Resource-bounded Knowledge (Extended Abstract)

Yoram Moses

Department of Applied Mathematics
The Weizmann Institute of Science
Rehovot, 76100 ISRAEL

Abstract: Traditional treatments of knowledge in distributed systems have not been able to account for processors' limited computational resources. This paper presents definitions of resource-bounded knowledge, belief, and common knowledge that in a precise sense capture the behavior of resource-bounded processors. Subtle properties of the resulting notions are discussed, and they are successfully applied to two problems in distributed computing.

What is knowledge? Can anybody answer this question?
Socrates

1 Introduction

A crucial problem in developing a rigorous theory of knowledge that faithfully models agents of interest is in appropriately capturing the fact that the agents, whether they are humans or computing agents, have limited resources. The traditional ***possible worlds*** semantics for knowledge (see [H]) is well known to suffer from what Hintikka terms the *logical omniscience* problem: An agent necessarily knows all consequences of the facts it knows. Such semantics definitely do not account for the fact that our agents' computational resources are limited. While there have been a number of attempts to overcome the logical omniscience problem in the literature (see [E,MH,C,Ra,RB,Ko,FH]), none of them provides machinery with which to capture specific resource bounds on the agents' computational abilities (or even just the fact that agents are restricted to computing computable functions!). Our purpose in this work is to provide a basis for a rigorous theory of resource-bounded knowledge and belief.

How can we define knowledge in a way that will account for agents' resource limitations? Consider, for example the case in which the agents in question are polynomial-time bounded processors. One "logical" approach would be to start out with some reasonable set of axioms and inference rules and represent the agent's local information by a set of assertions. We can then say that the agent "knows" a given fact if the fact is provable by a derivation of polynomial length. Such a definition immediately raises a number of critical problems: First, what could constitute a "reasonable" set of axioms and inference rules? Slightly different choices yield extremely different results. For example, adding as an axiom a theorem that requires exponential time to derive from the other axioms would add a large number of "known" facts that were previously not known. Along the same lines, what does the set of assertions corresponding to a polynomial-time agent's local information consist of? How can we map an agent's local state into such a set of assertions? An additional important point to notice here is that whether or not the agent knows a given fact according to such a definition depends on the *existence* of a polynomial length proof of this fact – an inherently nondeterministic event. This would make the above definition a troublesome candidate for capturing (deterministic) polynomial-time knowledge even in the unlikely event that the first two problems could be overcome.

Our intention is to develop concepts that will be useful for the design and analysis of distributed protocols and plans. In such circumstances knowledge is a precondition for various actions and the result of others. So that whereas the agents' actions depend on whether they know certain relevant facts, the agents themselves do not necessarily reason in a logical language or explicitly manipulate logical formulas. Rather, they determine what actions to perform based on various computations they perform. Their resource bounds restrict the complexity of the computations they can perform. Intuitively, the basic idea behind the definition we will propose is that a resource-bounded agent knows a particular fact at a given point only if it can actually compute that the fact holds. But we have to be careful in making this intuition precise. Consider the case in which a fact φ holds at a given point. There are clearly many algorithms the agent can run at that point that will accept (say "yes"). Which of them can be said to compute that φ holds at that point? Clearly we need to consider the algorithm's behavior at other points as well. Intuitively, we will say that an algorithm computes the truth of φ if it accepts whenever φ holds and rejects when φ does not hold. The agent can *actually* compute the truth of φ only if this computation does not require resources beyond the agent's capabilities. What resources the agent has (or can afford for a given task) can vary from one context to another, and we will thus consider the complexity of performing computations as a function of parameters that may depend on the context (e.g., the number of agents involved, number or size of messages accepted, number of failures that have occured, etc.). Roughly speaking, we will say that an agent has resource-bounded knowledge of φ if there is an algorithm that accepts at the current point and, when executed by the agent at any point whatsoever, will always run within the resource bounds and compute the truth of φ. We will make this more precise in Section 3.

Defined this way, resource-bounded knowledge has a number of interesting properties. For one, it does not suffer from logical omniscience. It is perfectly consistent (and very common) for an agent to know a fact φ, know that $\varphi \supset \psi$, and *not* know ψ. More importantly, this notion allows us to capture interesting subtle situations. For example, if we assume that factoring cannot be performed in polynomial time, then a polynomial-time processor that receives two large primes, multiplies them to get a number N, and then erases the primes, knows the value of N but does not know the factors of N.

