Verification of Resource Requirements of Distributed Reasoning Agents [1]

Natasha Alechina, Brian Logan, Nguyen Hoang Nga, Abdur Rakib[2]

School of Computer Science
University of Nottingham
Nottingham 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.

Keywords: Formalisms, logics, resource bounds, model checking.

1 Introduction

Distributed problem solving involves multiple agents combining their knowledge and computational resources to solve problems which no single agent could solve alone or to solve problems more effectively. For a given problem, different multiagent systems will prefer different solution strategies, depending on the relative costs of computational and communication resources. The tradeoffs between time, memory and communication 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 reasoning 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 agents literature on distributed problem solving in general (see, for example, [8,9,10]) and on distributed reasoning in particular, (see, for example [2,6]). Much of this work analyses the time and communication complexity of distributed reasoning algorithms. In this paper we present a framework for reasoning

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

[2] Email:{nza,bsl,hnn,rza}@cs.nott.ac.uk

This paper is electronically published in
Electronic Notes in Theoretical Computer Science
URL: www.elsevier.nl/locate/entcs

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 time (number of inference steps), memory (number of formulas or symbols) and the number of messages exchanged by the agents, allowing the investigation of tradeoffs between different resources. The framework is based on a novel epistemic logic, *BMCL*, for specifying resource-bounded reasoners. *BMCL* 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. (See [1] for a detailed description of *BMCL* and proofs of its soundness and completeness.) Using simple resolution examples, we show how to encode systems of distributed reasoning agents specified in *BMCL* in a model checker, and verify some example properties.

2 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 [4].[3] 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 [7]. 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.

Our model of communication complexity is based on [11], 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.

[3] We deviate from [4] 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.

3 Verifying Resource Bounds

The logic $BMCL$ allows us to express precisely how the beliefs of a set of resource-bounded agents change over time, and, given a memory and communication bound for each agent, to verify formulas stating that a certain belief will or will not be acquired within a finite number of computation steps.

Model Checker Encoding We have used the Mocha model checker [5] to encode a *BMCL* model and to verify resource bounds for a system of agents which reason using resolution. States of the *BMCL* 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. Let P be the set of propositional variables in K_i. Each cell is represented by a pair of bitvectors, each of length $k = |P|$, representing the positive and negative literals in the clause in some standard order (e.g., lexicographic order). For example, if P 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 values and transition relation for a group of related state variables. Reading a premise simply sets the bitvectors representing an arbitrary cell in agent i's memory to the appropriate values for the clause α. Resolution of two clauses α_1 and α_2 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 of a clause α from agent j to agent i 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 action is disabled.

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 the *BMCL* formula $EF\ B_1\{\}$, where EF means 'some time in the future' and $B_1\{\}$ means 'agent 1 believes the empty clause'.

Using the model checker, we can show that the derivation of the empty clause by a single agent requires a memory bound of 3 and 8 time steps (see Table 1).[4] 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. The tradeoffs between memory, communication and time bounds for two agents are also summarised in Table 1 for symmetric and asymmetric problem distributions. In the symmetric case each agent has all the premises $\sim A_1 \vee \sim A_2$; in the asymmetric case 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 $A_1 \vee \neg A_2$. The entries in the *Memory* column indicate the memory bounds for each agent, and those in the *Communication* column indicate how many times each agent communicated with the other. See [1] for a detailed analysis of the results.

[4] The space required for problems of this form is known to be logarithmic in the number of premises [7].

# 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

4 Discussion and Future Work

In this paper, we sketched a framework for verifying properties of systems of resource-bounded agents and illustrated its application using simple examples. While the examples are simple, they serve to illustrate the interaction between memory, time and communication bounds, and between the resource distribution and the problem distribution. In future work, we plan to enhance our logic to be able to express resource bounds as formulas in the language rather than axioms, allowing agents to reason about each other's resource limitations. We would also like to consider agents reasoning in simple epistemic or description logics.

Acknowledgements

This work was partially supported by EPSRC project number EP/E031226/1.

References

[1] N. Alechina, B. Logan, H. N. Nguyen, and A. Rakib. Verifying time, memory and communication bounds in systems of reasoning agents In *Proc. of the Seventh International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2008)*, Estoril, Portugal.

[2] P. Adjiman, P. Chatalic, F. Goasdoué, M.-C. Rousset, and L. Simon. Distributed reasoning in a peer-to-peer setting. In R. L. de Mántaras and L. Saitta, editors, *Proc. of the 16th European Conference on Artificial Intelligence, ECAI'2004*, pages 945–946. IOS Press, 2004.

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

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

[5] 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.

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

[7] J. L. Esteban and J. Torán. Space bounds for resolution. In *Proc. STACS 99, 16th Annual Symposium on Theoretical Aspects of Computer Science*, volume 1563 of *LNCS*, pages 551–560. Springer, 1999.

[8] B. Faltings and M. Yokoo. Introduction: Special issue on distributed constraint satisfaction. *Artificial Intelligence*, 161(1-2):1–5, 2005.

[9] 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*, volume 3690 of *LNCS*, pages 418–429. Springer, 2005.

[10] 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.

[11] A. C.-C. Yao. Some 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.