Verifying time, memory and communication bounds in systems of reasoning agents*

Natasha Alechina, Brian Logan, Nguyen Hoang Nga and Abdur Rakib
School of Computer science, University of Nottingham
Nottingham NG8 1BB, UK
{nza,bsl,hnn,rza}@Cs.Nott.AC.UK

Abstract

We present a framework for verifying systems composed of heterogeneous reasoning agents, in which each agent may have differing knowledge and inferential capabilities, and where the resources each agent is prepared to commit to a goal (time, memory and communication bandwidth) are bounded. The framework allows us to investigate, for example, whether a goal can be achieved if a particular agent, perhaps possessing key information or inferential capabilities, is unable (or unwilling) to contribute more than a given portion of its available computational resources or bandwidth to the problem.

1. Introduction

A distributed approach to problem solving involves the collective effort of multiple agents combine their knowledge and information to solve problems which no single agent could solve alone or to solve problems more effectively. For a given problem, for different multi agent systems, different solution strategies will be preferred depending on the relative costs of computational and communication resources for each agent. These tradeoffs may be different for different agents (e.g., reflecting their computational capabilities or network connection) and may reflect the agent's commitment to a particular problem. For a given set of agents with specified inferential abilities and resource bounds it may not be clear whether a particular problem can be solved at all, or, if it can, what computational and communication resources must be devoted to its solution by each agent.

There has been considerable work in the agent literature on distributed problem solving in general e.g., [15, 20, 22] and on distributed reasoning in particular [1, 8]. Much of this work analyses the time and communication complexity of distributed reasoning algorithms. In this paper we present a framework for reasoning about tradeoffs between time, memory and communication in systems of distributed reasoning agents. In contrast to previous work, e.g., [3] which focused primarily on memory limitations of single reasoners, our approach allows us to specify bounds on the number of messages the agents can exchange, allowing the investigation of tradeoffs between different resources. We introduce a novel epistemic logic, $BMCL$, for specifying resource-bounded reasoners. Critically, the logic allows upper bounds on the resource commitments (time, memory and communication) of each agent in the system to be specified. The logic is sound and complete and admits efficient model-checking. Using simple resolution examples, we show how to encode systems of distributed reasoning agents specified in the logic in a model checker, and verify some example properties.

2 Distributed Reasoners

We define the 'shape' of a proof in terms of the maximum space requirement at any step in the proof and the number of inference steps it contains. The lower bound on space for a given problem is then the least maximum space requirement of any proof, and the lower bound on time is the least number of inference steps of any proof. In general, a minimum space proof and a minimum time proof will be different (have different shapes). Bounding the space available for a proof will typically increase the number of inference steps required and bounding the number of steps will increase the space required.

We define the bounds on a reasoning agent in terms of its available resources expressed in terms of memory, time and communication. We assume that the memory required for a particular proof can be taken to be its space requirement (e.g., the number of formulas that must be simultaneously held in memory) times some constant. For a single threaded

*An extended version of this paper will be presented at AAMAS 2008, Estoril, Portugal.

agent, the number of inference steps executed times some constant can be taken as a measure of the time necessary to solve the problem. The communication requirement of a proof is taken to be the number of messages exchanged with other agents. In what follows, we ignore the constants and assume that the units of problem size and resources are the same.

In the distributed setting we distinguish between *symmetric problem distributions*, where all agents have the same premises and the same rules of inference, and *asymmetric problem distributions* where different premises may be assigned to different agents and/or the agents use different rules of inference. Similarly, we can distinguish between *symmetric resource distributions* (when all agents have the same resource bounds) and *asymmetric resource distributions*, (when different agents have different resource bounds).

Distribution does not necessarily change the shape (maximum space requirement and number of inference steps) of a proof. However, in a distributed setting the tradeoffs between memory and time bounds is complicated by communication. Unlike memory and time, communication has no direct counterpart in the proof. However like memory, communication can be substituted for time (e.g., if part of the proof is carried out by another agent), and, like time, it can be substituted for memory (e.g., if a lemma is communicated by another agent rather than having to be remembered). In the distributed setting, each agent has a minimum memory bound which is determined by its inference rules and which may be smaller than the minimum space requirement for the problem. If the memory bound for all agents taken individually is less than the minimum space requirement for the problem, then the communication bound must be greater than zero.

In the next section, we present measures of space, time and communication for distributed reasoning agents which allow us to make these tradeoffs precise.

3 Measuring Resources

We assume a set of n agents where each agent i has a set of propositional inference rules R_i (for example, R_i could contain conjunction introduction and modus ponens, or it could contain just a single rule of resolution) and a set of premises or a knowledge base K_i. The notion of a derivation, or a proof of a formula G from K_i is standard. We view the process of producing a proof of G as a sequence of configurations or states of a reasoner, starting from an empty configuration, and producing the next configuration by one of four operations: **Read** copies a formula from K_i into the current configuration; **Infer** applies a rule from R_i to formulas in the current configuration; **Skip** leaves the configuration unchanged; and **Copy** copies a formula α into the next configuration of agent j if α is in the current configuration of agent $i, j \neq i$. Note that **Read**, **Infer** and **Copy** may overwrite a formula from the previous configuration. The goal formula is derived if it occurs in the configuration of one of the agents.

We take the time complexity of a derivation to be the length of the sequence of configurations. Space complexity is taken to be the size of configurations as in [6].[1] The size of a configuration can be measured either in terms of the maximal number of formulas appearing in any configuration or in terms of the number of symbols required to represent a configuration. Clearly, for some inference systems, for example, where the set of inference rules contains both conjunction introduction and conjunction elimination, the first size measure results in constant space usage. However, for other systems, such as resolution, counting formulas results in non-trivial space complexity [13]. In this paper, we take the size of a configuration to be the maximal number of formulas, since all the reasoning systems we consider have a non-trivial space complexity for this measure.

