Verifying time and communication costs of rule-based reasoners

Natasha Alechina, Brian Logan, Nguyen Hoang Nga, and Abdur Rakib *

University of Nottingham, Nottingham, UK
(nza,bsl,hnn,rza)@cs.nott.ac.uk

Abstract. We present a framework for the automated verification of time and communication requirements in systems of distributed rule-based reasoning agents which allows us to determine how many rule-firing cycles are required to solve the problem, how many messages must be exchanged, and the trade-offs between the time and communication resources. We extend CTL* with belief and communication modalities to express bounds on the number of messages the agents can exchange. The resulting logic, $\mathcal{L}_{CRB}$, can be used to express both bounds on time and on communication. We provide an axiomatisation of the logic and prove that it is sound and complete. Using a synthetic but realistic example system of rule-based reasoning agents which allows the size of the problem and the distribution of knowledge among the reasoners to be varied, we show the Mocha model checker [7] can be used to encode and verify properties of systems of distributed rule-based agents. We describe the encoding and report results of model checking experiments which show that even simple systems have rich patterns of trade-offs between time and communication bounds.

1 Introduction

A key application of multi-agent systems research is distributed problem solving. Distributed approaches to problem solving allow groups of agents to collaborate to solve problems which no single agent could solve alone (e.g., because no single agent has all the information necessary to solve the problem), and/or to solve problems more effectively, e.g., in less time than a single agent. For a given problem and system of reasoning agents, many different solution strategies may be possible, each involving different commitments of computational resources and communication by each agent. 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 example, an agent may be unable to commit more than a given portion of its available computational resources or its available communication bandwidth to a particular problem. For a given system of agents with specified inferential abilities and resource bounds it may not be clear whether a particular problem can be solved at

* This work was supported by the Engineering and Physical Sciences Research Council [grant number EP/E031226].

all, or, if it can, what computational and communication resources must be devoted to its solution by each agent. For example, we may wish to know 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.

There has been considerable work in the agent literature on distributed problem solving in general (for example, [1–4]) and on distributed reasoning in particular ([5, 6]). Much of this work analyses the time and communication complexity of distributed reasoning algorithms. However, while we have upper (and some lower) bounds on time requirements for reasoning in distributed systems, possible trade-offs between resources such as time and communication are less clear. In previous work, e.g., [9–11, 8] we have investigated time vs. memory trade-offs for single reasoners, and in [12], we investigated resource requirements for time, memory and communication for systems of distributed resolution reasoners.

In this paper, we focus on a more detailed investigation of time and communication trade-offs for *rule-based reasoners*. We present a framework for the automated verification of time and communication requirements in systems of distributed rule-based reasoning agents. We extend CTL* with belief and communication modalities to express bounds on the number of messages the agents can exchange. Communication modalities are a novel logical concept (the only related work we are aware of is [8] which introduced nullary modalities for expressing the number of formulas in agent's memory), and considerably simplify the logic expressing communication bounds presented in [12]. The resulting logic, $\mathcal{L}_{CRB}$, can be used to express both bounds on time and on communication. We provide an axiomatisation of the logic and prove that it is sound and complete. Using $\mathcal{L}_{CRB}$ to specify bounds on the number of messages the agents can exchange, we can investigate trade-offs between time and communication resources, and we show how the Mocha model checker [7] can be used to encode and verify properties of such systems.

The structure of the paper is as follows. In section 2 we describe systems of communicating rule-based reasoners that we want to verify. In section 3 we introduce the epistemic logic $\mathcal{L}_{CRB}$. We describe the Mocha encoding of the transition systems which are models of the logic in section 4. Model-checking experiments are described in section 5 and we conclude in section 6.

2 Systems of communicating rule-based reasoners

In this section, we describe the systems of communicating rule-based agents which we investigate.

The system consists of n_A agents, where $n_A \geq 1$. We will assume that each agent has a number in $\{1, \ldots, n_A\}$, and use variables i and j over $\{1, \ldots, n_A\}$ to refer to agents. Each agent has a *program*, consisting of propositional Horn clause rules, and a working memory, which contains facts (propositions).[1] If an agent i has a rule

[1] The restriction to propositional rules is not a very drastic assumption: if the rules do not contain functional symbols and we can assume a fixed finite set of constant symbols, then any set of first-order Horn clauses and facts can be encoded as propositional formulas.

$A_1, \ldots, A_n \rightarrow B$, the facts $A_1, \ldots, A_n$ are in the agent's working memory and B is not in the agent's working memory in state s, then the agent can fire the rule which adds B to the agent's working memory in the successor state s'.

Time	Agent 1	Agent 2
t_0	$\{A_1, A_2, A_3, A_4\}$	$\{A_5, A_6, A_7, A_8\}$
operation:	RuleB2	RuleB4
t_1	$\{A_1, A_2, A_3, A_4, B_2\}$	$\{A_5, A_6, A_7, A_8, B_4\}$
operation:	RuleB1	RuleB3
t_2	$\{A_1, A_2, A_3, A_4, B_1, B_2\}$	$\{A_5, A_6, A_7, A_8, B_3, B_4\}$
operation:	RuleC1	RuleC2
t_3	$\{A_1, A_2, A_3, A_4, B_1, B_2, C_1\}$	$\{A_5, A_6, A_7, A_8, B_3, B_4, C_2\}$
operation:	Idle	Copy (C_1 from agent 1)
t_4	$\{A_1, A_2, A_3, A_4, B_1, B_2, C_1\}$	$\{A_5, A_6, A_7, A_8, B_3, B_4, C_1, C_2\}$
operation:	Idle	RuleD1
t_5	$\{A_1, A_2, A_3, A_4, B_1, B_2, C_1\}$	$\{A_5, A_6, A_7, A_8, B_3, B_4, C_1, C_2, D_1\}$