This abstract is organized as follows. The next section sketches the model of computation we will use. Section 3 defines resource-bounded knowledge and discusses some of its properties. Sections 4 and 5 prove two nontrivial theorems that provide additional evidence of the appropriateness of our definitions. The first, in Section 4, shows the relationship between resource-bounded knowledge and the implementability of knowledge-based protocols within given resource bounds. Roughly speaking, we show that once we replace all tests for knowledge in a knowledge-based protocol by tests for resource-bounded knowledge we obtain a protocol that is equivalent to the original one if and only if the original protocol can be implemented by agents restricted to the given resource bounds. In Section 5, we define resource-bounded belief, common knowledge, and common belief, and use these notions to prove a new impossibility result in fault-tolerant computing. More specifically, we show that no polynomial-time protocol for simultaneous Byzantine agreement can stop as early as any other polynomial-time protocol, in all runs of the generalized omissions failure model. This impossibility result can, in hindsight, be given a direct combinatorial proof. However, the fact that it was provable (and first proved) using our definitions is a strong statement in favor of our definitions. Section 6 provides some concluding remarks.

2 About the model

We now sketch the essential properties of our model of computation. For simplicity, we will restrict attention to synchronous systems here, although our treatment does not depend on synchrony. Our model is closely related to that of [MT].

We consider distributed systems consisting of a finite collection of processors (or agents) denoted by $i, j, \ldots$. A *run* r of such a system is an infinite sequence of global states. In each global state, every processor is in a unique *local state*. A *protocol* $\mathcal{P}$ specifies what actions each processor should take (local actions and communication-related actions) at any given point, as a function of its local state. In every global state each processor first computes what actions to perform, and then performs the actions. The next global state is the result of all actions taken in the current global state. The state of the run r when it is in its k^{th} global state is usually denoted by the pair (r, k); such pairs are called *points*. We will also denote points by c, c', etc. A *system* is identified with a set R of runs. Intuitively, the points in the runs of a system correspond to the "possible worlds" the system may be in.

3 Resource-bounded knowledge

Before presenting our definitions for resource-bounded knowledge, let us briefly review the "standard" definitions of (information-based) knowledge in a system (cf. [CM,FI,HM,PR,RK]). Within a given system, a *fact* φ is identified with a subset φ^M of the points of the system (intuitively, the points of which this fact is true). Formally, we say that a point c *satisfies* a fact φ, denoted $c \models \varphi$, if $c \in \varphi^M$. It is standard to start out with some basic set Φ of *ground* formulas, which are specified explicitly as particular facts about the system (subsets of the points), and extend them to a logical language with boolean connectives and (possibly) temporal or other modal operators (see [HM]). A formula φ is said to be ***valid*** with respect to a given system, denoted $\models \varphi$, if it holds at all points of the system. We close the language under modal operators K_i for all agents i. The formula $K_i\varphi$ should be read ***agent i (information-theoretically) knows*** φ. Information-theoretic knowledge in a given system is defined as follows: $c \models K_i\varphi$ exactly if $c' \models \varphi$ for all points c' of the system in which agent i has the same local state as it does in c. Under such a definition, knowledge has the properties of the modal system S5 (cf. [HM2]), which in particular means that it satisfies the ***knowledge axiom*** $K_i\varphi \supset \varphi$, and the ***distribution axiom*** $(K_i\varphi \wedge K_i(\varphi \supset \psi)) \supset K_i\psi$.

We are now in a position to define resource-bounded knowledge in a given system along the lines suggested in the introduction. We start out with a list of complexity parameters $\bar{y}$ that is defined at all points of the system, whose explicit values may vary from one point to another. These parameters may include, for example, the number of processors a given processor is interacting with, the number of messages it received, the amount of free memory it has, or any other parameters relevant to the particular application. We assume that the computations the processors can perform at a given local state can consume resources bounded by some function $\beta(\bar{y})$ from a complexity class $\mathcal{B}(\bar{y})$. These bounds may be on the time used in the computation, the amount of memory, whether the computations are deterministic, probabilistic, etc. A protocol is said to be ***$\mathcal{B}$-bounded*** with respect to a given system, if there is a function $\beta \in \mathcal{B}$ such that at every point c in the system the processors use resources bounded by $\beta(\bar{y}(c))$. Processors perform computations based exclusively on their local state. Intuitively, the only kind of tests that a $\mathcal{B}$-bounded processor can base its actions on are $\mathcal{B}$-bounded computations. Using K_i to denote information-based knowledge defined above, there is a natural sense in which we can consider a processor to be able to compute whether it knows a particular fact: Let A be a decision algorithm (i.e., one whose output is either *accept* or *reject*); then we say that ***A computes the truth of $K_i\varphi$*** if, for all points in the system, A's computation starting from i's local state at the point c accepts if $c \models K_i\varphi$, and rejects otherwise.[1] A fact $K_i\varphi$ is said to be ***$\mathcal{B}$-computable*** if it is computable using an algorithm A which consumes resources bounded by $\mathcal{B}$. We now define resource-bounded knowledge, denoted by $K_i^{\mathcal{B}}$, as follows:

$$c \models K_i^{\mathcal{B}}\varphi \quad \text{iff} \quad \begin{array}{l}\text{(a) } c \models K_i\varphi, \text{ and}\\ \text{(b) } K_i\varphi \text{ is } \mathcal{B}\text{-computable.}\end{array}$$

