Verification within the KARO Agent Theory*

Ullrich Hustadt[1], Clare Dixon[1], Renate A. Schmidt[2], Michael Fisher[1], John-Jules Meyer[3], and Wiebe van der Hoek[3]

[1] Centre for Agent Research and Development, Manchester Metropolitan University, UK
{U.Hustadt,C.Dixon,M.Fisher}@doc.mmu.ac.uk
[2] Department of Computer Science, University of Manchester, UK
schmidt@cs.man.ac.uk
[3] Department of Computer Science, University of Utrecht and Amsterdam, The Netherlands
{jj,wiebe}@cs.uu.nl

Abstract. This paper discusses automated reasoning in the KARO framework. The KARO framework accommodates a range of expressive modal logics for describing the behaviour of intelligent agents. We concentrate on a core logic within this framework, in particular, we describe two new methods for providing proof methods for this core logic, discuss some of the problems we have encountered in their design, and present an extended example of the use of the KARO framework and the two proof methods.

1 Introduction

The spread of computer technology has meant that more dynamic, complex and distributed computational systems are being developed. Understanding and managing such complexity is notoriously difficult and it is to support this that the agent-based systems paradigm was introduced. In order to reason about agent-based systems, a number of theories of rational agency have been developed, for example the BDI [20] and KARO [23] frameworks.

The KARO logic (for Knowledge, Abilities, Results and Opportunities) falls within a tradition of the use of *modal logic(s)* for describing the behaviour of intelligent agents. As such it builds upon an even longer tradition of philosophical logic where modal logic is employed to describe intensional notions such as knowledge, belief, time, obligation, desire, etc. These very notions are deemed to be appropriate for specifying, designing and implementing intelligent agents [25]. When following first-order logic approaches to the specification of these intensional concepts (such as the situation calculus) one has to build in the relevant properties of these concepts in an ad hoc manner, whereas in modal logic

* This research was supported by a travel grant of the Netherlands Organization for Scientific Research (NWO) and the British Council under the UK-Dutch Joint Scientific Research Programme JRP 442.

this can be done systematically using modal correspondence theory. Moreover, the availability of modal operators yields expressions in modal logic that are more concise than their first-order counterparts. Besides this natural adequacy of modal logic in this context, also computational issues are important: generally, propositional modal logics may be viewed as decidable fragments of full first-order logic, and the computational properties of (uncombined) modal logics are well-investigated [6, 11].

A popular approach to formal verification, particularly of temporal properties, concerns model-checking [4]. This is appropriate when a (finite-state) structure exists (or can be generated efficiently) upon which logical formulae can be tested. In particular, model checking verifies that the structure is a model for the required formula. In our case, we are interested in verifying properties of logical specifications, rather than finite-state structures. Although such models can be built directly from formulae, this is usually expensive. In addition, little work has been carried out concerning model-checking for the types of complex multi-modal logics we require (an exception being the work described in [2]). Consequently, we are here interested in carrying out verification via logical proof.

Thus, the aim of this paper is to examine proof methods for the KARO framework [23]. In particular, we study two approaches to the problem of proof which will be presented in Sections 3 and 4:

- proof methods for the fusion of $\mathcal{PDL}$ and $\mathsf{S5}_{(m)}$ based upon translation to classical logic and first-order resolution; and
- representation of KARO in terms of the fusion of CTL and $\mathsf{S5}_{(m)}$ and proof methods by direct clausal resolution on this combined logic.

Both approaches provide decision procedures for the particular combination of logics under consideration. Thus, unlike approaches which make use of full first-order logic, *unprovability* of a formulae with respect to a agent specification can be shown by each of two approaches automatically without reliance on model-theoretic consideration outside the actual proof method.

We illustrate how the KARO framework and the two approaches can be used by studying a small block world example in Section 5. We conclude with a discussion of the relative strength of the two approaches and a possible reconciliation of the two approaches.

2 Basic KARO Elements

We base our formal methods on the KARO logic [16, 22], a formal system that may be used to *specify*, *analyse* and *reason about* the behaviour of rational agents. In this paper we concentrate on one particular variant of the KARO framework and define a core subsystem for which we are able to provide sound, complete, and terminating inference systems.

Formally, the language of the KARO framework is defined over three primitive types: (i) a set of countably infinite atomic propositional variables, (ii) a set

Ag of agent names (a finite subset of the positive integers), and (iii) a set $\mathsf{Ac_{at}}$ of countably infinite atomic action variables.

Formulae are defined inductively as follows.

- $\top$ is an atomic propositional formula;
- $\varphi \vee \psi$ and $\neg\varphi$ are propositional formula provided so are φ and ψ;
- $\mathbf{K}_i\varphi$ (knowledge), $\langle \mathtt{do}_i(\alpha)\rangle\varphi$ (achievement of results by actions), $\mathbf{A}_i\alpha$ (ability), $\mathbf{O}_i\alpha$ (opportunity), $\mathbf{W}_i^s\varphi$ (selected wish), and $\Diamond_i\varphi$ (implementability) are propositional formulae, provided i is an agent name, α is an action formula and φ is a propositional formula;
- $\mathtt{id}$ (skip) is an atomic action formula;
- $\alpha \vee \beta$ (non-deterministic choice), $\alpha\,;\beta$ (sequencing), $\varphi!$ (confirm), $\alpha^{(n)}$ (bounded repetition), and $\alpha^\star$ (unbounded repetition) are action formulae, provided α and β are action formulae, φ is a propositional formula, and n is a natural number.

Implicit connectives include $\bot$, $\wedge$, $\rightarrow$, ... for propositional formulae, the duals of $\mathbf{K}_i$ and $\langle \mathtt{do}_i(\alpha)\rangle$ (denoted by $[\mathtt{do}_i(\alpha)]$), as well as $\mathbf{P}_i(\alpha,\varphi) = \langle \mathtt{do}_i(\alpha)\rangle\varphi \wedge \mathbf{A}_i\alpha$.