Fig. 1. Example 1

In addition to firing rules, agents can exchange messages regarding their current beliefs. We assume that there is a bound on communication for each agent i which limits agent i to at most $n_C(i)$ messages. Each agent has a communication counter, c_i, which starts at 0 and is not allowed to exceed the value $n_C(i)$. The exchange of information between agents is modelled as an abstract *Copy* operation: if a fact A is in agent i's working memory in state s, A is not in the working memory of agent j, and agent j has not exceeded its communication bound ($c_j < n_C(j)$) then in the successor state s', A can be added to agent j's working memory, and c_j incremented. Intuitively, this corresponds to the following operations rolled into one: j asking i for A, and i sending A to j. This is guaranteed to succeed and takes one tick of system time. The only agent which pays the communication cost is j. These assumptions are made for simplicity; it is straightforward to modify our definition of communication so that the 'cost' of communication is paid by both agents, communication takes more than one tick of time, and communication is non-deterministic. An agent can also perform an Idle operation (do nothing).

A problem is considered to be solved if one of the agents has derived the goal. The time taken to solve the problem is taken to be the total number of steps by the whole system (agents firing their rules or copying facts in parallel, at most one operation executed by each agent at every step). The communication cost for each agent is the value of communication counter for that agent.

As an example, consider a system of two agents, 1 and 2. The agents share the same set of rules:

RuleB1 $A_1, A_2 \rightarrow B_1$
RuleB2 $A_3, A_4 \rightarrow B_2$
RuleB3 $A_5, A_6 \rightarrow B_3$
RuleB4 $A_7, A_8 \rightarrow B_4$

RuleC1 $B_1, B_2 \rightarrow C_1$
RuleC2 $B_3, B_4 \rightarrow C_2$
RuleD1 $C_1, C_2 \rightarrow D_1$

The goal is to derive D_1. Figure 1 gives a simple example of a run of the system starting from a state where agent 1 has A_1, A_2, A_3 and A_4 in its working memory, and agent 2 has A_5, A_6, A_7, A_8. In this example, the agents require one communication and five time steps to derive the goal. (In fact, this is an optimal use of resources for this problem, as verified using Mocha, see section 5).

Throughout the paper, we will use variations on this synthetic 'binary tree' problem, with A_is being the leaves and the goal formula being the root of the tree, as examples (see Figure 2). We vary the number of rules and the distribution of 'leaf' facts be-

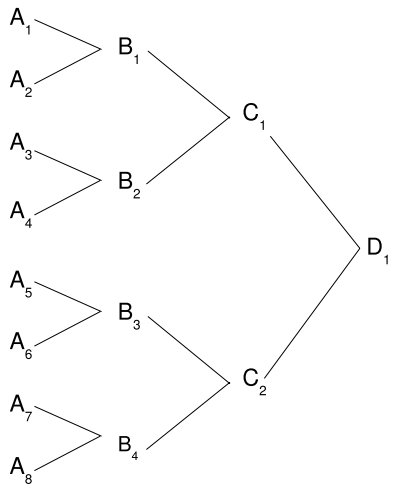

Fig. 2. Binary tree example

tween the agents. For example, a larger system can be generated using 16 'leaf' facts $A_1, \ldots, A_{16}$, adding extra rules to derive B_5 from A_9 and A_{10}, etc., and the new goal E_1 derivable from D_1 and D_2. We will refer to it as '16 leaf example'. Similarly, we will consider systems with 32, 64, 128 etc. leaf facts. We have chosen this form of example because it is typical of distributed reasoning problems and can be easily parameterised by the number of leaf facts and the distribution of facts to the agents.

3 Extending CTL* with belief operators and communication counters

In this section we introduce the formal models of the systems informally described in the previous section. Essentially, they correspond to infinite tree structures (representing branching time), where each state consists of the states of the agents (and the state of each agent corresponds to the contents of its working memory and a record of the number of copy actions it has performed). Transitions between states correspond to all agents executing one of their possible transitions in parallel, where possible transitions include applicable rule firings, copy actions, or idling.

Such structures can be directly encoded in Mocha, by encoding the contents of each agent's memory as boolean variables and the communication counter as an enumeration type variable, as described in section 4. However, this does not give us a precise logical description of systems of distributed rule-based reasoners and an appropriate logical language to describe such systems. We have therefore developed a language in which we can express the properties of the system, including agent's beliefs and communication bounds. This language is an extension of CTL*, and contains a belief operator for each agent and communication modalities. We provide an axiomatisation of the tree structures described above in this logical language. This gives us a precise way of reasoning about resource bounds in the resulting logic, $\mathcal{L}_{CRB}$. In particular we can reason about the interaction of temporal, belief and communication modalities, and the logical properties of communication modalities. It also provides us with a high-level specification language which can be translated in CTL* (in fact, all the properties of interest are expressible in CTL, but for technical reasons we based our axiomatisation on CTL*).