[1] A probabilistic algorithm executed at a given local state might accept in some executions and reject in others. In such a context, we should require the algorthm to accept on sufficiently many (e.g., with probability at least 2/3) if $c \models K_i\varphi$, and reject sufficiently often if $c \not\models K_i\varphi$.

We restrict attention to the computability of the form $K_i\varphi$ because any fact that agent i could possibly compute the truth of must be equivalent to a fact of this form. This follows from the fact that the algorithm A computes based solely on agent i's local state. It is interesting to note that our definition is in a form consistent with that of the logic of belief and awareness of [FH]. Using their terminology, the agent i in our definition is considered to be *aware* of φ if $K_i\varphi$ is $\mathcal{B}$-computable.

Thus, for example, a processor polynomial-time knows a given fact if the processor can test in polynomial time whether it (information-theoretically) knows the fact, and in this particular case the test would succeed (i.e., the processor *does* know the fact). Notice that we do not assume that the processor's protocol actually makes use of (or has explicit access to) the particular algorithm in question. What is important is that a related resource-bounded protocol *could* make use of this algorithm, and base its actions on whether the fact is known.

Let us now consider some of the properties of resource-bounded knowledge. First of all, $K_i^{\mathcal{B}}$ clearly satisfies the knowledge axiom $K_i^{\mathcal{B}}\varphi \supset \varphi$, since both $K_i^{\mathcal{B}}\varphi \supset K_i\varphi$ and $K_i\varphi \supset \varphi$ are valid. Furthermore, $K_i^{\mathcal{B}}$ satisfies the generalization rule: If $\models \varphi$ then $\models K_i^{\mathcal{B}}\varphi$. This means that all tautologies are resource-boundedly known. The reason is that all valid formulas are equivalent to (and hence have the same extension as) the fact **true**, and there is a trivial (constant time) algorithm that computes the truth of the tautology $K_i(\mathbf{true})$. Still, it can be argued, if an agent receives a graph from another agent, is it reasonable to say that it polynomial-time knows whether this graph is Hamiltonian? Notice that whereas, for a fixed G, the fact "*G is Hamiltonian*" is either a tautology or the negation of one, the fact "*the graph just received is Hamiltonian*" will in general be neither a tautology nor the negation of one (if different kinds of graphs can be received). So, whereas the agent will know that "*the graph just received is G*", and (say) that "*G is Hamiltonian*", it would only know the (potentially crucial) fact "*the graph just received is Hamiltonian*" if it can compute within its limited resources whether graphs it receives are Hamiltonian. Clearly, a protocol that should always act based on whether a fixed predetermined tautology is true (or false) can have the answer hard wired, and in any case should not compute its truth repeatedly in each invocation. The type of facts we will be most interested in will generally depend on the particular context and will be neither tautologies nor their negations.

We now address the question of logical omniscience. As expected, resource-bounded knowledge is, in general, *not* closed under deduction. This means that

$$\not\models (K_i^{\mathcal{B}}\varphi \wedge K_i^{\mathcal{B}}(\varphi \supset \psi)) \supset K_i^{\mathcal{B}}\psi.$$