#	Configuration	Operation
1	$\{\ \}$	
2	$\{A_1\}$	**Read**
3	$\{A_1, A_2\}$	**Read**
4	$\{A_1, A_1 \wedge A_2\}$	**Infer**
5	$\{A_1 \wedge A_2, A_1 \wedge A_2 \rightarrow B_1\}$	**Read**
6	$\{A_1 \wedge A_2, B_1\}$	**Infer**

Figure 1. Example derivation using $\bigwedge_I$ and MP

As an illustration, Figure 1 shows the space and time complexity of the derivation of the formula B_1 from $A_1,\ A_2,\ A_1 \wedge A_2 \rightarrow B_1$ in an inference system which contains only conjunction introduction and modus ponens. The length of the proof is 6 and the space usage is 2 (at most 2 formulas need to be present in the configuration at any given time). It is worth observing that the inference system consisting of just conjunction introduction and modus ponens does not have constant space complexity when space is measured as the number of formulas; a sequence of derivation examples requiring (logarithmically) growing space can easily be constructed starting from the example above, and continuing with a derivation of C_1 from $A_1, A_2, A_3, A_4, A_1 \wedge A_2 \rightarrow B_1, A_3 \wedge A_4 \rightarrow B_2, B_1 \wedge B_2 \rightarrow C_1$, etc.

Most research in time and space complexity of proofs

[1] We deviate from [6] in that we do not have an explicit *erase* operation, preferring to incorporate erasing (overwriting) in the *read* and *infer* operations. This obviously results in shorter proofs; however including an explicit erase operation gives proofs which are no more than twice as long as our proofs if we don't require the last configuration to contain *only* the goal formula.

#	Configuration	Operation
1	$\{\ \}$	
2	$\{A_1 \vee A_2\}$	**Read**
3	$\{A_1 \vee A_2, \neg A_1 \vee A_2\}$	**Read**
4	$\{A_1 \vee A_2, A_2\}$	**Infer**
5	$\{A_2, A_1 \vee \neg A_2\}$	**Read**
6	$\{A_2, A_1 \vee \neg A_2, \neg A_1 \vee \neg A_2\}$	**Read**
7	$\{A_2, \neg A_2, \neg A_1 \vee \neg A_2\}$	**Infer**
8	$\{\emptyset, \neg A_2, \neg A_1 \vee \neg A_2\}$	**Infer**

Figure 2. Example derivation using resolution

	Agent 1		Agent 2	
#	Configuration	Op.	Configuration	Op.
1	$\{\}$		$\{\}$	
2	$\{A_1 \vee A_2\}$	**Read**	$\{A_1 \vee \neg A_2\}$	**Read**
3	$\{A_1 \vee A_2, \neg A_1 \vee A_2\}$	**Read**	$\{\neg A_1 \vee \neg A_2, A_1 \vee \neg A_2\}$	**Read**
4	$\{A_1 \vee A_2, A_2\}$	**Infer**	$\{\neg A_2, A_1 \vee \neg A_2\}$	**Infer**
5	$\{A_1 \vee \neg A_2, A_2\}$	**Read**	$\{\neg A_2, A_2\}$	**Copy**
6	$\{A_1, A_2\}$	**Infer**	$\{\{\}, A_2\}$	**Infer**

Figure 3. Example derivation using resolution with two agents

has focused on the lower bounds for the inference system as a whole. While we are interested in the lower bounds, we are also interested in the trade-offs between time and space usage for particular derivations. For example, consider a set of premises $A_1, A_2, A_3, A_4, A_1 \wedge A_2 \rightarrow B_1, A_3 \wedge A_4 \rightarrow B_2$, $B_1 \wedge B_2 \rightarrow C_1$ and a goal formula $A_1 \wedge A_2 \wedge C$. It is possible to derive the goal from the premises using conjunction introduction and modus ponens and configurations of size 3 in 17 steps (deriving $A_1 \wedge A_2$ twice). On the other hand, with configurations of size 4 the proof is 3 steps shorter.

Different inference systems have different complexity and different tradeoffs. Figure 2 illustrates the (non-trivial) space complexity of resolution proofs in terms of the number of formulas in a configuration. The example, which is due to [13], shows the derivation of an empty clause by resolution from the set of all possible clauses of the form

$$\sim A_1 \vee \sim A_2 \vee \ldots \vee \sim A_n$$

(where $\sim A_i$ is either A_i or $\neg A_i$), for $n = 2$. Its space usage is 3 and the length of the proof is 8.

In the multiagent case, when several reasoners can communicate to derive a common goal, an additional resource of interest is how many messages the reasoners must exchange in order to derive the goal. In the distributed setting, we assume that each agent has its own set of premises and inference rules and its own configuration, and that the reasoning of the agents proceeds in lock step.

The goal formula is derived if it occurs in the configuration of one of the agents. Our model of communication complexity is based on [25], except that we count the number of formulas exchanged by the agents rather than the number of bits exchanged. The communication complexity of a joint derivation is then the (total) number of **Copy** operations in the derivation.

In general, in a distributed setting, trade-offs are possible between the number of messages exchanged and the space (size of a single agent's configuration) and time required for a derivation. The total space use (the total number of formulas in all agent's configurations) clearly cannot be less than the minimal configuration size required by a single reasoner to derive the goal formula from the union of all knowledge bases using all of the available inference rules, however this can be distributed between the agents in different ways, resulting in different numbers of exchanged messages. Similarly, if parts of a derivation can be performed in parallel, the total derivation will be shorter, though communication of the partial results will increase the communication complexity. As an illustration, figure 3 shows one possible distribution of the resolution example in figure 2. As can be seen, two communicating agents can solve the problem faster than a single agent.

4 A Bounded Memory and Communication Logic BMCL

In this section we present a temporal epistemic logic $BMCL$ which allows us to describe a set of reasoning agents with bounds on memory and on the number of messages they can exchange. In this logic, we can express statements like 'the agents will be able to derive the goal formula in n inference steps'. The bounds on memory and communication ability are expressed as axioms in the logic. In this paper, as an example, we have chosen to axiomatise a set of agents reasoning using resolution. Other reasoning systems can be axiomatised in a similar way, and we briefly sketch how to add model conditions and axioms for reasoners which reason using conjunction introduction and modus ponens to our logic at the end of this section.

Let the set of agents be $AG = \{1, 2, .., n_{AG}\}$. For simplicity, we assume that they agree on a finite set $PROP$ of propositional variables (this assumption can easily be relaxed, so that only some propositional variables are shared). Since each agent uses resolution for reasoning, we assume that all formulas of the internal language of the agents are in the form of *clauses*. For convenience, we define a clause as a set of *literals* in which a literal is a propositional variable or its negation. Then the set of literals is defined as $LPROP = \{p, \neg p | p \in PROP\}$. If L is a literal, then $\neg L$ is $\neg p$ if L is a propositional variable p, and p if L is of the form $\neg p$. Let Ω be the set of all possible clauses over

$PROP$, i.e., $\Omega = \wp(LPROP)$. Note that Ω is finite.

