}) = dc(\{x_{i_1} u_{i_1}^* ... x_{i_{n+1}}\}), \quad A_i \not\subseteq A_j \text{ for } i \neq j$$

$\Sigma_K = \{K_1, ..., K_n\}$, $x_{i_j} \in \Sigma_K^*$, $u_{i_j} \subseteq \Sigma_K$. □

2.6.2 Levels of knowledge in L_C

In the full language L_C we can characterize levels of knowledge by listing its (finitely many) maximal elements. Now $\Sigma_C = \{C_U\}_{U \subseteq N}$, downward closure is with respect to $\preceq$.

Theorem 9 : X is downward closed iff there are A_i, $i = 1, ..., m$, s.t.

$$X = \bigcup_{i=1}^{m} A_i$$

where A_i is of the form $dc(\{C_{U_1} \dots C_{U_{k_i}}\})$ for all i.

Proof: Obviously if X has such representation, it must be downward closed.

Let's suppose that X is downward closed. X must be regular (see theorem 3). Now we can easily prove the existence of the required representation by an induction on complexity of the regular expression describing X (using facts 6,7,8,11). □

Corollary 1 : Every level of knowledge L in Σ_C^* is characterized by a finite set of maximal elements $M = \{m_1, ..., m_k\}$.

Proof: Let M be a set of these A_i from the last theorem, which are maximal with respect to $\preceq$. Elements in M are incomparable because $\preceq$ is a partial order (fact 4). □

2.7 Realizing Levels

Since we now know that every level of knowledge is a downward closed set of strings, the interesting question arises whether given a downward closed set of strings L we can find some formula A and some run of a distributed system (history H) s.t. $L = L(A, H)$. The answer, it turns out, depends on the kind of communication available in the system. In a system with unreliable delivery time (asynchronous system) we are able to realize only finite levels of knowledge (this slightly generalizes the result of Halpern and Moses [HM] that no common knowledge can be achieved in an asynchronous system). In systems where all the communications are instantaneous broadcasts with at least 3 processors - synchronous systems - we are able to realize all levels of knowledge. In case there are only 2 processors, and "broadcasts" are the only medium of communication, we are not able to realize finite levels containing strings longer then 1. In the full systems, where there are two communication media: synchronous broadcast, and asynchronous send and receive, we are able to realize all levels.

So in the following sections we analyse separately these three kinds of systems:

1) Systems where only asynchronous communications are available.

2) Systems where only synchronous communications are available.

3) Systems where both synchronous and asynchronous communications are available.

Before we proceed with realizing levels of knowledge, we say something about the properties of formulae, which we will look at.

Definition: A formula A is *persistent* if whenever $H \models A$ and H' extends H, then $H' \models A$.

Theorem 10 (j)**:** If A is persistent then so is $K_i(A)$ for any i.

Proof: Suppose $H \models K_i(A)$ and H' extends H. Suppose that for some H'', $\Phi_i(H'')$ equals $\Phi_i(H')$. Then for some initial segment H_t of H'', $\Phi_i(H_t)$ equals $\Phi_i(H)$. Hence H_t satisfies A and by the persistence of A, so does H''. Since H'' was arbitrary with $\Phi_i(H'') = \Phi_i(H')$, H' satisfies $K_i(A)$. □

Theorem 11: Every formula A which is a boolean combination of P_i's is persistent. □

Theorem 12: Every formula of the form xA where A is a boolean combination of P_i's, and x is a string of knowledge operators is persistent. □

From now on we will look only at formulae of the form as in the last corollary.

2.7.1 Asynchronous Case

We now look at the question of the levels of knowledge in asynchronous systems. Only possible communications are via send and receive, where the arrival time of the message is not guaranteed. In the formal model we exclude broadcasts, so no event is local to more then one processor. This model corresponds to Chandy & Misra's model [CM].

What are the consequences of the fact, that messages although eventually delivered may remain in the mail (buffer) for an unbounded amount of time?

Suppose that a process i sends a message to a process j that a certain formula A (whose truth value is invariant over time) is true. Then when process j receives the message, it knows A and that i knows A. Thus $K_j(K_i(A))$ is true. However, i does not know that j has received the message and hence $K_i(K_j(K_i(A)))$ does not hold until i receives an acknowledgement from j.

Let's see that this fact is valid in our model. We'll take only a 2 processor system, the processors are 1 and 2. Let I be an initial configuration of the system in which the first bit of an input of the first processor is 1 (and this single bit is the whole input of 1). So $I = (v_1, v_2)$, $v_1 = 1$. Let P_1 say the input value of the processor 1, so $P_1 = Q_{1,1}$. Slightly abusing notation we will write that $I \models P_1$, instead of $I \models P_1 = Q_{1,1}$. P_1 is private to 1. Now let H be a history which starts in the initial configuration I and in which 1 sends a message to 2 informing him that P_1 is $Q_{1,1}$ and this message is received by 2 in the next instant of time. $H = I; (s(1,2,P_1), null); (null, r(1,2,P_1))$, and $H \models K_2K_1P_1$.

Let H' be a history in which the same message is sent, but it is delayed, it is not received by 2 in the second instant of time: $H' = I; (s(1,2,P_1), null); (null, null)$

Because of the extension rule H' must be in the protocol.

Clearly $H \approx_1 H'$.

Let H'' be a history with the same input configuration, but in which 1 is slow and hasn't sent anything yet: $H'' = I; (null, null); (null, null)$ (again because of the extension rule H'' must be in the protocol).

$H'' \approx_2 H'$.

Let $H''' = I_1$; where I_1 is an initial configuration in which 2 receives the same

input as in H, but 1's input is 0 instead of 1. H''' is in the protocol because of the first closure condition (all inputs are possible).

$H''' \approx_2 H''$, and we get:

$H''' \approx_2 H'' \approx_2 H' \approx_1 H$, so $H''' \approx_2 H' \approx_1 H$, so since $H''' \not\models P_1$ then $H \not\models K_1K_2P_1$, therefore also $H \not\models K_1K_2K_1P_1$.

It seems reasonable to suppose that a back and forth interchange of messages will only make a finite amount of difference and we proceed to show now that this is indeed the case.

Theorem 13 : Let A be a formula of the form xB, where x is a string of knowledge operators and B is a boolean combination of private facts P_i. Let $H \not\models K_jA$. Then if $H'' = H; H'$ and there is no receive in $\Phi_j(H')$ (neither of the form of $r(l, j, m)$ for some l, nor $bc(l, U, m)$ where $j \in U$, $l \neq j$), then $H'' \not\models K_jA$. Informally: a process may learn something about the others only when it receives a message.

Proof: Proof by induction on the length of H'. If length of H' is 0 then the theorem clearly holds. So suppose that H' is $H'_0; G$. If $H; H'_0 \not\models K_jA$ then there is some $H_1 \approx_j H; H'_0$ s.t. $H_1 \not\models A$. By the splitting rule if G is not a broadcast, we can split it: there are G_2, G_1 s.t. $H; H'_0; G_2; G_1$ is in $\mathbf{P}$ where $(G_1)_j = (G)_j$, $(G_1)_i = null$ for $i \neq j$; $(G_2)_i = (G)_i$, for $i \neq j$, $(G_2)_j = null$. $H; H'_0; G_2 \approx_j H; H'_0$, so $H_1 \approx_j H; H'_0; G_2$ By the extension rule $H_1; G_1$ must be in $\mathbf{P}$ (G_1 was not a receive!). Furthermore $H_1; G_1 \approx_j H; H'$. If x was empty (A is a boolean combination of private facts), then if $H_1 \not\models A$, then $H_1; G_1 \not\models A$. If x is K_sy, then by the induction hypothesis since s has not received any message in G_1 he didn't learn anything, so if $H_1 \not\models A$, then

$H_1; G_1 \not\models A$. So in any case $H; H' = H'' \not\models K_j A$. □

As a consequence we get the following theorem, which corresponds exactly to Chandy and Misra's [CM] theorem 5.

Theorem 14 [Chandy, Misra]: If for some histories H, H', s.t. H is an initial segment of H':

$$H' \models K_1 K_2 ... K_n A \text{ and } H \not\models K_n A$$

then in $H' - H$ there must be a sequence of messages: $m_{n-1}, m_{n-2}, \ldots, m_1$ s.t. m_{n-1} is sent by n and reaches $n-1$ (maybe via some other processes),...,m_1 is sent by 2 and (maybe indirectly) reaches 1 (messages may be different but they all must imply A). Moreover if A doesn't depend on any local event of n (its truth value depends on some event $e \notin E_n$) then there must be some event of the form $r(i, n, m)$ occurring after H but before $s(n, n-1, m_{n-1})$.

Proof: By induction on n using theorem 13. □

Theorem 15 (j)**:** Suppose communication is asynchronous and A is a persistent formula. Let H and H' be global histories such that H' extends H. Then $L(A, H')$ includes $L(A, H)$ and there is a finite X such that $L(A, H') \subseteq dc(X * L(A, H))$ (where $A * B = \{xy | x \in A,\ y \in B\}$).

Proof: The fact that the level *increases* with time follows from the fact that we deal with persistent formulae.

For the second part we use the theorem of Chandy and Misra.

Since there are only finitely many events G between H and H', there are only finitely many possible sequences of events as above and a finite set of strings x for which the conditions of theorem 14 are satisfied. Let X be the

set of these strings, then X satisfies the conditions of the theorem. □

We will try to characterize precisely how the level of knowledge grows when a message is received.

Theorem 16 : Suppose that a history H is exactly the following sequence of messages: $s(n, n-1, K_nP_n)$; $r(n, n-1, K_nP_n)$

$s(n-1, n-2, K_{n-1}K_nP_n)$; $r(n-1, n-2, K_{n-1}K_nP_n)$

...

$s(2, 1, K_2...K_nP_n)$; $r(2, 1, K_2...K_nP_n)$

where P_n is initially (in the empty history) known only to n; then

$$L(P_n, H) = dc\{K_1K_2...K_n\}$$

Proof: It is easy to notice that $dc\{K_1K_2...K_n\} \subseteq L(P_n, H)$.

The other inclusion we prove by induction on the length of H. If it is 0 then H is empty so P_n is known only to n. So assume that the theorem is true for histories up to length n.

Let's assume then that there is some y s.t. $H \models yP_n$ and $y \notin dc\{K_1K_2...K_n\}$. So y must be non-empty. Let then $y = K_iy'$. Then either i was not mentioned in the sequence of messages (i was not any of {1,...,n})- but in such a case i would have known P_n initially (it hasn't received any messages, see theorem 13) - which contradicts our assumption. So it must be that $i \in \{1, ..., n\}$. Then if we express $K_1K_2...K_n$ as zK_iz' where K_i picked is the leftmost occurrence of K_i in $K_1K_2...K_n$. Let's define H' as the initial segment of H up to the last receive by i. i doesn't learn anything in $H - H'$ (he doesn't receive any messages), so if $H \models K_iy'P_n$ then $H' \models K_iy'P_n$. $H' = H"; G$

where $(G)_i = r(s,i,m)$. By the splitting rule we can split G into G_1 and G_2, where G_1 is a receive on i, G_2 is *null* on i. $H"; G_1; G_2$ must be in the protocol. $H' \approx_i H"; G_1$, so $H" \models y'P_n$. Then by the induction hypothesis $y' \in dc(\{z'\})$, so $y' \leq z'$ and therefore $y \leq K_1K_2...K_n$. □

Theorem 17 : Suppose H realizes $L(A,H)$ and H' realizes $L(A,H')$ for some formula A which is a boolean combination of P_i. Moreover, H and H' are compatible and in both H and H' only strings of the form xA are communicated. Suppose that H'' is the concatenation of H and H', and initially A is known in H'' only to processes which knew A initially in H or in H'. Then:

$$L(A,H'') = L(A,H) \cup L(A,H')$$

Proof: Clearly $L(A,H) \cup L(A,H') \subseteq L(A,H'')$.

Let's suppose $H'' \models yA$.

If y is empty, then we are done, since $H \models A$.

Otherwise let yA be true in H'', where y is (say) K_1y'. Suppose that $K_1y' \notin L(A,H)$. Then there must be some point T s.t. $H; H_T \models K_1y'A$, $H; H_{T'} \not\models K_1y'A$ (for all $H_{T'}$ s.t. $H_{T'} \leq H_T \leq H'$). By theorem 13 H_T has as the last event some G s.t. $(G)_1 = r(l,i,zA)$. So $y' \leq lz$. But the event G was a part of H'. Message sent and received in H' must have been true in H'. So $H_T \models K_1y'A$, so $H' \models yA$. □

Note that H and H' could be executed in parallel. In fact we can take any minimal H'' s.t. both H and H' are embeddable in H''. □

We now show that *all* finite downward closed sets are actually attainable

as knowledge levels $L(A)$ of formulas in asynchronous systems.

Theorem 18 (j): Every finite downward closed set is the set $L(A,H)$ for an appropriate A and H in some asynchronous protocol.

Construction: Let X be a finite downward closed set of nonrepeating strings from Σ^*. We construct a formula A and a history H such that $L(A,H) = X$.

1) If X is empty then A is the formula *false*.

2) If X consists of the empty string, then A is the conjunction P_1 and P_2 where: P_1 is a predicate whose truth value is initially known only to process 1, P_2 is a predicate whose truth value is initially known only to process 2, and H is the empty history.

3) Otherwise X has nonempty strings and therefore the set of all maximal strings M in X is nonempty. Let $M = x_1, x_2, \ldots, x_r$. Let $x_i = K_{i(k_i)} \ldots K_{i(2)} K_{i(1)}$ for $i = 1, \ldots, r$. Then we take A to be a $\bigvee_{i=1}^{r} P_{i(1)}$.