For example, consider a system in which processor's local state (in all of the points in the system) encodes a graph. Furthermore, assume that all possible graphs can appear in the initial local state of a processor. Taking the number n of nodes in the graph encoded in a processor's state as the complexity parameter, imagine that the processors are restricted to polynomial-time computations in n. Let φ be the fact "*the graph is a ring*". Since checking whether a graph is a ring can easily be done in linear time, there is an obvious algorithm that detects whether a processor knows φ at any given point. Thus, in particular we have that $K\varphi$ is equivalent to $K^{\mathcal{B}}\varphi$. Whenever the graph is a ring the processor polynomial-time knows that it is a ring. Now consider the fact $\psi =$"*the graph is Hamiltonian*". It is easy to see that $\varphi \supset \psi$ is valid, since every ring is in particular a Hamiltonian graph. By the generalization rule, $K^{\mathcal{B}}(\varphi \supset \psi)$ is also valid. Recall that we assume that all possible graphs on n nodes may appear. Assuming P$\neq$NP, there is no polynomial-time algorithm that checks whether an arbitrary graph is Hamiltonian. Therefore, $K^{\mathcal{B}}\psi$ ***never*** holds. It follows that at points in which the graph is a ring both $K^{\mathcal{B}}\varphi$ and $K^{\mathcal{B}}(\varphi \supset \psi)$ hold, but $K^{\mathcal{B}}\psi$ does not hold. In a similar fashion, whenever the graph is Hamiltonian there is a particular Hamiltonian path in the graph which is and should obviously be polynomial-time known to be in the graph, while the graph is not known to be Hamiltonian. The algorithm checking for a particular fixed path can not be of great use in determining Hamiltonicity in general. Aside from showing that the distribution axiom fails, this example shows that our definition correctly handles the distinction between P and NP.

While it does not satisfy the distribution axiom, if $\mathcal{B}$ is closed under addition then resource-bounded knowledge does satisfy a slightly weaker property which we call the ***weak distribution axiom***:

$$\models (K_i^{\mathcal{B}}\varphi \wedge K_i^{\mathcal{B}}(\varphi \supset \psi)) \supset K_i^{\mathcal{B}}(\varphi \wedge \psi).$$

Thus, our disucussion above implies that when a processor's graph is a ring the processor ***does*** polynomial-time know that its graph is a Hamiltonian ring. The formulas φ and ψ from the above example also show that resource-bounded knowledge does not distribute over conjunction:

$$\not\models K_i^{\mathcal{B}}(\varphi \wedge \psi) \supset K_i^{\mathcal{B}}\psi.$$

However, if $\mathcal{B}$ is closed under addition, the converse does hold:

$$\models K_i^{\mathcal{B}}\varphi \wedge K_i^{\mathcal{B}}\psi \supset K_i^{\mathcal{B}}(\varphi \wedge \psi).$$

The $\mathcal{B}$-bounded algorithm that computes the truth of $K_i(\varphi \wedge \psi)$ is simply the conjunction of the algorithms for $K_i\varphi$ and $K_i\psi$.

Resource-bounded knowledge thus does not suffer from the logical omniscience property. A processor's knowledge is, in a precise sense, restricted by the processor's being limited to $\mathcal{B}$-bounded computations. Slightly extending the discussion above, we can show the following:

Proposition 1: $K_i^{\mathcal{B}}$ has the properties of S5, with the weak distribution axiom replacing the distribution axiom.

While resource-bounded knowledge was defined above with respect to a given system, we are often in situations in which we design a protocol that should work within particular resource bounds in a number of systems. A common case is in the design of parameterized protocols. A *parameterized protocol* is a function $\mathcal{P}(\cdot) : \bar{x} \mapsto \mathcal{P}(\bar{x})$, mapping parameter lists to fixed protocols. Intuitively, $\mathcal{P}(\bar{x})$ is the instantiation of $\mathcal{P}(\cdot)$ on parameter list $\bar{x}$. The list $\bar{x}$ will in general consist of uninstantiated variables in the protocol text, and possibly some initial common inputs to the system. For example, protocols for Byzantine agreement are usually parameterized by the number of processors n and a bound of t on the number of failures to be tolerated. The system parameters could, in other contexts, include a number N about whose properties the agents interact. Let $R(\bar{x})$ denote the system corresponding to all runs of $\mathcal{P}(\bar{x})$ and let $\mathcal{X}$ denote a set of parameter lists. We define the *class of systems defined by $\mathcal{P}$ and $\mathcal{X}$* to be $\{R(\bar{x}) : \bar{x} \in \mathcal{X}\}$. Our definition of resource-bounded knowledge immediately extends to classes of systems as well. Facts need to be given meaning in all the systems in the class, and the algorithms that compute the truth of a fact need to act correctly in all systems of the class. The complexity parameters $\bar{y}$ will often involve some of the parameters $\bar{x}$ that define the system. For example, a natural requirement of a protocol for Byzantine agreement is that it perform computations that are polynomial in n and t with respect to a class of systems. The design of protocols for simultaneous choice problems in [MT] and in Section 5 involves reasoning about classes of systems.

