DEPARTAMENTO DE CIÊNCIA DA COMPUTAÇÃO

Relatório Técnico

RT-MAC-9411

Towards Efficient Modelling of Distributed Knowledge Using Equational and Order-Sorted Logic

Renata Wassermann
Flávio S.C. da Silva

Junho 94

Towards Efficient Modelling of Distributed Knowledge Using Equational and Order-Sorted Logic

Renata Wassermann, Flávio S. Corrêa da Silva

Instituto de Matemática e Estatística da Universidade de São Paulo - Cid. Universitária "ASO" - PO Box 20570 - 01498-070 São Paulo (SP) - Brazil
email: {renata,fcs}@ime.usp.br

Abstract. Reasoning about multiple interacting agents is important for many areas of research such as distributed computing, artificial intelligence, game theory, decision theory, cognitive science, economics and psychology.
J. Y. Halpern and colleagues [HM90, HF89, HT93] have proposed the use of multiagent epistemic logics to formalise reasoning about multiple agents. In the present paper we extend their proposal to allow *grouping* of agents, and propose a strategy to build efficient resolution-based logic programs for automated reasoning about interacting agents.
Keywords: Automated reasoning, Distributed AI, Distributed Systems, Epistemic Logics.

1 Introduction

Following [HF89, HT93], a *distributed system* in the present paper is identified with a set of possible *runs*, where a *run* describes the way the system behaves over time. For example, if we think of a system consisting of only one process executing a sequential program, a run of this system can be seen as an automaton with a possibly infinite number of *internal states*. An *action* is then a function mapping one state into another. If $s_0, s_1, s_2, ...$ are internal states and $a_0, a_1, a_2, ...$ are actions, a run can be represented as a sequence $s_0 \xrightarrow{a_0} s_1 \xrightarrow{a_1} s_2 \xrightarrow{a_2} ...$

In a more general system with n processes, a *global state* is defined as a tuple $(s_e, s_1, s_2, s_3, ..., s_n)$, in which each s_i is the *local state* of process i and s_e is the state of the environment, which is intended to capture everything that is relevant to the system but cannot be deduced from the local states of individual processes. A run of one such system is a sequence of *global states* and *joint actions*. A *joint action* represents the actions performed by each process simultaneously. We can view a run as a mapping from time to global states[1]: $r(t)$ denotes the global state of the system in run r at instant t. A point in a run is defined as a pair (r, t), where r is a run and t is an instant. The local state of a process at a point (r, t) is given by the i-th position of the vector $r(t)$ (denoted as $r_i(t)$).

One way to reason about a distributed system is by viewing it as a collection of processes with variable knowledge according to actions being executed. In

[1] we consider "time" as a countable sequence of instants $t_0, t_1, t_2, ...$

order to be "fired" or executed, these actions depend on previous knowledge, and they may also modify it. In a certain sense, we are identifying local states of processes with the "knowledge" attained by these processes.

It should be remarked that we are not assuming with this that processes *do* reason. Rather, we are *attributing* knowledge to processes, i. e., we are using an "external" notion of knowledge.

In [HM90] it is defined that a process i knows a fact ψ at a point x (denoted $K_i\psi$) if ψ holds at every point in which the local state of i is the same as at point x. One important requirement ([HM90]) is that $K_i\psi \rightarrow \psi$, i. e., if process i knows a fact ψ then ψ is true. This property is known as the "knowledge axiom" and is used in philosophy to distinguish knowledge from belief.

We may want to reason about the knowledge of a group G of processes, such that $G \subseteq \{1, 2, ..., n\}$. The fact that everyone in G knows ψ is represented by $E_G\psi \equiv \wedge_{i \in G} K_i\psi$.

The statement "everyone in G knows that everyone in G knows that .... everyone in G knows that ψ" where the expression "everyone in G knows that..." appears k times in the sentence is represented by $E^k_G\psi$, defined by: $E^1_G\psi \equiv E_G\psi$ and for all k>1, $E^k_G\psi \equiv E_G E^{k-1}_G\psi$.

A fact ψ is said to be *common knowledge* in G ($C_G\psi$) if $E^k_G\psi$ for all k$\geq$1. Intuitively, this means that the fact ψ is *publicly known* among the elements of group G.