The semantics of the core KARO logic is based on *interpretations* $\mathcal{M} = (W, V, D, I, M)$, where (i) W is a non-empty set of worlds, (ii) V maps propositional variables to subsets of W, (iii) for every $i \in \mathsf{Ag}$ and every $a \in \mathsf{Ac_{at}}$, D contains a binary relation $r_{(i,a)}$ on W and a subset $c_{(i,a)}$ of W, (iv) I contains an equivalence relation K_i on W for each agent $i \in \mathsf{Ag}$, and (v) M contains a serial relation relation W_i on W for each agent $i \in \mathsf{Ag}$. Following the characterisation of agent theories in the introduction, D, I, and M comprise the dynamic, informational, and motivational components in the semantics of the core KARO logic. The relations r and sets c are extended to $\mathsf{Ag} \times \mathsf{Ac}$-sorted relations r^* and sets c^* in a way standard for dynamic logic.

The semantics of well-formed formulae of the KARO logic is defined as follows.

$$
\begin{array}{ll}
\mathcal{M}, w \models \top & \\
\mathcal{M}, w \models p & \text{iff } w \in V(p) \\
\mathcal{M}, w \models \neg\varphi & \text{iff } \mathcal{M}, w \not\models \varphi \\
\mathcal{M}, w \models \varphi \vee \psi & \text{iff } \mathcal{M}, w \models \varphi \text{ or } \mathcal{M}, w \models \psi \\
\mathcal{M}, w \models [\mathtt{do}_i(\alpha)]\varphi & \text{iff } \forall v \in W\ ((w,v) \in r^*_{(i,\alpha)} \rightarrow \mathcal{M}, v \models \varphi) \\
\mathcal{M}, w \models \mathbf{A}_i\alpha & \text{iff } w \in c^*_{(i,\alpha)} \\
\mathcal{M}, w \models \mathbf{O}_i\alpha & \text{iff } \mathcal{M}, w \models \langle \mathtt{do}_i(\alpha)\rangle\top \\
\mathcal{M}, w \models \mathbf{W}_i^s\varphi & \text{iff } \forall v \in W\ ((w,v) \in W_i \rightarrow \mathcal{M}, v \models \varphi) \\
\mathcal{M}, w \models \mathbf{K}_i\varphi & \text{iff } \forall v \in W\ ((w,v) \in K_i \rightarrow \mathcal{M}, v \models \varphi) \\
\mathcal{M}, w \models \Diamond_i\varphi & \text{iff } \exists k \in \mathbb{N}\ \exists a_1, \ldots, a_k \in \mathsf{Ac_{at}}\ \mathcal{M}, w \models \mathbf{P}_i(a_1; \ldots; a_k, \varphi)
\end{array}
$$

If $\mathcal{M}, w \models \varphi$ we say φ *holds at* w (in $\mathcal{M}$) or φ is *true in* w. A formula φ is *satisfiable* iff there is an interpretation $\mathcal{M}$ and a world w such that $\mathcal{M}, w \models \varphi$.

We refer to the logic defined above as the *KARO logic* even though it does not include all the features of the KARO framework. In this paper we make the following simplifying assumptions: (i) we assume $\mathbf{A}_i\alpha = \langle \mathtt{do}_i(\alpha)\rangle\top$, (ii) we

$$
\begin{array}{rlrl}
\neg\langle \mathtt{do}_i(\alpha)\rangle\psi & \Rightarrow [\mathtt{do}_i(\alpha)]\neg\psi & \neg[\mathtt{do}_i(\alpha)]\psi & \Rightarrow \langle \mathtt{do}_i(\alpha)\rangle\neg\psi \\
\langle \mathtt{do}_i(\alpha)\rangle(\psi \vee \phi) & \Rightarrow \langle \mathtt{do}_i(\alpha)\rangle\psi \vee \langle \mathtt{do}_i(\alpha)\rangle\phi & [\mathtt{do}_i(\alpha)](\psi \wedge \phi) & \Rightarrow [\mathtt{do}_i(\alpha)]\psi \wedge [\mathtt{do}_i(\alpha)]\phi \\
\langle \mathtt{do}_i(\alpha \vee \beta)\rangle\psi & \Rightarrow \langle \mathtt{do}_i(\alpha)\rangle\psi \vee \langle \mathtt{do}_i(\beta)\rangle\psi & [\mathtt{do}_i(\alpha \vee \beta)]\psi & \Rightarrow [\mathtt{do}_i(\alpha)]\psi \wedge [\mathtt{do}_i(\beta)]\psi \\
\langle \mathtt{do}_i(\alpha\,;\beta)\rangle\psi & \Rightarrow \langle \mathtt{do}_i(\alpha)\rangle\langle \mathtt{do}_i(\beta)\rangle\psi & [\mathtt{do}_i(\alpha\,;\beta)]\psi & \Rightarrow [\mathtt{do}_i(\alpha)][\mathtt{do}_i(\beta)]\psi \\
\langle \mathtt{do}_i(\mathtt{id})\rangle\psi & \Rightarrow \psi & [\mathtt{do}_i(\mathtt{id})]\psi & \Rightarrow \psi \\
\langle \mathtt{do}_i(\phi!)\rangle\psi & \Rightarrow \phi \wedge \psi & [\mathtt{do}_i(\phi!)]\psi & \Rightarrow \neg\phi \vee \psi \\
\langle \mathtt{do}_i(\alpha^{(0)})\rangle\psi & \Rightarrow \psi & [\mathtt{do}_i(\alpha^{(0)})]\psi & \Rightarrow \psi \\
\langle \mathtt{do}_i(\alpha^{(n+1)})\rangle\psi & \Rightarrow \langle \mathtt{do}_i(\alpha)\rangle\langle \mathtt{do}_i(\alpha^{(n)})\rangle\psi & [\mathtt{do}_i(\alpha^{(n+1)})]\psi & \Rightarrow [\mathtt{do}_i(\alpha)][\mathtt{do}_i(\alpha^{(n)})]\psi
\end{array}
$$

Fig. 1. Transformation rules for the core KARO logic

exclude the unbounded repetition operator $\alpha^\star$ and wishes $\mathbf{W}_i^s\varphi$ from the language, and (iii) there is no interaction between the dynamic and informational component. This fragment of the KARO logic is called the *core KARO logic*.