The only rule of inference that each agent has is the *resolution rule* which is defined as follows:

$$\frac{\alpha \ni L \qquad \beta \ni \neg L}{(\alpha \setminus \{L\}) \cup (\beta \setminus \{\neg L\})} \text{ Res}$$

which states that if there are two clauses α and β such that one contains a literal L and the other contains $\neg L$, then we can derive a new clause $(\alpha \setminus \{L\}) \cup (\beta \setminus \{\neg L\})$.

Each agent i has a memory of size $n_M(i)$ where one unit of memory corresponds to the ability to store an arbitrary clause. Agent i can read clauses from its set of premises K_i. We assume that each K_i is finite. The communication ability of the agents is expressed by the *copy* action which copies a clause from another agent's memory. The limit on each agent's communication ability is $n_C(i)$: in any valid run of the system, agent i can perform at most $n_C(i)$ copy actions.

4.1 Syntax of $BMCL$

The syntax of $BMCL$ is defined inductively as follows.

- $\top$ is a well-formed formula (wff) of $BMCL$.
- If α is a clause, then $B_i^r\alpha$ is a wff of $BMCL$, for all $i \in AG$.
- If α is a clause, then $B_i^c\alpha$ is a wff of $BMCL$, for all $i \in AG$.
- If ϕ and ψ are wff, then so are $\neg\phi$, $\phi \wedge \psi$.
- If ϕ and ψ are wff, then so are $X\phi$, $\phi U\psi$, $Y\phi$, $\phi S\psi$ and $A\phi$.

Classical abbreviations for $\vee$, $\rightarrow$, $\leftrightarrow$ and $\bot$ are defined as usual.

The language contains both temporal and epistemic modalities. For the temporal part of $BMCL$, we have $PCTL^*$, a branching time temporal logic with the past operator.[2] Intuitively, $PCTL^*$ describes infinite trees, or all possible runs of the system, over discrete time. In the temporal logic part of the language, X stands for next step, U for until, Y for previous step, S for since and A for 'on all paths'. We will also use abbreviations $F\phi \equiv \top U\phi$ for some time in the future, $P\phi \equiv \top S\phi$ for some time in the past, $E\phi \equiv \neg A\neg\phi$ for on some path and $start \equiv \neg Y\top$ for the starting state of the system. The epistemic part of the language consists of belief modalities $B_i^r\alpha$, which means that agent i has read α from its knowledge base or derived it, and $B_i^c\alpha$, which means that i has copied α from another agent. We define $B_i\alpha$ (agent i believes α) to be $B_i^r\alpha \vee B_i^c\alpha$.

[2]The reason we use $PCTL^*$ rather than CTL^* is that we need the past operator to express the bound on agent communication.

4.2 Semantics of $BMCL$

The semantics of $BMCL$ is defined by $BMCL$ tree-like transition systems. A $BMCL$ transition system $M = (S, R, V^r, V^c)$ is defined as follows.

- S is a non-empty set of states.
- $R \subseteq S \times S$ is a total binary relation, i.e. for any $s \in S$, there exists $t \in S$ such that $(s,t) \in R$. Moreover, it is also required that (S, R) is a tree-frame. A branch σ is an infinite sequence $(s_0, s_1, ..)$ such that $(s_i, s_{i+1}) \in R$ for all $i \geq 0$, σ_i denotes the element s_i of σ and $\sigma_{\leq i}$ is the prefix $(s_0, s_1, .., s_i)$ of σ. The set of all branches is denoted as BR. Note that since (S, R) is a tree-frame every state s has a unique past $past(s) = \sigma_{\leq i}$ where $\sigma_i = s$.
- $V^r : S \times AG \rightarrow \wp(\Omega)$, is a mapping that defines for each state which formulas an agent believes due to reading or inference.
- $V^c : S \times AG \rightarrow \wp(\Omega)$, is a mapping that defines for each state which formulas an agent copied from the memories of other agents.

The truth of a $BMCL$ formula in a state at point n of a path σ of M is defined inductively as follows:

- $M, \sigma, n \models B_i^r\alpha$ iff $\alpha \in V^r(\sigma_n, i)$,
- $M, \sigma, n \models B_i^c\alpha$ iff $\alpha \in V^c(\sigma_n, i)$,
- $M, \sigma, n \models \neg\phi$ iff $M, \sigma, n \not\models \phi$,
- $M, \sigma, n \models \phi \wedge \psi$ iff $M, \sigma, n \models \phi$ and $M, \sigma \models \psi$,
- $M, \sigma, n \models X\phi$ iff $M, \sigma, n+1 \models \phi$,
- $M, \sigma, n \models \phi U\psi$ iff $\exists m \geq n$ such that $\forall k \in [n, m)$ $M, \sigma, k \models \phi$ and $M, \sigma, m \models \psi$,
- $M, \sigma, n \models Y\phi$ iff $n > 0$ and $M, \sigma, n-1 \models \phi$,
- $M, \sigma, n \models \phi S\psi$ iff $\exists m \leq n$ such that $\forall k \in (m, n]$ $M, \sigma, m \models \phi$ and $M, \sigma, k \models \psi$,
- $M, \sigma, n \models A\phi$ iff $\forall \sigma' \in BR$ such that $\sigma'_{\leq n} = \sigma_{\leq n}$, $M, \sigma', n \models \phi$.

Now we describe conditions on the models. The first set of conditions refers to the accessibility relation R. The intuition behind the conditions is that R corresponds to the agents executing actions $\langle a_1, \ldots, a_{n_{AG}} \rangle$ in parallel, where action a_i is a possible action (transition) for the agent i in a given state. Actions of each agent i are: $Read_{i,\alpha,\beta}$ (reading a clause α from the knowledge base and erasing β), $Res_{i,\alpha_1,\alpha_2,L,\beta}$ (resolving α_1 and α_2 and erasing β), $Copy_{i,\alpha,\beta}$ (copying α from another agent and erasing β),

$Erase_{i,\alpha}$ (erasing α), and $Null_i$ (doing nothing), where $\alpha, \alpha_1, \alpha_2, \beta \in \Omega$ and $L \in LPROP$.[3] Intuitively, β is an arbitrary clause which may or may not be in the agent's memory, which gets overwritten in this transition. If the agent's memory is full ($|V^r(s,i)| + |V^c(s,i)| = n_M(i)$), then β has to be in $V^r(s,i) \cup V^c(s,i)$, otherwise we cannot add an extra formula to it (this would violate the condition on memory defined below). Not all actions are possible in any given state (for example, to perform a resolution step from state s, the agent has to have two resolvable clauses assigned in s). Let us denote the set of all possible actions by agent i in state s by $R_i(s)$.