We begin by defining an internal language for each agent. This language includes all possible formulas that the agent can store in its working memory. Let $\mathcal{A} = \{1, \ldots, n_A\}$ be the set of all agents, and P a finite common alphabet of facts. Let Π be a finite set of rules of the form $p_1, \ldots, p_n \rightarrow p$, where $n \geq 0$, $p_i, p \in P$ for all $i \in \{1, \ldots, n\}$ and $p_i \neq p_j$ for all $i \neq j$. For convenience, we use the notation $pre(\rho)$ where $\rho \in \Pi$ for the set of premises of ρ and $con(\rho)$ for the conclusion of ρ. For example, if $\rho = p_1, \ldots, p_n \rightarrow p$, then $pre(\rho) = \{p_1, \ldots, p_n\}$ and $con(\rho) = p$. The internal language IL, then, includes all the facts $p \in P$ and rules $\rho \in \Pi$. We denote the set of all formulas of IL by $\Omega = P \cup \Pi$. Note that Ω is finite.

The syntax of $\mathcal{L}_{CRB}$ includes the temporal operators of CTL^* and is defined inductively as follows:

- $\top$ (tautology) and $start$ (a propositional variable which is only true at the initial moment of time) are well-formed formulas (wff) of $\mathcal{L}_{CRB}$,
- $cp_i^{=n}$ (which states that the value of agent i's communication counter is n) is a wff of $\mathcal{L}_{CRB}$ for all $n \in \{0, \ldots, n_C(i)\}$ and $i \in \mathcal{A}$,
- $B_i p$ (agent i believes p) and $B_i \rho$ (agent i believes ρ) are wffs of $\mathcal{L}_{CRB}$ for any $p \in P$, $\rho \in \Pi$ and $i \in \mathcal{A}$,
- If φ and ψ are wffs of $\mathcal{L}_{CRB}$, then so are $\neg\varphi$ and $\varphi \wedge \psi$,
- If φ and ψ are wffs of $\mathcal{L}_{CRB}$, then so are $X\varphi$ (in the next state φ), $\varphi U \psi$ (φ holds until ψ), $A\varphi$ (on all paths φ).

Other classical abbreviations for $\bot$, $\vee$, $\rightarrow$ and $\leftrightarrow$, and temporal operations: $F\varphi \equiv \top U\varphi$ (at some point in the future φ) and $G\varphi \equiv \neg F\neg\varphi$ (at all points in the future φ), and $E\varphi \equiv \neg A\neg\varphi$ (on some path φ) are defined as usual. For convenience, we also introduce the following abbreviations: $CP_i = \{cp_i^{=n} \mid n = \{0, \ldots, n_C(i)\}\}$ and $CP = \bigcup_{i\in\mathcal{A}} CP_i$.

The semantics of $\mathcal{L}_{CRB}$ is defined by $\mathcal{L}_{CRB}$ transition systems which are based on ω-tree structures. Let (T, R) be a pair where T is a set and R is a binary relation on T. (T, R) is a ω-tree frame iff the following conditions are satisfied.

1. T is a non-empty set.
2. R is total, i.e. for all $t \in T$, there exists $s \in T$ such that tRs.
3. Let $<$ be the strict transitive closure of R, namely $\{(s, t) \in T \times T \mid \exists n \geq 0, t_0 = s, .., t_n = t \in T$ such that $t_i R t_{i+1} \forall i = 0, \ldots, n-1\}$.
4. For all $t \in T$, the past $\{s \in T \mid s < t\}$ is linearly ordered by $<$.
5. There is a smallest element called the root, which is denoted by t_0.
6. Each maximal linearly $<$- ordered subset of T is order-isomorphic to the natural numbers.

A branch of (T, R) is an ω-sequence $(t_0, t_1, \ldots)$ such that t_0 is the root and $t_i R t_{i+1}$ for all $i \geq 0$. We denote $B(T, R)$ to be the set of all branches of (T, R). For a branch $\sigma \in B(T, R)$, σ_i denotes the element t_i of σ and $\sigma_{\leq i}$ is the prefix $(t_0, t_1, \ldots, t_i)$ of σ.

A $\mathcal{L}_{CRB}$ transition system M is defined as a triple (T, R, V) where:

- (T, R) is a ω-tree frame,
- $V : T \times \mathcal{A} \rightarrow \wp(\Omega \cup CP)$ such that for all $s \in T$ and $i \in \mathcal{A}$: $V(s, i) = Q \cup \{cp_i^{=n}\}$ for some $Q \in \wp(\Omega)$ and $cp_i^{=n} \in CP_i$. We denote $V^*(s, i) = V(s, i) \setminus CP_i$.

The truth of a $\mathcal{L}_{CRB}$ formula at a point n of a path $\sigma \in B(T, R)$ is defined inductively as follows:

- $M, \sigma, n \models \top$,
- $M, \sigma, n \models start$ iff $n = 0$,
- $M, \sigma, n \models B_i\alpha$ iff $\alpha \in V(s, i)$,
- $M, \sigma, n \models cp_i^{=m}$ iff $cp_i^{=m} \in V(s, i)$,
- $M, \sigma, n \models \neg\varphi$ iff $M, \sigma, n \not\models \varphi$,
- $M, \sigma, n \models \varphi \wedge \psi$ iff $M, \sigma, n \models \varphi$ and $M, \sigma, n \models \psi$,
- $M, \sigma, n \models X\varphi$ iff $M, \sigma, n + 1 \models \varphi$,
- $M, \sigma, n \models \varphi U \psi$ iff $\exists m \geq n$ such that $\forall k \in [n, m)$ $M, \sigma, m \models \varphi$ and $M, \sigma, m \models \psi$,
- $M, \sigma, n \models A\varphi$ iff $\forall \sigma' \in B(T, R)$ such that $\sigma'_{\leq n} = \sigma_{\leq n}$, $M, \sigma', n \models \varphi$.