Let $H_i = s(i(1), i(2), K_1 A); r(i(1), i(2), K_1 A); s(i(2), i(3), K_2 K_1 A);$
$\ldots s(i(k-1), i(k), K_{i(k-1)} \ldots K_{i(2)} K_{i(1)} A);$
$r(i(k-1), i(k), K_{i(k-1)} \ldots K_{i(2)} K_{i(1)} A).$

Clearly $L(A, H_i) = dc(x_i) \cup \{j(1) | j = 1, \ldots, r\}$. Now by theorem 5 if $H = H_1; H_2; \ldots H_k$ then $L(A,H) = \bigcup_{i=1}^{k} dc(x_i)$.

Note that H could be any permutation of H_i's, in fact all H_i's could be executed in parallel. □

2.7.2 Synchronous Case

The situation with synchronous communication is much better as we see next. We start with a preliminary result.

Theorem 19 (j): The set $L(A)$ is infinite iff it includes common knowledge of A between two distinct processes i and j.

Proof: One direction is clear, as common knowledge between i and j includes all strings in $\{K_i, K_j\}^*$.

Conversely, suppose that $L(A)$ does not include common knowledge of any two distinct i,j. Then, since $L(A)$ is downward closed, for any such i,j, there must be a maximum $Alt(i,j)$ number of alternations between K_i and K_j in any string in $L(A)$. Let m_A be the largest of these $Alt(i,j)$ and let p be $m_A * n^2$. Now any nonrepeating string of length greater than $p+1$ has at least $p+1$ alternations, and hence more then m_A for some *specific* alternation between K_i and K_j. Thus no string in $L(A)$ can have length greater then $p+1$ and $L(A)$ is finite. □

So how can we create infinite levels? We need synchronous communication.

Theorem 20 : If we broadcast "xA" where x is a string of knowledge operators and A is a boolean combination of propositions P_i among the group of processes U, then the created level of knowledge of A increases by the downward closure of $\{C_U\}$. Formally: If $H = H'; G$, $\forall j \in U\ (G)_j = bc(i, U, xA)$, $\forall j' \notin U\ (G)_{j'} = null$, then:

$$L(A,H) = L(A,H') \cup dc(\{C_U\})dc(\{x\})$$

Proof: Let H be as in the premises of this theorem. Then $H \models A$ (our processes are honest and A may be broadcasted). Moreover since it is a property of all histories in the protocol, it is common knowledge that formulae sent (or broadcasted) are true and remain true (all formulae persistent):

$$\models C_N(H = H'; G; H'' \text{ and } (G)_j = bc(i, U, xA) \rightarrow H \models xA)$$

this implies that for every $U \subseteq N$:

$$\models C_U(H = H'; G; H'' \text{ and } (G)_j = bc(i, U, xA) \rightarrow H \models xA)$$

so in order to prove that $H \models C_U xA$ it is enough (since common knowledge is closed under modus ponens) to show that $H \models C_U(H = H'; G; H'' \text{ and } (G)_j = bc(i, U, xA)$. But it is a property of $\mathbf{P}$ that if for some $j \in U$, $(G)_j = bc(i, U, B)$ then for all $j \in U$, $(G)_j = bc(i, U, B)$. So $\forall H_0 \approx_{U^*} H$ $(H = H'; G; H''$ and $(G)_j = bc(i, U, xA)$ $) \rightarrow$ $(H_0 = H_0'; G; H_0''$ and $(G)_j = bc(i, U, xA))$. Therefore $H \models C_U xA$ so $H \models dc(\{C_U\})dc(\{x\})$, $H \models dc(\{C_U x\})A$. The theorem follows since no process outside of U received any message. □

We now prove that in the presence of synchronous communication *every* downward closed set of strings without repetitions is the level of knowledge of some formula under some history.

Definition : Given strings $x_1, ..., x_k$, not containing repetitions, let $N(x_1, ..., x_k)$ be the set $\{y | \forall i \leq k,\ x_i \not\leq y\}$.

Theorem 21 (j):

(a) $N(x_1, ..., x_k) = \bigcap N(x_i) : i \leq k$.

(b) If $x = a_1 ... a_m$ then $N(x) = (\Sigma - a_1)^* ... (\Sigma - a_m)^*$.

Proof : The first part is obvious.

To see (b) we use induction on m.

First of all, it is clear that $N(x)$ includes the right hand side. For suppose $y \in (\Sigma - a_1)^*...(\Sigma - a_m)^*$. If y has no a_1, then it is certainly in $N(x)$. Else the first occurrence of an a_1 in y must be from the $(\Sigma - a_2)^*$ at the earliest. y is then of the form $y_1 a_1 y_2$ where y_1 contains no a_1 and y_2 is in $(\Sigma - a_2)^*...(\Sigma - a_m)^*$. By induction hypothesis $a_2...a_m \not\leq y_2$ and so $x \not\leq y$.

Suppose now that y is in $N(x)$. We want to show that $y \in (\Sigma - a_1)^*...(\Sigma - a_m)^*$. If $m = 1$ then this is immediate. Otherwise we know that $x \not\leq y$.

If $a_1 \not\leq y$, then $y \in (\Sigma - a_1)^*$ and hence $y \in (\Sigma - a_1)^*...(\Sigma - a_m)^*$.

If $a_1 \leq y$ then $y = y_1 a_1 y_2$, where $a_1 \not\leq y_1$ and $a_2...a_m \not\leq y_2$. In that case $y_2 \in (\Sigma - a_2)^*...(\Sigma - a_m)^*$ by the induction hypothesis and $y_1 \in (\Sigma - a_1)^*$. Since $a_1 \neq a_2$, the result follows. □

The next theorem will show us how to realize $N(x)$. We take a sequence of broadcasts to $N - 1$ processes at a time. Let's assume that at time T, group U_T receives a broadcast and at time $T + 1$ group U_{T+1} (both of $N - 1$ processors). Then the processor sending broadcast at $T + 1$ (a_{T+1}) will be one of the processors in $U_{T+1} \cap U_T$, and the message sent will be the string of common knowledge operators $C_{U_T} \ldots C_{U_1}$ followed by a fixed formula P_1 private to a_1.

Theorem 22: Let H be a history in which the private information of s_1 is P_{s_1} and every event G^T occuring in H at time T for $T = 1, ..., l$, (where l =length(H)) is of the form: $(G^T)_i = bc(a_{s_T}, N - \{a_T\}, m_T)$ for all $a_{s_T} \neq a_T$, $a_{s_T} \neq a_{T-1}$, $(G^T)_i = null$ otherwise. Where $m_1 = P_{s_1}$, $m_{T+1} = C_{N-\{a_T\}} m_T$.

Then:

$$L(P_{s_1}, H) = dc(C_{N-\{a_l\}} \dots C_{N-\{a_1\}})$$

Proof: Induction on the length of H. Clearly if $l = 1$ then $L(P_{s_1}, H) = C_{N-\{a_1\}}$.

Suppose that for histories up to the length $l-1$ theorem is true. Now we can use theorem 19: if $H = H'; G^l$ where $(G^l)_i = bc(a_{s_l}, N - \{a_l\}, m_l)$ then $L(P_{s_1}, H) = L(P_{s_1}, H') \cup dc(C_{N-\{a_l\}} C_{N-\{a_l\}} \dots C_{N-\{a_1\}})$.
Since by I.H. $L(P_{s_1}, H') = dc(C_{N-\{a_{l-1}\}} \dots C_{N-\{a_1\}})$ then
$L(P_{s_1}, H) = dc(C_{N-\{a_l\}} \dots C_{N-\{a_1\}})$ □

Corollary 1 (j): In a system with at least 3 processors, for every x in Σ_C^*, there exists a history H and a formula A such that $L(A, H)$ is just the set of strings without repetitions in $N(x)$.

Proof: By fact 9, $U^* = dc(C_U)$. Therefore by induction on m, and fact 8 ($dc(A; B) = dc(A); dc(B)$), $U_1^* \dots U_m^* = dc(C_{U_1} \dots C_{U_m})$. So in particular $N(a_l \dots a_1) = L(P_{s_1}, H)$. □

Theorem 23 (j): Every downward closed set L of strings without repetitions is $L(A, H)$ for suitable A and H in a synchronous system with at least 3 processors.

Proof : Let $\{x_1, ..., x_k\}$ be the set of minimal elements of the set of strings without repetitions which are not in L. Then L is $N(x_1, ..., x_k)$.

If $k = 0$ then all strings without repetitions are in L and L is just common knowledge, which can be achieved by taking the formula A to be a tautology.

Otherwise for each x_j we can find a P_j and a history H_j such that

$L(P_j, H_j)$ is exactly $N(x_j)$. Now let H be $H_1...H_k$ and A be the conjunction of the P_j. (We assume the P_j are all independent so that if $i \neq j$ then H_i conveys no information about P_j). Then $y \in L(A, H)$ iff $H \models yA$ iff for all j, $H \models yP_j$ iff for all j, $H_j \models yP_j$ iff for all j, $y \in N(x_j)$ iff $y \in L$. □

Theorem 24: In a two processor system with only synchronous communication available, no finite level containing strings of length ≥ 2 can be achieved for any formula A.

Proof: Suppose that $H \models K_1K_2A$, and $H \not\models C_{\{1,2\}}A$. If A persistent and in the empty history 1 knows that 2 knows A, then A must be true in all histories and therefore must be common knowledge. So 1 must have learned that K_2A in H, so there must have been a communication between 1 and 2 to that effect, but there are only synchronous communications in the system, and these create common knowledge - contradiction. □

2.7.3 Both Kinds of Communication Available

If we have both kinds of communications in the system, we can directly realize every level. We show first how to realize a level in normal form. Later we will also show how to realize the level given by the set of minimal strings of the complement. As we will see from the examples in a following section, if we have some specification of the level of knowledge to be achieved, which is incomplete (doesn't specify a unique level), then the two constructions will give us different levels of knowledge.

Construction:

$$L = \bigcup_{i=1}^{k} dc(X_i)$$

where $X_i = x_{i(m)} \dots x_{i(1)}$, $x_{i(j)} \in \Sigma_C^*$.

We take $A = \bigvee_{i=1}^{k} P_{s(i)}$ where $s(i) \in U_j$, $s.t.$ $x_{i(1)} = C_{U_j}$. We create H_i to realize the level of knowledge $dc(X_i)$ of a formula $P_{s(i)}$ for every i, and then we take H to be the concatenation of all H_i's.

Let $X_i = C_{U_m} \dots C_{U_1}$.

(i) If U_1 is not a singleton then $s(i)$ initially sends a broadcast $P_{s(i)}$ to all the processes in U_1. Otherwise it does nothing.

(ii) Sender of the last broadcast (to U_j) sends asynchronously all that he knows about $P_{s(i)}$ to one of the processes in U_{j+1} (if there was no broadcast yet, $s(i)$ is a sender).

(iii) Recipient of the last asynchronous message broadcasts all that he knows about $P_{s(i)}$ to all processes in U_{j+1}. □

We can also construct any level specified by a set of minimal strings of a complement.

Construction:

$$L = N(x_1, ..., x_k)$$

where $x_i \in \Sigma_C$.

We take $A = \bigwedge_{i=1}^{k} P_{s(i)}$ where $s(i) \in U_j$, $s.t.$ $x_i = yC_{U_j}$. We create H_i to realize the level of knowledge $N(x_i)$ of a formula $P_{s(i)}$ for every i, then we take H to be a concatenation of all H_i's.

Let $x_i = C_{U_m} \ldots C_{U_1}$.

(i) $s(i)$ initially sends a broadcast $P_{s(i)}$ to all the processes *not* in U_1.

(ii) Sender of the last broadcast (to $N - U_j$) sends asynchronously all what he knows about $P_{s(i)}$ to one of the processes *not* in U_{j+1} (if there was no broadcast yet, $s(i)$ is a sender).

(iii) Recipient of the last asynchronous message broadcasts all what he knows about $P_{s(i)}$ to all processes in $N - U_{j+1}$. □

2.7.4 Limited n-casts

If in our system we have only a limited broadcast capability, then not all levels of knowledge can be achieved. Precisely:

Theorem 25: If in a system of N processes all the broadcasts are to groups of n processes ($n < N$), then no $C_U A$ can be realized for any $|U| > n$, for any formula A which is not true in all the histories.

Proof: Broadcast $bc(i, U, m)$ creates common knowledge of m among U. A sequence of broadcasts of up to n processes (say to $U_1, ..., U_k$) will create $dc(C_{U_k} C_{U_{k-1}} \ldots C_1)$ where all U_i for $i = 1, ..., k$ have at most n elements. Let U' be a set of n' elements where $n' > n$. Then $C_{U'} \notin dc(C_{U_k} C_{U_{k-1}} \ldots C_1)$, so it is not realized by a sequence of k n-casts. □

2.8 Examples

Every level of knowledge can be generated in an appropriate system. Practically there may be two kinds of reasons we want to obtain some particular level of knowledge:

1) We want to have *enough* knowledge in the system so we specify which strings L we want to have included in a realized level $L(A, H)$.

2) We want to prevent certain processes from knowing some facts about the others, in such situation we would specify a set L' of the strings we don't want to include in $L(A, H)$.

Let's consider now a pair (L, L') as a knowledge specification for our system, and let's look how we could realize it.

1. If $L' \cap dc(L)$ is not empty then (L, L') cannot be realized for any formula.

   For example if $L = \{K_1, K_2, K_3\}$, $L' = \{K_1, K_3\}$ then there is no system which realizes (L, L').

2. So a necessary condition for realizability of (L, L') is:

$$\forall m \in L \; \forall m' \in L' \;\; \neg(m' \leq m)$$

   Now we have two possibilities:

   If $dc(L') - L' = dc(L)$ then (L, L') uniquely specifies the level, which can be realized for an appropriate formula in an appropriate system.

   Let's look at an example. We have some fact F, and we want to create a history H, s.t. $H \models K_A K_B F$, $H \models K_B K_C F$, $H \models K_C K_A F$ (these three facts already imply that $H \models K_A F$, $H \models K_B F$, $H \models K_C F$,

but we also want $H \models \neg(K_A K_C F)$, $H \models \neg(K_B K_A F)$, and $H \models \neg(K_C K_B F)$. So our specification is (L, L'), $L = \{K_A K_B, K_B K_C, K_C K_A\}$, $L' = \{K_A K_C, K_B K_A, K_C K_B\}$. Notice that in this case, since we exclude the possibility of common knowledge between any pair of processes, our level of knowledge must be finite. In fact these requirements characterize the level of knowledge completely. The level we are looking for is exactly the set $L(A, H) = \{K_A K_B, K_B K_C, K_C K_A, K_A, K_B, K_C, \epsilon\}$.