The proof-theoretical requirements induced by the implementability operator are quite strong, and both approaches which will presented in the following two sections will weaken these requirements in different ways.

3 Proof by Translation

The translation approach to modal reasoning is based on the idea that inference in (combinations of) modal logics can be carried out by translating modal formulae into first-order logic and using conventional first-order theorem proving. There are various different translation morphisms for modal logics whose properties vary with regards the extent to which they are able to map modal logics into first-order logic, the decidability of the fragments of first-order logic into which modal formulae are translated, and the computational behaviour of first-order theorem provers on these fragments [6, 13, 15, 21].

In the following we present a decision procedure for the satisfiability problem in the core KARO logic consisting of three components: (i) a normalisation function which reduces complex action formulae to atomic action subformulae, (ii) a particular translation of normalised formulae into a fragment of first-order logic, and (iii) a resolution-based decision procedure for this fragment.

The normalisation function maps any formula φ of the core KARO logic to its normal form $\varphi\downarrow$ under the rewrite rules given in Figure 1. It is straightforward to see that the rewrite relation defined by these rules is confluent and terminating. Thus, the normal form $\varphi\downarrow$ of φ is logically equivalent to φ, it is unique, and in the absence of the unbounded repetition operator, $\varphi\downarrow$ contains no non-atomic action formulae.

Lemma 1. *The core KARO logic excluding the operator $\Diamond_i$ equivalently reduces to the fusion of multi-modal $\mathsf{K}_{(m)}$ and $\mathsf{S5}_m$.*

The particular translation we use has only recently been proposed by de Nivelle [5] and can be seen as a special case of the T-encoding introduced by Ohlbach [18]. It allows for conceptually simple decision procedures for extensions of $\mathsf{K4}$ by ordered resolution without any reliance on loop checking or similar techniques.

Without loss of generality we assume that the modal formulae under consideration are normalised and in negation normal form. We define a translation function π as follows.

$$\begin{aligned}
\pi(\langle \mathtt{do}_i(a)\rangle\varphi, x) &= \exists y\,(\mathtt{do}_i^a(x,y) \wedge \pi(\varphi, y)) & \pi(\top, x) &= \top \\
\pi(\mathbf{K}_i\varphi, x) &= q_{\mathbf{K}_i\varphi}(x) & \pi(p, x) &= q_p(x) \\
\pi(\mathbf{O}_i\alpha, x) &= \pi(\langle \mathtt{do}_i(\alpha)\rangle\top, x) & \pi(\neg\varphi, x) &= \neg\pi(\varphi, x) \\
\pi(\mathbf{A}_i\alpha, x) &= \pi(\langle \mathtt{do}_i(\alpha)\rangle\top, x) & \pi(\varphi \vee \psi, x) &= \pi(\varphi, x) \vee \pi(\psi, x) \\
\pi(\Diamond_i\varphi, x) &= \exists y\,\pi(\varphi, y)
\end{aligned}$$

a is an atomic action, p is a propositional variable, q_p is a unary predicate symbol uniquely associated with p, $q_{\mathbf{K}_i\varphi}$ is a predicate symbol uniquely associated with $\mathbf{K}_i\varphi$, and $\mathtt{do}_i^a$ is a binary predicate symbol associated with a and i which represent the relation $r_{(i,a)}$ in the semantics.

Finally, let $\Pi(\psi)$ be the formula

$$\exists x\,\pi(\psi, x) \wedge \textstyle\bigwedge_{\mathbf{K}_i\varphi \in \Gamma_{\mathbf{K}}(\psi)} \mathrm{Ax}(\mathbf{K}_i\varphi),$$

where $\Gamma_{\mathbf{K}}(\psi)$ is the set of subformulae of the form $\mathbf{K}_i\varphi$ in ψ, and $\mathrm{Ax}(\mathbf{K}_i\varphi)$ is the formula

$$\begin{aligned}
&\forall x\,(q_{\mathbf{K}_i\varphi}(x) \leftrightarrow \forall y\,(K_i(x,y) \rightarrow \pi(\varphi, y))) \\
\wedge\ &\forall x, y\,((q_{\mathbf{K}_i\varphi}(x) \wedge K_i(x,y)) \rightarrow q_{\mathbf{K}_i\varphi}(y)) \\
\wedge\ &\forall x, y\,((q_{\mathbf{K}_i\varphi}(y) \wedge K_i(x,y)) \rightarrow q_{\mathbf{K}_i\varphi}(x)) \wedge \forall x\, K_i(x,x).
\end{aligned}$$

Based on the close correspondence between the translation morphism Π and the semantics of the core KARO logic it is possible to prove the following.

Theorem 2. *Let ψ be a formula in the core KARO logic excluding the operator $\Diamond_i$. Then $\Pi(\psi)$ is first-order satisfiable iff there exists a model $\mathcal{M}$ and a world w such that $\mathcal{M}, w \models \psi$.*

Proof. The only problem in this theorem is caused by the fact that $\Pi(\psi)$ does not ensure that the relations K_i in a first-order model of $\Pi(\psi)$ are not necessarily equivalence relations while this is the fact for the corresponding relations K_i in the modal model. This problem can be overcome along the lines of de Nivelle [5] or Hustadt et al. [14].

Using certain structual transformation techniques, described for example in [6, 19], $\Pi(\psi)$ can be embedded into a number of solvable clausal classes, for example, the classes $\mathcal{S}^+$ [9] and DL^* [6]. In the following, by $\mathrm{CL}_{\mathrm{DL}^*}(\Pi(\psi))$ we denote an embedding of $\Pi(\psi)$ into the class DL^*. A decision procedures for DL^* can be formulated in the resolution framework of Bachmair and Ganzinger [1] using an ordering refinement of resolution.