The models of $\mathcal{L}_{CRB}$ satisfy a set of constraints on the accessibility relation. Intuitively, each R is composed of an n_A-tuple of agents' actions performed in parallel. We will next define precisely the set of actions that each agent can perform. They are $Rule_{i,\rho}$, $Copy_{i,\alpha}$ and $Idle_i$ where $i \in \mathcal{A}$, $\rho \in \Pi$ and $\alpha \in \Omega$. $Rule_{i,\rho}$ is the action of an agent i firing ρ; $Copy_{i,\alpha}$ the action of copying α from another agent and $Idle_i$ is when agent i does nothing and moves to the next state.

We set constraints on the set of models such that the two following conditions are satisfied: (i) any transition between two states of the model corresponds to the effect of

actions done by all agents in $\mathcal{A}$ and (ii) for any action of an agent in $\mathcal{A}$ that is applicable at a state s of the model, then there exists another state s' and a transition from s to s' which corresponds to the effect of the action. To formalise those two conditions, we have the following definitions.

Definition 1. *Let (T, R, V) be a tree model. The set of effective transitions R_a for an action a is defined as a subset of R and satisfies the following conditions, for all $(s,t) \in R$*

1. $(s,t) \in R_{Rule_{i,\rho}}$ *iff* $\rho \in V(s,i)$, $V(s,i) \supseteq pre(\rho)$, $con(\rho) \notin V(s,i)$ *and* $V(t,i) = V(s,i) \cup \{con(\rho)\}$. This condition says that s and t are connected by agent i's *rule-fired* transition if the following is true: ρ is a rule of i, $V(s,i)$ contains all premises of ρ but not its conclusion and the conclusion of ρ is added to the next state t of i.
2. $(s,t) \in R_{Copy_{i,\alpha}}$ *iff* $\alpha \in V(s,j)$ *where some* $j \in \mathcal{A}$ *and* $j \neq i$, $cp_i^{=n} \in V(s,i)$ *such that* $n < n_C$, $\alpha \notin V(s,i)$ *and* $V(t,i) = V(s,i) \setminus \{cp_i^{=n}\} \cup \{cp_i^{=n+1}\} \cup \{\alpha\}$. In this condition, s and t are connected by a $Copy$ transition of agent i iff i has copied so far at most $n_C(i) - 1$ messages from other agents, at s, i does not have α in its working memory while another agent j does and at the next state t, α is added into the working memory of i and its message counter is increased by one.
3. $(s,t) \in R_{Idle_i}$ *iff* $V(t,i) = V(s,i)$. The $Idle$ transition does not change the state.

Below, we specify when an action is applicable. Note that we only enable deriving a formula if this formula is not already in the agent's working memory.

Definition 2. *Let (T, R, V) be a tree model. The set $Act_{s,i}$ of applicable actions that an agent i can perform at a state $s \in T$ is defined as follows:*

1. $Rule_{i,\rho} \in Act_{s,i}$ *iff* $\rho \in V(s,i)$, $pre(\rho) \subseteq V(s,i)$ *and* $con(\rho) \notin V(s,i)$.
2. $Copy_{i,\alpha} \in Act_{s,i}$ *iff* $n < n_C(i)$ *where* n *is from* $cp_i^{=n} \in V(s,i)$, $\alpha \notin V(s,i)$, $\alpha \in V(s,j)$ *for some* $j \in \mathcal{A}$.
3. *It is always the case that* $Idle_i \in Act_{s,i}$.

Finally, the definition of the set of models corresponding to a system of rule-based reasoners is given below:

Definition 3. *$M(n_C)$ is the set of models (T, R, V) which satisfies the following conditions:*

1. $cp_i^{=0} \in V(t_0, i)$ *where t_0 is the root of (T, R) for all $i \in \mathcal{A}$.*
2. $R = \bigcup_{\forall a} R_a$.
3. *For all $s \in T$, $a_i \in Act_{s,i}$, there exists $t \in T$ such that $(s,t) \in R_{a_i}$ for all $i \in \mathcal{A}$.*

Below are some abbreviations which will be used in the axiomatisation:

- $ByRule_i(p,n) = \neg B_i p \wedge cp_i^{=n} \wedge \bigvee_{\rho \in \Pi \wedge con(\rho)=p} (B_i \rho \wedge \bigwedge_{p \in pre(\rho)} B_i p)$. This formula describes the state before the agent comes to believe formula p by the $Rule$ transition. n is the value of i's communication counter.
- $ByCopy_i(\alpha, n) = \neg B_i \alpha \wedge B_j \alpha' \wedge cp_i^{=n-1}$.

Let us now introduce the axiomatisation systems.

A1. All axioms and inference rules of CTL^* [13].

A2. $B_i\rho \wedge \bigwedge_{p\in pre(\rho)} B_i p \wedge cp_i^{=n} \wedge \neg B_i con(\rho) \rightarrow EX(B_i con(\rho) \wedge cp_i^{=n})$ for all $\rho \in \Pi$ and $i \in \mathcal{A}$.
Intuitively, this axiom says that it is always possible to make a transition to a state where agent i believes the conclusion of a rule ρ in its working memory. In addition, the communication counter of the agent does not increase.
The next axiom **A3** similarly describes transitions made by copy with communication counter increased).

A3. $cp_i^{=n} \wedge \neg B_i\alpha \wedge B_j\alpha \rightarrow EX(B_i\alpha \wedge cp_i^{=n+1})$ for any $\alpha \in \Omega$, $j \in \mathcal{A}$, $j \neq i$, $n < n_C(i)$.

A4. $EX(B_i\alpha \wedge B_i\beta) \rightarrow B_i\alpha \vee B_i\beta$.
This axiom says that at most one new belief is added in the next state.

