Artificial Intelligence for Engineering Design, Analysis and Manufacturing (1999), *13*, 119–132. Printed in the USA.

An algorithmic approach to knowledge evolution

ALESSIO LOMUSCIO AND MARK RYAN

School of Computer Science, University of Birmingham, Birmingham B15 2TT, UK

(RECEIVED July 24, 1998; REVISED August 11, 1998; ACCEPTED November 16, 1998)

Abstract

Intelligent agents must update their knowledge base as they acquire new information about their environment. The modal logic S5^n has been designed for representing knowledge bases in societies of agents. Halpern and Vardi have proposed the notion of refinement of S5^n Kripke models in order to solve multi-agent problems in which knowledge evolves. We argue that there are some problems with their proposal and attempt to solve them by moving from Kripke models to their corresponding *trees*. We define refinement of a tree with a formula, show some properties of the notion, and illustrate with the muddy children puzzle. We show how some diagnosis problems in engineering can be modelled as knowledge-based multi-agent systems, and hence how our approach can address them.

Keywords: Knowledge Representation; Modal Logic; Multi-agent Systems.

1. INTRODUCTION

1.1. Temporal epistemic modal logics and their potential for applications

In the last few years there is been a growing trend towards applying logical theories (and Multi-Agent theories in general) to the specification and analysis of engineering products. The reason behind this trend is that logic is a precise and unambiguous language, and it is increasingly seen a useful tool for specifying, reasoning about and validating complex systems.

Agent theories (see Wooldridge & Jennings, 1995, for a review) aim to represent key properties of an intelligent entity such as its knowledge, beliefs, intentions, desires, actions and most importantly its temporal evolution in a changing environment. Although much literature has been published in all of these areas, there is a general consensus about using epistemic and temporal modal logics, in which much progress has been made.

Epistemic modal logics (Hintikka, 1962; Fagin, Halpern, Moses & Vardi, 1995; and Meyer & van der Hoek, 1995) aim to represent the state of knowledge of an agent and to study what properties the state of knowledge should satisfy. This is formally done by using a classical modal operators defined on a Kripke style semantics Kripke (1959). Temporal modal logics (Gabbay, Hodkinson & Reynolds, 1993; Clarke & Emerson, 1981; Emerson, 1990; Lamport, 1994; Pnueli, 1977; and Shoham, 1987) use a similar technical tool to represent the temporal evolution of a system and investigate properties of this evolution.

In this paper we try to develop further part of this tool and we suggest that this can be proven useful in a practical example, specifically fault diagnosis in a distributed robotics situation. We examine a quite well known puzzle studied in computer science known as the muddy children puzzle and demonstrate that this example is conceptually equivalent to integrity self-checking in a robotics plant. We propose a general algorithm that can be applied in similar situations involving distributed knowledge among a group of agents.

1.2. The theoretical background

The logic S5^n models a community of *ideal* knowledge agents. Ideal knowledge agents have, among others, the properties of veridical knowledge (everything they know is true), positive introspection (they know what they know) and negative introspection (they know what they do not know). The modal logic S5^n (see, for example, Popkorn, 1994; and Goldblatt, 1992) can be axiomatised by taking all the propositional tautologies; the schemas of axioms $\Box_i(\phi \Rightarrow \psi) \Rightarrow \Box_i\phi \Rightarrow \Box_i\psi$, $\Box_i\phi \Rightarrow \phi$, $\Box_i\phi \Rightarrow \Box_i\Box_i\phi$, $\Diamond_i\phi \Rightarrow \Box_i\Diamond_i\phi$, where

Note: A presentation of part of this theory was given at PRR-98, a satellite workshop of ECAI-98.

Reprint requests to: Mark Ryan, School of Computer Science, University of Birmingham, Birmingham B15 2TT, United Kingdom. E-mail: M.D.Ryan@cs.bham.ac.uk; URL: www.cs.bham.ac.uk/~mdr

$i \in A$ represents an agent in the set of agents $A = \{1, \ldots, n\}$; and the inference rules Modus Ponens and Necessitation.

The logic $S5^n$ has also been extended to deal with properties that arise when we investigate the state of knowledge of the group. Subtle concepts like common knowledge and distributed knowledge have been well investigated (as in Fagin et al., 1995). The logic $S5^n$ is a successful tool for the agent theorist also because, even in its extensions to common knowledge and distributed knowledge, it has important meta-properties like closure under substitution, completeness and decidability (Meyer and van der Hoek, 1995).

The standard (*consequence relation)* approach to using $S5^n$ is to describe a situation as a set of formulas Γ, and to attempt to show that the situation satisfies a property ϕ by establishing $\Gamma \vdash \phi$ or $\Gamma \models \phi$. Establishing $\Gamma \vdash \phi$ involves finding a proof of ϕ from Γ, while establishing $\Gamma \models \phi$ involves reasoning about all (usually infinitely many) Kripke models satisfying Γ to show that they also satisfy ϕ. The completeness of $S5^n$ shows that these two notions are equivalent. However, experience has shown that this approach is computationally very expensive.

To overcome the intractability of this approach, Halpern and Vardi (1991) have proposed to use *model checking* as an alternative to theorem proving Halpern and Vardi (1991). In the model checking approach, the situation to be modelled is codified as a single Kripke model M rather than as a set of formulas Γ. The task of verifying that a property ϕ holds boils down to checking that M satisfies ϕ, written $M \models \phi$. This task is computationally much easier than the theorem proving task, being linear in the size of M and the size of ϕ Halpern and Vardi (1991).

Halpern and Vardi informally illustrate their approach by modelling the muddy children puzzle. In that puzzle, there are n children and n atomic propositions $p_1, p_2, \ldots, p_n$ representing whether each of the children have mud on their faces or not. Various announcements are made, first by the father of the children and then by the children themselves. The children thus acquire information about what other children know, and after some time the muddy ones among them are able to conclude that they are indeed muddy. We describe the problem in greater detail below.

Halpern and Vardi propose the following way of arriving at the model M to be checked. They start with the most general model for the set of atomic propositions at hand. To deal with the announcements made, they successively *refine* the model with formulas expressing the announcements made. This refinement process consists of removing some links from the Kripke model. At any time during this process, they can check whether child i knows p_i (for example), by checking whether the current model satisfies $\Box_i p_i$.

This method is illustrated in the paper Halpern and Vardi (1991) and the book Fagin et al. 1995), but a precise definition of the refinement operation is not given. Our original aim for this paper was to provide such a definition and explore its properties. However, we soon came to the opinion that there is no definition of model refinement on arbitrary $S5^n$ Kripke structures that will have intuitively acceptable properties. We explain our reasons for this view in Section 2. We believe the refinement and model checking ideas can still be made to work, however. In Section 3 we introduce a structure derived from a Kripke model, which we call a *Kripke tree*, and define the refinement operation on Kripke trees. We illustrate this notion using the muddy children example in Section 4. We prove some some properties of the refinement operation on Kripke trees in Section 5 and we conclude with some discussion in Section 6.

This is mainly a theoretical paper. However, we argue that scenarios conceptually equivalent to the muddy children puzzle can occur in robotics. We describe one of these scenarios in Section 2 and we solve it in Section 4 by applying the technical machinery we develop in Section 3.

1.3. Syntax and semantics

We assume finite sets P of *propositional atoms*, and A of *agents*. Formulas are given by the usual grammar:

$$\phi ::= p \mid \neg\phi \mid \phi_1 \wedge \phi_2 \mid \Box_i \phi \mid C\phi$$

where $p \in P$ and $i \in A$. Intuitively the formula $\Box_i \phi$ represents the assertion that the agent i knows the fact represented by the formula ϕ. The other propositional connectives can be defined in the usual way. The modal connectives $\Diamond_i$, E and B are defined as:

$$\begin{array}{lll} \Diamond_i \phi & \text{means} & \neg\Box_i\neg\phi \\ E\phi & \text{means} & \bigwedge_{i\in A}\Box_i\phi \\ B\phi & \text{means} & \neg C\neg\phi \end{array}$$

$\Diamond_i \phi$ means "it is consistent with i's knowledge that ϕ", $E\phi$ means that everyone knows ϕ, while $C\phi$ is the much stronger statement that ϕ is common knowledge. In a multiagent setting, a formula ϕ is said to be common knowledge if it is known by all the agents, and moreover that each agent knows that it is known by all the agents; and moreover, each agent knows that fact, and that one, etc. An announcement of ϕ results in common knowledge of ϕ among the hearers, because as well as hearing ϕ they also see that the others have heard it too (we assume throughout that all the agents are perceptive, intelligent, truthful). If one agent secretly informs all the others of ϕ, the result will be that everyone knows ϕ, but ϕ will not be common knowledge. B is the dual of C. Although not particularly useful intuitively, we will need it for technical reasons.

We will also need the following definitions.