Below is the definition of $R_i(s)$:

Definition 4.1 [Available actions] For every state s and agent i,

1. $Read_{i,\alpha,\beta} \in R_i(s)$ iff $\alpha \in K_i$ and $\beta \in \Omega$, or if $|V^r(s,i)| + |V^c(s,i)| = n_M(i)$ then $\beta \in V^r(s,i) \cup V^c(s,i)$.
2. $Res_{i,\alpha_1,\alpha_2,L,\beta} \in R_i(s)$ iff $\alpha_1, \alpha_2 \in \Omega$, $\alpha_1 \ni L$, $\alpha_2 \ni \neg L$, $\alpha_1, \alpha_2 \in V^r(s,i) \cup V^c(s,i)$, $\alpha = (\alpha_1 \setminus \{L\}) \cup (\alpha_2 \setminus \{\neg L\}) \notin K_i$ and β is as before.
3. $Copy_{i,\alpha,\beta} \in R_i(s)$ iff there exists $j \neq i$ such that $\alpha \in V^r(s,j) \cup V^c(s,j)$ and $past(s)$ does not contain more than $n_C(i) - 1$ transitions of the form $Copy_{i,\beta}$, and β is as before.[4]
4. $Null_i$ is always in $R_i(s)$.
5. There are no conditions on $Erase_{i,\alpha} \in R_i(s)$.

◁

Now we define effects of actions on the agent's state (assignments $V^r(s,i)$ and $V^c(s,i)$).

Definition 4.2 [Effects of actions] For each $i \in AG$, the result of performing an action a in state s is defined if $a \in R_i(s)$ and has the following effect on the assignment of clauses to i in the successor state t:

1. if a is $Read_{i,\alpha,\beta}$: $V^r(t,i) = (V^r(s,i) \setminus \{\beta\}) \cup \{\alpha\}$ and $V^c(t,i) = V^c(s,i) \setminus \{\beta\}$.
2. if a is $Res_{i,\alpha_1,\alpha_2,L,\beta}$: $V^r(t,i) = (V^r(s,i) \setminus \{\beta\}) \cup \{\alpha\}$ and $V^c(t,i) = V^c(s,i) \setminus \{\beta\}$, where $\alpha = (\alpha_1 \setminus \{L\}) \cup (\alpha_2 \setminus \{\neg L\})$.
3. if a is $Copy_{i,\alpha,\beta}$: $V^c(t,i) = (V^c(s,i) \setminus \{\beta\}) \cup \{\alpha\}$ and $V^r(t,i) = V^r(s,i) \setminus \{\beta\}$.
4. if a is $Null_i$: $V^r(t,i) = V^r(s,i)$ and $V^c(t,i) = V^c(s,i)$
5. if a is $Erase_{i,\alpha}$ then $V^r(t,i) = V^r(s,i) \setminus \{\alpha\}$ and $V^c(t,i) = V^c(s,i) \setminus \{\alpha\}$, where $\alpha \in \Omega$.

◁

Definition 4.3 $BMCM(K_1, .., K_{n_{AG}}, n_M, n_C)$ is the set of models $M = (S, R, V, C)$ such that:

1. For every s and t, $R(s,t)$ iff for some tuple of actions $\langle a_1, \ldots, a_{n_{AG}} \rangle$, $a_i \in R_i(s)$ and the assignment in t satisfies the effects of a_i for every i in $\{1, \ldots, n_{AG}\}$.
2. For every s and a tuple of actions $\langle a_1, \ldots, a_{n_{AG}} \rangle$, if $a_i \in R_i(s)$ for every i in $\{1, \ldots, n_{AG}\}$, then there exists $t \in S$ such that $R(s,t)$ and t satisfies the effects of a_i for every i in $\{1, \ldots, n_{AG}\}$.
3. The bound on each agent's memory is set by the following constraint on the mappings V^r and V^c:

   $|V^r(s,i)| + |V^c(s,i)| \leq n_M(i)$ for all $s \in S$ and $i \in AG$

◁

Note that the bound $n_C(i)$ on each agent i's communication ability (no branch contains more than $n_C(i)$ $Copy$ actions by agent i) follows from the fact that $Copy_i$ is only enabled if i has performed fewer than $n_C(i)$ copy actions in the past.

4.3 Axiomatisation of $BMCL$

Before we give an axiomatisation for the set of models defined above, we need the following abbreviations for expressing that i has performed at least k copy actions in the past. A successful copying of a clause α by agent i from an agent j is defined by the following formula:

$$copied(i,j,\alpha) \equiv B_j\alpha \wedge \neg B_i\alpha \wedge XB_i^c\alpha$$

Copying of any clause from any agent by agent i is defined as follows:

$$copied_i \equiv \bigvee_{j \in AG,\ \alpha \in \Omega} copied(i,j,\alpha)$$

So to say that there are at least k copy actions in agent i's past, we can use

$$C_i^{\geq}(k) = \underbrace{(YP(copied_i \wedge YP(copied_i \wedge \ldots YP(copied_i)\ldots)}_{k \text{ times}}$$

[3] The $Erase_{i,\alpha}$ action is introduced for purely technical reasons, to obtain a simpler axiomatisation of the system. The optimal sequences of actions found by the system when verifying properties of agents will contain no $Erase$ actions so will not affect the verification process.

[4] Assume that the state contains a communication counter for each agent i, which is set to 0 in the start state and is incremented every time i performs a copy action. After the counter reaches $n_C(i)$, agent i cannot perform any more copy actions.

and to say that there are fewer than k copy actions in agent i's past, we can say

$$C_i^{\leq}(k) = \underbrace{\neg(YP(copied_i \wedge YP(copied_i \wedge \ldots YP(copied_i)\ldots)}_{k+1 \text{ times}}$$

To define exactly k copies, we can use $C_i(0) = C_i^{\leq}(0)$ and $C_i(k) = C_i^{\geq}(k) \wedge C_i^{\leq}(k)$ for $k > 0$.

Consider the following set of axiom schemata:

A1 Axioms and rules of $PCTL^*$ as given in [24].

A2 $\bigwedge_{\alpha_q \in Q} B_i\alpha_q \wedge C_i(n) \rightarrow EX(\bigwedge_{\alpha_q \in Q} B_i\alpha_q \wedge C_i(n) \wedge B_i^r\alpha)$ for all $\alpha \in K_i$, $i \in AG$, $Q \subseteq \Omega$ with $|Q| < n_M(i)$, and $n \geq 0$.