A5. $B_i\alpha \rightarrow AXB_i\alpha$ for any $\alpha \in \Omega$.
This axiom says that an agent always believes in what it already believed before.

A6. $EX(B_i\alpha \wedge cp_i^{=n}) \rightarrow B_i\alpha \vee ByRule_i(\alpha, n) \vee ByCopy_i(\alpha, n)$ for any $\alpha \in \cup\Omega$.
This axiom says that a new belief can only be added by one of the valid reasoning actions.

A7. $start \rightarrow cp_i^{=0}$ for all $i \in \mathcal{A}$.
At the start state, the agent has not performed any $Copy$ actions.

A8. $\bigvee_{n=0\ldots n_C} cp_i^{=n}$ for all $i \in \mathcal{A}$.
There is always a number n between 0 and n_C corresponding to the number of $Copy$ actions agent i has performed.

A9. $cp_i^{=n} \rightarrow \neg cp_i^{=n'}$ for all $i \in \mathcal{A}$ and $n' \neq n$.
The number of previous $Copy$ actions by i in each state is unique.

A10. $\varphi \rightarrow EX\varphi$.
It is always possible to make a transition back to the same state (essentially an $Idle$ transition by all agents)

A11. $\bigwedge_{i\in\mathcal{A}} EX(\bigwedge_{\alpha\in Q_i} B_i\alpha \wedge cp_i^{=n_i}) \rightarrow EX \bigwedge_{i\in\mathcal{A}}(\bigwedge_{\alpha\in Q_i} B_i\alpha \wedge cp_i^{=n_i})$ for any $Q_i \subseteq \Omega$.
If each agent i can separately reach a state where it believes formulas in Q_i, then all agents together can reach a state where for each i, agent i believes formulas in Q_i.

Let us now define the logic obtained from the above axiomatisation system.

Definition 4. *$L(n_C)$ is the logic defined by the axiomatisation **A1** - **A11**.*

We have the following result.

Theorem 1. *$L(n_C)$ is sound and complete with respect to $M(n_C)$.*

Proof Sketch As usual, soundness is proved by showing that all axioms are valid and inference rules preserve validity. The proofs for axioms and rules included in **A1** are given in [13]. The validity of axioms **A2**-**A11** can be proved using the properties of models in $M(n_C)$. In the following, we provide the proof for **A2**. The proofs for other axioms are similar.

Let $M = (T, V, R) \in M(n_C)$, $\sigma \in B(T, R)$ and $n \geq 0$. Assume that $M, \sigma, n \models B_i\rho \wedge \bigwedge_{p \in pre(\rho)} B_i p \wedge cp_i^{=m} \wedge \neg B_i con(\rho)$ for some $\rho \in \Pi$. Then $p \in V(\sigma_n, i)$ for all $p \in pre(\rho)$ and $con(\rho) \notin V(\sigma_n, i)$. This means that $Rule_{i,\rho} \in Act_{\sigma_n, i}$. According to the definition of $M(n_C)$, there exists a $t' \in T$ such that $\sigma_n R t'$ and $V(t', i) = V(\sigma_n, i) \cup \{con(\rho)\}$. Let σ' be a branch in $B(T, R)$ such that $\sigma'_{\leq n} = \sigma_{\leq n}$ and $\sigma'_{n+1} = t'$. Then we have that $M, \sigma', n+1 \models B_i con(\rho) \wedge cp_i^{=m}$. It is obvious, then, that $M, \sigma, n \models EX(B_i con(\rho) \wedge cp_i^{=m})$.

Completeness is shown by constructing a tree model for a consistent formula φ. The construction is the one introduced in [13]. Since the initial state of all agents does not restrict the set of formulas they may derive in the future, for simplicity we conjunctively add to φ a tautology that contains all the potentially necessary formulas and message counters, in order to have enough sub-formulas for the construction. We construct a model $M = (T, R, V)$ for

$$\varphi' = \varphi \wedge \bigwedge_{\alpha \in \Omega} (XB_i\alpha \vee \neg XB_i\alpha) \wedge \bigwedge_{n=0 \ldots n_C, i \in \mathcal{A}} (Xcp_i^{=n} \vee \neg Xcp_i^{=n})$$

We then prove that M is in $M(n_C)$ by showing that it satisfies all properties listed in Definition 3.

By axiom **A8**, it is straightforward that at a state t of M there exists $cp_i^{=n}$ for some $n \in \{0, \ldots, n_C\}$ and any $i \in \mathcal{A}$ such that $cp_i^{=n} \in V(t, i)$. Moreover, **A9** ensures that one and only one such n can be presented in $V(t, i)$.

At the root t_0 of (T, R), the construction of the model implies that there exists a MCS[2] Γ_0 such that $\Gamma_0 \supseteq V(t_0, i)$ and $start \in \Gamma_0$. By axiom **A7**, it is trivial that $cp_i^{=0} \in V(t_0, i)$.

We then need to prove that at a state t of M, if an action a_i of agent $i \in \mathcal{A}$ is applicable, then there exists $t' \in M$ such that tRt' and $V(t', i)$ is the result of $V(t, i)$ after i performs action a_i. The proof is done by induction on the cases of a_i. Let us consider the case when a_i is $Rule_{i,\rho}$ for some $\rho \in \Pi$. Since $Rule_{i,\rho}$ is applicable at t, $con(\rho) \notin V(t, i)$ and $p \in V(t, i)$ for all $p \in pre(\rho)$. Therefore there exists a MCS Γ such that $\Gamma \supseteq V(t, i)$. Then we obtain $\bigwedge_{p \in pre(\rho)} B_i p \wedge cp_i^{=n} \wedge \neg B_i con(\rho) \in \Gamma$ for some $n \in \{0, \ldots, n_C\}$. By axiom **A2** and **MP**[3], $EX(B_i con(\rho) \wedge cp_i^{=n}) \in \Gamma$. Therefore, according to the construction, there exists $t' \in T$ such that tRt' and $V(t', i) \subseteq \Gamma'$ for some Γ' such that $B_i con(\rho) \wedge cp_i^{=n} \in \Gamma'$. Therefore $V(t', i) = V(t, i) \cup \{con(\rho)\}$.