Definition 1.1. A formula is *universal* if it has only the modalities $C, E, \Box_i$ and no negations outside them. Formally take

$$\phi ::= p \mid \neg\phi \mid \phi_1 \wedge \phi_2 \mid \phi_1 \vee \phi_2$$

https://doi.org/10.1017/S0890060499132062 Published online by Cambridge University Press

and define a formula ψ to be universal if it follows the following syntax:

$$\psi ::= \phi \mid \psi_1 \vee \psi_2 \mid \psi_1 \wedge \psi_2 \mid \Box_i \psi \mid E\psi \mid C\psi. \quad \blacksquare$$

DEFINITION 1.2. A formula is *safe* if it is universal and no $\Box_i$ and no C appears in the scope of $\vee$. Formally take

$$\phi ::= p \mid \neg\phi \mid \phi_1 \wedge \phi_2 \mid \phi_1 \vee \phi_2$$

and define a formula ψ to be safe if it follows the following syntax:

$$\psi ::= \phi \mid \psi_1 \wedge \psi_2 \mid \Box_i \psi \mid E\psi \mid C\psi. \quad \blacksquare$$

DEFINITION 1.3. A formula is *disjunction-free* if it is universal and has no $\vee$. Formally take

$$\phi ::= p \mid \neg\phi \mid \phi_1 \wedge \phi_2$$

and define a formula ψ to be disjunction-free if it follows the following syntax:

$$\psi ::= \phi \mid \psi_1 \wedge \psi_2 \mid \Box_i \psi \mid E\psi \mid C\psi. \quad \blacksquare$$

DEFINITION 1.4. Given a set A of agents, an *equivalence Kripke model* $M = (W, \sim, \pi, w)$ is given by:

1. A set W, whose elements are called *worlds*;
2. An A-indexed family of relations $\sim = \{\sim_i\}_{i \in A}$. For each $1 \le i \le n$, $\sim_i$ is an equivalence relation on W ($\sim_i \subseteq W \times W$), called the *accessibility relation*;
3. A function $\pi : P \to 2^W$, called the *interpretation*;
4. A world $w \in W$, the *actual world*.

See Figure 1 for an illustration.

Let $x \in W$. We define the relation of satisfaction of ϕ by M at x, written $M \models_x \phi$, in the usual way:

$$\begin{aligned}
M \models_x p &\quad \text{iff } p \in \pi(x) \\
M \models_x \neg\phi &\quad \text{iff } M \not\models_x \phi \\
M \models_x \phi \wedge \psi &\quad \text{iff } M \models_x \phi \text{ and } M \models_x \psi \\
M \models_x \Box_i \psi &\quad \text{iff for each } y \in W,\ x \sim_i y \text{ implies } M \models_y \psi \\
M \models_x C\psi &\quad \text{iff for each } k \ge 0 \text{ and } i_1, i_2, \ldots, i_k \in A, \\
&\quad \text{we have } M \models_x \Box_{i_1} \ldots \Box_{i_k} \psi \quad \blacksquare
\end{aligned}$$

We say that y is *reachable in* k steps from x if there are $w_1, w_2, \ldots w_{k-1} \in W$ and $i_1, i_2, \ldots i_k$ in A such that $x \sim_{i_1} w_1 \sim_{i_2} w_2 \ldots \sim_{i_{k-1}} w_{k-1} \sim_{i_k} y$. We also say that y is *reachable* from x if there is some k such that it is reachable in k steps. The following fact is useful for understanding the technical difference between E and C.

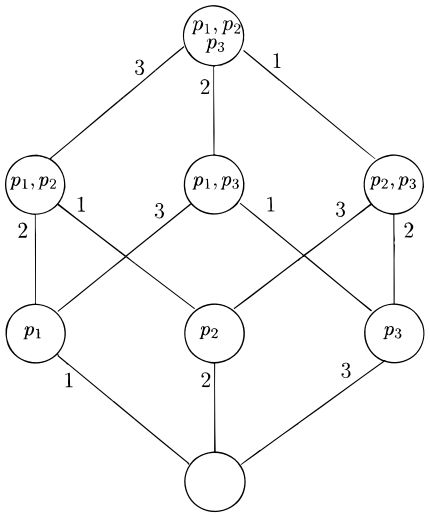

Fig. 1. M_1: The Kripke model for the muddy children puzzle with $n = 3$.

THEOREM 1.5. (Fagin et al., 1995)

1. *$M \models_x E^k\phi$ if and only if for all y that are reachable from x in k steps, we have $M \models_y \phi$.*
2. *$M \models_x C\phi$ if and only if for all y that are reachable from x, we have $M \models_y \phi$.* $\blacksquare$

2. REFINING KRIPKE MODELS

Halpern and Vardi propose to refine Kripke models in order to model the evolution of knowledge. They illustrate their method with the muddy children puzzle. This example is particularly important in the literature. We report it in the following.

2.1. The muddy children puzzle

There is a large group of children playing in the garden. A certain number (say k) get mud on their foreheads. Each child can see the mud (if present) on others but not on his own forehead. If $k > 1$ then each child can see another with mud on its forehead, so each one knows that at least one in the group is muddy. The father first announces that at least one of them is muddy (which, if $k > 1$, is something they know already); and then he repeatedly asks them "Does any of you know whether you have mud on your own forehead?" The first time they all answer "no". Indeed, they go on answering "no" to the first $k-1$ questions; but at the kth those with muddy foreheads are able to answer "yes."

At first sight, it seems rather puzzling that the children are eventually able to answer the father's question positively. The clue to understanding what goes on lies in the notion of common knowledge. Although everyone knows the content of the father's initial announcement, the father's saying it makes it common knowledge among them, so now

https://doi.org/10.1017/S0890060499132062 Published online by Cambridge University Press

they all know that everyone else knows it, etc. Consider a few cases of k.

- $k = 1$, that is, just one child has mud. That child is immediately able to answer "yes," because she has heard the father and does not see any other child with mud.
- $k = 2$, say a and b have mud. Everyone answers "no" the first time. Now a thinks: because b answered "no" the first time, he must see someone with mud. Well, the only person I can see with mud is b, so if b can see someone else it must be me. So a answers "yes" the second time. b reasons symmetrically about a, and also answers "yes."
- $k = 3$, say a, b, c. Everyone answers "no" the first two times. But now a thinks: if it was just b and c with mud, they would have answered "yes" the second time. So there must be a third person with mud; since I can only see b, c having mud, the third person must be me. So a answers "yes" the third time. For symmetrical reasons, so do b, c.

And similarly for other cases of k.

To see that it was not common knowledge before the father's announcement that one of the children was muddy, consider again $k = 2$, say a, b. Of course a and b both know someone is muddy (they see each other), but, for example, a doesn't know that b knows that someone is dirty. For all a knows, b might be the only dirty one, and therefore not be able to see a dirty child.

2.2. An engineering example

The muddy children puzzle, together with its many variants like the three wise men puzzle, etc. is popular among computer scientists. The reason is that it encodes subtle properties about reasoning, while also being applicable to real-life scenarios. We can imagine an example in which an engineering system could benefit from being able to cope with muddy-children-like situations.

Consider a factory in which similar robots collectively manufacture an object while moving in group in a large space. The robots can roughly be thought of being made of two components: the reasoning module and the mechanical actuators, effectively operating on the object. We want to design a fault detection system for the actuators. Given the large area the robots can be in, the installation of cameras to monitor the operational status of the robots' arms is not an option.

Let us suppose that the robots have a visual system directed toward the other robots that can detect faults in their mechanical arms. Note this is quite a reasonable assumption because it is often problematic to have visual systems that can do self-monitoring as well as monitoring the environment. Suppose now that the factory has a quality-control mechanism that can detect if something went wrong during the production of the object and assume this device broadcasts an alarm every time it notices a defect in the production.

This robotic scenario complies with the muddy children example: the children are now robots, the role of the father is taken by the fault detection system. Note that the assumption of communication being common knowledge is not violated because messages are assumed to be broadcasted to all the agents. The task of the robots is then to reason about their status and stop their operation in case they come to know that their mechanical arm is faulty. The evolution of their knowledge proceeds exactly as the case of the muddy children example where we assume the robots to operate synchronously.

Assuming the robots have a reasoning module able to handle the muddy children problem, the group of robots is then effectively able to do collective diagnosis. In the following we refer our discussion to muddy children, but the above scenario can serve equally well.

2.3. Halpern and Vardi's formulation

Suppose $A = \{1, \ldots n\}$ and $P = \{p_1, \ldots, p_n\}$; p_i means that the ith child has mud on its forehead. Suppose $n = 3$. The assumption of this puzzle is that each child can see the other children but cannot see itself, so each child knows whether the others have mud or not, but does not know about itself. Under these assumptions, Halpern and Vardi propose the Kripke structure of Figure 1 to model the initial situation.

Let w be any world in which there are at least two muddy children (i.e. w is one of the four upper worlds). In w, every child knows that at least one of the children has mud. However, it is not the case that it is common knowledge that each child has mud, since the world at the bottom of the lattice is reachable (cf. Theorem 1.5).

To model the father's announcement, Halpern and Vardi refine the model M_1 in Figure 1, arriving at M_2 in Figure 2 (these figures also appear in Halpern & Vardi, 1991; and Fagin et al., 1995). The refinement process is not precisely defined in Halpern and Vardi (1991), Fagin et al. (1995), though arguments in favour of the transformation from M_1 to M_2 are given.

Suppose now that the father asks the children whether they know whether they are muddy or not, and the children answer simultaneously that that they do not. Halpern and Vardi (1991) argue that this renders all models in which there is only one muddy child inaccessible, resulting in M_3 (Fig. 3).