A3 $\bigwedge_{\alpha_q \in Q} B_i\alpha_q \wedge C_i(n) \wedge B_i\alpha_1 \wedge B_i\alpha_2 \rightarrow EX(\bigwedge_{\alpha_q \in Q} B_i\alpha_q \wedge C_i(n) \wedge B_i\alpha)$ for any α_1 and α_2 such that $\alpha_1 \ni L$ and $\alpha_2 \ni \neg L$ for some literal L, $\alpha = (\alpha_1 \setminus \{L\}) \cup (\alpha_2 \setminus \{\neg L\}) \notin K_i$, $Q \subseteq \Omega$ with $|Q| < n_M(i)$, and $n \geq 0$.

A4 $\bigwedge_{\alpha_q \in Q} B_i\alpha_q \wedge C_i(n) \wedge B_j\alpha \wedge C_i^{\leq}(n_C(i)) \rightarrow EX(\bigwedge_{\alpha_q \in Q} B_i\alpha_q \wedge C_i(n+1) \wedge B_i^c\alpha)$ for all $i \neq j \in AG$, $Q \subseteq \Omega$ with $|Q| < n_M(i)$, and $n \geq 0$.

A5 $EX(B_i\alpha_1 \wedge B_i\alpha_2) \rightarrow B_i\alpha_1 \vee B_i\alpha_2$

A6 $EX(\neg B_i\alpha_1 \wedge \neg B_i\alpha_2) \rightarrow (\neg B_i\alpha_1 \vee \neg B_i\alpha_2)$

A7 $EX(B_i^r\alpha \wedge C_i(n)) \rightarrow B_i^r\alpha \vee (\neg B_i^r\alpha \wedge C_i(n))$ for all $\alpha \in K_i$

A8 $EX(B_i^r\alpha \wedge C_i(n)) \rightarrow B_i^r\alpha \vee (\neg B_i^r\alpha \wedge \bigvee_{(\alpha_1,\alpha_2) \in Res(\alpha)}(B_i\alpha_1 \wedge B_i\alpha_2 \wedge C_i(n)))$ for all $\alpha \notin K_i$ and $Res(\alpha) = \{(\alpha_1, \alpha_2) \in \Omega \times \Omega | \alpha_1 \ni L, \alpha_2 \ni \neg L \text{ and } \alpha = (\alpha_1 \setminus \{L\}) \cup (\alpha_2 \setminus \{\neg L\})\}$ for some literal L and $n \geq 0$

A9 $EX(B_i^c\alpha \wedge C_i(n)) \rightarrow B_i^c\alpha \vee (\neg B_i^c\alpha \wedge C_i(n-1) \wedge (\bigvee_{j \in AG} B_j\alpha))$

A10 $B_i\alpha_1 \wedge .. \wedge B_i\alpha_{n_M} \rightarrow \neg B_i\alpha_{n_M+1}$ where $i = 1, .., n_{AG}$, and $\alpha_i \neq \alpha_j$ for all $i \neq j$

A11 $C_i^{\leq}(n_C(i))$

A12 $\bigwedge_{j \in J} EX(\bigwedge_{q \in Q} B_j\alpha_q \wedge C_j(k_j)) \rightarrow EX \bigwedge_{j \in J}(\bigwedge_{q \in Q} B_j\alpha_q \wedge C_j(k_j))$ where $J \subseteq AG$ and all indices j are distinct, and $Q \subseteq \Omega$.

A13 $\phi \rightarrow EX\phi$

Let $BMCL(K_1, .., K_{n_{AG}}, n_M, n_C)$ be the logic defined by the our axiomatization. Then we have the following result.

Theorem 4.4 *$BMCL(K_1, .., K_{n_{AG}}, n_M, n_C)$ is sound and weakly complete with respect to $BMCM(K_1, .., K_{n_{AG}}, n_M, n_C)$.*

Proof. The proof of soundness is standard. Due to lack of space, we only prove validity of the first $BMCL$ axiom.

Let us consider **A2** and a model $M = (S, R, V^r, V^c)$ of $BMCM$ $(K_1, \ldots, K_{n_{AG}}, n_M)$. Let $\sigma = (s_0, s_1, \ldots) \in BR$, it is required to prove for any m, that if $M, \sigma, m \models \bigwedge_{\alpha_q \in Q} B_i\alpha_q \wedge C_i(n)$, then $M, \sigma, m \models EX(\bigwedge_{\alpha_q \in Q} B_i\alpha_q \wedge C_i(n) \wedge B^r\alpha)$ where $\alpha \in K_i$, $i \in AG$, $Q \subseteq \Omega$ with $|Q| < n_M(i)$ and $n \geq 0$. Since $\alpha \in K_i$, $Read_{i,\alpha,\beta} \in R_i(\sigma_n)$ for some $\beta \in \Omega \setminus Q$. Therefore, there exists $t \in S$ such that $R(s,t)$ and t satisfies the effects of $Read_{i,\alpha,\beta}$. In other words, we obtain $V^r(t,i) = V^r(s,i) \cup \{\alpha\} \setminus \{\beta\}$ and $V^c(t,i) = V^c(s,i) \setminus \{\beta\}$, this shows $V^r(t,i) \ni \alpha$. Since $M, \sigma, m \models \bigwedge_{\alpha_q \in Q} B_i\alpha_q$, $V^r(s,i) \cup V^c(s,i) \ni \alpha_q$ for all $q \in Q$. Moreover, since $|Q| < n_M(i)$, we have $\beta \in \Omega \setminus Q$, therefore $V^r(t,i) \cup V^c(t,i) \ni \alpha_q$. Then, $M, \sigma', m+1 \models \bigwedge_{\alpha_q \in Q} B_i\alpha_q \wedge B^r\alpha$ for some $\sigma' \in BR$ such that $\sigma'_{\leq m+1} = (\sigma_1, \ldots, \sigma_m, t)$.

Since $M, \sigma, m \models C_i(n)$, we have that agent i has performed exactly n copy actions on the prefix $(\sigma_1, \ldots, \sigma_m)$. Moreover, the action that agent i performs between σ_m and t is to read α from K_i, therefore it has still performed exactly n copy actions on the prefix $(\sigma_1, \ldots, \sigma_m, t)$. Then, it is straightforward that $M, \sigma', m+1 \models C_i(n)$. That gives us $M, \sigma', m+1 \models \bigwedge_{\alpha_q \in Q} B_i\alpha_q \wedge B^r\alpha \wedge C(i,n)$. Since $\sigma'_{\leq m} = \sigma_{\leq m}$ and $R(\sigma_m, t)$, we obtain $M, \sigma, m \models EX(\bigwedge_{\alpha_q \in Q} B_i\alpha_q \wedge C_i(n) \wedge B^r\alpha)$.