The possibility of using various modal operators that is characteristic of multimodal logics can be useful to reason about knowledge ([DEL92]). Since we are considering systems with a finite number of processes, the multimodal systems we need consider only have a finite number of pairs of modal operators $\mu_i =< \Diamond_i, \Box_i >$, each one declared with some "modal type", and a binary accessibility relation R_i between worlds with the properties corresponding to the modal type associated.

The main idea of *epistemic logics* is that we can formalise the expression "agent i knows that..." using modal operators $\Box_i$. The modal type to be assigned depends on the notion of knowledge in use. In [DEL92] it is suggested that the most common modal types for epistemic logics are KT (reflexive), KT4 (reflexive and transitive), KD4 (serial and transitive) and KT5 (reflexive and symmetric). We will see that S5 (reflexive, symmetric and transitive) is a good system to formalise the intended notion of knowledge in distributed systems.

A *multimodal system* $S = (\Sigma, M_1, M_2, ..., M_n, <)$ consists of: a signature Σ containing function and predicate symbols, each one declared to be rigid or flexible[2]; for each $i \in \{1, ..., n\}$ a pair $\mu_i =< \Box_i, \Diamond_i >$ of modal operators and a modal system type M_i; and a set of declarations $\mu_i < \mu_j$ for some distinct $i, j \in \{1, 2, ..., n\}$. Terms and formulas are defined in this system in a standard way, using the connectives $\vee, \wedge, \neg$, the quantifiers $\exists$ and $\forall$, the modal operators $\Box_i$ and $\Diamond_i$ and a set $\mathcal{V}$ of variables.

An *interpretation* I for this system (*a Kripke structure*) consists of: a set W

[2] a function or predicate symbol is said to be rigid if its meaning is independent from the world considered, and flexible otherwise

of worlds; a set of binary relations $\{R_1, R_2, ..., R_n\}$ on the elements of W, called ***accessibility relations***; a set D called the ***discourse domain***; for every function symbol f of arity n, a function $f^I : W \times D^n \to D$ if f is flexible, a function f^I: $D^n \to D$ if f is rigid; and for every predicate symbol p of arity n, a function $p^I : W \times D^n \to \{0,1\}$ if p is flexible, a function $p^I : D^n \to \{0,1\}$ if p is rigid.

Each relation R_i is supposed to have the properties associated to the corresponding modal system type M_i, that is, reflexivity if M_i is KT, reflexivity and transitivity if M_i is KT4, etc. Also, if we have $\mu_i < \mu_j$ then $R_i \subset R_j$.

A ***Kripke structure*** can be viewed as a labelled directed graph, whose nodes are the worlds in W and worlds w_1 and w_2 are joined by an edge labelled i iff $(w_1, w_2) \in R_i$.

A formula $\Box_i\varphi$ ***holds in a world*** w if φ holds in every world w' such that wR_iw', while $\Diamond_i\varphi$ holds in w if there exists at least one world w' such that wR_iw', and φ holds in w'.

Following [HM90], it is natural to assume in many systems involving multiple interacting agents (e.g. connected machines in an automated manufacturing system) that the relations R_i characterise equivalence relations among points. Hence, it is reasonable to assume that the corresponding modal system types for each modal operator K_i are S5. On the other hand, the modal types corresponding to the operators E_G are not S5, since they fail to satisfy transitivity. Since C_G is the transitive closure of E_G, then modal types corresponding to operators C_G are also S5.

Assume that our agents can be grouped according to some hierarchy. Besides enabling us to focus our attention on critical subsystems if we wish to, such a grouping often leads to an improvement upon the performance of automated reasoning systems for multiple interacting agents, since it imposes a sort structure that can be exploited during search and unification([Coh89, AKP91]). This is the subject we explore in the following sections.

In section 2 we introduce a prototypical problem we shall use throughout the paper. This is a variation of the well known "cheating husbands" or "muddy children" problem, described in [HM90, HF89]. In section 3 we show how the language used can be "translated" to a first-order, equational, order-sorted theory. In section 4 we present a natural extension to the language to admit groups of agents, and review our example to illustrate the benefits of grouping the agents.

We close the article with some general discussion and proposals for future development.

2 An Example

We now introduce a simple problem: the problem of detecting the number of faulty devices in an automated manufacturing system:

> In a totally automated factory, each machine emits a signal when it is not working well. Each machine receives the signals emitted by the others, but it cannot perceive its own signal. Besides the machines, there is in

the factory an "automated supervisor", i.e., a machine that can tell if there is a signal but cannot distinguish how many. If the supervisor's light is on, then there is at least one machine out of order in the factory. We want the machines to know exactly how many of them are out of order, so that help can be called.