4 Knowledge-based protocols

Knowledge-based protocols were first defined by Halpern and Fagin in [HF]. Roughly speaking, a *knowledge-based protocol* is a protocol in which processors' actions explicitly depend on their knowledge. Such a protocol contains commands of the form `if` $K\varphi$ `then do` S, or `if` $\neg K\psi$ `then do` S'. We leave a more formal definition of knowledge-based protocols to the full paper. Specifically, we will think of such a protocol as a program text in which some of the tests depend on the processors' knowledge. One way of arguing for the appropriateness of our definitions of resource-bounded knowledge is by relating them to the implementability of a knowledge-based protocol within particular resource bounds. Roughly speaking, it would be pleasing to be able to show that a knowledge-based protocol is implementable within resource bounds $\mathcal{B}$ if and only if it is equivalent to the *resource-bounded knowledge-based protocol* we get by replacing all formulas of the form $K\varphi$ in the protocol by $K^{\mathcal{B}}\varphi$. By *equivalent* here we mean that there is a one to one and onto mapping between the set of runs corresponding to the original protocol and the set corresponding to the resource-bounded one, such that in corresponding runs exactly the same messages are sent and the same internal actions are performed at the same times. Thus, intuitively, the tests for knowledge in the original protocol and the corresponding tests for resource-bounded knowledge are interchangeable. As we shall soon see, a slight variant of this statement holds.

Consider the situation in the rings vs. Hamiltonian graphs in the previous section. Again, we take $\varphi =$"*the graph is a ring*", $\psi =$"*the graph is Hamiltonian*", and $\mathcal{B}(n)$ is polynomial time. Imagine a knowledge-based protocol that reads as follows:

```
if $\neg K\varphi$ then do $S$
      else if $K\psi$ then do $S'$
```

This protocol is easily implementable in polynomial time, since testing the first condition $\neg K\varphi$ (which holds exactly if the graph is not a ring) can be done in linear time, and when the second test, for $K\psi$, is actually performed, it is guaranteed to hold. The reason, of course, is that the second test is reached only in cases in which the graph is a ring, which in particular implies that it is Hamiltonian, and hence that $K\psi$ holds. However, as we have seen in the previous section, $K^{\mathcal{B}}\psi$ never holds, since there is no polynomial-time algorithm to test whether the graph is Hamiltonian. So, strictly speaking, the theorem we had in mind does not hold! But it fails for a good reason. The fact that the execution of the protocol has reached the test for $K\psi$ carries quite a bit of nontrivial information. And this information is what makes testing for $K\psi$ trivial in this particular case. This leads us to a slight variant of the original theorem we had in mind, which does indeed hold. Given a knowledge-based protocol $\mathcal{P}$, let the *labeled* version of the protocol $\mathcal{P}^{\ell}$ be the result of labelling each test for knowledge in the protocol by a distinct label ℓ. At some points c the protocol will perform the test labeled by ℓ, and at others it won't reach that test. For every label ℓ' in $\mathcal{P}^{\ell}$, we define the fact "**at** ℓ'" to hold at a given point c, denoted $c \models$ **at** ℓ', exactly if the test labeled ℓ' is reached in the computation the processor performs at c. We now define the resource-bounded version of $\mathcal{P}^{\ell}$, denoted $\mathcal{P}^{\mathcal{B}}$, to be the protocol resulting from replacing every test of the form $K\varphi'$ in the protocol $\mathcal{P}$ by a test for $K^{\mathcal{B}}(\textbf{at } \ell_j \wedge \varphi')$, where ℓ_j is the label of $K\varphi'$ in $\mathcal{P}^{\ell}$. We now have:

Proposition 2: A knowledge-based protocol $\mathcal{P}$ is implementable within resource bounds $\mathcal{B}$ iff $\mathcal{P}$ is equivalent to $\mathcal{P}^{\mathcal{B}}$.

5 Simultaneous Byzantine agreement

The initial motivation for this paper came from work with Mark Tuttle on [MT]. That paper deals with the problem of performing simultaneous choice problems, a large class of problems one of which is simultaneous Byzantine agreement (denoted SBA).[2] Roughly speaking, that paper shows that in a protocol for SBA that is optimal in all runs in the generalized omissions failure model correct processors must perform actions as soon as they (information-theoretically) know that particular relevant facts are common knowledge. Furthermore, testing whether a processor knows that these facts are common knowledge is shown to be NP-hard. Part of the conclusions to that paper reads as follows:

> ... *Since it is unreasonable to expect processors to perform NP-hard computations between consecutive rounds of communication, it is natural to ask what is the earliest time at which such actions can be performed by* resource-bounded *processors ... the information-based definition of knowledge ... is not appropriate for reasoning about such questions. A major challenge motivated by this is the elaboration of the definition of knowledge ... to include notions of resource-bounded knowledge that would provide us with appropriate tools [notions such as* polynomial-time knowledge *and* polynomial-time common knowledge*] for analyzing such questions ...*

Our definitions of resource-bounded knowledge can indeed be used to analyze this question, yielding a proof that there can be no polynomial-time protocols for simultaneous Byzantine agreement that are optimal in all runs with respect to polynomial-time protocols. In this section we will only present a rough sketch of the technical development leading to this result. Complete details will appear in the full paper. The precise definitions of all of the terms we use here can be found in [MT]. Aside from that, the section is self-contained.