Theorem 3 (Soundness, completeness, and termination [6]). *Let ψ be a formula of the core KARO logic excluding $\Diamond_i$ and let $N = \mathrm{CL}_{\mathrm{DL}^*}(\Pi(\psi))$. Let $\succ$ be any ordering which is compatible with the strict subterm ordering and let $\mathsf{R}^{\succ}$ be the ordered resolution calculus restricted by $\succ$. Then:*

1. *Any derivation from N in $\mathsf{R}^{\succ}$ terminates in double exponential time.*
2. *φ is unsatisfiable iff the empty clause can be derived from N.*

All the results presented also hold for the core KARO logic including $\Diamond_i$ under our assumption that $\mathbf{A}_i\alpha = \langle \mathtt{do}_i(\alpha)\rangle\top$. If we drop this assumption, then the translation by π and Π does no longer preserve satisfiability. Although it is possible to devise alternative translations π' and Π' (into second-order logic) such that Theorem 2 holds for the core KARO logic including $\Diamond_i$, Theorem 3 would be invalid for Π'. This is not difficult to see if we consider the problem of showing that $\Diamond_i\varphi$ is not true at a world w in an interpretation $\mathcal{M}$. This is the case iff $\forall k \in \mathbb{N}\ \forall a_1, \ldots, a_k \in \mathsf{Ac_{at}}\ \mathcal{M}, w \not\models \mathbf{P}_i(a_1; \ldots; a_k, \varphi)$. Due to the (universal) quantification over $\mathbb{N}$, proving that $\Diamond_i\varphi$ is not true at a world w requires *inductive theorem proving*. This is outside the scope of first-order resolution and termination is not guaranteed.

Since our emphasis is on practical inference methods to support the deductive verification of agents, we have opted for a simplified core logic.

4 Proof by Clausal Temporal Resolution

Here, we use the simple observation that the use of $\mathcal{PDL}$ in the KARO framework is very similar to the use of branching-time temporal logic. Thus, we attempt to use a simple CTL branching-time temporal logic to represent the dynamic component of the core KARO logic. Dynamic operators and implementability are replaced by CTL formulae by the following rules: for example,

$$\begin{array}{rcl} \langle \mathtt{do}_i(a)\rangle\varphi & \text{is replaced by} & \mathbf{E}\bigcirc(\mathrm{done}_i(a) \wedge \varphi), \\ [\mathtt{do}_i(a)]\varphi & \text{is replaced by} & \mathbf{A}\bigcirc(\mathrm{done}_i(a) \Rightarrow \varphi) \text{ and} \\ \Diamond_i\varphi & \text{is replaced by} & \mathbf{E}\Diamond\varphi, \end{array}$$

where $\mathrm{done}_i(a)$ is a propositional variable uniquely associated with agent i and atomic action a. Initially we work in the logic CTL, with multi-modal S5, the semantics of which are given in, for example [12], with no interactions.

Formulae in the fusion of CTL and $\mathsf{S5}_{(n)}$ can rewritten into a normal form, called SNF_{karo}, that separates temporal and modal aspects (as is done in [8]). Formulae in SNF_{karo} are of the general form $\mathbf{A}\Box^* \bigwedge_i T_i$ where $\mathbf{A}\Box^*$ is the universal relation (which can be defined in terms of the operators "everyone knows" and "common knowledge") and each T_i is a *clause* of one of the following forms.

$$\begin{array}{lll} \mathbf{start} \Rightarrow \bigvee_{k=1}^{n} L_k & & (\textit{initial} \text{ clauses}) \\ \bigwedge_{j=1}^{m} L'_j \Rightarrow \mathbf{A}\bigcirc \bigvee_{k=1}^{n} L_k & \bigwedge_{j=1}^{m} L'_j \Rightarrow \mathbf{E}\bigcirc(\bigvee_{k=1}^{n} L_k)_{\langle \mathsf{c}_i\rangle} & (\textit{step} \text{ clauses}) \\ \bigwedge_{j=1}^{m} L'_j \Rightarrow \mathbf{A}\Diamond L & \bigwedge_{j=1}^{m} L'_j \Rightarrow \mathbf{E}\Diamond L_{\langle \mathsf{c}_i\rangle} & (\textit{sometime} \text{ clauses}) \\ \mathbf{true} \Rightarrow \bigvee_{k=1}^{n} M_k^i & & (\mathbf{K}_i \text{ clauses}) \\ \mathbf{true} \Rightarrow \bigvee_{k=1}^{n} L_k & & (\text{literal clauses}) \end{array}$$

where L'_j, L_k, and L are literals and M^i_k are either literals, or modal literals involving the modal operator $\mathbf{K}_i$. Further, each $\mathbf{K}_i$ clause has at least one disjunct that is a modal literal. $\mathbf{K}_i$ clauses are sometimes known as *knowledge clauses*. Each step and sometime clause that involves the **E**-operator is labelled by an index of the form $\langle \mathsf{c}_i \rangle$ similar to the use of Skolem constants in first-order logic. This index indicates a particular path and arises from the translation of formulae such as $\mathbf{E}(L \mathcal{U} L')$. During the translation to the normal form such formulae are translated into several **E** step clauses and a **E** sometime clause (which ensures that L' must actually hold). To indicate that all these clauses refer to the same path they are annotated with an index. The outer '$\mathbf{A}\Box^*$' operator that surrounds the conjunction of clauses is usually omitted. Similarly, for convenience the conjunction is dropped and we consider just the set of clauses T_i. We denote the normalisation function into SNF_{karo} by τ.

In the following we present a resolution-based calculus for SNF_{karo}. In contrast to the translation approach described in the previous section, this calculus works directly on SNF_{karo} formulae. The inference rules are divided into initial resolution rules, knowledge resolution rules, step resolution rules, and temporal resolution rules, which will be described in the following. We present sufficient rules to understand the following example, the full set of knowledge rules can be found in [8] and the full set of step and temporal resolution rules are given in [3]. In the following, if L is a literal, then $\sim L$ denotes A if $L = \neg A$ and it denotes $\neg L$, otherwise.

Initial Resolution. A literal clause may be resolved with an initial clause (IRES1) or two initial clauses may be resolved together (IRES2) as follows

$$\text{[IRES1]}\quad \frac{\mathbf{true} \Rightarrow (C \vee L) \qquad \mathbf{start} \Rightarrow (D \vee \sim L)}{\mathbf{start} \Rightarrow (C \vee D)} \qquad \text{[IRES2]}\quad \frac{\mathbf{start} \Rightarrow (C \vee L) \qquad \mathbf{start} \Rightarrow (D \vee \sim L)}{\mathbf{start} \Rightarrow (C \vee D)}$$

where C and D are disjunctions of literals.