It is important to understand the role the supervisor plays. If there is only one machine out of order, this machine doesn't receive any signal, so, it will never know whether there is a machine out of order or not without the supervisor's help. Suppose now that there are two machines out of order, $m1$ and $m2$. Each of them receives one signal, but $m1$ considers the possibility that $m2$ doesn't receive any signal, so that it considers that there might be only one machine out of order ($m2$). If the supervisor's light is on, both machines know (and they know that they know) that at least one of them is out of order (in fact, it is common knowledge that at least one of them is out of order). Since $m1$ knows that it is impossible that $m2$ doesn't receive any signal, $m1$ concludes that it must be one of the machines out of order.

In [HM90] we have the result that if k machines are out of order, the "minimum degree" of knowledge the machines need to solve the problem is E^k. In our example, we are assuming that all of the machines are sensitive to the supervisor's signal, so that when the supervisor's light is on, it becomes common knowledge (and therefore, E^k for every $k \geq 1$) in the group of the machines that at least one of them is out of order.

This is what happens: If the supervisor's light is not on, there is nothing to do. If it is on, it is common knowledge that at least one of the machines is out of order. So, at the first instant, if any machine is not receiving any signal, it can deduce that it is out of order and the problem is solved. If there is no such machine, at the following instant all of the machines are able to conclude that there must be at least two machines out of order, otherwise the problem would have been solved at the previous instant. If any of the machines receives only one signal, it can deduce to be out of order and solve the problem. If all of them receive more than one signal, at the following instant it will be common knowledge that there are at least three machines out of order, and so on until the problem is solved.

We can describe the rules used to solve the problem using a multimodal system S consisting of a signature containing the flexible predicate $al(num, t)$, which has the intended meaning that there are at least num machines out of order at instant t, the rigid predicate $e(num)$ meaning that there are exactly num machines out of order, and $n + 1$ pairs of modal operators of type S5, where n is the total number of machines in the factory. We will use the modal operators K_i for $i \in \{1, ..., n\}$ meaning "agent i knows that..." and C_G meaning "it is common knowledge for group G that...".

For example, assume that we have 60 machines $m1, m2, ..., m60$. We can express that "if at instant t there is a machine i that knows that there are at least k machines that are out of order, and if it is common knowledge at that instant that at least k machines are out of order, then, in the following instant, it is common knowledge that there are exactly $k+1$ machines out of order and all of the machines know this number" as follows:

$$\exists i : G\forall t, k : NK_i al(k,t) \wedge C_G al(k,t) \rightarrow C_G e(k+1) \wedge \forall j : GK_j al(k+1, t+1) \quad (1)$$

We can express that "if at instant t it is common knowledge that there are at least k machines out of order, and all of the machines know there are more than k machines out of order, then it is common knowledge at the following instant that there are at least $k+1$ machines out of order" by:

$$\forall i : G\forall t, k : N\neg K_i al(k,t) \wedge C_G al(k,t) \rightarrow C_G al(k+1, t+1) \quad (2)$$

The fact that the supervisor's light is on at instant 0 can be expressed by $C_G al(1,0)$.

3 Translating from multimodal to equational and order sorted logics

We will now see how to rewrite a set of multimodal formulas as first-order, equational, order-sorted formulas.

An interpretation for a multimodal logic includes a relational structure $< W, R_1, R_2, ..., R_n >$ where W is a set of worlds and each R_i is an accessibility relation associated to a pair of modal operators $< \Diamond_i, \Box_i >$. The constraints on this structure define the types of the modal systems.

The first key idea for the method described in [DEL92] is to replace this relational structure by an algebraic one, $< W, A_1, A_2, ..., A_n, ! >$, where W is, as before, a set of worlds, each A_i is a set of operators and ! is a function from $W \times \bigcup_{\{1,...,n\}} A_i$ to W. Given an algebraic structure $< W, A_1, ..., A_n, ! >$ we can define a relational one by:

- wR_iw' iff there is an operator $a \in A_i$ such that $w!a = w'$

Considering $< W, R_i >$ as a graph, let A_i be a set of labels such that for every node w and every a in A_i, there is one and only one edge with source w labelled by a. Thus, $w!a$ is defined as the only world w' such that (w, w') is labelled by a.