If there are precisely two children with mud (i.e., the actual world is one of the three in the second layer), then each of the muddy children now knows it is muddy. For suppose the actual world is the left one of those three, i.e. w with $\pi(w) = \{p_1, p_2\}$. We easily verify that $M_3 \models_w \Box_1 p_1$ and $M_3 \models_w \Box_2 p_2$.

If all three children are muddy, i.e. the actual world w is the top one, then we are not yet done, for we do not have $M_3 \models_w \Box_i p_i$ for any i. The father again asks each of the chil-

https://doi.org/10.1017/S0890060499132062 Published online by Cambridge University Press

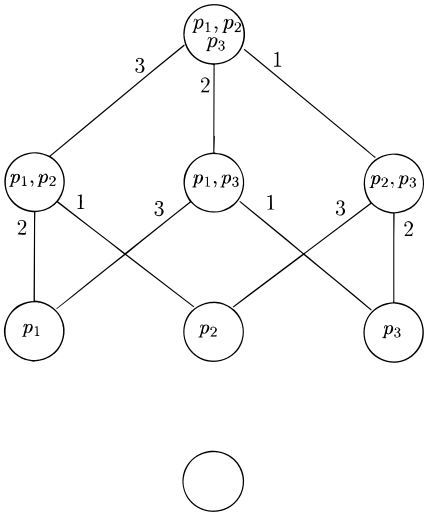

Fig. 2. M_2: The Kripke structure after the father speaks.

dren if they know if they are muddy, and the model is refined again according to their answer "no", resulting in M_4 which is M_3 with the last remaining links removed. (M_4 is not illustrated.) We can easily check that $M_4 \models_w \Box_i p_i$ for each i.

In summary, the method proposed by Halpern and Vardi (1991) for solving muddy-children-type puzzles is the following. Start with a suitably general model M_1 reflecting the initial set-up of the puzzle. Refine it successively by the announcements made. At the end of the announcements, check formulas against the refined model. In the example above, we refined M_1 first by $\phi_1 = C(p_1 \vee p_2 \vee p_3)$ (the father's announcement), and then twice by

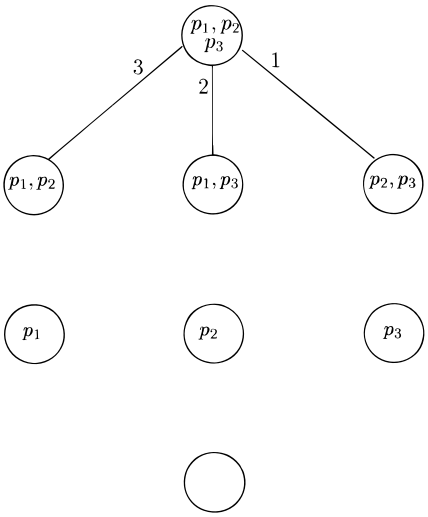

Fig. 3. M_3: The Kripke structure after the children announce that they do not know whether they are muddy.

$$\phi_2 = C(\neg\Box_1 p_1 \wedge \neg\Box_1 \neg p_1) \wedge C(\neg\Box_2 p_2 \wedge \neg\Box_2 \neg p_2) \wedge C(\neg\Box_3 p_3 \wedge \neg\Box_3 \neg p_3)$$

which corresponds to each of the three children announcing that they do not know whether they are muddy or not.

Halpern and Vardi (1991) do not precisely define what refinement by a formula means. The intuition they give is that refinement removes a minimal set of links of the model, so that the model satisfies the formula at the actual world. Removing links means that epistemic possibilities are removed, that is, knowledge is gained, so this seems intuitively the right thing to do.

2.4. Problems with refinement of Kripke models

Let us write $M * \phi$ to denote the result of refining the model M by the formula ϕ. Thus, in the example above, $M_2 = M_1 * \phi_1$, etc.

The muddy-children example discussed above naturally lead us to the question of whether it is possible to make precise the notion of refinement of a Kripke model by a formula, and of what properties this would have. Essentially any refinement procedure will remove the links to the states that are responsible for the nonsatisfaction of the formula we are refining with. However, some unexpected problems of any natural procedure operating on Kripke models can be found.

Consider the following examples.

EXAMPLE 2.1. Let M_5 be the Kripke model illustrated in Figure 4, with the left-hand world w the actual world, and consider refining by $\Box_1 p$. The definitions we examined differed in subtle cases involving quite complex formulas and models, but they all agreed in this one: the resulting model must be M_6 (see Fig. 4). What happens is that agent 1 gains the knowledge of p, and so must eliminate the epistemic possibility of $\neg p$ by removing the link.

The counterintuitive property of this example is that $M_5 \models_w \Box_3 \Diamond_1 p$, while $M_6 \not\models_w \Box_3 \Diamond_1 p$. Thus, in M_5, agent 3 knows that p is consistent with 1's knowledge. But after 1 learns p for sure in M_6, 3 no longer knows this! ■

EXAMPLE 2.2. Figure 5 shows a model and (the only) two outcomes one could consider for its refinement by $\Box_1 \Box_2 (p \vee q)$. One must remove either the 1 link or the 2 link in order to prevent the 1–2 path to the world exhibiting $\neg(p \vee q)$. The choice is which link to remove. Both outcomes reveal undesirable properties of the refinement operator. In the first case, removing the 1 link adds too much to 1's knowledge (he learns p), while the second case gives

Fig. 4. M_5 and M_6 (Example 2.1).

https://doi.org/10.1017/S0890060499132062 Published online by Cambridge University Press

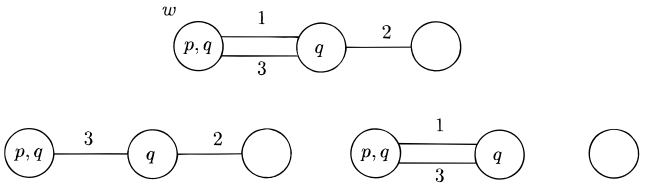

Fig. 5. Two outcomes for refinement of the top model by $\Box_1\Box_2(p \vee q)$ (Example 2.2).

us a situation in which a model satisfies $\Box_3\Diamond_2\neg q$ but its refinement by $\Box_1\Box_2(p \vee q)$ does not. It is counterintuitive that 3's knowledge should change in this way when we refine by $\Box_1\Box_2(p \vee q)$.

The second case at least has the desirable property that a minimal change of the knowledge of agents at the actual world w is made, since the set of reachable states from w is maximized (cf. Theorem 1.5). ■

EXAMPLE 2.3. Refinement by universal formulas (definition 1.1) ought to be cumulative, and such formulas ought to commute with each other (i.e. $M * \phi * \psi = M * \psi * \phi$). However, another example shows that this will be hard to achieve. Consider the model M_7 shown at the top of Figure 6, and let $\phi = \Box_1 p$ and $\psi = \Box_1\Box_2(p \vee q)$. Whatever way one thinks about defining $*$, the result in the left-hand branch seems clear. Note that $M_7 * \Box_1 p$ already satisfies $\Box_1\Box_2(p \vee q)$ and therefore $M_7 * \Box_1 p * \Box_1\Box_2(p \vee q) = M_7 * \Box_1 p$.

An argument for the stated result of $M_7 * \Box_1\Box_2(p \vee q)$ was given in Example 2.2, and further refining by $\Box_1 p$ leaves little room for maneuver. However, the resulting models differ on whether they satisfy (for example) $\Box_3\Box_2 q$. ■

Example 2.3 shows that even universal formulas, do not enjoy commutativity in any reasonable refinement setting.

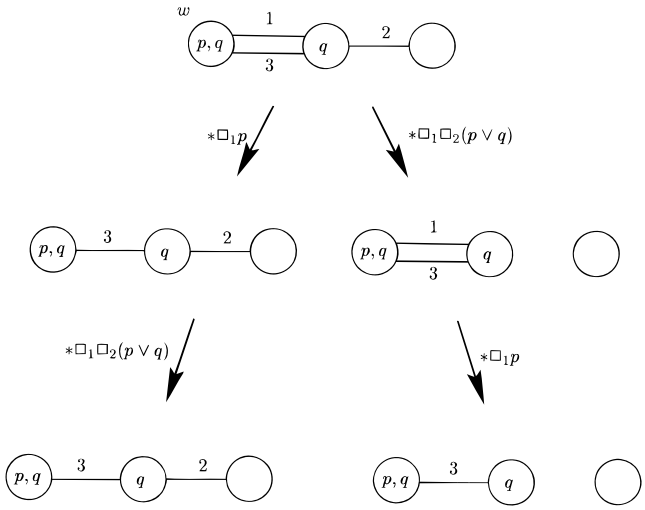

Fig. 6. Two evolutions of M_7 (Example 2.3), showing that $M * \phi * \psi = M * \psi * \phi$.

However, commutativity for universal formulas seems intuitively correct: the order in which ideal agents acquire information should not matter. Nonuniversal formulas are a different matter because they can express absence of knowledge, and this will not commute with the acquisition of new knowledge.