We can attain (L, L') by a protocol in which (for example) all A, B, C independently learn about F, A sends a message F to C, C sends F to B, B sends F to A, and all messages are sent asynchronously. In a synchronous system we can achieve (L, L') by a protocol in which first we use a broadcast to bring about common knowledge of the fact $(p \wedge q \wedge r) \rightarrow F$. Then A broadcasts p to $\{A, B\}$. Later B broadcasts $C_{\{A,B\}}p$ to $\{B, C\}$. Similarly first C broadcasts q to $\{B, C\}$, later C broadcasts $C_{\{B,C\}}q$ to $\{B, C\}$, finally A broadcasts r to $\{A, C\}$, and then A broadcasts $C_{\{A,C\}}r$ to $\{A, B\}$,

If (L, L') can be realized but there is $x \in \Sigma^*$ s.t. $x \notin dc(L)$ and $\neg(\exists m' \in L'\ m' \leq x)$ then the level is not uniquely specified (we can include x in (L, L') but we don't have to!).

The smallest level realizing (L, L') is $dc(L)$ and the largest one is $N(L')$ (or $dc(\Sigma^* - L')$).

Let for example $L = \{C_{\{1,2\}}K_3\}$, $L' = \{K_3 K_2\}$ In order to realize (L, L') we can take just $dc(L)$. This can be done by taking formula A to be P_3, and a history: $s(3, 1, K_3 P_3); r(3, 1, K_3 P_3); bc(1, \{1, 2\}, K_3 P_3)$.

Another possibility is to realize $N(K_3K_2)$.

2.9 Applications

An example of a practical situation where there is a necessity of a certain finite level of knowledge was given by Halpern and Zuck [HZ], who used the concept of knowledge for proving correctness of certain protocols.

Suppose that a sequence of bits is communicated in an asynchronous system where messages can be delayed (or lost), but if they are received - they must be received in order. Bit sending protocol is correct if all the bits sent are received. In [HZ] it is proved that a necessary and sufficient condition for the correctness of a bit sending protocol is $K_sK_rK_sK_r(\text{"}value\ of\ the\ first\ bit\text{"})$ meaning - "sender knows that the receiver knows that the sender knows that the receiver knows the value of the first bit", whenever the sender sends a second bit.

So in order to implement this protocol it is enough to achieve a specified level of knowledge.

There are a lot of examples showing that common knowledge is necessary for correctness of certain protocols.

More interestingly, Moses and Tuttle [MT] proved that certain synchronized action problems not only require common knowledge, but also that there is always a most efficient solution which is an implementation of a simple knowledge based algorithm (algorithm where there are explicit tests of knowledge).

Using higher level "knowledge" based language in designing consensus algorithms is possible because of one very important property of common knowledge: it is always achieved by all the processes involved simultaneously. The synchronization algorithm simply says: "repeat .... until $C_U A$" for a certain formula A.

Chapter 3

Consensus

3.1 Introduction

So far we have been considering an increase in the level of knowledge of a given formula A in the system, where A was communicated (or A preceded by a sequence of knowledge operators).

Now we will look at the situation when we are interested in the knowledge among processes of one particular fact - what is the real global history H. If some process i knows that the global history is H, then clearly $H \models K_i A$ for all A s.t. $H \models A$. If our language is rich enough to distinguish all histories - it corresponds to looking at knowledge of some formula φ which is true in H and false in all other histories.

We will consider the situation when we have some fixed finite global history H which is a result of a run of some distributed system, where everybody knows everybody else's program and this fact is common knowledge among

all processes. The situation is static - running the protocol is not a part of a history. Processes are trying to determine H, and this process of determining H is not a part of H. Because of that I will talk about worlds or states rather then histories.

Let's look at the situation where W is the set of possible worlds and processes' information is given as a partition of W. Moreover these partitions are common knowledge. We will assume that all the partitions are finite. This is reasonable since our processes are finitary.

We will assume from now on that communication is synchronous.

It is easy to notice that in such a situation when two processes communicate, and are able to send exactly what they know about H, i.e. a complete description of the set of all histories compatible with his local history, then everything that the processes *can* learn, they *will* learn after one round. The situation will stabilize and what they both will know - is that the real world is one of the worlds in the appropriate element of the coarsest common refinement of both partitions.

The interaction between both processes in that case will proceed as follows: process 1 sends an element of his partition containing the real world to the process 2, process 2 takes the intersection of the set he has gotten from 1 and an element of his own partition containing the real world and sends this set to 1, 1 takes the intersection of this set and the element of his partition. This set will turn out to be equal to the set he has just received. From now on both processes will be sending the same set.

Not always processes communicate all what they know. There are natural

situations where agents instead of communicating all what they know about the set of possible worlds, communicate values of a certain function whose domain is the set of sets of possible worlds and codomain is some space of messages. This may be because the capacity of communication channels is limited, or because agents are competing and they do not want to reveal all their information. Good example of such a situation is when agents received different information about the item being auctioned and based on these private information they are willing to bid. Their bids are prices, amount of money they are willing to pay, not what they have learned about the item. In such case price is a function communicated.

We are going to investigate situations of that type. We will assume that all processes are communicating the same function, moreover this fact is common knowledge among them.

This function may be not 1-1. In that case in general we cannot expect that the knowledge of both processes will be the same, as we can see from the following trivial example:

Example 1 : Take $W = \{a, b\}$ and let $P_1 = \{\{a, b\}\}$, $P_2 = \{\{a\}, \{b\}\}$ and f is the constant function 0. Then processor 1 will know nothing, while 2 will know what the real world is. Nobody will learn anything from communication. Constant function doesn't give any information. □

Since we cannot reasonably require the same knowledge of two processes, perhaps at least we will be able to achieve consensus among them regarding the value of f. After all, the function f may represent the only available description (code) of the set of worlds (histories).

This model was investigated by economists. Consider the following interpretation: W is some set of possible results of an experiment, and two processes (now persons or agents) receive some information about the result. There is common knowledge between them of what *kind* of information they both receive, but the actual information is unknown. Both persons are interested in computing the probability of a certain event. There is some prior probability distribution on W given, and it is shared by both agents. Without any additional information they would both have the same value - $p(E)$. But if person 1 learns that the result w is in $P_1(w)$, then he can compute a new probability as $p(E|P_1(w))$. This is his *posterior* probability of E. Similarly, 2 can compute his posterior probability $p(E|P_2(w))$.

There is no a priori reason why $p(E|P_1(w)) = p(E|P_2(w))$, but surprisingly Robert Aumann [A], has shown in 1976 that when the posteriors are common knowledge between two processes, then they must indeed be the same. Like-minded agents cannot "agree to disagree".

Aumann didn't address the question how the agents computed their posteriors and how they could become common knowledge. Geanakoplos and Polemarchakis [GP] first investigated the procedure in which one agent computes his posterior, sends the value to the second agent, who in turn computes his new posterior excluding from his set of possible worlds all the worlds in which the first agent would have sent a different value; sends this new posterior back to the agent one, and so on - this process continues until (as they have proved) - both agents' posteriors will converge to the same value, which will happen after $k + l$ rounds where k, l are numbers of sets in partitions of

agents 1 and 2 respectively.

This process of reaching consensus may be illustrated by the example in figure 3.1.

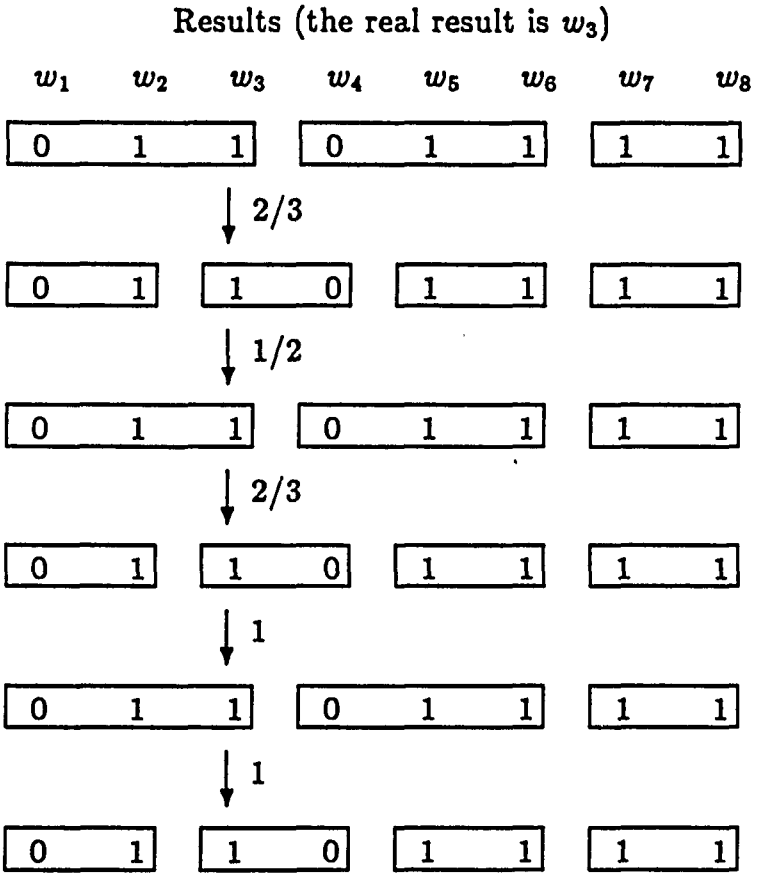
Cave [C] and Bacharach [B] noticed that the property of the conditional probability function that was crucial in both Aumann's and Geanakoplos-Polemarchakis' proofs is that if the conditional probability of a certain event is the same in two disjoint sets, then the conditional probability of this event in the union of the two sets will be the same as it was in both of them. This property was called *union consistency* by Cave and the *"sure - thing" principle* by Bacharach. Cave and subsequently Bacharach proved that if two agents communicate values of some function satisfying the sure-thing principle back and forth revising their sets of possible worlds and therefore recomputing values of the function - they will end up with the same values.

All these results generalize to the case when there are $n > 2$ agents and all the agents in turn communicate their values of a function so all of them hear all these values - and there is common knowledge among them of this fact. This was also investigated by Cave.

The situation where every announcement is public corresponds in computer science to the case of a broadcast from reliable source. In economics it corresponds to an auction.

But prices are not always set up via auctions. Communication in the system is not always synchronous n-cast. Very often there are private communications between agents, inaccessible to others.

Here we will show how conditional probabilities of an event are communicated back and forth between two agents. We assume that the space consists of 8 points. Every point can be either 0 or 1. This can be interpreted as 8 possible results of an experiment, results marked 1 will support some hypothesis, results 0 - not. Agents are interested in the probability that the hypothesis is true. We assume that all the results are initially equally likely. Note that the prior probability (if no information is received by agents) is 3/4. Now let's look at the protocol:



and from now on both agents will send 1.

Figure 3.1: Example of Exchanging Conditional Probabilities Among 2 Agents

In fact, the situation when there are more then two agents involved, and agents communicate using person-to-person links inaccessible to other agents (binary channels) is much more difficult to analyse. This is because even when there are 3 agents, and their information partitions are common knowledge among them, if they communicate values of f in a ring (1 to 2, 2 to 3, 3 to 1 and so on) then since 3 doesn't know what 2 has heard from 1 it usually can imagine more then one scenario which made 2 send the value he has sent, so it is more difficult for him to exclude worlds.

We will give the general formula for updating agents knowledge. We will also define a class of *fair* protocols, protocols in which all the agents' information will affect all other agents' knowledge. This will be necessary if we hope for consensus among agents. Of course if an agent doesn't communicate with anybody - we cannot expect consensus between this agent and the others! It will turn out that for $n > 2$ the sure-thing principle is not enough. We will show an example where $n = 3$ and the function communicated satisfies the sure-thing principle but consensus is not reached. We will define a stronger condition - weak convexity, which will turn out to be sufficient for $n \leq 3$ but will not work for $n > 3$ (we will show an example). Finally, a still stronger condition will be defined (strong convexity) which will force consensus for all n.

Also we will give examples of important functions (especially 0-1 functions) which are not strongly convex but still bring consensus for any n and we will discuss some necessary conditions for guaranteeing consensus regardless of the partitions and the real world.

3.2 Basic Notions

We assume given a space W, the space of states (or of possible worlds) and n participants with *finite* partitions P_i of W.

Let $P_i(x)$ be the equivalence class of world x in partition P_i. Let $x \equiv y$ mean that for all i, $P_i(x) = P_i(y)$. We will say that a subset X of W is *closed* if x in X and $x \equiv y$ imply that y is also in X. If P^+ is the common refinement (join) of the P_i, then X is closed iff it is a union of P^+ equivalence classes. Note that since the P_i are finite, so is P^+. Similarly, we will use the expression *i-closed* to describe a union of elements of the partition P_i. Note that an i-closed set is closed, but not necessarily vice versa.

In our model time is discrete, all agents have access to a global clock (so we have *common* clock), which is set to 0 before communications start. Now by a *protocol* we mean a pair of functions $s(t), r(t)$ from the natural numbers (≥ 0) to the set {1,...,n} . Here t stands for time and $s(t)$, $r(t)$ are, respectively, the sender and the recipient at time t. So $s(t) = i$, $r(t) = j$ means that at time t i communicated with j (i sent a message which was received by j).

Definition : Given a protocol $(s(t), r(t))$ consider the directed graph whose vertices are the participants { 1,...,n} and there is an edge from i to j if there are infinitely many t such that $s(t) = i$ and $r(t) = j$. Then the protocol is *fair* if the graph above is strongly connected, i.e. if there is a path of directed edges which passes through every vertex at least once, returning to its origin.