The second key idea is that the properties of the relations R_i can be captured by equational constraints on the set A_i of operators. The relation R_i is functional if the set A_i contains only one operator, it is reflexive if there is a unit element 1 in A_i such that $w!1 = w$ for all w, it is transitive if there is a composition operation * defined on A_i with $w!(a * a') = (w!a)!a'$ and symmetric if there is a function inv defined on $A_i$such that $a * inv(a) = 1$ for every a in A_i.

Supposing we have a multimodal system S=(F, P, $M_1, M_2, ..., M_n$, <) as in section1, we define a signature Σ(S) = (($S, \leq$), F', P', Dec) for an order-sorted logic as follows:

- $\mathcal{S} = \{\mathcal{W}, \mathcal{A}_1, \mathcal{A}_2, ..., \mathcal{A}_n, \mathcal{D}\}$ is the set of sort symbols, where $\mathcal{W}$ is the sort for worlds, each $\mathcal{A}_i$ is a sort for operators on worlds and $\mathcal{D}$ is the sort for elements of the discourse domain
- $\mathcal{A}_i < \mathcal{A}_j$ iff $\mu_i < \mu_j$
- $P' = P, F' = F \cup \{\varepsilon, !, *, 1, inv\} \cup \{a_i / M_i = KF\}$ (the a_i's are fresh symbols)
- Dec contains the following declarations :
  for every rigid function symbol f of arity $n, f : \mathcal{D}^n \rightarrow \mathcal{D}$,
  for every rigid predicate symbol p of arity $n, p : \mathcal{D}^n \rightarrow \{0, 1\}$,
  for every flexible function symbol f of arity $n, f : \mathcal{W} \times \mathcal{D}^n \rightarrow \mathcal{D}$,
  for every flexible predicate symbol p of arity $n, p : \mathcal{W} \times \mathcal{D}^n \rightarrow \{0, 1\}$,
  $! : \mathcal{W} \times \cup_{\{1,...,n\}} \mathcal{A}_i \rightarrow \mathcal{W}$,
  $* : \mathcal{A}_i \times \mathcal{A}_i \rightarrow \mathcal{A}_i$, for every i such that M_i is KD4 or KT4,
  $\varepsilon : \mathcal{W}, 1 : \mathcal{A}_i$ for every i such that M_i is KT or KT4,
  $a_i : \mathcal{A}_i$ for every i such that M_i is KF
  $inv : \mathcal{A}_i \rightarrow \mathcal{A}_i$ for every i such that M_i is KT5 or S5.

The language built on Σ(S) is called the language of the path theory associated with S. The only terms built in this language having sort $\mathcal{W}$ are the terms of the general form $\varepsilon!a^1!a^2!...a^k$ where each of the a^j has sort $\mathcal{A}_i$ for some i. A natural interpretation for these terms is that they are denoting some world that can be reached from the world ε by following the transitions labelled $a^1, a^2, ..., a^k$.

Now, we associate a set of equation to every modal type:

- $E(KT) = \{w!1 = w\}$
- $E(KD4) = \{w!(a * a') = (w!a)!a', (a * a') * a'' = a * (a' * a'')\}$
- $E(KT4) = E(KT) \cup E(KD4) \cup \{a * 1 = a, 1 * a = a\}$
- $E(S5) = E(KT4) \cup \{a * inv(a) = 1\}$

Every a, a' and a'' appearing in $E(M_i)$ has sort $\mathcal{A}_i$ and w has sort $\mathcal{W}$.

$E(S) = \cup_{\{1,..,n\}} E(M_i)$ is called the path theory associated to a multimodal system S.

A Σ(S)-E(S)-interpretation is an interpretation for $\Sigma(S)$ that satisfies E(S). A formula ψ on $\Sigma(S)$ is E(S)-satisfiable if it is satisfied in some $\Sigma(S)$-E(S)-interpretation.

In [DEL92], F. Debart and colleagues suggest the function T defined below that translates a multimodal formula into a path formula, such that if S is a multimodal system and ψ is a multimodal formula, ψ is S-satisfiable iff $T(\psi)$ is E(S)-satisfiable.