3. REFINING KRIPKE TREES

Some of the problems exhibited by the three examples at the end of the preceding section seem to be due to the following fact: when we remove a link in a Kripke model to block a certain path, we also block other paths that used that link. To overcome this problem, we would like to unravel Kripke models into trees, in which each link participates in just one path. At first sight this looks like it will destroy the finiteness of our models, a feature on which effective refinement operators and model checking operators rely. To retain finiteness, we will need to limit in advance the maximum nesting of boxes that is allowed, and construct a tree to depth greater than this number. Semantic structures similar to Kripke trees have been defined in Hughes and Cresswell (1984). Our definition differs in detail from the one in Hughes and Cresswell (1984), but it largely agrees with it in spirit.

In this section we define the notion of Kripke tree, show a translation of equivalence Kripke models into Kripke trees and define an algorithm for refining knowledge structures.

3.1. Kripke trees: Basic definitions

DEFINITION 3.1. (Kripke tree) A *Kripke tree* $T = (V, E, \sigma)$ is

- a set V; elements of V are called *vertices*;
- an A-indexed family $E = \{E_i\}_{i \in A}$ of *edges* $E_i \subseteq V \times V$, such that the structure (V, E) forms a tree, that is,
  - there is a unique vertex $v_0 \in V$ such that for all $v \in V$ and $i \in A$, $(v, v_0) \notin E_i$. The vertex v_0 is called the *root* of T.
  - for every vertex v there is a unique and finite *path* from the root to v, i.e. unique sequences $(v_0, v_1, \ldots, v_k)$ and $(i_1, \ldots, i_k)$ such that $(v_j, v_{j+1}) \in E_{i_{j+1}}$ $(0 \leq j < k)$ and $v_k = v$.
- a function $\sigma : P \to 2^V$, called *interpretation*. ■

We write E^* to mean the transitive closure of the union of relations in E, that is, $(v, v') \in E^*$ if there is a path from v to v', i.e. sequences $(v_0, v_1, \ldots, v_k)$ and $(i_1, \ldots, i_k)$ such that $(v_j, v_{j+1}) \in E_{i_{j+1}}$, with $0 \leq j < k$, $v_0 = v$ and $v_k = v'$.

We also allow the empty tree $(\emptyset, \emptyset, \emptyset)$ which we write as $\perp$. It has no root.

https://doi.org/10.1017/S0890060499132062 Published online by Cambridge University Press

DEFINITION 3.2. (Generated Kripke tree) Let $M = (W, \sim, \pi, w_0)$ be an equivalence Kripke model. The Kripke tree $T_M = (V, E, \sigma)$ *generated by* M is given as follows:

- The set of *vertices* is the set of paths in M:

$$V = \{(w_0, i_1, w_1, \ldots, w_{k-1}, i_k, w_k) \mid (w_j, w_{j+1}) \in \sim_{i_{j+1}}, \; (0 \leq j < k)\}$$

- E is an A-indexed family of edges. For $s, s' \in V$, there is an i-edge between s, s', written $(s, s') \in E_i$, if s' equals s extended by an i-link, i.e. $s = (w_0, i_1, w_1, \ldots, w_k)$, $s' = (w_0, i_1, \ldots, w_k, i, w)$ for some w.
- The *valuation* σ *is defined by* $\sigma(p) = \{(w_0, i_1, w_1, \ldots, w_k) \mid w_k \in \pi(p)\}$. ■

The vertex $w_0 \in V$ is the root of the tree.

When the model M is clear from the context or not relevant we will simply indicate the tree as T.

Generated Kripke trees are irreflexive, intransitive, anti-symmetric, anti-convergent and serial.

If the model M has at least two distinct worlds related by some relation $\sim_i$, then the tree T_M is infinite. For our purposes of model refinement, we usually want to deal with finite trees. The tree T_M^k is the tree T_M with paths truncated at length k. Obviously by truncating the tree we will lose seriality.

Definition 3.3. (Truncated tree of depth k) Given a tree $T = (V, E, \sigma)$, the truncated tree of depth k is defined as $T^k = (V', E', \sigma')$, where

- $V' = \{v \in V \mid$ the distance of v from the root is less or equal than $k\}$.
- $E' = E|_{V'}$ is the restriction of E to V',
- $\sigma' = \sigma|_{V'}$ is the restriction of σ to V'.

Infinite and finite trees satisfy modal formulae in the expected way: ■

DEFINITION 3.4. (Interpretation) Let ϕ be a formula, and T a tree. The satisfaction of ϕ by T at vertex v, written $T \models_v \phi$, is inductively defined as follows:

- $T \models_v p$ if $v \in \sigma(p)$;
- $T \models_v \neg\phi$ if not $T \models_v \phi$;
- $T \models_v \phi \wedge \psi$ if $T \models_v \phi$ and $T \models_v \psi$;
- $T \models_v \Box_i \phi$ if for all $v' \in V$, $(v, v') \in E_i$ implies $T \models_{v'} \phi$;
- $T \models_v C\phi$ if for all $v' \in V$, $(v, v') \in E^*$ implies $T \models_{v'} \phi$.

The tree T satisfies ϕ, written $T \models \phi$, if it satisfies ϕ at its root. The empty tree $\perp$ satisfies no formula. ■

An infinite tree T_M is semantically equivalent to its generating model M as the following shows:

LEMMA 3.5. *Let* $M = (W, \sim, \pi, w_0)$ *be an equivalence Kripke model and* $T_M = (V, E, \sigma)$ *its associated Kripke tree. Let* $v = (w_0, i_1, w_1, \ldots, w)$ *be any vertex ending in* w*, and* ϕ *any formula. Then*:

$$M \models_w \phi \quad \text{if and only if} \quad T_M \models_v \phi. \qquad ■$$

Proof: By induction on the structure of ϕ. The result holds for atoms by construction. For the inductive case observe that there is a one-to-one correspondence between paths in M from w and extensions of the path represented by vertex v. ■

In particular, Lemma 3.5 applies to valuations at the root of the tree, corresponding to the actual world of the model. into the following:

COROLLARY 3.6. $M \models \phi$ *if and only if* $T_M \models \phi$.

For the case of truncated tree, Lemma 3.5 is not valid. However, we can prove a related result for formulae up to a certain level of modal nesting.

We inductively define the rank of a formula as follows: ■

DEFINITION 3.7. (Rank of a formula) The rank $\text{rank}(\phi)$ of a formula ϕ is defined as follows:

- $\text{rank}(p) = 0$, where p is a propositional atom.
- $\text{rank}(\neg\phi) = \text{rank}(\phi)$.
- $\text{rank}(\phi_1 \wedge \phi_2) = max\{\text{rank}(\phi_1), \text{rank}(\phi_2)\}$.
- $\text{rank}(\phi_1 \vee \phi_2) = max\{\text{rank}(\phi_1), \text{rank}(\phi_2)\}$.
- $\text{rank}(\Box_i \phi) = \text{rank}(\phi) + 1$.
- $\text{rank}(C\phi) = \infty$. ■

The rank of a formula ϕ intuitively represents the maximum number of nested modalities that occur in ϕ. If an operator C occurs in ϕ we take the value of $\text{rank}(\phi)$ to be infinite. The rank of a formula reflects the maximal length of any path that needs to be explored to evaluate ϕ on an infinite tree. In other words, to evaluate a formula ϕ of rank k at w_0 we need not examine worlds whose distance from w_0 is greater than k.

LEMMA 3.8. *If* $\text{rank}(\phi) \leq k$, $M \models \phi$ *if and only if* $T_M^k \models \phi$. ■

Proof: By corollary 3.6, $M \models \phi$ if and only if $T_M \models \phi$, but, by induction, the evaluation of a formula of $\text{rank}(\phi) \leq k$ does not involve the evaluation of nodes of depth greater than k. So $T_M \models \phi$ if and only if $T_M^k \models \phi$, which gives the result. ■

In the following we shift our attention from an equivalence Kripke model to its truncated generated tree. Truncated generated trees satisfy $S5_n$-axioms provided that the rank of the formulae is sufficiently small compared to the size on the tree. The following clarifies under which circumstances $S5_n$-axioms are satisfied at the root of the tree and that $S5_n$-inference rules are sound.

LEMMA 3.9. *Let* M *be an equivalence model and* T_M^k *its generated tree truncated at* k.

https://doi.org/10.1017/S0890060499132062 Published online by Cambridge University Press

1. $T_M^k \models \phi$, *where* ϕ *is a tautology, and* $\text{rank}(\phi) \leq k$.
2. $T_M^k \models \Box_i(\phi \Rightarrow \psi) \Rightarrow \Box_i\phi \Rightarrow \Box_i\psi$, *where max* $\{\text{rank}(\phi), \text{rank}(\psi)\} \leq k-1$.
3. $T_M^k \models \Box_i\phi \Rightarrow \phi$, *where* $\text{rank}(\phi) \leq k-1$.
4. $T_M^k \models \Box_i\phi \Rightarrow \Box_i\Box_i\phi$, *where* $\text{rank}(\phi) \leq k-2$.
5. $T_M^k \models \Diamond_i\phi \Rightarrow \Box_i\Diamond_i\phi$, *where* $\text{rank}(\phi) \leq k-2$.
6. *If for every vertex* $v \in V$ *of* T_M^k, $T_M^k \models_v \phi$, *then for every* $v \in V$, $T_M^k \models_v \Box_i\phi$, *for any* $i \in A$.
7. *If for every vertex* $v \in V$ *of* T_M^k, $T_M^k \models_v \phi$, *and* $T_M^k \models_v \phi \Rightarrow \psi$ *then* $T_M^k \models_v \psi$. ■

Proof: We prove item number 4; the others can be done similarly. Suppose $T_M^k \not\models \Box_i\phi \Rightarrow \Box_i\Box_i\phi$, where $\text{rank}(\phi) \leq k-2$. Since T_M^k is generated by M, and $\text{rank}(\Box_i\phi \Rightarrow \Box_i\Box_i\phi) \leq k$, then by Lemma 3.8 we have $M \not\models \Box_i\phi \Rightarrow \Box_i\Box_i\phi$. But by hypothesis M is an equivalence model. This is absurd. ■

Before we proceed further, we introduce a few basic definitions and operations on subtrees.