SBA requires the nonfaulty processors to perform particular simultaneous actions under certain conditions. Faulty processors' actions are not specified. The situation therefore turns out to be best analyzed by using a notion of (information-based) belief that closely corresponds to information-based knowledge. We define *agent i believes φ*, denoted $B_i\varphi$, as follows:

[2]Briefly, the SBA problem is one in which n processors, at most t of which are faulty, start out with initial values $x_i \in \{0,1\}$. The system is synchronous and the correct processors are required to all decide simultaneously on an identical value v, with the restriction that if all of the x_i's are 0 (resp. 1) then $v = 0$ (resp. 1); cf. [DM]. While we will focus on SBA in this section, essentially everything we do here applies to simultaneous choice problems in general.

$c \models B_i\varphi$ iff $c' \models \varphi$ for all points c' in which agent i both is nonfaulty *and* has the same local state as at c.

Denoting the set of nonfaulty processors by $\mathcal{N}$, the formula $B_i\varphi$ is equivalent to the formula $K_i(i \in \mathcal{N} \supset \varphi)$ (cf. [MT]). Intuitively, an agent's beliefs here are based on the assumption that it (the agent) is nonfaulty (an agent is not guaranteed to know whether it is faulty). This definition has the property that nonfaulty processors' beliefs (and hence actions based on them) are always true, while for faulty processors $B(\mathbf{false})$ is satisfiable(!). We define $B_i^{\mathcal{B}}$, the corresponding notion of resource-bounded belief, in exactly the same way as we did for knowledge in Section 3, except that the algorithm A is now required to correctly detect whether $B_i\varphi$ holds only when executed at points in which agent i is nonfaulty. Again, facts that *nonfaulty* processors resource-bounded believe will be guaranteed to be true.

In order to define common belief and resource-bounded common belief, we first define *every nonfaulty agent believes* (resp. *resource-bounded believes*) φ, denoted $E_{\mathcal{N}}\varphi$ (resp. $E_{\mathcal{N}}^{\mathcal{B}}\varphi$), to be $\bigwedge_{i\in\mathcal{N}} B_i\varphi$ (resp. $\bigwedge_{i\in\mathcal{N}} B_i^{\mathcal{B}}\varphi$). The definitions of common belief and resource-bounded common belief are now given in terms of fixed points along the lines of [HM]. Common belief, denoted $C_{\mathcal{N}}\varphi$ is defined to be $\nu X.E_{\mathcal{N}}(\varphi \wedge X)$, where ν stands for the greatest fixed point operator. This notion of common belief coincides with the notion of common knowledge with respect to the nonfaulty agents defined in [MT]. Whereas the greatest fixed point here is well-defined, its resource-bounded analogue $\nu X.E_{\mathcal{N}}^{\mathcal{B}}(\varphi \wedge X)$ — is not well-defined. The greatest fixed point is no longer guaranteed to always exist. (We can show examples where it doesn't.) We thus define:

$$C_{\mathcal{N}}^{\mathcal{B}}\varphi = \begin{cases} \nu X.E_{\mathcal{N}}^{\mathcal{B}}(\varphi \wedge X) & \text{if it exists,} \\ \mathbf{false} & \text{otherwise.} \end{cases}$$

Three crucial properties of common belief and resource-bounded common belief are that $\models C_{\mathcal{N}}^{\mathcal{B}}\varphi \supset C_{\mathcal{N}}\varphi$ and the *fixed point axioms*: $\models C_{\mathcal{N}}\varphi \equiv E_{\mathcal{N}}(\varphi \wedge C_{\mathcal{N}}\varphi)$, and similarly $\models C_{\mathcal{N}}^{\mathcal{B}}\varphi \equiv E_{\mathcal{N}}^{\mathcal{B}}(\varphi \wedge C_{\mathcal{N}}^{\mathcal{B}}\varphi)$. We discuss additional properties of resource-bounded common knowledge (which is defined in the same manner) and resource-bounded common belief in the full paper. The following proposition relates simultaneous actions and resource-bounded common belief:

Proposition 3: Let $\mathcal{C}$ be a class of systems of $\mathcal{B}$-bounded processors, and let a be a simultaneous action for $\mathcal{N}$ in $\mathcal{C}$. Then $c \models$ "a is being performed by $\mathcal{N}$" holds iff $c \models C_{\mathcal{N}}^{\mathcal{B}}$("$a$ is being performed by $\mathcal{N}$").