$T(\psi) = t(\varepsilon, \psi)$

and the intermediate function t is defined recursively as:

$t(\pi, x) = x$ if x is a variable with sort $\mathcal{D}$

$t(\pi, f(\tau_1, \tau_2, ..., \tau_n)) = f(t(\pi, \tau_1), t(\pi, \tau_2), ..., t(\pi, \tau_n))$ if f is rigid

$t(\pi, f(\tau_1, \tau_2, ..., \tau_n)) = f(\pi, t(\pi, \tau_1), t(\pi, \tau_2), ..., t(\pi, \tau_n))$ if f is flexible

$t(\pi, p(\tau_1, \tau_2, ..., \tau_n)) = p(t(\pi, \tau_1), t(\pi, \tau_2), ..., t(\pi, \tau_n))$ if p is rigid

$t(\pi, p(\tau_1, \tau_2, ..., \tau_n)) = p(\pi, t(\pi, \tau_1), t(\pi, \tau_2), ..., t(\pi, \tau_n))$ if p is flexible

$t(\pi, \neg\psi) = \neg t(\pi, \psi)$

$t(\pi, \psi_1 \vee \psi_2) = t(\pi, \psi_1) \vee t(\pi, \psi_2)$
$t(\pi, \psi_1 \wedge \psi_2) = t(\pi, \psi_1) \wedge t(\pi, \psi_2)$
$t(\pi, \forall x \psi) = \forall x : \mathcal{D}\ t(\pi, \psi)$
$t(\pi, \exists x \psi) = \exists x : \mathcal{D}\ t(\pi, \psi)$
$t(\pi, \Diamond_i \psi) = \exists a : A_i\ t(\pi!a, \psi)$, where a is not in $Var(\pi)$
$t(\pi, \Box_i \psi) = \forall a : A_i\ t(\pi!a, \psi)$, where a is not in $Var(\pi)$

Now, let's go back to our multimodal logic where all of the modal operators $(K_1, ..., K_n, C_G)$ are of type S5. As the accessibility relation associated to each one of the K_i's is an equivalence relation, the last line of the translation function becomes:

$t(\pi, K_i \psi) = \forall \sigma : \mathcal{W}$ such that $\sigma \equiv_K \pi\ t(\sigma, \psi)$

And for the modal operator C_G we have:

$t(\pi, C_G \psi) = \forall \sigma : \mathcal{W}$ such that $\sigma \equiv_C \pi\ t(\sigma, \psi)$

The expressions 1 and 2 from the example in section 2 can be presented as:

$$\forall k, t : N \exists i : Gal(i, k, t) \wedge al(k, t) - e(G, k + 1) \wedge \forall j : Gal(j, k + 1, t + 1) \qquad (3)$$

$$\forall k, t : N \forall i : G \neg al(i, k, t) \wedge al(k, t) - al(k + 1, t + 1) \qquad (4)$$

An input to the problem should be the number of signals received by each machine and whether the supervisor's light is on or not. For example:

al(1,25,t); al(2,25,t); al(3,25,t); al(4,24,t); ...; al(60,24,t)

al(1,0) /* supervisor's light on at instant 0*/

The resolution for the query $- e(K)$ would require 24 steps and the problem would be solved after 24 instants.

4 Introducing Agent Grouping

We now introduce a new approach, where the agents are divided in groups, according to some hierarchy. One of the natural advantages of the method, besides the increase in time efficiency, is that it allows us to obtain more descriptive results.

By dividing the agents into groups, we are dividing the discourse domain and introducing new sorts in our logic. This can be exploited during resolution, through order-sorted unification.

The amount of information the agents have can be greatly increased if this hierarchy is somehow made transparent for them, for example, if a group can access the knowledge of its subgroups. This fact can be very useful as we will show, also in cases where not all of the system is reliable.

Going back once more to our example of finding the number of faulty devices in an automated factory, we divide now our sixty machines in eleven groups named A, B, C, ..., K, according to their importance in the factory or their chance of not working well, each group having its own supervisor. The hierarchy can be seen as a semi-lattice, where one group is the union of its "children" and the intersection of its "parents". We have at the top a group (K) containing all of the machines in the factory.

In the end we obtain a more descriptive answer, because the intermediate results (i.e. the number of out of order machines in each group) are preserved. Also, because of the semi-lattice structure, sometimes we are able to solve the problem satisfactorily even when the "supervisors" don't always detect the problems.

During the resolution, we associate with each group an interval in which the number of out of order machines must lay. After solving one group, we can refine the intervals associated with the groups above the solved one, that is, the groups containing it.