Knowledge Resolution. During knowledge resolution we apply the following rules which are based on the modal resolution system introduced by Mints [17]. In general we may only apply a (knowledge) resolution rule between two literal clauses, a knowledge and a literal clause, or between two knowledge clauses relating to the same modal operator e.g. two $\mathbf{K}_1$ clauses.

$$\text{[KRES1]}\ \frac{\mathbf{true} \Rightarrow C \vee M \qquad \mathbf{true} \Rightarrow D \vee \sim M}{\mathbf{true} \Rightarrow C \vee D} \qquad \text{[KRES4]}\ \frac{\mathbf{true} \Rightarrow C \vee \neg \mathbf{K}_i L \qquad \mathbf{true} \Rightarrow D \vee L}{\mathbf{true} \Rightarrow C \vee \mathrm{mod}(D)}$$

The function $\mathrm{mod}(D)$ used in KRES4 is defined on disjunctions D of literals or modal literals, as follows.

$$\mathrm{mod}(\mathbf{K}_i L) = \mathbf{K}_i L \qquad \mathrm{mod}(\neg \mathbf{K}_i L) = \neg \mathbf{K}_i L$$
$$\mathrm{mod}(A \vee B) = \mathrm{mod}(A) \vee \mathrm{mod}(B) \qquad \mathrm{mod}(L) = \neg \mathbf{K}_i \sim L$$

Two further rules KRES2 and KRES3 allow resolution between $\mathbf{K}_i L$ and $\mathbf{K}_i{\sim}L$, and between $\mathbf{K}_i L$ and ${\sim}L$.

Finally given a clause involving a disjunction of literals or modal literals of the form $\mathbf{K}_i L$ we can remove the $\mathbf{K}_i$ operators (KRES5) obtaining a literal clause for use during step and temporal resolution. For the complete set of modal resolution rules see [8].

Step Resolution. 'Step' resolution consists of the application of standard classical resolution to formulae representing constraints at a particular moment in time, together with simplification rules for transferring contradictions within states to constraints on previous states. Simplification and subsumption rules are also applied. In the following $\mathbf{P}$ is either path operator.

$$\text{[SRES2]}\ \frac{\begin{array}{r} P \Rightarrow \mathbf{E}{\bigcirc}(F \vee L)_{\langle \mathsf{c}_i \rangle} \\ Q \Rightarrow \mathbf{A}{\bigcirc}(G \vee {\sim}L) \end{array}}{(P \wedge Q) \Rightarrow \mathbf{E}{\bigcirc}(F \vee G)_{\langle \mathsf{c}_i \rangle}} \qquad \text{[SRES4]}\ \frac{Q \Rightarrow \mathbf{P}{\bigcirc}\mathbf{false}}{\mathbf{true} \Rightarrow {\sim}Q}$$

We also allow resolution between two $\mathbf{A}$ step clauses (SRES1) and two $\mathbf{E}$ step clauses (SRES3) with the same index. A step clause may be resolved with a literal clause (where G is a disjunction of literals) and any index is carried to the resolvent to give the following resolution rules.

$$\text{[SRES5]}\ \frac{\begin{array}{r} P \Rightarrow \mathbf{A}{\bigcirc}(F \vee L) \\ \mathbf{true} \Rightarrow (G \vee {\sim}L) \end{array}}{P \Rightarrow \mathbf{A}{\bigcirc}(F \vee G)} \qquad \frac{\begin{array}{r} P \Rightarrow \mathbf{E}{\bigcirc}(F \vee L)_{\langle \mathsf{c}_i \rangle} \\ \mathbf{true} \Rightarrow (G \vee {\sim}L) \end{array}}{P \Rightarrow \mathbf{E}{\bigcirc}(F \vee G)_{\langle \mathsf{c}_i \rangle}}$$

The complete set of step rules can be found in [3]

Temporal Resolution. During temporal resolution the aim is to resolve one of the sometime clauses, $Q \Rightarrow \mathbf{P}{\Diamond}L$, with a set of clauses that together imply ${\Box}{\sim}L$ along the same path, for example a set of clauses that together have the effect of $F \Rightarrow {\bigcirc}{\Box}{\sim}L$. However the interaction between the '$\bigcirc$' and '$\Box$' operators makes the definition of such a rule non-trivial and further the translation to SNF_{karo} will have removed all but the outer level of $\Box$-operators. So, resolution will be between a sometime clause and a *set* of clauses that together imply an $\Box$-formula that occurs on the same path, which will contradict the $\Diamond$-clause. The details of these rules can be found in [3].

Theorem 4 (Soundness, completeness, and termination). *Let φ be a formula of the core KARO logic excluding the operator $\Diamond\!\!\!\!-_i$ and let $N = \tau(\varphi)$. Then:*

1. *Any derivation from N terminates.*
2. *φ is unsatisfiable iff N has a refutation by the temporal resolution procedure described above.*

The proofs are analogous to those in [7, 8, 10].

Again we have excluded the operator $\Diamond\!\!\!\!-_i$ in Theorem 4. The transformation τ which replaces $\Diamond\!\!\!\!-_i \varphi$ by $\mathbf{E}{\Diamond}\varphi$ does not preserve satisfiability. We are currently investigating an alternative transformation τ' which expands $\Diamond\!\!\!\!-_i \varphi$ with

$\varphi \vee \mathbf{E}(\bigvee_{a \in \mathsf{Ac_{at}}}(c_i^a \wedge \bigcirc done_i^a)))\mathcal{U}\varphi)$ which seems to reflect the semantics of $\Diamond_i\varphi$ more faithfully if we drop the assumption that $\mathbf{A}_i\alpha = \langle \mathtt{do}_i(\alpha)\rangle\top$.