For other cases of a_i, the proofs are similar by using **MP** and axioms **A3** and axiom **A10**. Then, axiom **A11** enables us to show that, for any tuple of actions $(a_1, \ldots, a_{n_A})$ such that all a_i are applicable at a state t of M, there exists $t' \in T$ such that $V(t', i)$ is the result of performing a_i at t for all $i \in \mathcal{A}$. The proof is similar to that above, except that each case under consideration is a tuple of actions, and by using axiom **A11** and **MP**.

Finally, we prove that for any $t' \in T$ such that tRt', there exists a tuple of actions $(a_1, \ldots, a_{n_A})$ and $V(t', i)$ is the result of $V(t, i)$ when agent i performs a_i for all $i \in \mathcal{A}$.

[2] MCS stands for *maximally consistent set*.

[3] **MP** stands for Modus Ponens.

By axioms **A4** and **A5**, $V^*(t', i)$ is different from $V^*(t, i)$ by at most one formula added and no formula removed. If no formula is added (and no formula is removed), we set a_i to be $Idle_i$. Let us now consider the case where a formula α is added. By axiom **A6**, if $cp_i^{=n} \in V(t, i)$ for some $n \in \{0, \ldots, n_C\}$ then either cp_i^n or $cp_i^{n+1} \in V(t', i)$. If $cp_i^n \in V(t', i)$ then set a_i to be $Rule_{i,\rho}$ for some $\rho \in V(t, i)$ such that $\alpha = con(\rho)$ (this must happen according to **A6**). If $cp_i^{n+1} \in V(t', i)$ then set a_i to be $Copy_{i,\alpha}$ (this must happen according to **A6** that $\alpha \in V(t, j)$ for some $j \in \mathcal{A}$). Thereby, we have proved the existence of the tuple $(a_1, \ldots, a_{n_A})$ for tRt'. Then, we conclude that $M \in M(n_C)$. □

4 Mocha encoding

It is straightforward to encode a $\mathcal{L}_{CRB}$ model 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], due to the ease with which we can specify concurrently executing agents in *reactive modules*, the description language used by Mocha.

The state of the system is described by a set of state variables and each system state corresponds to an assignment of values to the variables. The presence or absence of each fact in the working memory of an agent is represented by a boolean state variable $a_i A_j$ which represents the fact that agent i believes fact A_j. The initial values of these variables determines the initial distribution of facts between agents.[4] In the experiments reported below (which used the binary tree example, see Figure 2), all derived (non-leaf) variables were initialised to *false*, and only the allocation of leaves to each agent was varied.

The actions of firing a rule, copying a fact from another agent and idling were encoded as a Mocha *atom* which describe the initial condition and transition relation for a group of related state variables. Inference is implemented by marking the consequent of a rule as present in working memory at the next cycle if all of the antecedents of the rule are present in working memory at the current cycle. A rule is only enabled if its consequent is not already present in working memory at the current cycle. Communication is implemented by copying the value representing the presence of a fact in the working memory of another agent at the current cycle to the corresponding state variable in the agent performing the copy at the next cycle. Copying is only enabled if the fact to be copied is not already in the working memory of the agent performing the copy. In the experiments, we assumed that all rules are believed by all agents in the initial state, and did not implement copying rules. However, this can be done in a straightforward way by adding an extra boolean variable to the premises of each rule, and implementing copying a rule as copying this variable. To express the communication bound, we use a counter for each agent which is incremented each time a copy action is performed

[4] We can also leave the initial allocation of facts undetermined, and allow the model checker to find an allocation which satisfies some property, e.g., that there is a proof which takes less than 7 steps. However for the experiments reported here, we specified the initial assignment of facts to agents.

by the agent. To allow an agent to idle at any cycle, the atoms which update working memory in each agent are declared to be *lazy*.

The evolution of the system's state is described by an initial round followed by an infinite sequence of update rounds. The variables are initialised to their initial values in the initial round and new values are assigned to the variables in the subsequent update rounds. At each update round, Mocha non-deterministically chooses between the enabled rules and copy operations and idling.

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 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 $EX^n\, tr(B_i\alpha)$, where EX^n is EX repeated n times, and $tr(B_i\alpha)$ is a state variable encoding of the fact that α is present in the agent's working memory (e.g. $tr(B_i\alpha) = a_iA_j$ if $\alpha = A_j$). To obtain the actual derivation, we can verify an invariant which states that $tr(B_i\alpha)$ is never true, and use the counterexample trace to show how the system reaches the state where α is proved. To bound the number of messages used, we can include a bound on the value of the message counter of one or more agents in the property to be verified. For example, $EX^n\, (tr(B_i\alpha) \wedge tr(cp_i^{=0} \vee cp_i^{=1}))$, where $tr(cp_i^{=0} \vee cp_i^{=1})$ is translated to the statement $a_i\_counter < 2$, bounds the number of messages used by agent i to be at most 1. The encoding of the models and translation of the properties from $\mathcal{L}_{CRB}$ into the Mocha specification language does not involve a significant overhead in comparison to other model-checking problems.