The state of each group can be ***ignorant***, ***active***, ***solved*** or ***waiting*** at each instant. A group is classified as ignorant when its supervisor and all the ones below it are off, meaning that we have no information about the existence of faulty machines in it. A group is active when all the others below it are solved or ignorant with at least one solved, meaning that there is a problem in it. A group is said to be solved when we know the exact number of out of order machines in it, and it is classified as waiting if it is waiting for a group below it to be solved, that is, if there is an active group below it.

When a group is active, the machines in it receive the signals of the others in the group.

We suppose the existence of two predicates $child(G, G1)$ and $descend(G, G1)$ meaning respectively that group G appears immediately below or below group $G1$ in the hierarchy, and a function $card(G)$ that gives the cardinality of group G.

Our example would be encoded as follows:

A = {m1, m2, m3, ..., m10}
B = {m11, m12, ..., m15}
C = {m36, ..., m50}
D = {m16, ..., m35}
E = {m51, ..., m60}
F = {m1, ..., m15}
G = {m16, ..., m50}
H = {m16, ..., m35, m51, ..., m60}
I = {m1,..., m50}
J = {m16, ..., m35, m36,..., m50, m51, ..., m60}
K = {m1, ..., m60}

When a group is active, each of its elements receives all the signals within the group, except for its own.

For every group there must be n rules of the following form, where n is the number of elements in the group:

al(m, g, t) : quantity of signals received by machine m in group g at instant t
state(g, t, lb, ub) : state of group g at instant t.
[*lb..ub*] denotes the interval in which the number of faulty machines lay

/*Group A */

al(m1, A, t) = 3 ← state(A, t, _, _) = active
al(m2, A, t) = 3 ← state(A, t, _, _) = active
al(m3, A, t) = 3 ← state(A, t, _, _) = active
al(m4, A, t) = 2 ← state(A, t, _, _) = active
al(m5, A, t) = 2 ← state(A, t, _, _) = active
al(m6, A, t) = 2 ← state(A, t, _, _) = active
al(m7, A, t) = 3 ← state(A, t, _, _) = active
al(m8, A, t) = 3 ← state(A, t, _, _) = active
al(m9, A, t) = 3 ← state(A, t, _, _) = active
al(m10, A, t) = 3 ← state(A, t, _, _) = active
(Similarly for groups B, ..., K)

We use a predicate sup(*group*) to indicate that the light of the supervisor of *group* is on.

We can have, for example, sup(A) and sup(H).

The initial states of the groups are given by:

∀ t, g, gl state(g,t,0,card(g)) = ignorant ← ¬sup(g), descend(gl,g), ¬ sup(gl)
∀ g, gl state(g,0,1,card(g)) = waiting ← descend(gl,g), sup(gl)
∀ g, gl state(g,0,1,card(g)) = active ← sup(g), descend(gl,g), ¬sup(gl)

A group g is waiting if it was already waiting at the previous instant and there exists at least one subgroup of g that is active:

∀g,t ∃gl state(g,t+1,_,_) = waiting ← state(g,t,_,_) = waiting, descend(gl,g), state(gl,t,_,_) = active

The state of a group g turns into active if it was waiting in the previous instant, none of its subgroups was active and at least one of its "children" $g1$ or $g2$ is solved, the other being either solved or ignorant.

The interval associated to g, [*lb..ub*], is obtained from the intervals associated to g1 and g2, [lb1..ub1] and [lb2..ub2], respectively:

∀g,t ∃g1,g2,lb1,ub1,lb2,ub2
state(g,t+1,lb,ub) = active ← state(g,t,_,_) = waiting, child(g1,g), child(g2,g), g1≠g2,

state(g1,t,lb1,ub1) = solved,
state(g2,t,lb2,ub2) = {solved; ignorant}
lb = max(lb1,lb2),
ub = card(g) - max(card(g1)-ub1,card(g2)-ub2)

A group that is already active at instant t goes on being active at instant $t+1$ if all of its members receive more signals than the lower bound (lb) and less than the upper bound (ub) of the interval associated. The interval can then be refined:

∀m:g, ∀g,t
state(g,t+1,lb+1,ub-1) = active ← state(g,t,lb,ub) = active,
al(m,g,t)>lb, al(m,g,t)<ub

The problem is solved in a group if the group was active in the previous instant and there is a member of the group that received a number not grater than lb or not less than ub of signals. The associated interval is then refined to contain only the number of out of order machines in the solved group:

∀g,t ∃m:g
state(g,t+1,lb+1,lb+1) = solved ← state(g,t,lb,ub) = active,
a(m,g,t) ≤ lb

∀g,t ∃ m:g
state(g,t+1,ub,ub) = solved ← state(g,t,lb,ub) = active,
al(m,g,t) ≥ ub

Comparing the two solutions to the problem, we see that in the case where we had only one supervisor for the sixty machines, it took them 24 instants to find out the number of faulty ones. When the machines are divided in groups, it takes 22 instants for them to solve the problem, a small gain in this particular case, but something that could be relevant in larger examples.