Proposition 3 shows that resource-bounded common belief is a necessary condition for performing simultaneous actions when the agents are resource bounded. This is analogous to the results in [DM] and [MT] that show that common belief is a necessary condition for similar actions performed by arbitrary (not necessarily resource-bounded) agents. We denote the fact that a simultaneous action a is allowed according to the problem specification by $\mathit{enabled}(a)$. In SBA, the simultaneous actions are deciding 0 and deciding 1, and $\mathit{enabled}(\text{decide } v)$ is equivalent to the existence of at least one initial value $x_i = v$. The distribution axiom for common belief gives as a corollary that when an action a is performed it is common belief that $\mathit{enabled}(a)$ holds. However, like resource-bounded knowledge and belief, resource-bounded common belief does not satisfy the distribution axiom. Thus, a similar corollary does not hold in the resource-bounded case. But a weaker version of this corollary *does* hold and will be instrumental for our impossibility result. Before we can state it we need to make a few definitions. In [DM] and [MT] the *full information protocol* $\mathcal{F}$ is shown to play an instrumental role in designing optimal protocols. This is a protocol in which every processor sends a complete description of its local state to all others in every round. In the generalized omissions failure model this description can be succinctly represented, as shown in [MT]. Finally, two runs of different protocols for a simultaneous choice problem are said to be *corresponding* runs if the input to the system and the pattern of failures is the same in both (i.e., the adversary's behavior is the same in both). Setting $\mathcal{B}$ to be polynomial time in n and t, we can now show:

Theorem 4: Let $C_{\mathcal{F}}$ be the class of systems of the runs of $\mathcal{F}$ in the generalized omissions model; Then $c \models C_{\mathcal{N}}^{\mathcal{B}}\psi$ for some ψ satisfying $\models \psi \supset \mathit{enabled}(\text{decide } v)$ iff there exists a polynomial-time protocol for SBA that decides v at time k in the run corresponding to r.

A protocol is said to be *polynomial-time optimal (in all runs)* if in all runs it halts as soon as any other polynomial-time protocol does in a corresponding run. As a result of the above theorem, we now have:

Corollary 5: Assuming P$\neq$NP, there is no polynomial-time optimal protocol for SBA in the generalized omissions model.

The proof of this fact applies Theorem 4 to the construction in the NP-hardness lower bound proof of [MT]. In hindsight, the proof of Corollary 5 follows from that lower bound proof via combinatorial means that can be explained without needing the full terminology of Theorem 4. However, the fact that it was provable (and in fact first proved) using this theorem and the related concepts developed in this paper is a strong statement in favor of our definitions.

6 Conclusions

This paper presents definitions of resource-bounded knowledge for distributed systems applications, and argues that they capture essential aspects of resource-bounded distributed computing. While our examples in Section 3 pointed out subtle properties of the definitions, the results in the last two sections showed that these definitions can be successfully applied to nontrivial problems in distributed computing. A natural question at this point is to what extent our definitions truly capture the notion of resource-bounded knowledge. Will there be cases in which these definitions will be insufficient for the analysis of resource-bounded distributed protocols? We feel that our definitions are useful, well motivated and robust. Nevertheless, generalizations of our notions and additional notions will be necessary for certain applications. One contribution of our work is in providing a useful framework for defining such notions.

Joe Halpern points out that since we require an algorithm computing the truth of a formula to be correct at all possible points, it might be difficult or somewhat unintuitive to model in our framework a situation in which an agent knows a certain fact, such as that the graph it received is Hamiltonian, once it is given a proof of this fact (e.g., shown a Hamiltonian path), and does not know this before. Given our definitions, the agent in this scenario would never polynomially know that *"the graph is Hamiltonian"*. However, once it is given the proof of Hamiltonicity, the agent *would* know that *"the graph is Hamiltonian* **and** *that message contains a proof of it"*, which, of course, implies the former. A somewhat intuitionistic aspect of our definition: A fact that is hard to verify within the given resource bounds can become known only in conjunction with its proof. To what extent this resolves the original problem in a satisfactory way is a matter of taste. We see no cleaner way out.