5 Experimental results

In this section we describe the results of experiments for different sizes of the binary tree example (see Figure 2) and different distributions of leaves between the agents. The experiments were designed to investigate trade-offs between the number of steps and the number of messages exchanged (a shorter derivation with more messages or a longer derivation with fewer messages).

First, as a 'base case' and also to get an idea of the size of examples which can be model-checked in a reasonable time using our Mocha encoding, we ran experiments with just one agent, varying the size of the tree. The results are shown in Figure 3. As one would expect, the number of steps equals to the total number of rules in the example. While for our binary tree example the results are unsurprising, in a less uniform rule-based system such a result may be difficult to establish by a simple inspection of rules.

We then investigated different distributions of leaf facts between the agents. Figure 4 shows the number of derivation steps and the number of messages for each agent for varying distributions of 8 leaves. Note that there are several optimal (non-dominated) derivations for the same initial distribution of leaves between the agents. For example, when agent 1 has all the leaves apart from A_8, and agent 2 has A_8, the obvious solution

Case	# leaves	# steps
1.	8	7
2.	16	15
3.	32	31
4.	64	63
5.	128	127

Fig. 3. Resource requirements for one agent

Case	Agent 1	Agent 2	# steps	# messages agent 1	# messages agent 2
1.	$A_1 - A_8$		7	-	-
2.	$A_1 - A_7$	A_8	6	0	3
3.	$A_1 - A_7$	A_8	6	1	2
4.	$A_1 - A_7$	A_8	7	1	1
5.	$A_1 - A_7$	A_8	8	1	0
6.	$A_1 - A_6$	A_7, A_8	6	0	2
7.	$A_1 - A_6$	A_7, A_8	6	1	1
8.	$A_1 - A_6$	A_7, A_8	7	1	0
9.	$A_1 - A_4$	$A_5 - A_8$	5	1	0
10.	A_1, A_3, A_5, A_7	A_2, A_4, A_6, A_8	7	2	3
11.	A_1, A_3, A_5, A_7	A_2, A_4, A_6, A_8	11	0	4

Fig. 4. Resource requirements for optimal derivation in 8 leaves cases

is case 5, where agent 1 copies A_8 from agent 2, and then derives the goal in 7 steps, as in case 1. This derivation requires 8 time steps and one message. However, the agents can solve the problem in fewer steps by exchanging more messages. For example, case 2 describes the situation when agent 2 copies A_7 from agent 1, while agent 1 derives B_3 (step 1). Then agent 2 derives B_4 while agent 1 derives B_2 (step 2). Then agent 2 copies B_3 from agent 1, while agent 1 derives B_1 (step 3). At the next step agent 1 derives C_1 and agent 2 derives C_2 (step 4). Then agent 2 copies C_1 from agent 1 (step 5) and agent 1 idles; finally at step 6 agent 2 derives D_1. The effect of the bound on messages varies with the distribution, as can be seen in cases 10 and 11: if agent 1 has all the odd leaves and agent 2 all the even leaves, then to derive the goal either requires 7 steps and 5 messages, or 11 steps and 4 messages.

Similar trade-offs are apparent for a problem with 16 leaves, as shown in Figure 5. However in this case there are a larger number of possible distributions of leaves, and, in general, more trade-offs for each distribution. For example, when one of the agents has all the leaves but one, we again have the obvious solution where agent 1 copies the missing leaf and derives the goal on its own, which takes 16 steps and 1 message (case 7). In addition there are 15, 14, 13 and 12 step derivations, where the shorter the derivation the more messages the agents have to exchange (cases 2-7). We also see interesting trade-offs when agent 2 has two leaves (cases 8-13) or four leaves in the same subtree (cases 14-17). When agent 1 has 3 leaves in each subtree and agent 4 the fourth leaf in each subtree, there is again an obvious derivation in which agent 1 copies the 4 missing leaves and completes the derivation in 19 steps and 4 copy operations, and a more interesting one which takes 13 steps and the agents exchange more messages

(agent 2 copies 3 leaves to complete a part of the proof, and then copies variables from higher up in the tree). The difference is also more marked in the 'odd and even' case (cases 20 and 21), where agent 1 has all the odd leaves and agent 2 all the even leaves, where increasing the message bound by 1 reduces the length of the proof by 10 steps.

Although these examples are very simple, they point to the possibility of complex trade-offs between time and communication bounds in systems of distributed reasoning agents. For more complex examples, we would anticipate that such trade-offs would be harder to predict *a priori*, and our framework would be of correspondingly greater utility.