5 Conclusions

The initial motivation for this research were the observation that order-sorted and feature logics [BS92, AKP91] were powerful tools for knowledge representation and inference, and the search for sample problems to illustrate this view.

In this paper we focused on the utilization of such logics to perform efficient reasoning about multiple interacting agents, and we showed - by means of an example - that order-sorted languages not only can make the computations involved in unification less costly (as presented in [Coh89, AKP91]) but also allow that additional relevant information about the problem be encoded (the grouping of agents in our example), which also contributes to improving the efficiency of reasoning about the corresponding systems.

The example presented in this paper was implemented and tested as a PROLOG program.

Immediate future work includes implementing our example using a language based on order-sorted unification (possibly LIFE [AKP91]), and exploring further the relation between this and other examples and feature logics and feature-based unification and resolution.

Acknowledgements: The first author would like to thank CAPES for financial support. The second author is partially supported by FAPESP grant 93/0603-01, and CNPq grant 300041/93-4

References

[AKP91] Hassan Ait-Kaci and Andreas Podelski. Towards a meaning of LIFE. In *Lecture Notes in Computer Science*, volume 528, pages 255–274. Springer-Verlag, 1991.

[BS92] Rolf Backofen and Gert Smolka. A complete and recursive feature theory. Technical Report RR-92-30, DFKI - German Centre for Artificial Intelligence, September 1992.

[Coh89] A. G. Cohn. Taxonomic reasoning with many-sorted logics. *Artificial Intelligence Review*, 3:89–128, 1989.

[DEL92] Françoise Debart, Patrice Enjalbert, and Madeleine Lescot. Multimodal logic programming using equational and order-sorted logic. *Theoretical Computer Science*, 105:141–166, 1992.

[HF89] Joseph Y. Halpern and Ronald Fagin. Modelling knowledge and action in distributed systems. *Distributed Computing*, 3:159–177, 1989.

[HM90] Joseph Y. Halpern and Yoram Moses. Knowledge and common knowledge in a distributed environment. *Journal of the Association for Computing Machinery*, 37 no.3:549–587, 1990.

[HT93] Joseph Y. Halpern and Mark R. Tuttle. Knowledge, probability and adversaries. *Journal of the association for Computing Machinery*, 40 no.4:917–962, 1993.

This article was processed using the LaTeX macro package with LLNCS style

RELATÓRIOS TÉCNICOS

DEPARTAMENTO DE CIÊNCIA DA COMPUTAÇÃO

Instituto de Matemática e Estatística da USP

A listagem contendo os relatórios técnicos anteriores a 1992 poderá ser consultada ou solicitada à Secretaria do Departamento, pessoalmente, por carta ou e-mail(mac@ime.usp.br).

J.Z. Gonçalves, Arnaldo Mandel
COMMUTATIVITY THEOREMS FOR DIVISION RINGS AND DOMAINS
RT-MAC-9201, Janeiro 1992, 12 pp.

J. Sakarovitch
THE "LAST" DECISION PROBLEM FOR RATIONAL TRACE LANGUAGES
RT-MAC 9202, Abril 1992, 20 pp.

Valdemar W. Setzer, Fábio Henrique Carvalheiro
ALGOTRITMOS E SUA ANÁLISE (UMA INTRODUÇÃO DIDÁTICA)
RT-MAC 9203, Agosto 1992, 19 pp.

Claudio Santos Pinhanez
UM SIMULADOR DE SUBSUMPTION ARCHITECTURES
RT-MAC-9204, Outubro 1992, 18 pp.

Julio M. Stern
REGIONALIZAÇÃO DA MATRIZ PARA O ESTADO DE SÃO PAULO
RT-MAC-9205, Julho 1992, 14 pp.

Imre Simon
THE PRODUCT OF RATIONAL LANGUAGES
RT-MAC-9301, Maio 1993, 18 pp.