DEFINITION 3.10. (Rooted-subtrees) Let $T' = (V', E', \sigma')$, $T = (V, E, \sigma)$ be trees with roots v_0', v_0. The tree T' is a *rooted subtree* of T, written $T' \leq T$, if $v_0 \in V'$, $V' \subseteq V$, $E|_{V'} = E'$, and $\sigma|_{V'} = \sigma'$. ■

DEFINITION 3.11. (Intersection of trees) Let $T' = (V', E', \sigma')$ and $T = (V, E, \sigma)$ be trees such that $\sigma|_{V' \cap V} = \sigma'|_{V' \cap V}$. The intersection of T and T' is $T \sqcap T' = (V' \cap V, E' \cap E, \sigma'|_{V' \cap V})$. ■

It is easy to see that definition 3.11 (when applicable) defines a tree.

DEFINITION 3.12. (Restriction of trees) Let $T = (V, E, \sigma)$ be a tree with root v, and V' a subset of V. The restriction of T to V', written $T|_{V'}$, is the largest rooted subtree of T generated by v whose vertices are in V'. If the root of T is not in V', then $T|_{V'} = \bot$. ■

3.2. Kripke trees: refinement

In Section 2.4, we discussed the difficulties that arise when using equivalence Kripke models as knowledge structures for refinement. Example 2.3 showed that any straightforward procedure to refine an equivalence Kripke model will be noncommutative even for universal formulae, that is, there will be universal α, β, such that $M * \alpha * \beta \not\equiv M * \beta * \alpha$.

Commutativity for universal formulae can be achieved by shifting to Kripke trees. Before we can show this, we must define refinement on Kripke trees.

The typical working scenario in which we operate is the same one as that advocated by Halpern and Vardi (1991), except that we refine T_M^k instead of M. It can be described as follows: we are given an initial configuration of a multi-agent system (MAS), and a set of formulae $\{\phi_1, \ldots, \phi_m\}$ that represent the update of the scenario. The question is whether the updated configuration will validate a set of formulae $\{\psi_1, \ldots, \psi_l\}$. We assume every ψ to have finite rank, i.e. we cannot check a formula containing the operator of common knowledge. There is no restriction on the ϕs.

Our method operates as follows:

1. Start from the most general equivalence Kripke model M that represents the MAS.
2. Generate the infinite tree T_M, as given in Definition 3.2.
3. Generate from T_M^k, the truncated tree of depth k, for some sufficiently large k.
4. Sequentially refine T_M^k with $\{\phi_1, \ldots, \phi_m\}$.
5. Check whether the resulting tree structure satisfies $\{\psi_1, \ldots, \psi_l\}$.

The method described above needs some further explanation. First, what is the most general Kripke model representing a MAS configuration? How are we to build it? Our answer is the same as that given by Halpern and Vardi. Assume the set of atoms P is finite, as we set it to be in Section 1.3. We take the model whose universe W is equal to 2^P with an interpretation that covers all the possible assignments to the atoms. We take the relations $\sim_i, i \in A$ to be the universal relations on $W \times W$, and w_0 to be the actual world of the given MAS.

In general we will require that M is more specific than the most general model, for example, some agent will have a certain knowledge about the world. We can add all the formulae that need be satisfied to the set of updates $\{\phi_1, \ldots, \phi_m\}$. For example in the muddy children example we can start from the model with universal relations and add

$$\bigwedge_{\substack{i,j \in \{1,2,3\} \\ i \neq j}} C(p_i = K_j\, p_i)$$

to the set of updates.

We have already explained how to execute steps 1, 2, 3, and 5. We now present a notion of refinement to execute step 4.

DEFINITION 3.13. (Refinement of Kripke tree structures) Given a truncated Kripke tree $T = (V, E, \sigma)$, a point $v \in V$, and a formula ϕ, the result $T' = (T, v) * \phi$ of refining T by ϕ at v is procedurally defined as follows. We assume that the negation symbols in ϕ apply only to atomic propositions (to achieve this, negations may be pushed inwards using de Morgan laws and dualities $\Box/\Diamond$ and C/B).

- If $T = \bot$, then $T' = \bot$.
- If $T \models_v \phi$, then $T' = T$.
- Otherwise the result is defined inductively on ϕ:
  - $\phi = p$. $T' = \bot$.
  - $\phi = \neg p$. $T' = \bot$.

https://doi.org/10.1017/S0890060499132062 Published online by Cambridge University Press

– $\phi = \psi \wedge \chi$. $T' = ((T,v) * \psi) \sqcap ((T,v) * x)$.

– $\phi = \psi \vee \chi$. If $(T,v) * \psi \leq (T,v) * \chi$ then $T' = (T,v) * \chi$, and if $(T,v) * \chi \leq (T,v) * \psi$ then $T' = (T,v) * \psi$. Otherwise T' is nondeterministically given as $(T,v) * \psi$ or $(T,v) * \chi$.

– $\phi = \Box_i \psi$. T' is given by computing as follows:
$T' := T$;
for each v' such that $(v,v') \in E_i$ do
IF: $(T',v') * \psi = \bot$
THEN: $T' := T'|_{V-\{v'\}}$
ELSE: $T' := (T',v') * \psi$

– $\phi = \Diamond_i \psi$. Let X be the set $X = \{(T,v') * \psi \,|\, (v,v') \in E_i\}$.
IF: $X = \varnothing$
THEN: $T' = \bot$,
ELSE: T' is non-deterministically chosen to be a $\leq$-maximal element of X.

– $\phi = C\psi$. T' is given by computing as follows:
$T' := T$;
for each v' such that $(v,v') \in E^*$ do
IF: $(T',v') * \psi = \bot$
THEN: $T' := T'|_{V-\{v'\}}$
ELSE: $T' := (T',v') * \psi$

– $\phi = B\,\psi$. Let X be the set $X = \{(T,v') * \psi \,|\, (v,v') \in E^*\}$.
IF: $X = \varnothing$
THEN: $T' = \bot$,
ELSE: T' is non-deterministically chosen to be a $\leq$-maximal element of X. ■

$T * \phi$ means $(T,v) * \phi$, where v is the root of T.

LEMMA 3.14. *Given a tree T, a formula α and a point v, $(T,v) * \alpha$ is a tree.* ■

Proof: It follows from the fact that if T is a tree then $T|_{V'}$ is also a tree. ■

The intuition behind $(T,v) * \phi$ is that it is obtained by removing as small a set of links from T as possible, in order to satisfy ϕ. Note that, due to the clauses for the connectives $\vee, \Diamond_i, B$, we have that the tree $(T,v) * \phi$ is not uniquely defined. However, we will see that running the procedure on the muddy children example does not introduce nondeterminism.

4. THE MUDDY CHILDREN PUZZLE USING KRIPKE TREES

In Section 2.1, we described the muddy children puzzle and we reported the formalisation that was given in Fagin et al. (1995), Halpern and Vardi (1991). The aim of the present section is to solve an instance of it [where the actual situation is coded by the tuple p_1, p_2, p_3 that we equivalently write as (1,1,1); all the children are muddy] by using Kripke trees and the methods we introduced in Section 3.

We start with the most general model to represent the puzzle: this is the model M_1 of Figure 1.[1] Given M_1, we generate the infinite tree T_{M_1} for M_1 and then the truncation T_1 of T_{M_1}. In this example we truncate at level, say, ten. The starting tree and the three successive refinements are in Figure 7 and 8 (shown to three levels). Let $\phi_1 = C(p_1 \vee p_2 \vee p_3)$ (this is the father's announcement), and $\phi_2 = C(\neg\Box_1 p_1 \wedge \neg\Box_1 \neg p_1) \wedge C(\neg\Box_2 p_2 \wedge \neg\Box_2 \neg p_2) \wedge C(\neg\Box_3 p_3 \wedge \neg\Box_3 \neg p_3)$ (the children's simultaneous reply that they don't know whether or not they are muddy). We now sequentially update T_1 by ϕ_1 and then by ϕ_2 three times. Note that since all children are muddy, they will have to speak three times before everyone knows he is muddy.

Consider the algorithm of Definition 3.13 and T_1. Following the algorithm, the refined tree $T_1 * \phi_1 = T_2$ in Figure 7 is T_1 in which the links to states where no children are muddy have been removed. The tree $T_3 = T_2 * \phi_2$ (shown in Figure 8) is then constructed by isolating worlds that do not see two worlds for every relation. In fact, only in this case one of the formulae $\Diamond_i p_i \wedge \Diamond_i \neg p_i$ can fail on a point of T_2. We can now obtain T_4 similarly.