Recall that we have observed in the previous section that the core KARO logic excluding $\Diamond_i$ reduces to the fusion of multi-modal $\mathsf{K}_{(m)}$ and $\mathsf{S5}_m$, while in this section we have used a reduction to the fusion of CTL and $\mathsf{S5}_m$. This may seem unreasonable, since the satisfiability problem in $\mathsf{K}_{(m)}$ is (only) PSPACE-complete, while the satisfiability problem in CTL is EXPTIME-complete. However, as τ is a polynomial reduction mapping, the complexity of testing the satisfiability of $\tau(\varphi)$ is the same as testing the satisfiability of φ. Moreover, the apparent possibility to provide a satisfiability equivalence preserving mapping of $\Diamond_i\varphi$ into SNF_{karo} provides a justification.

5 Eve in a Blocks World

Consider two agents, Adam and Eve, living in a blocks world containing three blocks b, c, and d. We use $\mathrm{on}(X, Y)$ and $\mathrm{clear}(X)$ to describe that a block Y is on top of a block X and that no block is on top of X, respectively. A tower consists of three distinct blocks X_1, X_2, X_3 such that X_3 is clear, X_3 is on X_2, and X_2 is on X_1 (axiom (C_1)). We allow only one atomic action: $\mathrm{put}(X, Y)$, which has the effect of Y being placed on X. Eve has the ability of performing a $\mathrm{put}(X, Y)$ action if and only if X and Y are clear, Y is not identical to X, and Y is not equal to c (axiom (A_1)). The axiom (E_1) describes the effects of performing a put action: After any action $\mathrm{put}(X, Y)$ the block Y is on X and X is no longer clear. The axioms (N_1) to (N_3) describe properties of the blocks world which remain unchanged by performing an action. For example, if block Z is clear and not equal to some block X, then putting some arbitrary block Y (possibly identical to Z) on X leaves Z clear (axiom (N_1)). Additionally, the axioms themselves, except for (I_1), remain true irrespective of the actions which are performed.

$$(A_1) \quad \mathbf{A}_E\mathrm{put}(X,Y) \equiv (\mathrm{clear}(X) \wedge \mathrm{clear}(Y) \wedge X \neq Y \wedge Y \neq c)$$

$$(E_1) \quad \rightarrow [\mathtt{do}_i(\mathrm{put}(X,Y))](\mathrm{on}(X,Y) \wedge \neg\mathrm{clear}(X))$$

$$(N_1) \quad (\mathrm{clear}(Z) \wedge Z \neq X) \rightarrow [\mathtt{do}_i(\mathrm{put}(X,Y))](\mathrm{clear}(Z))$$

$$(N_2) \quad (\mathrm{on}(V,Z) \wedge Z \neq Y) \rightarrow [\mathtt{do}_i(\mathrm{put}(X,Y)))](\mathrm{on}(V,Z))$$

$$(N_3) \quad (X = Y) \wedge (U \neq V) \rightarrow [\mathtt{do}_i(\alpha)](X = Y \wedge U \neq V)$$

$$(C_1) \quad \mathrm{tower}(X_1,X_2,X_3) \equiv \textstyle\bigwedge_{i\neq j}(X_i \neq X_j) \wedge \mathrm{on}(X_1,X_2) \wedge \mathrm{on}(X_2,X_3) \wedge \mathrm{clear}(X_3)$$

$$(I_1) \quad \mathbf{K}_E\mathrm{clear}(c) \wedge \mathbf{K}_E\mathrm{clear}(d)$$

In the axioms above i is an element of $\{A, E\}$ where A and E denote Adam and Eve. Recall that in the core KARO logic we identify $\mathbf{A}_i\alpha$ with $\langle \mathtt{do}_i(\alpha)\rangle\top$. Consequently, the axiom (A_1) becomes

$$(A_3') \quad \langle \mathtt{do}_E(\mathrm{put}(X,Y))\rangle\top \equiv (\mathrm{clear}(X) \wedge \mathrm{clear}(Y) \wedge X \neq Y \wedge Y \neq c)$$

In the following we will prove that the axioms (A_1) to (C_1) together with (I_1) imply that if Eve knows that Adam puts block c on block b, then she knows that she can implement the tower (b, c, d), that is, we show that the assumption

$$(K_1) \qquad \mathbf{K}_E\langle \mathtt{do}_A(\mathrm{put}(b,c))\rangle\top \wedge \neg\mathbf{K}_E\Diamond_E \mathrm{tower}(b,c,d)$$

leads to a contradiction.

Although the problem is presented in a first order setting, as we have a finite domain we can easily form all ground instances of the axioms in our specification. Thus, in the following, an expression 'on(b, c)' denotes a propositional variable uniquely associated with the atom on(b, c) in our specification. Due to axiom (N_3) which states that equality and inequality of blocks remains unaffected by Eve's actions, we can eliminate all equations from the instantiated axioms.

Solving the Eve Example By Translation. We will first show how we obtain a refutation for the specification of Eve's blocks world using the translation approach. Let ψ be the conjunction of the axioms (A_1) to (C_1), (I_1), and (K_1). Then $\mathrm{CL}_{\mathrm{DL}^*}(\Pi(\psi))$ contains amongst others the following clauses which will be used in our refutation. The axioms from which a particular clause originates are indicated in square brackets to the left of the clause. Recall that $\pi(p, x) = q_p(x)$ where q_p is a unary predicate symbol uniquely associated with the propositional variable p. To simplify our notation we will write 'on(b, c, x)' instead of '$q_{\mathrm{on}(b,c)}(x)$'. Note that the translation of the axiom (A_3') and the left conjunction of (K_1) contain existential quantifiers which lead to the introduction of Skolem functions during the transformation to clausal normal form. Consequently, the clauses (1) and (14) contain unary Skolem functions g_c^d and g_b^c, respectively. These Skolem functions are associated with particular actions, namely, put(c, d) and put(b, c), respectively. In addition, the Skolem constant ϵ is introduced by Π itself.