To prove completeness, a satisfying model for a consistent formula is constructed as in the completeness proof of $PCTL^*$ from [24]. Then we use the axioms to show that this model is in $BMCM(K_1, \ldots, K_{n_{AG}}, n_M, n_C)$. QED

4.4 Systems of Heterogeneous Reasoners

Changing the logic to accommodate reasoners which reason using a different set of inference rules rather than resolution is relatively straightforward. As an illustration, we show how to add model conditions and axioms for reasoners which use modus ponens and conjunction introduction. We assume that the knowledge base of these reasoners contains literals and implications of the form $L_1 \wedge \ldots \wedge L_n \rightarrow L$.

First of all, we need to change the conditions on models so that instead of using the Res action, a reasoner could change the state by performing MP and AND actions. Let i be an (MP, AND) reasoner. Define Ω_i as K_i closed under subformulas and the following conjunction introduction: if Q is a set of distinct literals from K_i, then $\wedge Q \in \Omega_i$. An agent i has actions $Read_{i,\phi,\beta}$ for any formula ϕ in K_i,

$Copy_{i,\phi,\beta}$ for any formula $\phi \in \Omega_i$, $Null_i$, $Erase_i$, and instead of Res it has $MP_{i,\phi_1,\phi_1\rightarrow\phi_2,\beta}$ and $AND_{i,\phi_1,\phi_2,\beta}$.

Definition 4.5 [Availability of MP and AND] For any $s \in S$:

1. $MP_{i,\phi_1,\phi_1\rightarrow\phi_2,\beta} \in R_i(s)$ iff $\phi_1, \phi_1 \rightarrow \phi_2 \in V^r(s,i) \cup V^c(s,i)$ and $\beta \in \Omega_i$.
2. $AND_{i,\phi_1,\phi_2,\beta} \in R_i(s)$ iff $\phi_1, \phi_2 \in V^r(s,i) \cup V^c(s,i)$ and $\beta \in \Omega_i$.

◁

Definition 4.6 [Effects of MP and AND] For every $s \in S$, the result of performing action a is defined if $a \in R_i(s)$ and has the following effect on the resulting state t:

1. if a is $MP_{i,\phi,\phi\rightarrow\phi_2,\beta}$ then $V^r(t,i) = V^r(s,i) \cup \{\phi_2\} \setminus \{\beta\}$ and $V^c(s,i) = V^c(t,i) \setminus \{\beta\}$.
2. if a is $AND_{i,\phi_1,\phi_2,\beta}$ iff $V^r(t,i) = V^c(s,i) \cup \{\phi_1 \wedge \phi_2\} \setminus \{\beta\}$ and $V^c(s,i) = V^c(t,i) \setminus \{\beta\}$.

◁

The corresponding axioms for the (MP, AND) reasoner are as follows:

A13 $\bigwedge_{q\in Q} B_i\phi_q \wedge C_i(n) \wedge B_i\phi_1 \wedge B_i(\phi_1 \rightarrow \phi_2) \rightarrow EX(\bigwedge_{q\in Q} B_i\phi_q \wedge C_i(n) \wedge B_i\phi_2)$ where $Q \subseteq \Omega_i$ with $|Q| < n_M(i)$

A14 $\bigwedge_{q\in Q} B_i\phi_q \wedge C_i(n) \wedge B_i\phi_1 \wedge B_i\phi_2 \rightarrow EX(\bigwedge_{q\in Q} B_i\phi_q \wedge C_i(n) \wedge B_i(\phi_1 \wedge \phi_2))$ where $Q \subseteq \Omega_i$ with $|Q| < n_M(i)$, and $\phi_1, \phi_2 \in \Omega_i$.

A15 $EX(B_i(\phi_1 \wedge \phi_2) \wedge C_i(n)) \rightarrow (B_i(\phi_i \wedge \phi_2) \vee (\neg B_i(\phi_i \wedge \phi_2) \wedge B_i\phi_1 \wedge B_i\phi_2 \wedge C_i(n)))$

A16 $EX(B_i\phi_2 \wedge C_i(n)) \rightarrow (B_i\phi_2 \vee (\neg B_i\phi_2 \wedge C_i(n) \wedge \bigvee_{\phi_1\rightarrow\phi_2 \in K_i}(B_i\phi_1 \wedge B_i(\phi_1 \rightarrow \phi_2))))$ for all $\phi_2 \notin K_i$.

Now we can add the conditions and axioms for the (MP, AND) reasoner to the system for resolution reasoners and obtain an axiomatisation for the heterogeneous system of reasoners.

5 Verifying Resource Bounds

The logic $BMCL$ allows us to express precisely how beliefs of a set of resource-bounded agents change over time, and, given a memory and communication bound for each agent, to verify formulas which state that a certain belief will or will not be acquired within a certain number of steps. For example, given a system of two agents with premises $K_1 = \{\{p_1, p_2\}, \{\neg p_1, p_2\}\}$ and $K_2 = \{\{p_1, \neg p_2\}, \{p_1, \neg p_2\}\}$, with bounds $n_M(1) = 2$, $n_M(2) = 2$ (both agents have 2 memory cells) and $n_C(1) = 0$, $n_C(2) = 1$ (agent 1 cannot copy anything and agent 2 can copy one clause), we can prove that $start \rightarrow EX^5B_2(\{\})$ (i.e., from the start state, the agents can derive the empty clause in 5 steps).

However, rather than deriving such properties by hand, it is more convenient to use an automatic method to verify them. In this section, we describe how the models in $BMCM(K_1, .., K_{n_{AG}}, n_M, n_C)$ can be encoded as an input to a model-checker to allow the automatic verification of the properties expressing resource bounds.

5.1 Model Checker Encoding

It is straightforward to encode a $BMCM$ model of such a system for a standard model checker, and to verify resource bounds using existing model checking techniques. For the examples reported here, we have used the Mocha model checker [7].