Flávio Soares C. da Silva
AUTOMATED REASONING WITH UNCERTAINTIES
RT-MAC-9302, Maio 1993, 25 pp.

Flávio Soares C. da Silva
ON PROOF-AND MODEL-BASED TECHNIQUES FOR REASONING WITH UNCERTAINTY
RT-MAC-9303, Maio 1993, 11 pp.

Carlos Humes Jr.,Leônidas de O.Brandão,Manuel Pera Garcia
A MIXED DYNAMICS APPROACH FOR LINEAR CORRIDOR POLICIES
(A REVISITATION OF DYNAMIC SETUP SCHEDULING AND FLOW CONTROL IN MANUFACTURING SYSTEMS)
RT-MAC-9304, Junho 1993, 25 pp.

Ana Flora P.C.Humes e Carlos Humes Jr.
STABILITY OF CLEARING OPEN LOOP POLICIES IN MANUFACTURING SYSTEMS (Revised Version)
RT-MAC-9305, Julho 1993, 31 pp.

Maria Angela M.C. Gurgel e Yoshiko Wakabayashi
THE COMPLETE PRE-ORDER POLYTOPE: FACETS AND SEPARATION PROBLEM
RT-MAC-9306, Julho 1993, 29 pp.

Tito Homem de Mello e Carlos Humes Jr.
SOME STABILITY CONDITIONS FOR FLEXIBLE MANUFACTURING SYSTEMS WITH NO SET-UP TIMES
RT-MAC-9307, Julho de 1993, 26 pp.

Carlos Humes Jr. e Tito Homem de Mello
A NECESSARY AND SUFFICIENT CONDITION FOR THE EXISTENCE OF ANALYTIC CENTERS IN PATH FOLLOWING METHODS FOR LINEAR PROGRAMMING
RT-MAC-9308, Agosto de 1993

Flavio S. Corrêa da Silva
AN ALGEBRAIC VIEW OF COMBINATION RULES
RT-MAC-9401, Janeiro de 1994, 10 pp.

Flavio S. Corrêa da Silva e Junior Barrera
AUTOMATING THE GENERATION OF PROCEDURES TO ANALYSE BINARY IMAGES
RT-MAC-9402, Janeiro de 1994, 13 pp.

Junior Barrera, Gerald Jean Francis Banon e Roberto de Alencar Lotufo
A MATHEMATICAL MORPHOLOGY TOOLBOX FOR THE KHOROS SYSTEM
RT-MAC-9403, Janeiro de 1994, 28 pp.

Flavio S. Corrêa da Silva
ON THE RELATIONS BETWEEN INCIDENCE CALCULUS AND FAGIN-HALPERN STRUCTURES
RT-MAC-9404, abril de 1994, 11 pp.

Junior Barrera; Flávio Soares Corrêa da Silva e Gerald Jean Francis Banon
AUTOMATIC PROGRAMMING OF BINARY MORPHOLOGICAL MACHINES
RT-MAC-9405, abril de 1994, 15 pp.

Valdemar W. Setzer; Cristina G. Fernandes; Wania Gomes Pedrosa e Flavio Hirata
UM GERADOR DE ANALISADORES SINTÁTICOS PARA GRAFOS SINTÁTICOS SIMPLES
RT-MAC-9406, abril de 1994, 16 pp.

Siang W. Song
TOWARDS A SIMPLE CONSTRUCTION METHOD FOR HAMILTONIAN DECOMPOSITION OF THE HYPERCUBE
RT-MAC-9407, maio de 1994, 13 pp.

Julio M. Stern
MODELOS MATEMATICOS PARA FORMAÇÃO DE PORTFÓLIOS
RT-MAC-9408, maio de 1994, 50 pp.

Imre Simon
STRING MATCHING ALGORITHMS AND AUTOMATA
RT-MAC-9409, maio de 1994, 14 pp.

Valdemar W. Setzer e Andrea Zisman
*CONCURRENCY CONTROL FOR ACCESSING AND COMPACTING B-TREES**
RT-MAC-9410, junho de 1994, 21 pp.

Renata Wassermann e Flávio S. Corrêa da Silva
TOWARDS EFFICIENT MODELLING OF DISTRIBUTED KNOWLEDGE USING EQUATIONAL AND ORDER-SORTED LOGIC
RT-MAC-9411, junho de 1994, 15 pp.