$[A_3']$	(1)	$\neg\mathrm{clear}(c,y) \vee \neg\mathrm{clear}(d,y) \vee \mathtt{do}_E^{\mathrm{put}(c,d)}(x, g_c^d(x))_*$
$[E_1]$	(2)	$\neg\mathtt{do}_A^{\mathrm{put}(b,c)}(x,y)_* \vee \mathrm{on}(b,c,y)$
$[E_1]$	(3)	$\neg\mathtt{do}_E^{\mathrm{put}(c,d)}(x,y)_* \vee \mathrm{on}(c,d,y)$
$[N_1]$	(4)	$\neg\mathrm{clear}(c,x) \vee \neg\mathtt{do}_A^{\mathrm{put}(b,c)}(x,y)_* \vee \mathrm{clear}(c,y)$
$[N_1]$	(5)	$\neg\mathrm{clear}(d,x) \vee \neg\mathtt{do}_A^{\mathrm{put}(b,c)}(x,y)_* \vee \mathrm{clear}(d,y)$
$[N_1]$	(6)	$\neg\mathrm{clear}(d,x) \vee \neg\mathtt{do}_E^{\mathrm{put}(c,d)}(x,y)_* \vee \mathrm{clear}(d,y)$
$[N_2]$	(7)	$\neg\mathrm{on}(b,c,x) \vee \neg\mathtt{do}_E^{\mathrm{put}(c,d)}(x,y)_* \vee \mathrm{on}(b,c,y)$
$[C_1]$	(8)	$\neg\mathrm{on}(b,c,x) \vee \neg\mathrm{on}(c,d,x) \vee \neg\mathrm{clear}(d,x) \vee \mathrm{tower}(b,c,d,x)_*$
$[K_1]$	(9)	$q_{\mathbf{K}_E\langle \mathtt{do}_E(\mathrm{put}(b,c))\rangle\top}(\epsilon)$
$[K_1]$	(10)	$\neg q_{\mathbf{K}_E\Diamond_E \mathrm{tower}(b,c,d)}(\epsilon)$
$[K_1]$	(11)	$q_{\mathbf{K}_E\Diamond_E \mathrm{tower}(b,c,d)}(x) \vee K_E(x, h_{K_E}(x))_*$
$[K_1]$	(12)	$q_{\mathbf{K}_E\Diamond_E \mathrm{tower}(b,c,d)}(x) \vee \neg\mathrm{tower}(b,c,d,y)_*$
[Ax]	(13)	$\neg q_{\mathbf{K}_E\langle \mathtt{do}_E(\mathrm{put}(b,c))\rangle\top}(x) \vee \neg\mathbf{K}_E(x,y)_* \vee q_{\langle \mathtt{do}_E(\mathrm{put}(b,c))\rangle\top}(y)$
[Ax]	(14)	$\neg q_{\langle \mathtt{do}_E(\mathrm{put}(b,c))\rangle\top}(x) \vee \mathtt{do}_A^{\mathrm{put}(b,c)}(x, g_b^c(x))_*$
[Ax]	(15)	$\neg q_{\mathbf{K}_E\mathrm{clear}(c)}(x) \vee \neg\mathbf{K}_E(x,y)_* \vee \mathrm{clear}(c,y)$
[Ax]	(16)	$\neg q_{\mathbf{K}_E\mathrm{clear}(d)}(x) \vee \neg\mathbf{K}_E(x,y)_* \vee \mathrm{clear}(d,y)$

$[I_1]$ (17) $q_{\mathbf{K}_E \mathrm{clear}(d)}(\epsilon)$
$[I_1]$ (18) $q_{\mathbf{K}_E \mathrm{clear}(d)}(\epsilon)$

We have obtained the refutation of $\mathrm{CL}_{\mathrm{DL}^*}(\Pi(\psi))$ by using the first-order theorem prover SPASS 1.0.0 [24] which implements the resolution framework of Bachmair and Ganzinger [1]. As an ordering we used a recursive path ordering. Since any recursive path ordering is compatible with the strict subterm ordering, SPASS is a decision procedure by Theorem 3. In every non-unit clause we marked the maximal literal of the clause by an index $\cdot_*$. Thus, inference steps are restricted to these literals.

We observe that clause (12) consists of two variable-disjoint subclauses. This clause will be subject to splitting which introduces two branches into our search space: One on which the unit clause $q_{\mathbf{K}_E \Diamond_E \mathrm{tower}(b,c,d)}(x)$ is an element of the clause set and one on which the unit clause $\neg\mathrm{tower}(b,c,d,y)$ is an element of the clause set instead. For the first set of clauses we directly obtain a contradiction using clause (10). For the second set of clauses

[12.2] (19) $\neg\mathrm{tower}(b,c,d,y)_*$

replaces clause (12). We see that among the clause (1) to (17), only (1), (8), (14), and (11) contain a positive literal which is maximal and can thus serve as positive premises in resolution steps. We can derive among others the following clauses.

[11.2,13.2] (20) $q_{\mathbf{K}_E \mathrm{tower}(b,c,d)}(x) \vee \neg q_{\mathbf{K}_E \langle \mathtt{do}_E(\mathrm{put}(b,c)) \rangle \top}(x) \vee q_{\langle \mathtt{do}_E(\mathrm{put}(b,c)) \rangle \top}(h_{K_E}(x))_*$
[11.2,15.2] (21) $q_{\mathbf{K}_E \mathrm{tower}(b,c,d)}(x) \vee \neg q_{\mathbf{K}_E \mathrm{clear}(c)}(x) \vee \mathrm{clear}(c, h_{K_E}(x))$
[11.2,16.2] (22) $q_{\mathbf{K}_E \mathrm{tower}(b,c,d)}(x) \vee \neg q_{\mathbf{K}_E \mathrm{clear}(d)}(x) \vee \mathrm{clear}(d, h_{K_E}(x))$
[14.2, 2.2] (23) $\neg q_{\langle \mathtt{do}_E(\mathrm{put}(b,c)) \rangle \top}(x) \vee \mathrm{on}(b, c, g^c_b(x))_*$
[14.2, 4.2] (24) $\neg\mathrm{clear}(c,x) \vee \neg q_{\langle \mathtt{do}_E(\mathrm{put}(b,c)) \rangle \top}(x) \vee \mathrm{clear}(c, g^c_b(x))_*$
[14.2, 5.2] (25) $\neg\mathrm{clear}(d,x) \vee \neg q_{\langle \mathtt{do}_E(\mathrm{put}(b,c)) \rangle \top}(x) \vee \mathrm{clear}(d, g^c_b(x))_*$
[1.3, 3.1] (26) $\neg\mathrm{clear}(c,x) \vee \neg\mathrm{clear}(d,x) \vee \mathrm{on}(c, d, g^d_c(x))_*$
[1.3, 6.2] (27) $\neg\mathrm{clear}(c,x) \vee \neg\mathrm{clear}(d,x) \vee \mathrm{clear}(d, g^d_c(x))_*$
[1.3, 7.2] (28) $\neg\mathrm{clear}(c,x) \vee \neg\mathrm{clear}(d,x) \vee \neg\mathrm{on}(b,c,x) \vee \mathrm{on}(b, c, g^d_c(x))_*$
[8.4,19.1] (29) $\neg\mathrm{clear}(d,x) \vee \neg\mathrm{on}(b,c,x) \vee \neg\mathrm{on}(c,d,x)_*$