If the participants are observing a fair protocol, then not only will each participant be a recipient and a sender infinitely many times, but each participant will receive information from every other, possibly indirectly.

We will assume for simplicity that always $s(t+1) = r(t)$. However this assumption is not necessary.

The simplest example of a fair protocol is a "round-robin" protocol - where the first person sends value of f to the second person, second to third,...., and so on, until it reaches the last - nth person, who sends back to the first person. And this cycle is repeated forever. Formally if participants are labeled from 0 to n-1, then this protocol can be expressed as $s(t) = r(t-1) = t \bmod n$.

Let f be some function from a set of non-empty subsets of the state space W to some domain D (f is the function whose values are communicated)[1]. If f is intended to be a conditional probability, then we assume given a probability measure π on W and some event A contained in W. Then $f(X)$ for $X \subseteq W$ is just $\pi(A|X) = \pi(A \cap X)/\pi(X)$ where $\pi(X) \neq 0$

Suppose now that $p = (s(t), r(t))$ is some protocol. We define by induction on t the message $m(x,t)$ sent at time t, and $C(x,i,t)$, the set of possible states for i at time t, given that the real state is x.

$C(x,i,0) = P_i(x)$
If $i = r(t)$, then $C(x,i,t+1) = C(x,i,t) \cap \{y | m(y,t) = m(x,t)\}$
where $m(x,t) = f(C(x,s(t),t))$.

[1]From now on whenever we consider $f : 2^W \rightarrow D$, we will assume that f is defined for the non-empty sets only and f is defined for all closed subsets of W.

If $i \neq r(t)$, then $C(x,i,t+1) = C(x,i,t)$.

We let $v(x,i,t) = f(C(x,i,t))$ so that $m(x,t)$ is just $v(x,s(t),t)$

Theorem 1 (j): If $x \equiv y$ then for all i,t, $m(x,t) = m(y,t)$ and $C(x,i,t) = C(y,i,t)$.

Proof : Straightforward, by induction on t. □

Theorem 2 (j): If all P_i are finite, then there is T s.t.

$$\forall w,i\ \forall t,t' > T\ C(w,i,t) = C(w,i,t')$$

so for every i we have some limiting partition of W (moreover this partition is finite).

Proof: For all x,i, $C(x,i,t)$ is a non-increasing function of t. Also it is a nonempty union of P^+ equivalence classes. Since P^+ is finite, $C(x,i,t)$ is eventually constant. Let $t(x,i)$ be the least t such that for $t' > t$, the value of $C(x,i,t')$ is the same.

$$t(x,i) = Min\{t | \forall t' \geq t\ f(C(x,i,t')) = f(C(x,i,t))\}$$

Then $t(x,i)$ depends only on the P^+ equivalence class of x. Since P^+ has finitely many equivalence classes, and there are finitely many i, we can find T s.t.:

$$T = Max\{t(x,i) | i \leq n,\ x \in W\}$$

Clearly $C(x,i,t)$ is constant for all $t' > T$ regardless of x,i. T does of course depend on the protocol $(s(t),r(t))$. It follows also that $m(x,t) = f(C(x,s(t),t))$ depends only on x and $s(t)$ if $t > T$. We write $C(x,i,\infty)$ to indicate the limiting value of $C(x,i,t)$. Note also that there is finitely many possible limiting values (must be less then $|2^{P^+}|$ and P^+ is finite). □

Values communicated depend on the protocol. In the following example we will show that even in the simple case of two agents, the final values will change if we change who initiates communication (so we will have same partitions, same function and the same communication graph - but change in the protocol will result in a change of final values).

Example 2 : Let $W = \{x_1, x_2, x_3, x_4, x_5, x_6\}$. Let's assign weights to points in W: $w(x_1) = w(x_4) = 0$, $w(x_2) = w(x_3) = 1$, $w(x_5) = 3$, $w(x_6) = 4$. We will take our function $f(A)$ to be the average of the weights of points in A. So:

$$f(A) = \frac{\Sigma_{a \in A} w(a)}{|A|}$$

(later on we will see that function f will always lead to consensus in fair protocols). Let's define partitions:

$P_1 = \{\{x_1, x_2\}, \{x_3, x_4\}, \{x_5, x_6\}\}$

$P_2 = \{\{x_1, x_3, x_5\}, \{x_2, x_4, x_6\}\}$

If we take the real world to be x_1, and then consider the situation in which 1 starts communication - we will get consensus value 1/2. On the other hand if 2 starts communication - the consensus value will be 0 (see figure 3.2).

So change in the protocol changes consensus values. Now we will keep the protocol fixed (fair) and we will look how the changes in functions communicated will affect the final values (most important is of course which functions will give consensus). First, let's look when two functions will give us exactly the same information.

For subsets of W let's define the following relation (depending on f):

$$X \approx_f Y \text{ iff } f(X) = f(Y)$$

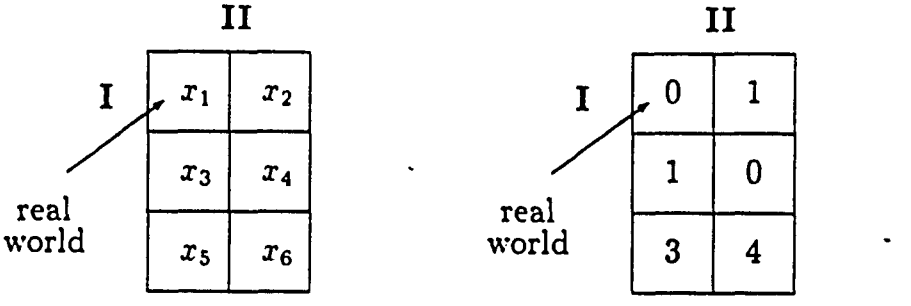


Figure 3.2: Example when Different Protocols give Different Consensus Values

Let's notice that if values of f and g are communicated according to the same protocol, and all partitions are the same, then:

$$\approx_f = \approx_g \Rightarrow C_f(x,i,t) = C_g(x,i,t)$$

for every i, x, t.

As a consequence, the final knowledge of all agents is the same when two functions defining the same equivalence relation $\approx_f$ are communicated:

$$\approx_f = \approx_g \Rightarrow C_f(x,i,\infty) = C_g(x,i,\infty)$$

for every i, x, t.

When two functions can distinguish the same sets of worlds, they carry the same amount of information and they will give agents the same knowledge. It seems reasonable then to order functions according to how informative they are. Before we give a formal definition, let's look at two extreme cases.

The least informative is a constant function as it doesn't carry any information. Every agent at the end of any protocol when values of a constant function are communicated will know exactly what he knew before the communication started - for every x, i, t, $C_f(x, i, t) = P_i(x)$ if f is constant.

The other extreme is a function which distinguishes every two subsets of W. In the following theorem we will show that in such a case what agents eventually learn is everything they could possibly know together.

Theorem 3: Let $f : 2^W \rightarrow S$ be 1-1 (injective), then:

$$C_f(x, i, \infty) = P^+(x)$$

for every x, i.

Proof: Notice that $\approx_f = \approx_{Id}$, where Id is the identity function on 2^W, and when we send $C(x, i, t)$ directly the final values will be $P^+(x)$. □

We have the following corollary:

Corollary 1 : When we communicate a 1-1 function according to a fair protocol, consensus on the value of the function among agents must be reached. □

Now we will formalize the notion of a function being *more or less informative* then another. Let $f \leq g$ iff $\approx_f \subseteq \approx_g$. Clearly $\leq$ is a partial order. Moreover constant functions are minimal elements of $\leq$, while 1-1 functions are maximal.

One important fact worth noticing here is that when no learning takes place in some world in a protocol in which a more informative function is communicated, there can be no learning if in the same world some less

informative function is communicated according to the same protocol:

Theorem 4 : If $f \leq g$, and for some world x, for all t, $C_g(x,i,t) = P_i(x)$, then for the same world x, for all t, $C_f(x,i,t) = P_i(x)$.

Proof: Induction on t. □

Unfortunately, as we will see from the following example, it may happen that a less informative function communicated in the same fair protocol with the same initial partitions may at the end bring more knowledge to participants.

Theorem 5: There is a 2-person protocol P and two functions f, g s.t. $f < g$ and when f, g are communicated according to P, then

$$C_f(x,i,\infty) \subset C_g(x,i,\infty)$$

for some x, i.

Proof: We will construct an appropriate example. Let $W = \{a,b,c,d\}$. Let

$P_1 = \{\{a,b\},\{c,d\}\}$

$P_2 = \{\{a,c\},\{b,d\}\}$

Both functions will be conditional probabilities of certain events E, F, so $f(X) = p(E|X)$, $g(X) = p(F|X)$ where initial probabilities are given as follows:

$p(E|a) = p(E|b) = p(F|a) = p(F|b) = 1/2$, $p(E|d) = p(F|d) = 1/4$, $p(E|c) = 3/4$, $p(F|c) = 1/3$

It is easy to check (see figure 3.3) that $f < g$, but $C_f(a,1,\infty) = \{a\}$, while $C_g(a,1,\infty) = \{a,b\}$. □

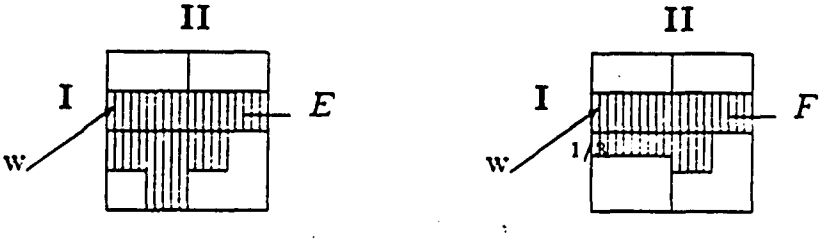


Figure 3.3: More Informative Function Gives Less Knowledge

This shows that the more informative functions do not necessarily bring more knowledge. Also, they are not necessarily more likely to bring about consensus (constant functions always bring consensus!). The question can be asked what are these important properties of functions that decide whether they bring consensus or not. We are interested in the situation when the protocol is fair, but the information given initially to participating agents may differ. The question is, for which functions f must consensus on the value of f be always reached, no matter how we choose the initial partitions.

3.3 Sure-Thing Principle

The case of two agents is well known - this was the case investigated by Geanakoplos and Polemarchakis and also by Bacharach. Let's define (following Bacharach) the "sure-thing" principle (Cave referred to the same condition as *union consistency*):

Definition: Function $f : 2^W \rightarrow D$ satisfies *the sure-thing principle* iff for every pair of disjoint sets $A, B \subseteq W$:

$$f(A) = f(B) \Rightarrow f(A \cup B) = f(A)$$

Note that the conditional probability function $f(X) = f(E|X)$ always satisfies the sure-thing principle.

Theorem 6 [Geanakoplos-Polemarchakis, Cave, Bacharach]: If two agents communicate values of a function satisfying the sure-thing principle, then they must reach consensus on the value of this function.

Proof: See e.g. [B]. □

It turns out that the sure-thing principle is not only a sufficient condition for reaching consensus among two agents, but it is also a necessary condition, as we can see from the following:

Theorem 7 : If f doesn't satisfy the sure-thing principle then there are partitions P_1, P_2 such that if agents communicate values of f, then no consensus is reached.

Proof: If f doesn't satisfy the sure-thing principle, there must be two disjoint sets $A, B \subset W$ s.t. $f(A) = f(B) = p$, $f(A \cup B) = q$, and $p \neq q$. Then let's define P_1, P_2 as follows:

$P_1 = \{A, B, W - (A \cup B)\}$

$P_2 = \{\{A \cup B\}, W - (A \cup B)\}$

We will show that for all worlds in $A \cup B$ no consensus can be reached. Let the real world w be in A (if $w \in B$ then the reasoning is the same). 1 sends p to 2. 2 cannot exclude any world because in all the worlds in $A \cup B$ 1 would

have sent p, so he sends $f(A \cup B) = q$ back to 1. This in turn will not give any information to 1, so his next message will be p as it was before and this process will continue ad infinitum. □

It turns out that when we consider even three agents communicating values of f according to some fair protocol - the sure-thing principle is too weak. As the following theorem shows, there are functions which satisfy the sure-thing principle, but they don't bring consensus in case of three agents.

Example 3 [j]: There is a function $f : 2^W \rightarrow R$ satisfying the sure-thing principle, a fair protocol P, and partitions P_1, P_2, P_3 of W s.t. when values of f are communicated according to P, then no consensus between agents is reached.

Let the space W equal the set { 1,2,3,4,5,6,7,8} . The function f satisfying the "sure thing" principle takes integer values and is defined as follows: let the subsets of W be numbered X_1,X_2,... and let $num(X_i)$ be i.

Now we let:

$f(\{1,2\}) = f(\{3,4\}) = f(\{1,2,3,4\}) = 1$

$f(\{5,6\}) = f(\{7,8\}) = f(\{5,6,7,8\}) = 2$

$f(\{1,3\}) = f(\{5,7\}) = f(\{1,3,5,7\}) = 3$

$f(\{2,4\}) = f(\{6,8\}) = f(\{2,4,6,8\}) = 4$

$f(\{1,5\}) = f(\{2,6\}) = f(\{1,2,5,6\}) = 5$

$f(\{3,7\}) = f(\{4,8\}) = f(\{3,4,7,8\}) = 6$

and for all other subsets X of W, $f(X) = num(X) + 6$.