Case	Agent 1	Agent 2	# steps	# copy 1	# copy 2
1.	$A_1 - A_{16}$		15	-	-
2.	$A_1 - A_{15}$	A_{16}	12	0	6
3.	$A_1 - A_{15}$	A_{16}	12	1	4
4.	$A_1 - A_{15}$	A_{16}	13	1	3
5.	$A_1 - A_{15}$	A_{16}	14	1	2
6.	$A_1 - A_{15}$	A_{16}	15	1	1
7.	$A_1 - A_{15}$	A_{16}	16	1	0
8.	$A_1 - A_{14}$	A_{15}, A_{16}	11	0	5
9.	$A_1 - A_{14}$	A_{15}, A_{16}	11	1	4
10.	$A_1 - A_{14}$	A_{15}, A_{16}	12	1	3
11.	$A_1 - A_{14}$	A_{15}, A_{16}	13	1	2
12.	$A_1 - A_{14}$	A_{15}, A_{16}	14	1	1
13.	$A_1 - A_{14}$	A_{15}, A_{16}	15	1	0
14.	$A_1 - A_{12}$	$A_{13}, A_{14}, A_{15}, A_{16}$	11	0	4
15.	$A_1 - A_{12}$	$A_{13}, A_{14}, A_{15}, A_{16}$	11	1	2
16.	$A_1 - A_{12}$	$A_{13}, A_{14}, A_{15}, A_{16}$	12	1	1
17.	$A_1 - A_{12}$	$A_{13}, A_{14}, A_{15}, A_{16}$	13	1	0
18.	$A_1 - A_3, A_5 - A_7, A_9 - A_{11}, A_{13} - A_{15}$	A_4, A_8, A_{12}, A_{16}	13	2	6
19.	$A_1 - A_3, A_5 - A_7, A_9 - A_{11}, A_{13} - A_{15}$	A_4, A_8, A_{12}, A_{16}	19	4	0
20.	$A_1, A_3, A_5, A_7, A_9, A_{11}, A_{13}, A_{15}$	$A_2, A_4, A_6, A_8, A_{12}, A_{14}, A_{16}$	13	4	5
21.	$A_1, A_3, A_5, A_7, A_9, A_{11}, A_{13}, A_{15}$	$A_2, A_4, A_6, A_8, A_{12}, A_{14}, A_{16}$	23	0	8

Fig. 5. Resource requirements for optimal derivation in 16 leaves cases

6 Conclusions

In this paper, we proposed an approach to modelling and verifying resource requirements of distributed rule-based reasoners. We showed how to reason about time and communication bounds in such systems, and defined a sound and complete logic, $\mathcal{L}_{CRB}$, in which such reasoning can be expressed. The models of the logic can be encoded as an input to a standard model-checker such as Mocha and properties of interest translated into CTL, without a significant overhead in comparison to other model-checking problems. We described results of some experiments on a synthetic example which show

interesting trade-offs between time required by the agents to solve the problem and the number of messages they need to exchange.

References

1. Faltings, B., Yokoo, M.: Introduction: Special issue on distributed constraint satisfaction. Artificial Intelligence **161** (2005) 1–5
2. Jung, H., Tambe, M.: On communication in solving distributed constraint satisfaction problems. In Pechoucek, M., Petta, P., Varga, L.Z., eds.: Multi-Agent Systems and Applications IV, 4th International Central and Eastern European Conference on Multi-Agent Systems, CEEMAS 2005, Budapest, Hungary, September 15-17, 2005, Proceedings. Volume 3690 of Lecture Notes in Computer Science., Springer (2005) 418–429
3. Provan, G.M.: A model-based diagnosis framework for distributed embedded systems. In Fensel, D., Giunchiglia, F., McGuinness, D.L., Williams, M.A., eds.: Proceedings of the 8th International Conference on Principles and Knowledge Representation and Reasoning (KR-02), Toulouse, France, April 22-25, 2002, Morgan Kaufmann (2002) 341–352
4. Wooldridge, M., Dunne, P.E.: On the computational complexity of coalitional resource games. Artif. Intell. **170** (2006) 835–871
5. Adjiman, P., Chatalic, P., Goasdoué, F., Rousset, M.C., Simon, L.: Distributed reasoning in a peer-to-peer setting. In de Mántaras, R.L., Saitta, L., eds.: Proceedings of the 16th European Conference on Artificial Intelligence, ECAI'2004, including Prestigious Applicants of Intelligent Systems, PAIS 2004, Valencia, Spain, August 22-27, 2004, IOS Press (2004) 945–946
6. Amir, E., McIlraith, S.A.: Partition-based logical reasoning for first-order and propositional theories. Artificial Intelligence **162** (2005) 49–88
7. Alur, R., Henzinger, T.A., Mang, F.Y.C., Qadeer, S., Rajamani, S.K., Tasiran, S.: MOCHA: Modularity in model checking. In: Computer Aided Verification. (1998) 521–525
8. Ågotnes, T., Alechina, N.: Knowing minimum/maximum n formulae. In Brewka, G., Coradeschi, S., Perini, A., Traverso, P., eds.: Proceedings of the 17th European Conference on Artificial Intelligence (ECAI 2006), IOS Press (2006) 317–321
9. Albore, A., Alechina, N., Bertoli, P., Ghidini, C., Logan, B., Serafini, L.: Model-checking memory requirements of resource-bounded reasoners. In: Proceedings of the Twenty-First National Conference on Artificial Intelligence (AAAI 2006), AAAI Press (2006) 213–218
10. Alechina, N., Bertoli, P., Ghidini, C., Jago, M., Logan, B., Serafini, L.: Verifying space and time requirements for resource-bounded age nts. In Stone, P., Weiss, G., eds.: Proceedings of the Fifth International Joint Conference on Aut onomous Agents and Multi-Agent Systems (AAMAS 2006), Hakodate, Japan, IEEE Press (2006) 217–219
11. Alechina, N., Bertoli, P., Ghidini, C., Jago, M., Logan, B., Serafini, L.: Verifying space and time requirements for resource-bounded agents. In Edelkamp, S., Lomuscio, A., eds.: Proceedings of the Fourth Workshop on Model Checking and Artif icial Intelligence (MoChArt-2006). (2006) 16–30
12. Alechina, N., Logan, B., Nga, N.H., Rakib, A.: Verifying time, memory and communication bounds in systems of reasoning agents. In Padgham, Parkes, Muller, Parsons, eds.: Proceedings of the Seventh International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2008), Estoril, Portugal (2008)
13. Reynolds, M.: An axiomatization of full computation tree logic. J. Symb. Log. **66** (2001) 1011–1057