States of the $BMCM$ models correspond to an assignment of values to state variables in the model-checker. The state variables representing an agent's memory are organised as a collection of 'cells', each holding at most one clause. For an agent i with memory bound $n_M(i)$, there are $n_M(i)$ cells. Each cell is represented by a pair of bitvectors, each of length $k = |PROP|$, representing the positive and negative literals in the clause in some standard order (e.g., lexicographic order). For example, if $PROP$ contains the propositional variables A_1, A_2 and A_3 with index positions 0, 1 and 2 respectively, the clause $A_1 \vee \neg A_3$ would be represented by two bitvectors: "100" for the positive literals and "001" for the negative literals. This gives reasonably compact states.

Actions by each agent such as reading a premise, resolution and communication with other agents are represented by Mocha *atoms* which describe the initial condition and transition relation for a group of related state variables. Reading a premise ($Read_{i,\alpha,\beta}$) simply sets the bitvectors representing an arbitrary cell in agent i's memory to the appropriate values for the clause α. Resolution ($Res_{i,\alpha_1,\alpha_2,L,\beta}$) is implemented using simple bit operations on cells containing values representing α_1 and α_2, with the results being assigned to an arbitrary cell in agent i's memory. Communication ($Copy_{i,\alpha,\beta}$) is implemented by copying the values representing α from a cell of agent j to an arbitrary cell of agent i. To express the communication bound, we use a counter for each agent which is incremented each time a copy action is performed by the agent. After the counter for agent i reaches $n_C(i)$, the $Copy_{i,\alpha,\beta}$ action is disabled.

Mocha supports hierarchical modelling through composition of *modules*. A module is a collection of atoms and a specification of which of the state variables updated by

# agents	Distrib.	Memory	Comm.	Time
1	Symmetric	3	–	8
2	Symmetric	2, 2	1, 0	6
2	Symmetric	3, 3	1, 0	6
2	Symmetric	3, 3	0, 0	8
2	Symmetric	2, 1	1, 1	9
2	Asymmetric	2, 2	2, 1	7
2	Asymmetric	3, 3	2, 1	7
2	Asymmetric	3, 1	1, 0	8

Table 1. Tradeoffs between resource bounds

those atoms are visible from outside the module. In our encoding, each agent is represented by a module. A particular distributed reasoning system is then simply a parallel composition of the appropriate agent modules.

The specification language of Mocha is ATL, which includes CTL. We can express properties such as 'agent i may derive belief ϕ in n steps' as $EF\ tr(B_i\alpha)$ where $tr(B_i\alpha)$ is a suitable encoding of the fact that a clause α is present in the agent's memory (either as a disjunction of possible values of cell bitvectors, or as a special boolean variable which becomes true when one of the cells contains a particular value, for example all 0s for the empty clause). To obtain the actual derivation we can verify the negation of a formula, for example $AG\ \neg tr(B_i\alpha)$—the counterexample trace will show how the system reaches the state where α is proved.

5.2 Examples

Consider a single agent (agent 1) whose knowledge base contains all clauses of the form $\sim A_1 \vee \sim A_2$ where $\sim A_i$ is either A_i or $\neg A_i$, and which has the goal of deriving the empty clause. We can express the property that agent 1 will derive the empty clause at some point in the future as $EF\ B_1\{\}$.

Using the model checker, we can show that deriving the empty clause requires a memory bound of 3 and 8 time steps (see Figure 2).[5] We can also show that these space and time bounds are minimal for a single agent; i.e., increasing the space bound does not result in a shorter proof.

With two agents and a symmetric problem distribution (i.e., each agent has all the premises $\sim A_1 \vee \sim A_2$), we can show that a memory bound of 2 (i.e., the minimum required for resolution) and a communication bound of 1 gives a proof of 6 steps (see Figure 3). Reducing the communication bound to 0 results in no proof, as, with a memory bound of 2 for each agent, at least one clause must be communicated from one agent to the other. Increasing the space bound to 3 (for each agent) does not shorten the proof, though it does allow the communication bound to be reduced to 0 at the cost of increasing the proof length to 8 (i.e., the single agent case). Reducing the total space bound to 3 (i.e., 2 for one agent and 1 for the other, equivalent to the single agent case) increases the number of steps required to find a proof to 9 and the communication bound to 1 for each agent. In effect, one agent functions as a cache for a clause required later in the proof, and this clause must be copied in both directions.

If the problem distribution is asymmetric, e.g., if one agent has premises $A_1 \vee A_2$ and $\neg A_1 \vee \neg A_2$ and the other has premises $\neg A_1 \vee A_2$ and $A_1 \vee \neg A_2$, then with a memory bound of 2 for each agent, we can show that the time bound is 7, and the communication bound is 2 for the first agent and 1 for the second. Increasing the memory bound for each agent to 3 does not reduce the time bound. However the memory bound can be reduced to 1 and the communication bound reduced to 1 for one agent and 0 for the other, if the time bound is increased to 8 (again this is equivalent to the single agent case, except that one agent copies the clause it lacks from the other rather than reading it). These tradeoffs are summarised in Table 1.

Increasing the size of the problem increases the number of possible tradeoffs, but similar patterns can be seen to the 2-variable case. For example, if the agent's knowledge base contain all clauses of the form $\sim A_1 \vee \sim A_2 \vee \sim A_3$, then a single agent requires a memory bound of 4 and 16 steps to achieve the goal. In comparison, two agents, each with a memory bound of 2, require 13 steps and 4 messages to derive the goal.

While extremely simple, these examples serve to illustrate the interaction between memory, time and communication bounds, and between the resource distribution and the problem distribution.

6 Related Work

There exist several approaches to epistemic logic which model reasoners as resource-bounded (not logically omniscient), including deduction model of belief [21], step logic and active logic [12, 17], algorithmic knowledge [18, 14, 23], and other syntactic epistemic logics [11, 2, 5, 19] where each inference step takes the agent into the next (or some future) moment in time. A logic where the depth of belief reasoning is limited is studied in [16].

A considerable amount of work has also been done in the area of model-checking multi-agent systems (see, e.g., [10, 9]). However, this work lacks a clear connection between the way agent reasoning is modelled in agent theory (which typically assumes that the agents are logically omniscient) and the formalisations used for model checking, and emphasises correctness rather than the interplay between time, memory, bounds on communications and the ability

[5] The space required for problems of this form is known to be logarithmic in the number of premises [13].

of agents to derive a certain belief.

The current paper extends the work of [3] which proposed a method of verifying memory and time bounds in a single reasoner which reasons in classical logic using natural deduction rather than resolution. We also extend the work in [4] which analyses a system of communicating rule-based reasoners and verifies time bounds for those systems, but assumes unlimited memory. As far as we are aware, the logic we propose in this paper is the first attempt to analyse time, space and communication bounds of reasoners in one logical system, and verify properties relating to all three resources using a model-checker.