This ensures that f satisfies the sure-thing principle. Now the participants are a,b,c and their partitions P_a, P_b, P_c are defined by

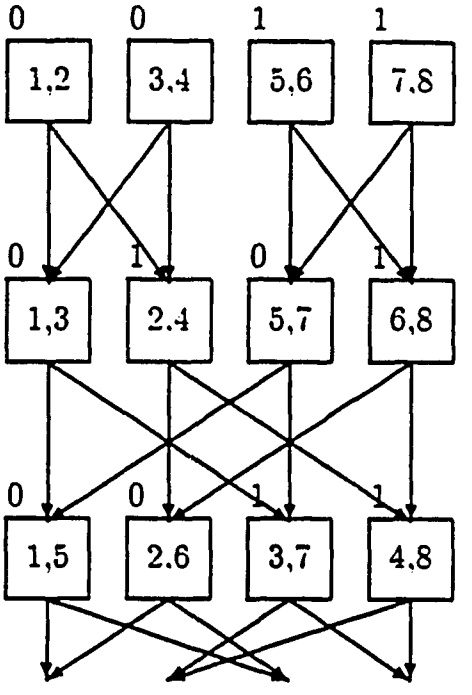


Figure 3.4: Three agents, Sure-Thing Principle Satisfied, No Consensus in a Fair Protocol

$P_a = \{\{1,2\},\{3,4\},\{5,6\},\{7,8\}\}$

$P_b = \{\{1,3\},\{2,4\},\{5,7\},\{6,8\}\}$

$P_c = \{\{1,5\},\{2,6\},\{3,7\},\{4,8\}\}$.

The real world is 1.

Notice that a sends a 1 to b so b knows that the real world is in {1,2,3,4}. He learns nothing since he knew anyway that it was in {1,3}. Similarly, b sends a 3 to c who learns nothing thereby and c sends a 5 to a who also learns nothing. Since the three participants have learned nothing in the first round, they will repeat their signals ad infinitum and consensus will not take place (see figure 3.4). □

We will now turn into the case of three agents.

3.4 Weakly Convex Functions

Now we will introduce a stronger condition than the sure-thing principle; this condition, weak convexity, implies the sure-thing principle, but not vice-versa. Weak convexity turns out to be a sufficient condition for reaching consensus among three agents.

Definition: A function f is *weakly convex* iff $f : 2^W \rightarrow R$ and for every $A, B \subset W$ disjoint,

$$Min(f(A), f(B)) \leq f(A \cup B) \leq Max(f(A), f(B))$$

All averaging functions are weakly convex, also conditional probability is weakly convex. Note that another equivalent way of expressing weak convexity is that f is weakly convex iff for every disjoint A, B there are numbers a, b such that $a, b \in [0, 1]$ (they are both in the closed interval 0-1) and $a + b = 1$ and

$$f(A \cup B) = af(A) + bf(B)$$

Theorem 8 (j): If there is a space W, finite partitions P_1,...,P_n of W and a weakly convex function f s.t. for some fair protocol Pr, the sets of possible limiting values are P^1,...,P^n, then there exist finite partitions P'_1,..., P'_n of W and protocol Pr' with the same graph as Pr such that executing Pr', we get the *same* set of limiting values, but no one gains any knowledge during the execution of Pr'. I.e. $C(w, i, 1) = C(w, i, \infty)$ for all i,w.

Proof: Simply take $P_i'(w)$ to be $C(w,i,\infty)$ and let $Pr'(t) = (s(t+T), r(t+T))$ where $Pr(t) = (s(t), r(t))$ and T is such that stability (no further change in the C's) is always reached by time T. □

In particular, if no consensus was reached in the first case, then in the second case also there will be no consensus, and moreover, no learning will ever take place.

Theorem 9 (j): Suppose there is a space W, a world $w \in W$ and a weakly convex function f such that when w is the actual world, and values of f are communicated according to a fair protocol, then no consensus takes place. Then there is another weakly convex function f', taking values in $\{0,1\}$ such that no consensus is reached when values of f' are communicated according to the same protocol.

Proof: To see that this theorem is true, assume by theorem 8, that the sets $C(x,i,\infty)$ are the same as the sets $P_i(x)$. Let x,i,j be such that i,j differ on the limiting value when the real world is x, and say $v(x,i,\infty) = f(P_i(x)) < v(x,j,\infty) = f(P_j(x))$. Let α be such that

$$f(P_i(x)) < \alpha < f(P_j(x))$$

and α is not a value of f. Now let, for all closed X:

$f'(X) = 0$ if $f(X) < \alpha$ and $f'(X) = 1$ if $f(X) > \alpha$.

The reader can easily check that f' is weakly convex and consensus does not take place. In fact, there is no learning with f' either. □

We note that for 2-valued functions, weak convexity is equivalent to the "sure-thing" principle. Note also that the last theorem does not hold for the

functions f satisfying only the "sure-thing" principle. For this reason the following theorem also needs the weak convexity assumption. The reduction from the general case to the 2-valued case cannot be carried out.

Theorem 10: If f is weakly convex and the protocol P is fair then if 3 participants communicate values of f according to P then consensus on the value of f must be reached.

Proof: To prove the theorem, assume that the values of a weakly convex function f are communicated between 1,2,3 using some fair protocol and consensus is not reached. By the previous two theorems we can assume that f is $\{0,1\}$ -valued and no knowledge is gained during the execution of the protocol.

Call a closed set X *unstable* iff for some persons i,j in a world w from X, i and j have different values. Let $U = \bigcup\{X | X \text{ is unstable}\}$. $U \neq \emptyset$ since we assumed that consensus is not reached. We can now express U in *two* ways:

$$\text{(i) } U = \bigcup\{C(w,i,\infty) | f(C(w,i,\infty)) = 0\ f(C(w,i \ominus 1,\infty)) = 1$$

$$\text{(ii) } U = \bigcup\{C(w,i,\infty) | f(C(w,i,\infty)) = 1\ f(C(w,i \ominus 1,\infty)) = 0$$

(where $x \ominus y = x - y \bmod 3$). The first representation is obtained by looking at a person who sends 0 and receives 1 (there must be such a person), and the second representation is obtained by looking at a person who sends 1 and receives 0.

Each union is disjoint. This is because suppose that a closed set X is unstable. Then let's take some world $w \in X$. In w if one person receives 0 and sends 1, other person receives 1 and sends 0 then the third person will

in w receive and send the same value. So w will be exactly in one set in the first representation and exactly in one set in the second representation.

The first representation forces us to assign $f(U) = 0$ while the second representation requires $f(U) = 1$ so we have a contradiction. This proves theorem 10. □

Note that for more then three people this proof breaks down, if we have four people communicating in a "round-robin" protocol, then if we have U - the set of all unstable elements, representing U using (i) and (ii) might give us non-disjoint sets, so we are not forced to assign any value to them. And as we will see soon, weak convexity doesn't guarantee consensus in case of more then three agents:

Example 4 [j]: [2] There is a weakly convex function f s.t. when values of f are communicated among 4 agents according to a fair protocol, then no consensus can be reached.

Proof: We give an example of a weakly convex function f, s.t. ' when values of f are communicated by 4 people using "round-robin" protocol then no consensus is reached.

Let the space W be the set $\{a, b, c, d, e\}$.
Let the function $f : 2^W \rightarrow \{0, 1\}$ be defined as follows:
$f(a) = f(b) = 1$
$f(c) = f(d) = f(e)$
$f(a \cup c) = f(a \cup d) = f(a \cup e) = f(b \cup c) = f(b \cup d) = f(b \cup e) = f(a \cup b) = 1$

[2] Ashok Chandra helped in constructing this example.

$f(c \cup d) = f(c \cup e) = f(d \cup e) = 0$

$f(a \cup c \cup d) = f(b \cup d \cup e) = f(c \cup d \cup e) = 0$

$f(b \cup c \cup d) = f(a \cup d \cup e) = f(a \cup c \cup e) = f(b \cup c \cup e) = f(a \cup b \cup c) =$

$f(a \cup b \cup d) = f(a \cup b \cup e) = 1$

$f(a \cup c \cup d \cup e) = f(b \cup c \cup d \cup e) = 0$

$f(a \cup b \cup c \cup d) = f(a \cup b \cup d \cup e) = f(a \cup b \cup c \cup e) = 1$

$f(a \cup b \cup c \cup d \cup e) = 1$

It can be easily checked that f is weakly convex. To complete our example we need to define partitions of W for all four participants:

$P_1 = \{\{a\}, \{b, c, d\}, \{e\}\}$

$P_2 = \{\{b\}, \{a, c, d\}, \{e\}\}$

$P_3 = \{\{b\}, \{a, d, e\}, \{c\}\}$

$P_4 = \{\{a\}, \{b, d, e\}, \{c\}\}$

If the real world is d then 1 sends value 1 to 2, but 2 learns nothing since $P_2(d) = \{a, c, d\} \subseteq \{w | f(P_1(w)) = 1\} = \{a, b, c, d\}$ so 2 sends value 0 to 3. Similarly 3 learns nothing upon receiving 0 from 2, and therefore sends value 1 to 4, 4 learns nothing and sends 0 to 1, 1 doesn't learn anything so the situation is stable. Nobody learns anything, and there is no consensus. In fact notice that for this particular example there is no world $w \in W$ s.t. the consensus will be reached in w. World d has an interesting property that values communicated "oscillate". □

Clearly weak convexity is not a necessary condition for reaching consensus, since there are some injective functions which are not weakly convex. We will investigate necessary conditions later on, but first let's see what kind of

sufficient condition we can find for the general case of any number of agents.

3.5 Strongly Convex Functions

It is enough to strengthen the convexity condition slightly to guarantee consensus in the general case. This stronger version we will call strong convexity, or simply convexity.

Definition: Function f, $f : 2^W \rightarrow R$ is *convex* iff f is weakly convex, and for every $A, B \subset W$ disjoint, if $f(A) \neq f(B)$ then:

$$Min(f(A), f(B)) < f(A \cup B) < Max(f(A), f(B))$$

Note that the conditional probability is convex. Also, like in the case of weak convexity there is another, equivalent way of expressing convexity: f is convex iff for every disjoint A, B there are numbers a, b such that $a, b \in (0, 1)$ (open interval) $a + b = 1$ and

$$f(A \cup B) = af(A) + bf(B)$$

Theorem 11 (j): For every strongly convex function f, if f is communicated according to a fair protocol, consensus on the value of f must be reached.

Proof: To proceed with the proof of the theorem, we assume for simplicity that $n = 3$, and that for all t, $s(t) = t \ mod \ 3$ and $r(t) = (t + 1) \ mod \ 3$. This protocol is evidently fair. Later on we will show how to generalize the proof to any n and to any fair protocol.

We will say that p is a *possible limiting value* for i if there is an x such that $p = f((C(x,i,\infty))$. Let the possible limiting values for individual 1 be $p_1,...,p_k$ with $p_1 < p_2 < ... < p_k$. Similarly $(q_1,...,q_r)$ are the values for 2 (in order) and $(r_1,...,r_m)$ are the limiting values for 3. We want to show that for all x, the limiting values generated by x, do not depend on i.

Let p be a possible limiting value for 1 and let $E(p)$ be the set of all x such that $f((C(x,1,\infty)) = p$. Then we can represent $E(p)$ as:

$$E(p) = \bigcup\{C(x,1,\infty)|f(C(x,i,\infty)) = p$$

hence $f(E(p)) = p$ by the "sure thing" principle. Call p and q *compatible* if there is an x such that $f(C(x,1,\infty)) = p$ and $f(C(x,2,\infty)) = q$. We now note that for such an x, for large t with $t = 1 \bmod 3$, 1 sends the value p to 2 but:

$$C(x,2,t+1) = C(x,2,t) = C(x,2,\infty)$$

and hence

$$C(x,2,\infty) \subseteq f(\{y|C(x,1,t) = p\}) \text{ thus } C(x,2,\infty) \subseteq E(p)$$

Thus $E(p)$ is a *union* of sets of the form $C(u,2,\infty)$ and hence p must be an average of those values q_i which are compatible with it.

Consider p_1, the smallest of the p's. The set of values q compatible with it must contain at least one element $\leq p_1$ and hence $q_1 \leq p_1$. By symmetry $r_1 \leq q_1$ and $p_1 \leq r_1$. Hence all are equal. (Notice here that this will also be true for f weakly convex.)

But now note that only q_1 can be compatible with p_1 since the other q_i are strictly greater and the average of q_1 with any other q_i would be strictly

greater than q_1. Similarly, only r_1 is compatible with q_1 and only p_1 with r_1. Thus, after time T if any of 1,2,3 is sending p_1, they all are, and if one of them isn't, then none of them is. (Here proof breaks down for functions which are weakly convex but not convex.)

Thus in particular, p_2 can be compatible only with q_i for $i \geq 2$, q_2 with r_i for $i \geq 2$ and r_2 with p_i for $i \geq 2$. We can now repeat our argument and show that $p_2 = q_2 = r_2$ and any of them forces the others. By induction we get, for all i, $p_i = q_i = r_i$ and all force each other.

We note now that our argument only used the fact that we are able to find a chain $t_1 < t_2 < ...., < t_p$, all greater than T and such that
(a) $s(t_1) = 1$
(b) the sender at t_{j+1} is the recipient at t_j and
(c) the chain passes through all participants, finally returning to 1. (1 may occur in between also). This is clearly implied by the condition of fairness so the theorem is proved. □

As a corollary we get the following:

Corollary 1 (j): If W is a probability space such that all elements of P^+ have nonzero probability, and A is some event, then the limiting probability of A is the same for all i, j, provided that the protocol is fair.

Proof of corollary: We note that in this case $P(A, i, t)$, the probability of A for i at time t, is $\pi(A|C(x, i, t))$ and $\pi(A|X)$ is a convex function. □

Example 5 [j]: Let's consider the case of statistical inference by Bayesian individuals. Suppose there are two dice, d_1 is fair but the other,

d_2 is fair only between {1,2,3,4,5} , never showing 6. d_1 is worth 1 dollar and d_2 is worth a hundred dollars. The two dice are put in a box which is shaken, and then one of the dice is taken out and tossed, showing a 2. Of two gamblers a, b, a is told that the result was under 4, while b is told that it is even. a now evaluates the probability that the die is d_2 as 6/11, and hence his expected value for the die will be $ 55. However, b thinks that the probability that the die is d_2 is only 4/9 and his expected value for the offered die will be $ 45. Thus if the die is offered at fifty dollars, clearly a would buy it, and b would not. What will happen if they communicate their values of the die to each other? Will these values converge? What if there are more than 2 gamblers?