Having made all the refinements, we can now check whether or not the muddy children know that they are muddy. This involves checking

$$T_4 \vDash \bigwedge_{i=1}^{3} (p_i \Rightarrow K_i p_i),$$

which is indeed the case.

Analogously we can prove that the procedure given in Section 3 produces solutions for the other cases of the muddy children.

Note that had we decided to consider the Kripke tree truncated at $n \geq 4$, the formula $\bigwedge_{i=1}^{3}(p_i \Rightarrow K_i p_i)$ would still be satisfied at the root after three refinements.

Let us now consider the example presented in Section 2.2. By following the above described procedure with the assumption of synchronicity, the k faulty robots will announce their fault and disconnect from the system after k rounds, allowing the system to start normal production again and substitute the faulty units.

[1] According the the notion of most general model as described in Section 3.2 the model M should actually be $M = (2^{\{p_1,p_2,p_3\}}, U, \pi, w)$, where U is a family of universal relations on $W \times W$, and $\pi(w) = \{p_1, p_2, p_3\}$. The model M_1 we analyse is the result of the update of M by

$$C(p_i \Rightarrow K_j p_i) : i \neq j;\ i,j \in \{1,2,3\},$$

where the formula above represents the fact that children can see each other. For brevity (as in Fagin et al., 1995; Halpern & Vardi, 1991) we start our analysis from M_1; i.e. rather than building the tree for M and update it first by $C(p_i \Rightarrow K_j p_i)$, we directly build the tree for M_1. The reader can check that this leads to the same result.

https://doi.org/10.1017/S0890060499132062 Published online by Cambridge University Press

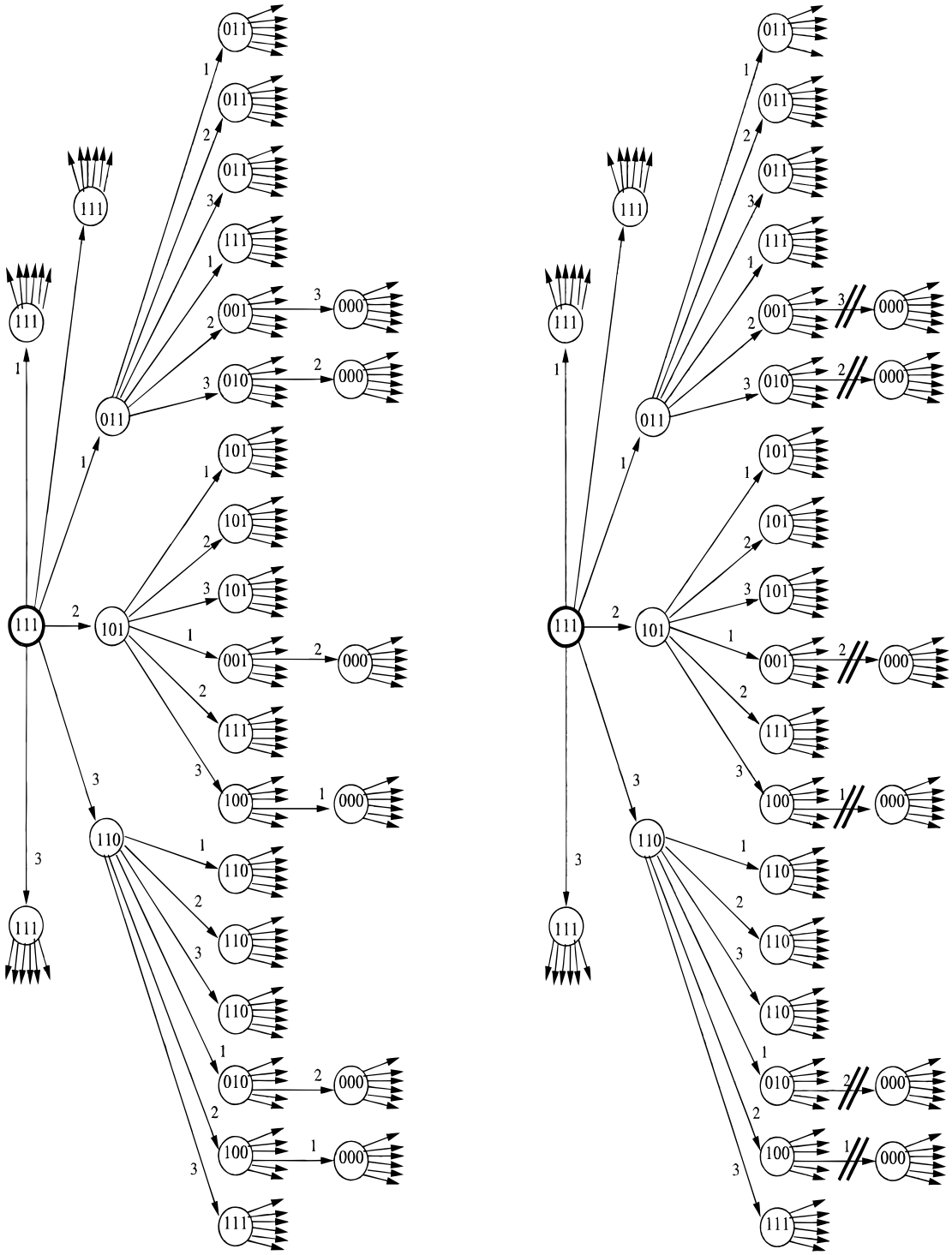

Fig. 7. T_1, T_2: The Kripke trees before and after the father speaks.

5. PROPERTIES OF REFINEMENT ON KRIPKE TREES

In the rest of this section we analyse some more properties of the refinement procedure that we defined in Definition 3.13.

The first remark that we should make is that refining a scenario by some agent's knowledge cannot affect other agents' knowledge, as was the case in Example 2.1 for Kripke models. This is because by unravelling a Kripke model we produce a tree whose leaves are in a bijection with paths of the original model. We formalize this as follows:

https://doi.org/10.1017/S0890060499132062 Published online by Cambridge University Press

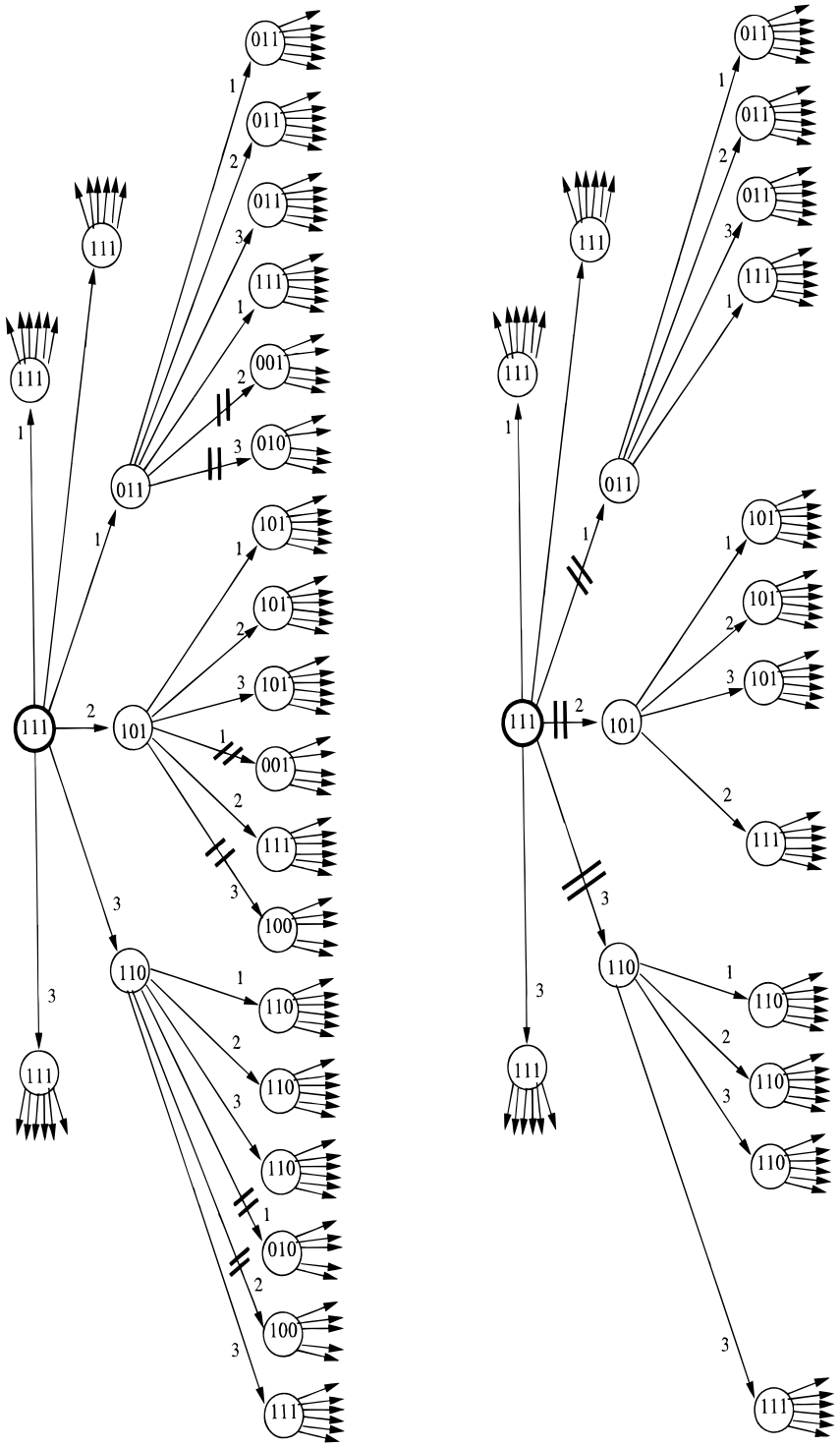

Fig. 8. T_3, T_4: The Kripke trees after the children speak the first and second time.