7 Conclusions and Future Works

In this paper, we analyse the time, space and communication resources required by a system of reasoning agents to achieve a goal. We give a rigorous definition of the measures for each of those resources, and introduce an epistemic logic $BMCL$ where we can express properties of a system of resource-bounded reasoning agents. In particular, we can express bounds on memory and communication resources as axioms in the logic. We axiomatise a system of agents which reason using resolution (other reasoning systems can be axiomatised in a similar way), prove that the resulting logic is sound and complete, and show how to express properties of the system of reasoning agents in $BMCL$. Finally, we show how $BMCL$ transition systems can be encoded as input to the Mocha model-checker and how properties, such as existence of derivations with given bounds on memory, communication, and the number of inference steps, can be verified automatically.

In future work, we plan to consider logical languages containing primitive operators which would allow us to state the agents' resource limitations as formulas in the language rather than axioms, and consider agents reasoning about each other's resource limitations. We also would like to consider agents reasoning in a simple epistemic or description logic.

Acknowledgements: This work was partially supported by the EPSRC project EP/E031226/1.

References

[1] P. Adjiman, P. Chatalic, F. Goasdoué, M.-C. Rousset, and L. Simon. Distributed reasoning in a peer-to-peer setting. In *Proceedings of the 16th European Conference on Artificial Intelligence (ECAI'2004)*, pages 945–946. IOS Press, 2004.

[2] T. Ågotnes and M. Walicki. Strongly complete axiomatizations of "knowing at most" in standard syntactic assignments. In *Proceedings of the* 6^{th} *International Workshop on Computational Logic in Multi-agent Systems (CLIMA VI)*, 2005.

[3] A. Albore, N. Alechina, P. Bertoli, C. Ghidini, B. Logan, and L. Serafini. Model-checking memory requirements of resource-bounded reasoners. In *Proceedings of the Twenty-First National Conference on Artificial Intelligence (AAAI 2006)*, pages 213–218. AAAI Press, 2006.

[4] N. Alechina, M. Jago, and B. Logan. Modal logics for communicating rule-based agents. In *Proceedings of the 17th European Conference on Artificial Intelligence (ECAI'2006)*, pages 322–326. IOS Press, 2006.

[5] N. Alechina, B. Logan, and M. Whitsey. A complete and decidable logic for resource-bounded agents. In *Proceedings of the Third International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS 2004)*, pages 606–613. ACM Press, 2004.

[6] M. Alekhnovich, E. Ben-Sasson, A. A. Razborov, and A. Wigderson. Space complexity in propositional calculus. *SIAM J. of Computing, 31(4)*, pages 1184–1211, 2002.

[7] R. Alur, T. A. Henzinger, F. Y. C. Mang, S. Qadeer, S. K. Rajamani, and S. Tasiran. MOCHA: Modularity in model checking. In *Computer Aided Verification*, pages 521–525, 1998.

[8] E. Amir and S. A. McIlraith. Partition-based logical reasoning for first-order and propositional theories. *Artificial Intelligence, 162(1-2)*, pages 49–88, 2005.

[9] M. Benerecetti, F. Giunchiglia, and L. Serafini. Model checking multiagent systems. *J. Log. Comput., 8(3)*, pages 401–423, 1998.

[10] R. Bordini, W. V. M. Fisher, and M. Wooldridge. State-space reduction techniques in agent verification. In *Proc. of the Third International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS-2004)*, pages 896–903. ACM Press, 2004.

[11] H. Duc. Reasoning about rational, but not logically omniscient, agents. *J. Log. Comput., 5*, pages 633–648, 1997.

[12] J. J. Elgot-Drapkin and D. Perlis. Reasoning situated in time I: Basic concepts. *J. of Experimental and Theoretical Artificial Intelligence, 2*, pages 75–98, 1990.

[13] J. L. Esteban and J. Torán. Space bounds for resolution. In *Proc. of the 16th Annual Symposium on Theoretical Aspects of Computer Science (STACS 99)*, pages 551–560. Springer, 1999.

[14] R. Fagin, J. Y. Halpern, Y. Moses, , and M. Y. Vardi. *Reasoning about Knowledge.* MIT Press, Cambridge, 1995.

[15] B. Faltings and M. Yokoo. Introduction: Special issue on distributed constraint satisfaction. *J. of Artif. Intell., 161(1-2)*, pages 1–5, 2005.

[16] M. Fisher and C. Ghidini. Programming Resource-Bounded Deliberative Agents. In *Proc. of the Sixteenth International Joint Conference on Artificial Intelligence (IJCAI'99)*, pages 200–206. Morgan Kaufmann, 1999.

[17] J. Grant, S. Kraus, and D. Perlis. A logic for characterizing multiple bounded agents. *Autonomous Agents and Multi-Agent Systems, 3(4)*, pages 351–387, 2000.

[18] J. Y. Halpern, Y. Moses, and M. Y. Vardi. Algorithmic knowledge. In *Proc. of the 5th Conference on Theoretical Aspects of Reasoning about Knowledge*, pages 255–266. Morgan Kaufmann, 1994.

[19] M. Jago. *Logics for Resource-Bounded Agents*. PhD thesis, University of Nottingham, 2006.

[20] H. Jung and M. Tambe. On communication in solving distributed constraint satisfaction problems. In *Multi-Agent Systems and Applications IV, Proc. 4th International Central and Eastern European Conference on Multi-Agent Systems, CEEMAS 2005*, pages 418–429. Springer, 2005.

[21] K. Konolige. *A Deduction Model of Belief.* Morgan Kaufmann, San Francisco, Calif., 1986.

[22] G. M. Provan. A model-based diagnosis framework for distributed embedded systems. In *Proc. of the Eighth International Conference on Principles and Knowledge Representation and Reasoning (KR-02)*, pages 341–352. Morgan Kaufmann, 2002.

[23] R. Pucella. Deductive algorithmic knowledge. In *AI&M 1-2004, Eighth International Symposium on Artificial Intelligence and Mathematics*, 2004.

[24] M. Reynolds. An axiomatization of $PCTL^*$. *Inf. Comput., 201(1)*, pages 72–119, 2005.

[25] A. C.-C. Yao. ome complexity questions related to distributive computing(preliminary report). In *Conference Record of the Eleventh Annual ACM Symposium on Theory of Computing*, pages 209–213. ACM, 1979.