To put this problem abstractly, Suppose that the random variable X takes values in a set W. X is distributed according to one of a finite number of probability distributions $W_1, W_2, ..., W_k$. We also have n individuals each of whom has his information partition P_i of the set W. The individuals share a utility function u:$\{W_1, W_2, ..., W_k\} \rightarrow R$, say $u(W_i)=u_i$. The result w of an experiment performed (to measure X) is (partially) communicated to each person by specifying $P_i(w)$.

The individuals communicate values of $f : 2^W \rightarrow R$ where for $A \subseteq W$, $f(A) = \sum_{i=1}^{k} p(W_i|A) \cdot u_i$. In order for f to be defined for all i-closed subsets of W, we need to assume that for $A \in P^+$, $p(A) > 0$. It is easy to check then that for disjoint, non-empty subsets A_1,A_2 of W, if $p(A_1 \cup A_2) \neq 0$, then $f(A_1 \cup A_2) = f(A_1) \cdot (p(A_2)/p(A_1 \cup A_2)) + f(A_2) \cdot (p(A_1)/p(A_1 \cup A_2))$. Hence f is convex so that when values of f are exchanged, then by the strong

convexity theorem, consensus will be reached. □

We will see later on that in the case described in the example consensus will be reached even if the only message communicated is whether the person is willing to buy a die or not.

3.6 Domination on Worlds

3.6.1 Cycles and the lack of consensus

So far the only necessary condition for reaching consensus was the sure-thing principle. Now we will try to find some stronger necessary conditions for the case of more then two agents communicating.

In order to find these necessary conditions we will look first at some examples of functions which do not guarantee consensus.

Robert Aumann gave the following colorful example showing why it may happen that consensus among communicating individuals is not reached despite the fact that they are "like-minded".

Suppose there are three detectives: Adams, Brown and Cox. In the detective school they all learned that adults are more reliable than teenagers (maturity), teenagers than children (accuracy) and children than adults (disingenuity). There are 3 material witnesses in the case: an adult, a teenager and a child. Testimony of every witness points to a different suspect. It is common knowledge that Adams interviews the adult and the teenager, Brown the child and the adult, and Cox the teenager and the child. Predictably,

Adams believes the adult, Brown the child, and Cox the teenager. Adams now tells Brown whom he is going to arrest, which Brown knew, since he also interviewed the adult, but then Brown prefers testimony of the child. He in turn tells Cox whom he is going to arrest and so on. Nobody gains any information from what he is told, and nobody's mind is changed by what he is told.

Crucial in this example was the cyclicity of the "reliability" relation on the set of witnesses (possible worlds). In this chapter I will try to formalize the concept of one world being more "believable" then the other, in a sense this more "believable" world like the more reliable witness "dominates" the other. This notion will be made precise in the following definition.

Definition : Let's define a *domination relation* ($\succ_f$) generated by f on 2^W as $A \succ_f B$ iff A,B disjoint, f(A)≠f(B) and $f(A \cup B) = f(A)$

Properties of $\succ_f$

Fact 1: If f is weakly convex, two-valued, then

$$C \succ_f A \cup B,\ A \cap B = \oslash,\ f(A) = f(B) \Rightarrow C \succ_f A \wedge C \succ_f B$$

Proof: Suppose otherwise. Let $C \not\succ_f A$. Then (since f is 2 valued) $A \succ_f C$. But we have $f(A \cup B \cup C) = f(C)$ (because $C \succ_f A \cup B$ but also $f(A \cup C) = f(A)$, so since $f(B) = f(A)$ then weak convexity forces us to assign $f(A)$ to $f(A \cup B \cup C)$ (notice that B and $A \cup C$ are disjoint). □

Fact 2: If f is weakly convex, two-valued, A, B, C are disjoint, then

$$f(A \cup B) = f(A \cup C) \neq f(A \cup B \cup C) \Rightarrow A \succ_f B \wedge A \succ_f C \wedge B \cup C \succ_f A$$

Proof : Clearly it is impossible that $f(A) = f(B) = f(C)$, so suppose that $f(A) = f(B) \neq f(C)$, but then $f(A \cup B) = f(A)$, so $f(A \cup B \cup C) = f(C)$ (since it is not equal to $f(A \cup B) = f(A)$). But then by fact 1 $C \succ_f A$ so $f(A \cup C) = f(C) \neq f(A) = f(A \cup B)$ - contradiction. Similarly it is impossible that $f(A) = f(C) \neq f(B)$. So it must be that $f(B) = f(C) \neq f(A)$. Then suppose that $A \not\succ_f B$, so $B \succ_f A$ therefore $f(A \cup B) = f(B) = f(C)$, so $f(A \cup B \cup C) = f(C)$ which is a contradiction. Hence $A \succ_f B$, and by similar argument $A \succ_f C$. Then clearly since $f(A \cup B \cup C) \neq f(A \cup B) = f(A)$ it must be that $B \cup C \succ_f A$. □

Fact 3: If f is weakly convex, two-valued, then

$$f(A) = f(B) \neq f(A \cup B) \Rightarrow$$

$$A \cap B \succ_f A - B \wedge A \cap B \succ_f A - C \wedge (A - B) \cup (A - C) \succ_f A \cap B$$

Proof : Apply fact 2 to sets A', B', C' where $A' = A \cup B$, $B' = A - B$, $C' = B - A$. □

Fact 4: If f is weakly convex, two-valued, A, B, C, D are disjoint then

$$A \succ_f B \succ_f C \succ_f D \Rightarrow A \succ_f D$$

Proof : By weak convexity $f(A \cup B \cup C \cup D) = f(A)$ (since we can express $A \cup B \cup C \cup D = (A \cup B) \cup (C \cup D)$ - union of two disjoint sets with the same value assigned by f). So suppose that $D \succ_f A$, but then by representing $A \cup B \cup C \cup D$ as $(A \cup D) \cup (B \cup C)$ it must be $f(A \cup B \cup C \cup D) = f(B) = f(C) \neq f(A)$ and this is a contradiction.

Definition: The relation $\succ_f$ is *cyclic* iff there is a sequence of sets of

worlds $A_1, ..., A_k$ s.t.

$$A_1 \succ_f A_2 \succ_f ... \succ_f A_k \succ_f A_1$$

This sequence we will call a *cycle* of length k.

If $\succ_f$ is not cyclic we will call it *acyclic*.

Note that the fact that $\succ_f$ is acyclic is equivalent to the fact that $< 2^W, \succ_f >$ is embeddable into $< R, >>$.

Fact 5: If f is 2-valued then all the cycles in $\succ_f$ are of even length.

Proof: Let's suppose that $A_1, ..., A_k$ form a cycle in $\succ_f$ where f is 2-valued. Without loss of generality we can assume that $f : 2^W \rightarrow \{0, 1\}$ and $f(A_1) = 1$. Then for all even i, $f(A_i) = 0$ and for all odd j, $f(A_j) = 1$ $(i, j \leq k)$. So since $f(A_k)$ must be 0 $(A_k \succ_f A_1)$ then we can conclude that k is even. □

Definition : A set of sets of worlds $\mathcal{A} \subseteq P(W)$ has *pairwise disjoint dominators with respect to f* iff

$$\forall A_1, A_2 \in \mathcal{A}\ f(A_1 - A_2) = f(A_1)$$

Intuitively we don't want two sets in our collection to share some "strong" (dominating) subset of worlds. Other way of expressing this condition is to say that

$$\forall A_1, A_2 \in \mathcal{A}\ A_1 \cap A_2 \not\succ_f A_1 - A_2$$

Theorem 12: If $\succ_f$ is cyclic, f is a two-valued function and moreover there is a cycle $A_1 \succ_f A_2 \succ_f ... \succ_f A_n \succ_f A_1$ which has pairwise disjoint

dominators with respect to f, then there exist partitions $P_1, ..., P_n$ of W s.t., when values of f are communicated in any world $w \in \bigcup A_i$ in an n-person round-robin protocol then no consensus is reached.

Proof: Suppose that we have a cycle in $\succ_f$ of length $n = 2m$ (see fact 5). $A_1 \succ_f A_2 \succ_f ... \succ_f A_n$, and $f(A_i - A_j) = f(A_i)$ for $i \neq j$. Let's assume that $f(A_{2k}) = 0$ and $f(A_{2k-1}) = 1$ for $k = 1, ..., n/2$. Now we will construct partitions $P_1, ..., P_n$. In P_i we define A'_j in the following way: $A'_j = A_j - (A_i \cup A_{i+1})$:

$P_1 = \{A_1 \cup A_2, A'_3, ..., A'_n, W - \bigcup A_i\}$

$P_2 = \{A_2 \cup A_3, A'_1, A'_4, ..., A'_n, W - \bigcup A_i\}$

$P_3 = \{A_3 \cup A_4, A'_1, A'_2, A'_5, ..., A'_n, W - \bigcup A_i\}$

...

$P_n = \{A_n \cup A_1, A'_2, A'_3, ..., A'_{n-1}, W - \bigcup A_i\}$

In any world $w \in A_i$, all people except the person $i - 1$ will send $f(A_i)$ while the person $i - 1$ will send the opposite value. □

Note also that the original example of a weakly convex 0-1 function for which consensus among 4 people is not reached was of this kind.

Corollary 1 : : If $\succ_f$ is cyclic, f is a two-valued function and moreover there is a cycle in $\succ_f$ built up of pairwise disjoint sets, then there exist partitions $P_1, ..., P_n$ of W s.t., when values of f are communicated in any world w in any of the sets forming the cycle in an n-person round-robin protocol then no consensus is reached.

Proof: If A_i, A_j disjoint, then $f(A_i - A_j) = f(A_i)$. □

Of course question is whether cyclicity of $\succ_f$ is a sufficient condition

for existence of partitions that might lead to a situation with no consensus reached. The answer to that is negative - as we will see from the following example - therefore the additional condition, that sets in a cycle have pairwise disjoint dominators, turns out to be essential.

Example 6 : This is a weakly convex function with cyclic domination relation s.t. in every world in every partition consensus on the value of f must be reached in every fair protocol.

$W = \{a, b, c, d, e\}$

$f(a) = f(b) = f(c) = f(d) = 1$

$f(e) = 0$ and $\forall i \in \{a, b, c, d\}\ f(\{e, i\}) = 0$

$f(\{b, d, e\}) = f(\{a, c, e\}) = 0$

for all other sets A, $f(A) = 1$.

Clearly $\succ_f$ is cyclic, since

$$\{a\} \succ_f \{b, e\} \succ_f \{d\} \succ_f \{c, e\} \succ_f \{a\}$$

First let's notice the following property of f: for any $B \subseteq A$, $\{e\} \succ_f A \rightarrow \{e\} \succ_f B$ (this is because e is the only element on which f gives value 0).

Now we will prove that f guarantees consensus; suppose otherwise, then there are some agents i and $i+1$ s.t. i sends 0 and $i+1$ sends 1 in w. If i sends 0 in w, then $e \in C(i, w, \infty)$. Nothing is learned, so

$$C(i+1, w, \infty) \subseteq \bigcup\{C(i, w, \infty) | f(C(i, w, \infty) = 0\} = C(i, w, \infty)$$

But then $e \in C(i+1, w, \infty)$, and by the property of f mentioned earlier, $f(C(i+1, w, \infty) = 0$. So we have a contradiction. □

3.7 Decision functions

So far we were dealing with 0-1 functions. These are the functions that carry the least amount of information, at most 1 bit. These functions are not only a source of examples where no consensus among agents is guaranteed. They are genuinely important since they are *decision* functions. They naturally model a situation in which the communication between agents is limited to a yes/no answer. In economics it can be for example whether agent is willing to bet (trade) or not. In computer science it may be the choice of an input value of a single bit.

Another interesting example is a truth value of a formula when the set W is a space of propositions.

There are some quite natural examples of decision functions, which when communicated in a fair protocol must create consensus among participants. Let's consider first the situation when our set W of possible worlds is finite. Now let's assume that we have probability distribution of a certain property E on W. One of the most interesting cases is when we want to decide "yes" iff $p(E|X) > \alpha$, where X is current set of possible worlds for an agent and α is some fixed threshold value.

We will prove that in such a case in fact consensus is guaranteed. This will be a consequence of the following theorem:

Theorem 13: If w is some weight distribution on W (finite), and if a decision function based on the weighted average is communicated in a fair

protocol; where f defined by:

$$f(A) = \begin{cases} 1 & \text{iff } [\Sigma_{a_i \in A} w(a_i)]/|A| > \alpha \\ 0 & \text{otherwise} \end{cases}$$

(for some α) then consensus is reached.

Proof: Let's suppose that consensus is not reached. Then when we shift w. to get $w' : W \to R$. $w'(A) = w(A) - \alpha$, we can express f as:

$$f(A) = \begin{cases} 1 & \text{iff } \Sigma_{a \in A} w'(a) > 0 \\ 0 & \text{iff } \Sigma_{a \in A} w'(a) < 0 \end{cases}$$

Note that:

$$\frac{\Sigma_{a \in A} w'(a)}{|A|} = \frac{\Sigma_{a \in A}(w(a) - \alpha)}{|A|} = \frac{\Sigma_{a \in A} w(a) - |A|\alpha}{|A|} = \frac{\Sigma_{a \in A} w(a)}{|A|} - \alpha$$

So in fact:

$$\frac{\Sigma_{a \in A} w'(a)}{|A|} > 0 \text{ iff } \frac{\Sigma_{a \in A} w(a)}{|A|} > \alpha$$

If consensus is not reached in some fair protocol when values of f are communicated then there must be some *unstable* worlds, worlds w for which there are two agents i, j s.t. in a stabilised situation (when nothing more can be learned) i sends 0 and j sends 1.