Another question is how our notion of resource-bounded knowledge is related to analyzing notions such as the recent *interactive proofs* and *zero knowledge* introduced by Goldwasser, Micali, and Rackoff in [GMR]. We have recently made progress on this problem in joint work with Joe Halpern and Mark Tuttle (see [HMT]). In that context we need to extend the notion of resource-bounded knowledge to that of resource-bounded knowledge *with respect to a promise* Ψ. The essential change in the definition is that the resource-bounded algorithm A computing the truth of $K_i^p\varphi$ is required to act correctly only when the promise Ψ holds. A may act arbitrarily when Ψ does not hold. The analysis in [HMT] also makes use of a notion of probabilistic knowldge along the lines of [FH2], and defines a notion of knowing *how* to perform various tasks efficiently. Much is still left to be done.

Acknowledgements

I'd especially like to thank Mark Tuttle for many insightful remarks on this work and a thorough reading of early drafts. Joe Halpern's comments motivated significant improvements in the exposition. I'd also like to thank Martin Abadi, Michael Ben-Or, Oded Goldreich, Joe Halpern, Yael Neumann, Adi Shamir, and Yoav Shoham for useful discussions on the topic of this paper.

Bibliography

[CM] K. M. Chandy and J. Misra, How processes learn, *Distributed Computing* **1**:1, 1986, pp. 40-52.

[C] M. J. Cresswell, *Logics and Languages*, Methuen and Co. 1973.

[DM] C. Dwork and Y. Moses, Knowledge and common knowledge in a Byzantine environment: The case of crash failures, *Theoretical Aspects of Reasoning About Knowledge: proceedings of the 1986 conference*, Monterey, 1986, J.Y. Halpern ed., Morgan Kaufmann, pp. 149-170. To appear, *Information and Computation*, 1988.

[E] R. A. Eberle, A logic of believing, knowing, and inferring, *Synthese* **26**, 1974, pp. 356-382.

[FH] R. Fagin and J. Y. Halpern, Belief, awareness, and limited reasoning, *Proceedings of the Ninth International Joint Conference on Artificial Intelligence*, 1985, pp. 491-501.

[FH2] R. Fagin, and J. Y. Halpern, Reasoning about knowledge and probability: Preliminary report, these proceedings.

[FI] M. J. Fischer and N. Immerman, Foundations of knowledge for distributed systems, *Theoretical Aspects of Reasoning About Knowledge: proceedings of the 1986 conference*, Monterey, 1986, J.Y. Halpern ed., Morgan Kaufmann, pp. 171–185.

[GMR] S. Goldwasser, S. Micali, and C. Rackoff, The knowledge complexity of interactive proof-systems, *Proceedings of the 17th Symposium on the Theory of Computing*, 1985, pp. 291–304.

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

[HM] J. Y. Halpern and Y. Moses, Knowledge and common knowledge in a distributed environment, Version of August 1987 is available as IBM RJ 4421 (second revision).

[HM2] J. Y. Halpern and Y. Moses, A guide to the modal logic of knowledge and belief, *Proceedings of the Ninth International Joint Conference on Artificial Intelligence*, 1985, pp. 480–490.

[HMT] J. Y. Halpern, Y. Moses, and M. R. Tuttle, A Knowledge-based Analysis of Zero Knowledge, to appear in *Proceedings of the Twentieth Annual ACM Symposium on Theory of Computing*, 1988.

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

[Kon] K. Konolige, Belief and incompleteness, *SRI Artificial Intelligence Note 319,* SRI International, Menlo Park, California, 1984.

[MH] B. Moore and G. Hendrix, Computational models of of beliefs and the semantics of belief sentences, SRI International technical Note 187, 1979.

[MT] Y. Moses and M. R. Tuttle, Programming simultaneous actions using common knowledge, *Proceedings of the 27th Annual Symposium on Foundations of Computer Science*, 1986, pp. 208–221. To appear, *Algorithmica* 1987/8.

[PR] R. Parikh and R. Ramanujam, Distributed processes and the logic of knowledge (preliminary report), *Proceedings of the Workshop on Logics of Programs*, 1985, pp. 256–268.

[Ra] V. Rantala, Impossible worlds semantics and logical consequence, *Acta Philosophica Fennica* **35**, 1982, pp. 106-115.

[RB] N. Rescher and R. Brandom, *The logic of inconsistency*, Rowman and Littlefield 1979.

[RK] S. J. Rosenschein and L. P. Kaelbling, The synthesis of digital machines with provable epistemic properties, *Theoretical Aspects of Reasoning About Knowledge: proceedings of the 1986 conference*, Monterey, 1986, J.Y. Halpern ed., Morgan Kaufmann, pp. 83-98.