THEOREM 5.1. *Let T be a tree, and ϕ,ψ two formulae, we have the following*:

$$\text{If } T \vDash \Box_i \phi \text{ then } T * \Box_j \psi \vDash \Box_i \phi, \text{ with } i \neq j.$$

Proof: Nodes of a Kripke tree are in a bijection with paths of the generating model. Therefore by removing some j-links we cannot affect the interpretation of any modality whose index is not j. The only problematic case would arise if $i = j$ and $T * \Box_j \psi = \bot$, but this is excluded by hypothesis. ■

Although the theorem above refers to infinite trees, an analogue version can be proved for truncated trees. In that

https://doi.org/10.1017/S0890060499132062 Published online by Cambridge University Press

case we need the rank of the formulae to be less or equal to the depth of the truncated tree minus 1.

The second point worth stressing is that Kripke trees solve the problem of Example 2.3, i.e. we can prove commutativity although the result is limited to safe formulae (Definition 1.2).

LEMMA 5.2. *Let $T_1 = (V_1, E_1, \sigma_1)$, $T_2 = (V_2, E_2, \sigma_2)$ be trees. The following hold.*

1. $(T,v) * \phi \leq T$.
2. *If α is disjunction-free, then $T_1 \leq T_2$ implies $(T_1,v) * \alpha \leq (T_2,v) * \alpha$, where $v \in V_1 \cap V_2$.*
3. *If α is universal then $T \vDash \alpha$, $T' \neq \bot$, $T' \leq T$ imply $T' \vDash \alpha$.*

Proof:

1. The procedure for obtaining $(T,v) * \phi$ only removes links or produces the empty tree. Therefore we have the result.
2. We perform structural induction on α. Let $T_1' = (T_1,v) * \alpha$ and $T_2' = (T_2,v) * \alpha$. Suppose α is of the form:
   - $\alpha = p$. If $v \in \sigma(p)$ then $T_1' = T_1$, $T_2' = T_2$; else $T_1' = T_2' = \bot$.
   - $\alpha = \neg p$. If $v \notin \sigma(p)$ then $T_1' = T_2' = \bot$; else $T_1' = T_1$, $T_2' = T_2$.
   - $\beta \wedge \gamma$.

$$\begin{aligned}(T_1,v) * \alpha &= (T_1,v) * \beta \sqcap (T_1,v) * \gamma \\ &\leq (T_2,v) * \beta \sqcap (T_2,v) * \gamma \\ &\quad \text{Induction hypothesis} \\ &= (T_2,v) * \alpha\end{aligned}$$

   - $\alpha = \Box_i \beta$. Set $T_1' = T_1$ and $T_2' = T_2$ and we execute the loops of Definition 3.13 ($\Box_i$-case) synchronously. We will show that $T_1' \leq T_2'$ is an invariant of the execution. Suppose $(v,v') \in E_{2i}$.
     - If $(v,v') \in E_{1i}$, then consider the following cases:
       - $(T_1',v') * \beta = \bot$ and $(T_2',v') * \beta = \bot$. $T_1' := T_1'|_{V-\{v'\}}$ and $T_2' := T_2'|_{V-\{v'\}}$, so $T_1' \leq T_2'$ is not violated.
       - $(T_1',v') * \beta = \bot$ and $(T_2',v') * \beta \neq \bot$. $T_1' := T_1'|_{V-\{v'\}}$ and $T_2' := (T_2',v') * \beta$; so $T_1' \leq T_2'$.
       - $(T_1',v') * \beta \neq \bot$ and $(T_2',v') * \beta = \bot$. Contradicts hypothesis that $T_1' \leq T_2'$.
       - $(T_1',v') * \beta \neq \bot$ and $(T_2',v') * \beta \neq \bot$. $T_1' := (T_1',v') * \beta$, $T_2' := (T_2',v') * \beta$, and $T_1' \leq T_2'$ by induction hypothesis.
     - If $(v,v') \notin E_{1i}$ then T_1' is unchanged by the body of the loop, while T_2' becomes one of $T_2' := T_2'|_{V-\{v'\}}$ and $(T'_2,v') * \beta$. In either case, we are removing links in T_2 which are not present in T_1, so $T_1' \leq T_2'$ is preserved.
   - $\alpha = E\beta$. It follows by induction hypothesis by noting that $E\beta = \bigwedge_{i=1}^{n} K_i \beta$.
   - $\alpha = C\beta$. Similar to $\Box_i \beta$, but with proofs related to E^*.
3. It follows from structural induction on α. ∎

THEOREM 5.3. (Success) *If α is universal, $(T,v) * \alpha = \bot$ or $(T,v) * \alpha \vDash_v \alpha$.* ∎

Proof: Induction on α. The cases $\alpha = p, \neg p, \psi \vee \chi, \Box_i \psi, E\phi, C\psi$ are straightforward; we prove the case $\alpha = \psi \wedge \chi$.

$(T,v) * (\psi \wedge \chi) = (T,v) * \psi \sqcap (T,v) * \chi$. But by induction hypothesis we have that $(T,v) * \psi \vDash \psi$ and that $(T,v) * \chi \vDash \chi$. Since $(T,v) * \psi \leq (T,v) * \psi \sqcap (T,v) * \chi$ and $(T,v) * \chi \leq (T,v) * \psi \sqcap (T,v) * \chi$, by part 3 of Lemma 5.2, we have that $(T,v) * \psi \sqcap (T,v) * \chi \vDash \psi$ and that $(T,v) * \psi \sqcap (T,v) * \chi \vDash \chi$. So we have that $(T,v) * \psi \sqcap (T,v) * \chi \vDash \phi \wedge \chi$. ∎

LEMMA 5.4 *If α is safe, then the outcome of $(T,v) * \alpha$ is deterministically defined.* ∎

Proof: Suppose ϕ contains no $\Box_i, C$ operators. Then it is an easy induction to see that $(T,v) * \phi$ is either T or $\bot$. Now consider $(T,v) * (\phi \vee \psi)$, where ϕ, ψ are $\Box_i, C$-free. We see that either $(T,v) * \phi \leq (T,v) * \psi$ or $(T,v) * \psi \leq (T,v) * \phi$, so the result is again T or $\bot$. The cases $\Box_i \phi, C\phi$ do not introduce nondeterminism. ∎

We show that, for universal formulae, the change made by a refinement is the minimal one possible in order to satisfy the formula:

THEOREM 5.5 *If α is safe, then the tree $(T,v) * \alpha$ is $\leq$-maximum in $\{T' \leq T \mid T' \vDash_v \alpha$ or $T' = \bot\}$.* ∎

Proof: Let $T' = (T,v) * \alpha$. By part 1 of Lemma 5.2 and Theorem 5.3, we know T' is in the set. To prove that it is maximum, take any T'' in the set; we will show $T'' \leq T'$. If $T'' = \bot$ the result is immediate; otherwise, we have $T'' \vDash_v \alpha$ and $T'' \leq T$. Since $T'' \leq T$, we get $(T'',v) * \alpha \leq (T,v) * \alpha$ by part 2 of Lemma 5.2. But $(T'',v) * \alpha = T''$ (since it is already $T'' \vDash_v \alpha$) and $(T,v) * \alpha = T'$; so $T'' \leq T'$. ∎

THEOREM 5.6. *If α, β are safe, then the tree $(T,v) * \alpha * \beta$ is maximum in $\{T' \leq T \mid T' \vDash_v \alpha \wedge \beta$ or $T' = \bot\}$.* ∎

Proof: Let $T' = (T,v) * \alpha * \beta$. By parts 1 and 3 of Lemma 5.2 and Theorem 5.3, we know T' is in the set. The argument that it is maximum is similar to the proof of Theorem 5.5. Take any T'' in the set; we will show $T'' \leq T'$. If $T'' = \bot$ the result is immediate; otherwise, we have $T'' \vDash_v$

https://doi.org/10.1017/S0890060499132062 Published online by Cambridge University Press

$\alpha \wedge \beta$ and $T'' \leq T$. Since $T'' \leq T$, we get $(T'',v) * \alpha * \beta \leq (T,v) * \alpha * \beta$ by part 2 of Lemma 5.2. But since $T'' \models_v \alpha$ we have $(T'',v) * \alpha = T''$ and since $T'' \models_v \beta$ we have $(T'',v) * \beta = T''$. So $(T'',v) * \alpha * \beta = T''$ and $(T,v) * \alpha * \beta = T'$. Therefore we have $T'' \leq T'$. ■

THEOREM 5.7. (Commutativity) *If α, β are safe, then* $T * \alpha * \beta = T * \beta * \alpha$. ■

Proof: By Theorem 5.6, $T * \alpha * \beta$ and $T * \beta * \alpha$ are maximum in the same set. Therefore they are equal. ■

It is worth mentioning an example of which non-universal formulae can make commutativity to fail, independently of non-determinism.