Let's call the set of unstable worlds U. U is not empty. We can express U in two ways: $U = U_0 = U_1$ where

$$U_0 = \bigcup_{i,w} \{C(i, w, \infty) | f(C(i, w, \infty)) = 0,\ f(C(i \ominus 1, w, \infty)) = 1\}$$

and

$$U_1 = \bigcup_{j,w'} \{C(j, w', \infty) | f(C(j, w', \infty)) = 1,\ f(C(j \ominus 1, w', \infty)) = 0\}$$

For U_0 we have a sequence of inequalities of the form

$$\sum_{a \in C(i,w,\infty)} w'(a) < 0$$

Adding up this sequence we'll get inequality

$$\beta_1 = \sum_{a \in U_0} k_a \cdot w'(a) < 0$$

where k_a is the number of sets $C(i, w, \infty) \subseteq U_0$ s.t. $a \in C(i, w, \infty)$. Similarly by looking at U_1 we'll get

$$\beta_2 = \sum_{a \in U_1} l_a \cdot w'(a) > 0$$

where l_a is the number of sets $C(j, w', \infty) \subseteq U_1$ s.t. $a \in C(j, w', \infty)$.

But now notice that if during the execution of one round of the protocol for a certain world w value communicated was changed l_w times from 0 to 1, then value communicated must have been changed the same number of times from 1 to 0. So $k_a = l_a$ for every a. Thus $\beta_1 = \beta_2$ and we get a contradiction. □

Remark 1: Notice that f is weakly convex, since if A, B are disjoint, then if $f(A) = 0$ and $f(B) = 0$ then

$$\Sigma_{a \in A} w(a) < \alpha |A|$$

$$\Sigma_{b \in B} w(b) < \alpha |B|$$

and adding these two equalities we get

$$\Sigma_{a \in A} w(a) + \Sigma_{b \in B} w(b) < \alpha(|A| + |B|)$$

so $f(A \cup B) = 0$. Similarly if $f(A) = f(B) = 1$. □

Remark 2: Not every decision function with an acyclic domination relation can be expressed as a decision based on weighted average. For example if we have

$$\{a\} \succ_f \{b\} \text{ and } \{a\} \succ_f \{c\} \text{ but } \{b,c\} \succ_f \{a\}$$

then we cannot assign weights to a, b, c so that f would be based on averages of weights. □

As a corollary to theorem 13 we will prove that every decision function based on conditional probability brings consensus when communicated according to a fair protocol.

Theorem 14 : If p_0 is a rational number, then the decision function d_E

$$d_E(A) = \begin{cases} 1 & \text{iff } p(E|A) > p_0 \\ 0 & \text{otherwise} \end{cases}$$

based on conditional probability of E brings consensus when communicated according to some fair protocol.

Proof: Let $|P^+| = n$. Let's order elements of P^+: $X_1, \ldots, X_n$. Let $W' = \{w_1, \ldots, w_n\}$. There is a mapping $m : P^+ \to W'$, $m(X_i) = w_i$. We can extend m to the all closed subsets of W in a natural way: $m(X) = \bigcup\{m(X_i) | X_i \in P^+\}$. Clearly if the values of the function f' are communicated according to the same protocol in W with partitions P_i and in W' with partitions P_i' defined by a mapping m (where $f'(m(X)) = f(X)$) then consensus on f' is reached iff consensus on f is reached. So, w.l.o.g. we can assume that our space W is finite.

Now let's notice that if all the worlds are equally likely, then when we take $p(E \cap \{w\})$ to be the weight of w, then $d_E(A)$ is just an average weight

of A, so our theorem follows directly from theorem 13.

If weights of the points are different, we can create another set W', and a mapping $\varphi : W' \to W$, s.t. weights of all points are the same in W', and for all sets in $P^{+}_{W'}$, $p_{W'}(E|A) = p_W(E|\varphi(A))$ (we extend φ to a mapping $P(W') \to P(W)$ in the obvious way). So if there is no consensus in general case, there would be no consensus in a case where all the points are equally likely. □

The last theorem deals with the case when agents are willing to bet on E iff their beliefs that E is true are above certain threshold value. If I assign value $p(E|A)$ to the set A, it means that the degree of my belief in E if all I know about the real world is A is $p(E|A)$.

This is the classical Bayesian approach. The degree of belief assigned by an individual to A is a sum of beliefs assigned to elements of A. This requires prior assignment of probabilities (masses) to all points in the domain (all possible worlds). Very often this is not a realistic assumption. Also, for the same reason if a Bayesian individual assigns p_0 to A, then he is forced to assign $1 - p_0$ to $\neg A$. But the value of p_0 may be low not because of individual's high disbelief in A but simply because of the lack of information. As a remedy to these problems with the classical approach, another way of computing beliefs was proposed by Dempster and later on developed by Shafer [D], [Sh]. Here instead of requiring the prior probability distribution on W we will require probability distribution on $P(W)$. If prior probability of E (I will call it "mass" of E - m_E) is assigned to a set A - it means that we are not willing to assign any part of it to any of the subsets of A (we

don't know where it should be located in A), but we *are* willing to assign the whole of it to A. In the Dempster-Shafer approach belief P_* is defined as:

$$P_*(E|A) = \Sigma_{B \subseteq A} m_E(B)$$

As we will see P_* is not necessarily weakly convex, therefore consensus on the value of P_* in a fair protocol is not guaranteed. This is shown as the following example:

Example 7 : P_* is a Dempster-Shafer belief function and values of P_* are communicated then the consensus on value of P_* is not guaranteed (even for 2 people).

Let $W = \{a, b\}$ and let P_* be defined using mass function m as follows:
$m_E(\{a\}) = m_E(\{b\}) = 1/4$, $m_E(\{a, b\}) = 1/2$
Now $P_*(E|A) = \Sigma_{B \subseteq A} m_E(B)$, so:
$P_*(E|\{a\}) = P_*(E|\{b\}) = 1/4$, $P_*(E|\{a, b\}) = 1$
Let's take the following partitions:
$P_1 = \{\{a\}, \{b\}\}$
$P_2 = \{\{a, b\}\}$

3.7.1 Voting Schemes

In Aumann's example (section 3.6.1) the set of possible worlds was interpreted as a set of witnesses and agents have to choose the most reliable one, and make a decision which follows the recommendation of the most reliable witness. This is a quite common situation. Very often, agents must make a decision based on a expertize - but experts disagree. We have to consider

the views of all the experts who (as far as we know) might be right, and then choose. We assume here that one expert is in fact right, and none of the agents will ever exclude the expert who is right from consideration. We can further interpret this as a vote among the experts. In case of a decision function we will interpret worlds as votes "for" (when value is 1) or "against" (when value is 0); a function which assigns our decision depending on the experts' votes we will call a *voting scheme*. Voting schemes are important in many applications, they are also sometimes called social choice functions. We will show that in both extreme cases - democracy and dictatorship - consensus must be reached.

First we will consider "democracy".

Theorem 15: If we have preassigned values from {0,1} to all elements of a finite set W, and our function f is defined as a fixed majority vote with fixed default value in case of a tie (say 0), then consensus on the value of f in a fair protocol must be reached.

Proof : If we have our f defined as:

$$f(A) = \begin{cases} 1 & \text{iff } |\{a|a \in A\ w(a) = 1\}| > p|\{a|a \in A\ w(a) = 0\}| \\ 0 & \text{otherwise} \end{cases}$$

then we can express f as:

$$f(A) = \begin{cases} 1 & \text{iff } [\Sigma_{a_i \in A} w(a_i)]/|A| > p \\ 0 & \text{otherwise} \end{cases}$$

□

Note that if we add the third value - say "undecided" which will mean e.g. that the vote must be repeated in case we have a tie, consensus also must be reached.

Theorem 16: If we have preassigned values {yes,no} to all elements of a finite set W, and our function f is defined as a fixed majority vote with value "undecided" in case of a tie, then consensus on the value of f in a fair protocol must be reached (it may be "undecided").

Proof : We can define f' as:

$$f'(A) = \begin{cases} 1 & \text{iff } [\Sigma_{a_i \in A} w(a_i)]/|A| > p \\ 1/2 & \text{iff } [\Sigma_{a_i \in A} w(a_i)]/|A| = p \\ 0 & \text{iff } [\Sigma_{a_i \in A} w(a_i)]/|A| < p \end{cases}$$

We interpret 1 as "yes", 0 as "no", 1/2 as "undecided". Notice that f' is weakly convex, and $\approx_f = \approx_{f'}$, so consensus is guaranteed for both f and f' or for neither of them (see theorem 3). Let's suppose that consensus is not reached for f'. But then consensus would not be reached for some function g of the form:

$$g(A) = \begin{cases} 1 & \text{iff } [\Sigma_{a_i \in A} w(a_i)]/|A| > p_0 \\ 0 & \text{otherwise} \end{cases}$$

which is impossible by the previous theorem. □

Now we will analyze "dictatorship" (or rather the strictly hierarchical case). Here I assume that there is an ordering of experts (worlds), s.t. for every two experts there is always a "preferred" one, or - if we cannot choose between them, they recommend the same thing (the hierarchy of experts is shared by all agents, and this fact is common knowledge). Then consensus among agents must be reached as the following theorem shows.

Theorem 17: Let $f : W \to S$, let $\leq$ be some ordering of W, s.t. $\leq$ is reflexive, transitive and for every pair w_1, w_2 of elements from W $w_1 \leq w_2$ or $w_2 \leq w_1$, if $w_1 \leq w_2$ and $w_2 \leq w_1$ then $f(w_1) = f(w_2)$; then when values of

f' are communicated in a fair protocol, then consensus among agents must be reached, where f' is defined as follows:
$f'(A) = f(a_0)$ where a_0 is the least element of A with respect to $\leq$.

Proof: Let's look at the situation when nothing is learned.

$$C(w, i+1, \infty) \subseteq \bigcup \{C(w', i, \infty) | f(C(w', i, \infty)) = p_i\}$$

where p_i is the value sent by i. By the definition of f we must have in each $C(w, i, \infty)$ some dominating world x_i, so:

$$\forall i \; \exists x_i \; x_i \in C(w, i, \infty)$$

$$\forall x \; x \in C(w, i, \infty) \; x \leq x_i$$

so we have:

$$x_n \leq x_{n-1} \leq \ldots \leq x_1 \leq x_n$$

therefore by the property of $\leq$, all $f(x_i)$'s must be equal. □

3.7.2 Other two-valued functions

There are other interesting and important decision functions which guarantee consensus in a fair protocol. First we will get a result which is a direct consequence of the previous theorem.

Theorem 18: Every function $f : W \rightarrow R$ (where W - finite) of the form:

$$f(A) = max(\{f(a) | a \in A\})$$

brings consensus in a fair protocol (the same is true for f defined as minimum).

Proof: Notice that if we define $w_1 \leq w_2$ iff $f(w_1) \leq f(w_2)$ then $\leq$ will have all the properties required in the previous theorem, moreover:

$$f(A) = f(a_0) \text{ s.t. } a_0 \in A,\ \forall a \in A\ a \leq a_0$$

so f can be expressed the way theorem.. requires. □

Corollary : If W is a space of propositions (finite) with a certain boolean valuation $v : W \rightarrow \{0,1\}$ and we define ..

$$f(A) = v'(\bigvee a \in A)$$

where v' is a natural extension of a valuation v then the consensus when values of f are communicated in a fair protocol must be reached. (Similarly for $f(A) = v'(\bigwedge a \in A)$)

Proof : Notice that $f(A) = max(\{v(a)|a \in A\})$. □

3.8 Summary

In this chapter we investigated the protocols, in which agents hide certain private information using common coding function. We showed how certain characteristics of coding function force the agents to reach consensus on the value of the function in every protocol in which everybody can influence everybody else's knowledge infinitely many times.

We were using examples from economics because the initial work in this area was done by economists. But this model has a natural computer science interpretation.

Let W' be a set of global histories (of fixed length) of a distributed system with a global clock accessible to all the processors. Every processor i has a partition of W' given by his local history - projection function Φ as:

$$\tilde{P}_i(H) = \{H' | \Phi_i(H') = \Phi_i(H)\}$$

let's also assume that there are finitely many elements in every partition. We assume that the programs run by the processors are common knowledge. Therefore all the partitions are common knowledge. Now the processors start the procedure to determine the global state (global history).

This procedure is not considered to be a part of the history. They send messages according to some fair protocol, where messages are some codes of their local histories (coding mechanism is common to all).

Coding may be necessary because of the limited capacity of the communication buffers, limit on the length of messages. It is irrelevant whether communications are synchronous 2-casts or asynchronous send and receive.

Since such an interpretation fits the framework described in this chapter, all the results will apply and the ability of the system to reach consensus among processors on the code of a global history will depend on the properties of the coding function exactly the way we found here.

Chapter 4

Open Problems

4.1 Problem of logical omniscience

There was a lot of criticism of S5 axioms for knowledge. Although we have obtained many interesting results using models satisfying these axioms, still it seems like they fail to capture some intuitive properties of knowledge. And they seem not to be well suited to describe certain properties of both human beings and computers.

The fact that knowledge is closed under modus ponens:

$$K_i(A \rightarrow B) \wedge K_i A \Rightarrow K_i B$$

implies that knowers are perfect reasoners[1]. For example if we take first order logic (or some first order theory) where modus ponens is the only

[1]The problem arises because possible worlds semantics is used, not because we assume that information partitions the set of possible worlds W. Suppose that we don't require our relation $\approx_i$ to be an equivalence, instead we take any binary relation R_i on W. When the semantics of a knowledge operator K_i is given by $w \models K_i A$ iff $\forall w'$, $(w, w') \in R_i$ $w' \models A$,

rule of inference; then if a knower knows the axioms, then he knows all the theorems.

Moreover[2], he knows what he doesn't know, so if some statement is false, he cannot know it, therefore he knows that he doesn't know it. So if our knower knows a complete set of axioms, he can decide the validity of every statement by "introspection". No open problems remain. Everybody is omniscient. And this clearly is neither an attribute of men nor machines.

This problem, known as a problem of *logical omniscience* still lacks a satisfactory solution, though several attempts have been made.

The main application of logic of knowledge to computer science so far seemed to be in proving correctness of distributed protocols by showing that they do in fact implement certain knowledge-based algorithms. Also sometimes it is much easier to write high-level, knowledge-based algorithms as a first step towards solving some practical problem in the area of distributed computing. In such an algorithm there may be instructions of the form:

if $K_i A$ *then do action1 else do action2*

including explicit tests of knowledge. If a processor i cannot compute the truth value of $K_i A$, the algorithm which includes a knowledge test becomes useless.