EXAMPLE 5.8. Commutativity can fail for arbitrary formulae. The problem is that if the formulae are non-universal, the order of updating can play a role in the outcome of the update and we might have that one of the two cases fail. The example we report here is the tree T_5, illustrated in Figure 9, where the root is the top vertex. Consider now $T_6 = T_5 * \Diamond_1 \neg p * \Box_1(p \vee \neg q)$, illustrated, and $T_7 = T_5 * \Box_1(p \vee \neg q) * \Diamond_1 \neg p = \bot$. ■

6. CONCLUSIONS AND FURTHER WORK

In this paper we have developed the proposal in Halpern and Vardi (1991) for model refinement and model checking. We argued that model refinement could not be defined satisfactorily on Kripke models, and proposed a definition on Kripke trees obtained from Kripke models instead.

The shift from Kripke models to Kripke trees let us achieve two main results. First, we showed that it is possible to refine trees by a formula expressing knowledge of a formula without affecting the knowledge of the other agents (Theorem 5.7)–this was not apparently possible on standard Kripke models (see Example 2.1). Second, while it seems impossible to obtain commutativity for even safe formulas on Kripke models, we showed this is possible for Kripke trees. Many of the issues we discussed in this note still need investigating and we refer the reader to Lomuscio and Ryan (1998) for a list of technical conjectures currently under analysis.

Although the attention in this paper is on theoretical issues, in Section 2.2 we proposed an example in which these ideas can be applied. This consisted in a collective diagnosis problem among a group of homogeneous robots working at a factory. It should be clear that the scenarios commonly analysed in collective diagnosis research (Fröhlich, Nejdl & Schroeder, 1998; Böttcher & Dressler, 1993; Jennings and Wittig, 1992; Schroeder, 1998) are somehow different from our example. Our example is much closer to scenarios coming directly from robotics.

Nowadays, robots (see McKerrow, 1991, for an introduction) regularly substitute humans in many tasks. Diagnosis and maintenance in hazardous environments is one of the many important areas in which robots can clearly offer valuable solutions (NEI, 1991). Indeed the use of robots in environmental monitoring and cleaning, especially in controlled radiation areas, and in steam generators has seen a substantial growth in the last 15 years (see NEI, 1992; and Gerriets 1992) and it is reasonable to assume that more advanced solutions will become increasingly available in the future.

In this context we believe that, although our example is not realistic at present (because it presupposes the availability of complex visual systems, etc.), it is likely and worth advocating that in the future engineering will be able to profit from techniques such as the one presented in this paper. Our short term research agenda includes an implementation of the algorithm exposed in this paper and further analysis of its underlying properties.

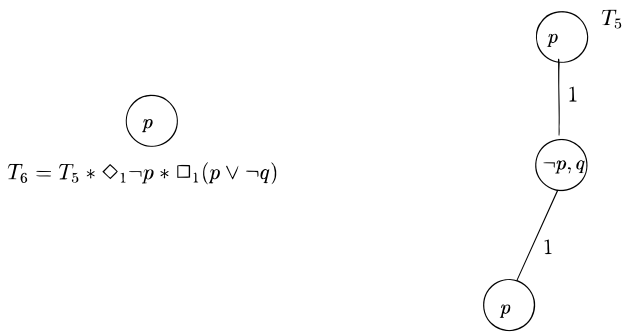

Fig. 9. T_5 and T_6 discussed in Example 5.9. While $T_6 = T_5 * \Diamond_1 \neg p * \Box_1(p \vee \neg q)$ is defined and shown above, $T_7 = T_5 * \Box_1(P \vee \neg q) * \Diamond_1 \neg p$ is undefined.

REFERENCES

Böttcher, C., & Dressler, O. (1993). Diagnosis process dynamics: Holding the diagnostic trackhound in leash. *Proceedings of the 13th International Joint Conference on Artificial Intelligence*, pp. 1460–1471. Morgan Kauffmann, Los Altos, California.

Clarke, E.M., & Emerson, E.A. (1981). Design and synthesis of synchronization skeletons using branching time temporal logic. *Proc. Workshop on Logics of Programs, Vol. 131 of Lecture Notes in Computer Science*, pp. 52–71. Springer, New York.

Emerson, E.A. (1990). Temporal and modal logic. In *Handbook of Theoretical Computer Science*, (van Leeuwen, J., Ed.), Elsevier Science Publishers, Chap. 16, pp. 996–1071. New York.

Fagin, R., Halpern, J.Y., Moses, Y., & Vardi, M.Y. (1995). *Reasoning about knowledge*. MIT Press, Cambridge.

Fröhlich, P., Nejdl, W., & Schroeder, M. (1998). Strategies in model-based diagnosis. *Journal of Automated Reasoning 20(1/2)*, 81–105.

Gabbay, D.M., Hodkinson, I.M., & Reynolds, M.A. (1993). *Temporal logic: Mathematical foundations and computational aspects, Volume 1: Mathematical Foundations*. Oxford University Press, Oxford, England.

Gerriets, W. (1992). TROD cleans up at Nine Mile Point 1. Nuclear Engineering International. London, England.

https://doi.org/10.1017/S0890060499132062 Published online by Cambridge University Press

Goldblatt, R. (1992). *Logics of time and computation, 2nd Ed. (Revised and Expanded, Vol. 7 of CSLI Lecture Notes)*, CSLI, Stanford.
Halpern, J., & Vardi, M. (1991). Model checking vs. theorem proving: A manifesto. In *Artificial Intelligence and Mathematical Theory of Computation*, pp. 151–176. Academic Press, Inc. New York.
Hintikka, J. (1962). *Knowledge and belief, an introduction to the logic of the two notions*. Cornell University Press, Ithaca, NY.
Hughes, G.E., & Cresswell, M.J. (1984). *A companion to modal logic*, Methuen, London.
Jennings, N.R., & Wittig, T. (1992). ARCHON: Theory and practice. In *Distributed Artificial Intelligence: Theory and Praxis*, (Avouris, N.M. and Gasser, L. Eds.), pp. 179–195. Kluwer Academic Press.
Kripke, S.A. (1959). Semantic analysis of modal logic (abstract). *Journal of Symbolic Logic 24*, 323–324.
Lamport, L. (1994). The temporal logic of actions. *ACM Transactions on Programming Languages and Systems 16(3)*, 872–923.
Lomuscio, A., & Ryan, M. (1998). Model checking and refinement for $S5_n$. *Proceedings of the ECAI98-workshop on Practical Reasoning and Rationality (PRR98)*.
McKerrow, P.J. (1991). *Introduction to robotics*. Addison-Wesley, Sydney, Australia.
Meyer, J.-J.C., & van der Hoek, W. (1995). *Epistemic logic for AI and computer science, Vol. 41 of cambridge tracts in theoretical computer science*, Cambridge University Press, New York.
NEI (1991). ROSA III: The Westinghouse workhorse. Nuclear Engineering International. London, England.
NEI (1992). A brief history of robots in the US. Nuclear Engineering International. London, England.
Pnueli, A. (1977). The temporal logic of programs. *Proceedings of the 18th IEEE Symposium on Foundations of Computer Science*, 45–57.
Popkorn, S. (1994). *First steps in modal logic*. Cambridge University Press, Cambridge, England.
Schroeder, M. (1998). *Autonomous, model-based diagnosis agents*. Kluwer Academic Publisher.
Shoham, Y. (1987). Temporal logics in AI: Semantical and ontological considerations. *Artificial Intelligence 33*, 89–104.
Wooldridge, M., & Jennings, N.R. (1995). Intelligent agents: Theory and practice. *Knowledge Engineering Review*.

Alessio Lomuscio (born 1969) is Research Assistant at the Department of Electronic Engineering, Queen Mary and Westfield College, University of London. He obtained a 'Laurea' in Electronic Engineering from Politecnico di Milano (I) in 1995. Between 1996 and 1998 he has been studying for a PhD at the School of Computer Science of the University of Birmingham (UK) under the supervision of Dr. Mark Ryan. He successfully defended his PhD thesis on *Knowledge sharing among ideal agents* in April 1999. His research interests focus on logic and its application in artificial intelligence. He has presented a number of articles at international conferences, he is member of several international research projects and has served as editor for the the special issue on electronic agents of the ACM journal *Crossroads*. For further details, see HTTP://www.cs.bham.ac.uk/~ar1.

Mark Ryan (born 1962) is Lecturer in Computer Science at the University of Birmingham (U.K.). He obtained B.A. and M.A. degrees from the University of Cambridge in 1986 and 1989, and a Ph.D. in Computer Science from Imperial College, University of London, in 1992. His research interests include logic and its applications in computer science and artificial intelligence. He is principal investigator on several national and international research projects, including *Feature Integration in Requirements Engineering* (funded by the European Union), *Automatic Verification of Randomized Distributed Algorithms* (EPSRC), and Feature Specification Languages (British Telecom). For further details, see www.cs.bham.ac.uk/~mdr.

https://doi.org/10.1017/S0890060499132062 Published online by Cambridge University Press