To exclude such situations we would like to say that i knows A iff A is true in all histories compatible with H for i and *i is able to compute*

---

closure of knowledge under modus ponens will remain sound. We may modify the possible worlds semantics so that the closure under Modus Ponens is not guaranteed anymore. One of possibilities is introducing "queer" possible worlds, worlds in which everything is possible, so nothing is known (see [Fi]). This approach certainly requires further investigation.

[2]Here assumption that $\approx_i$ is an equivalence becomes crucial.

K_iA. Depending on what "i is able to compute" means, we will get different notions of resource-bounded knowledge. We will describe some of these in the following section.

4.1.1 Resource-bounded reasoning

To avoid the property of logical omniscience Yoram Moses proposed the following approach [M]: keep the classical definition of knowledge with S5 axioms. This will be implicit knowledge. But we also need to consider resource-bounded knowledge (relative, explicit). His definition is: $H \models K_i^B A$ iff $H \models K_iA$ and there is a decision algorithm M for K_iA (complexity of M within bound B) which on input h_i, where $h_i = \Phi_i(H)$ gives the answer "true".

How does the knowledge generator M work? Input is the local history. So if the relevant fact involves other processes, then we first need to generate the set of all global histories compatible with h_i. Then we need to check that in fact every history generated in such a way satisfies A.

My objection is that it may happen that there is an appropriate resource-bounded algorithm to decide A , but the knower i might not know it. This would create paradoxical situations: say when it was proven that Linear Programming is in P then immediately all P-bounded knowers learned all the facts that required LP in their decision procedures. similarly, suppose that some day somebody finds a fast (P) factorization algorithm. Does it mean that factorization is effectively known today to every P-bounded knower?

Also Moses applied his definition only to the case of unreliable systems where complexity was in terms of a number of failures. It is not quite clear what the results of applying his definition to fully reliable systems (like the one described in the chapter 2) would be. The natural complexity measure seems to be in terms of the length of a local history. The complexity of knowledge algorithms in case of different protocols certainly seems worth studying.

Moses' approach was developed by Fischer and Zuck in [FZ], where the authors (probably to avoid paradoxes we have mentioned) introduced one more parameter in the definition of knowledge. Now processor i knows A with a resource bound B relative to the machine M iff there is a machine M accessible to i, which computes K_iA within bound B. Here processes might not know some A in spite of the existence of an algorithm to determine that in fact K_iA holds. This algorithm may be not known to i. Still properties of this new notion of knowledge remain to be examined.

A different solution to the problem of logical omniscience was proposed by Rohit Parikh in [P4]. He argues that when we talk about knowledge we usually mean one of 2 different notions: either a "linguistic knowledge" (an ability to consistently answer queries about A) or a "behavioral knowledge" which is an ability to consistently perform different actions depending on the truth value of A.

Later he gives a definition of a "knowledge algorithm" as a database together with a procedure which on input which is a pair (question, resource bound) answers to a question (yes - if the procedure can answer it positively

within a resource bound) and updates a database. This definition acknowledges the fact that by (even unsuccessful) trials of answering queries our knowledge may grow, so in the next attempt we use what we've learned earlier and we may answer positively (within a resource bound) something that couldn't be answered within the same resource bound before the database was updated. So far, the properties of knowledge defined in such a way also remain to be investigated.

4.1.2 Probabilistic Knowledge

When we look at distributed processes, information which is the basis for the processors' knowledge is obtained by a processor through the messages received from the others and through the observation of its own behavior (if processors are non-deterministic, observing local actions gives information to a processor).

So far we have been assuming that the information that processors receive is always perfect. What if there is some chance of an error? Then intuitively we have learned something, but we cannot *exclude* any of the possible worlds. What has happened is that some of the worlds have became less likely than the others.

If the probability of the message being correct is high, then we might say that with this high probability we know, what we would have known if the message is guaranted to be correct. This leads to a notion of probabilistic knowledge, which recently attracted a noticeable interest.

Another notion of probabilistic knowledge may be derived from analyzing the situations where messages are always correct, but their connection with the truth value of a formula is probabilistic. As an example of such a situation we can consider weather forecasting. Today's temperature, humidity etc can be recorded correctly and still based on all that information we are only able to talk about the probability of rain tomorrow.

There were some attempts to define logics which can handle both probability and knowledge (e.g. [LS], [HMc]), but knowledge there was usually independent of probability. In [HZ] probabilistic knowledge was defined in the context of the resource-bounded reasoning. There i knew A with probability p if i has a probabilistic machine for answering queries about A (probabilistic algorithm). The problem of defining what "i knows A with probability p" means so that this reflects properties of the world rather then the fact that i is using some particular probabilistic algorithm, remains open.

4.1.3 Restricted Views

So far we indicated what attempts were made to handle two idealizing assumptions about knowers:

1. unlimited computing power
2. perfect information

When we talk about the computing power, there are two problems involved: limited resources that the machine can use for computations (e.g. in terms of, if we use Turing Machine model, number of steps required, or number of squares of the worktape used) and also limited size of the machine itself (in

terms of the number of states).

Since our model assumed that every two local histories of i which *are* different, also *appear* different to i, we implicitly assume that i has an unbounded memory and unbounded number of states. This assumption can be lifted quite naturally without any major change of the model. In [HM] the authors gave the following semantics of the knowledge operator: $H \models K_i A$ iff $\forall H'$ s.t. $v_i(H) = v_i(H')$ $H' \models A$, where v_i is a view function for the ith processor. It never distinguishes more then its local history (if $\Phi_i(H) = \Phi_i(H')$ then $v_i(H) = v_i(H')$) but it may distinguish substantially less. In a possible worlds semantics such a definition defines an equivalence relation, so again we face the problem of logical omniscience. Here processors are perfect reasoners with unlimited computing power, but (possibly) with a limited number of states, which in turn may restrict their ability to remember local histories.

This kind of interpretation applies to our work described in the chapter 3. There communicating agents had their knowledge given as partitions, but what they were communicating wasn't what they knew, it was some function f which provides only partial information. Our f corresponds to the view function of [HM], but it models a limitation on the message size (channel capacity) rather then the limitations of processors themselves.

4.2 Lack of Knowledge

Often we need to know that somebody doesn't know something. This kind of "negative" knowledge is especially important in cryptography. Unfortunately formulae of the form $\neg K_i A$ are not persistent, so our procedures for realizing levels of knowledge cannot be used. Chandy and Misra [CM] showed that knowledge can be lost in asynchronous systems by sending a message. If a sender knew that a receiver didn't know a certain fact, then after he sends a message informing of this fact, he neither knows that the intended receiver knows the fact, since the message may be still in the buffer, nor does he know that the receiver doesn't know that fact, since the message could have been delivered promptly.

Still, the knowledge of formulae which are not persistent is not very well understood. There may be some formulae false upon sending which become true when received and vice versa. As an example of such a surprising formula consider a statement (due to Dexter Kozen) "You don't know it, but there is a bug crawling on your back".

Problems with the reasoning about the non-persistent formulae correspond to known problems with non-monotonic reasoning.

4.3 Other directions for further research

All the problems mentioned in this chapter require further research. Here I want to mention some other open problems in the area of application of

knowledge to computer science.

1. Knowledge-based semantics for programming languages (for distributed systems) leading to a knowledge-based proof systems. CSP seems to be a language suitable for such approach.

2. Comprehensive analysis of the properties of distributed protocols in terms of knowledge (properties like safety, liveness, fairness etc).

There is one important open problem left directly related to our research.

What is the complexity of the consensus algorithms (in terms of the number of rounds needed)?

We know that when 2 people reach an agreement, $k+l$ rounds are enough (where k, l are the number of elements in the partitions of both people). What is a sufficient number of rounds in case of n people?

Another natural extension of our research would be some notion of the distance between the levels of knowledge, and a corresponding analysis of how to create a level L' of A starting with a level L in a cheapest way (where presumably synchronous communication is more "expensive" then asynchronous communication).

Certainly, knowledge based analysis is a new and very promising area of theoretical computer science. Relatively very little was done so far, since the interest in using knowledge to reason about distributed systems is quite recent. New theorems wait to be discovered, new applications wait to be found.

Bibliography

[A] Aumann R., "Agreeing to Disagree", *Annals of Statistics*, 4 (1976) pp 1236-1239.

[B] Bacharach M., "Some Extensions of a Claim of Aumann in an Axiomatic Model of Knowledge", *J. Economic Theory*, 37 (1985) pp. 167-190.

[C] Cave J., "Learning to Agree", *Economics Letters*, 12 (1983) pp. 147-152

[CM] Chandy M. and Misra J., "How Processes Learn", *Proceedings of 4th ACM Conference on Principles of Distributed Computing* (1985) pp 204-214.

[Ch] Chuang Tsu, *Inner Chapters, a new translation by Gia-Fu Feng and Jane English*, Vintage Books, a division of Random House, New York, 1974

[CM2] Clark H. and Marshall C., "Definite Reference and Mutual Knowledge" in *Elements of Discourse Understanding* Ed. Joshi, Webber and Sag, Cambridge U. Press, 1962.

[FH] Fagin R. and Halpern J., "Belief, Awareness, and Limited Reasoning", *Proc. of the 9th IJCAI*, 1985, pp.491-501

[Fi] Fitting M., *Proof Methods for Modal and Intuitionistic Logic*, D.Reidel Publ. Company, Dordrecht, 1983

[FI] Fischer M. and Immerman N., "Foundations of Knowledge for Distributed Systems, *Yale Univ. Tech. Report* YALEU/DCS/TR-450, December 1985

[FZ] Fischer M. and Zuck L., "Relative Knowledge and Belief" (Extended Abstract), *Yale Univ. Tech. Report* YALEU/DCS/TR-589, December 1987

[GP] Geanakoplos J. and Polemarchakis H., "We Can't Disagree Forever", *J. Economic Theory*, 28 (1982) pp. 192-200.

[Ha] Halpern J., "Reasoning About Knowledge: An Overview", *Theoretical Aspects of Reasoning About Knowledge: Proc. of the 1986 Conference*, (J. Halpern, ed.),Morgan Kaufmann, 1986, pp.1-17.

[HMc] Halpern J. and McAllester D., "Knowledge, Likelihood and Probability", *Proceedings of AAAI-84*, (1984), pp.137-141.

[HM] Halpern J. and Moses Y., "Knowledge and Common Knowledge in a Distributed Environment", *Proc. 3rd ACM Conf. on Principles of Distributed Computing* (1984) pp. 50-61. (To appear in *Jour. Assoc. Comp. Mach.*)

[HM2] Halpern J. and Moses Y., "A Guide to the Modal Logics of Knowledge and Belief", *Proc. of the 9th IJCAI* (1985) pp. 480-490.

[HZ] Halpern J. and Zuck L., "A Little Knowledge Goes a Long Way: Simple Knowledge-Based Derivations and Correctness Proofs for a Family of Protocols", *Proc. 6th ACM Symp. on Principles of Distributed Computing* (1987), pp.269-280.

[H] Higman G., "Ordering by Divisibility in Abstract Algebras", *Proc. Lon. Math. Soc.* 2 (1952) pp 326-336.

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

[JP] de Jongh D.H.J. and Parikh R., "Well Partial Orderings and Hierarchies", *Proc. Kon. Ned. Akad. Sci* Ser. A, 80 (1977) pp 195-207.

[K] Kripke S., "Semantical Analysis of Modal Logic", *Zeitschrift fur Math. Logik und Grundlagen der Mathematik 9*, 1963, pp. 67-96.

[Lad] Ladner R.E., "The Computational Complexity of Provability in Systems of Modal Propositional Logic", *SIAM J. of Computing 6:3*, 1977, pp. 467-480.

[Lam] Lamport L., "Time, Clocks and the Ordering of Events in a Distributed System", *CACM* , vol. 21, #7, July 1978, pp. 558-565.

[LS] Lehmann D. and Shelah S., "Reasoning with Time and Chance", *Information and Control*, **53**, 1982, pp. 165-198.

[L] Lewis D., *Convention*, Harvard University Press, 1969.

[MDH] Moses Y., Dolev D. and Halpern J., "Cheating Husbands and Other Stories", *ACM-PODC 1985* pp. 215-223.

[M] Moses Y., "Resource-bounded Knowledge", *Theoretical Aspects of Reasoning About Knowledge: Proc. of the 1988 Conference*, (M. Vardi, ed.),Morgan Kaufmann, 1988, pp. 261-276.

[MT] Moses Y. and Tuttle M., "Programming Simultaneous Actions Using Common Knowledge", *Proc. of the 27th Annual Symp. on Foundations of Computer Science*, 1986, pp. 208-221. To appear, *Algorithmica*

[P1] Parikh R., "Knowledge and the Problem of Logical Omniscience", *ACM-SIGART ISMIS* , (Z. Ras and M. Zemankova, ed.), Elsevier Science Publ., 1987, pp. 432-439

[P2] Parikh R., "Logics of Knowledge, Games, and Dynamic Logic", *FST-TCS 84*, Springer LNCS #181, pp. 202-222.

[P3] Parikh R., "Some Recent Applications of Knowledge", *FST-TCS 87*, Springer LNCS #287, pp. 528-539.

[PK1] Parikh R. and Krasucki P., "Levels of Knowledge in Distributed Computing", (**to appear**), earlier version of the paper under the same title was published by R. Parikh in a proceedings of *IEEE Symposium on Logic in Computer Science*, Boston 1985, pp. 322-331.

[PK2] Parikh R. and Krasucki P., "Communication, Consensus and Knowledge", to appear in *The Journal of Economic Theory.*

[PR] Parikh R. and Ramanujam R., "Distributed Processes and the Logic of Knowledge", *Logics of Programs* Springer LNCS #193, pp 256-268.

[S] Stark R., "Logics of Knowledge", *Zeit. f. Math. Logik und Grundlagen der Math.* 27 (1981).