Intuitively, clause (29) says that there is no situation x in which the blocks b, c, and d form a tower. The remainder of the derivation shows that this assumption leads to a contradiction. We choose clause (26) to derive the following clause.

[26.3,29.3] (30) $\neg\mathrm{clear}(c,x) \vee \neg\mathrm{clear}(d,x) \vee \neg\mathrm{clear}(d, g^d_c(x)) \vee \neg\mathrm{on}(b, c, g^d_c(x))_*$

Note that in clause (30) all literals containing a Skolem term originate from the negative premise (29) while all the remaining literals originate from the positive premise (26). Intuitively, literals containing the Skolem term $g^d_c(x)$ impose constraints on the situation we are in after performing a $\mathrm{put}(c,d)$ action in a situation x, while the remaining literals which have x as their final argument impose constraints on situation x itself.

Since literals containing a Skolem term are deeper than the remaining literals, the ordering restrictions on the resolution inference rule restrict applications of resolution to these literals. In the following part of the derivation we consecutively eliminate these literals by resolution inferences with the clauses (27) and (28) and obtain

$$(31) \quad \neg\mathrm{clear}(c,x) \vee \neg\mathrm{clear}(d,x) \vee \neg\mathrm{on}(b,c,x)_* \qquad$$

Here the literal $\neg\mathrm{on}(b,c,x)$ is maximal and we choose clause (23) which is related to a $\mathrm{put}(b,c)$ action as positive premise.

$$[23.2,31.4] \quad (32) \quad \neg q_{\langle \mathtt{do}_E(\mathrm{put}(b,c))\rangle\top}(x) \vee \neg\mathrm{clear}(c, g^c_b(x)) \vee \neg\mathrm{clear}(d, g^c_b(x))$$

By inference steps with the clauses (24) and (25) we eliminate all literals containing Skolem terms and obtain

$$(33) \quad \neg q_{\langle \mathtt{do}_E(\mathrm{put}(b,c))\rangle\top}(x) \vee \neg\mathrm{clear}(c,x) \vee \neg\mathrm{clear}(d,x)$$

Using clause (20), (21), and (22) we obtain

$$[11.2,15.2] \quad (34) \quad \begin{array}{l} q_{\mathbf{K}_E\mathrm{tower}(b,c,d)}(x) \vee \neg q_{\mathbf{K}_E\langle \mathtt{do}_E(\mathrm{put}(b,c))\rangle\top}(x) \\ \qquad \vee \neg q_{\mathbf{K}_E\mathrm{clear}(c)}(x) \vee \neg q_{\mathbf{K}_E\mathrm{clear}(d)}(x) \end{array}$$

from which we can finally derive a contradiction by (10), (9), (17), and (18).

Solving the Eve Example By Temporal Resolution. The specification of the problem can be written as formulae in the normal form as follows. For example (E_1) instantiated where $X = c$ and $Y = d$ can be written as the following two rules.

$$\begin{array}{l} \mathbf{true} \rightarrow \mathbf{A}\bigcirc(\neg\mathrm{done}_E(\mathrm{put}(c,d)) \vee \mathrm{on}(c,d)) \\ \mathbf{true} \rightarrow \mathbf{A}\bigcirc(\neg\mathrm{done}_E(\mathrm{put}(c,d)) \vee \neg\mathrm{clear}(c)) \end{array}$$

The conjunction of initial conditions is rewritten by a new proposition v and the conjuncts $\mathbf{K}_E\mathrm{clear}(c)$ and $\mathbf{K}_E\mathrm{clear}(d)$ can be can be written as follows

$$\begin{array}{ll} (I_2) & \mathbf{start} \rightarrow v \\ (I_3) & \mathbf{true} \rightarrow \neg v \vee \mathbf{K}_E\mathrm{clear}(c) \\ (I_4) & \mathbf{true} \rightarrow \neg v \vee \mathbf{K}_E\mathrm{clear}(d) \end{array}$$

Translating (K_1) into the CTL formulation we obtain

$$\mathbf{K}_E\mathbf{E}\bigcirc(\mathrm{done}_A(\mathrm{put}(b,c))) \wedge \neg\mathbf{K}_E\mathbf{E}\Diamond\mathrm{tower}(b,c,d).$$

Next we rewrite into the normal form introducing new propositions w, x, y, z and replacing $\mathrm{tower}(b,c,d)$ with its definition. Thus w replaces the above conjunction (G_1), y replaces the subformula $\mathbf{E}\bigcirc(\mathrm{done}_A(\mathrm{put}(b,c)))$ (G_2), (G_3), and z replaces $\neg\mathbf{E}\Diamond\mathrm{tower}(b,c,d)$ (G_4) which is equivalent to $\mathbf{A}\Box\neg\mathrm{tower}(b,c,d)$. The latter formula is rewritten into SNF_{karo} using the new proposition x (G_5), (G_6), (G_7).

$$(G_1) \qquad \mathbf{start} \rightarrow w$$

(G_2) $\quad \mathbf{true} \rightarrow \neg w \vee \mathbf{K}_E y$

(G_3) $\quad y \rightarrow \mathbf{E}\bigcirc(\mathrm{done}_A(\mathrm{put}(b,c)))$

(G_4) $\quad \mathbf{true} \rightarrow \neg w \vee \neg\mathbf{K}_E \neg z$

(G_5) $\quad \mathbf{true} \rightarrow \neg z \vee x$

(G_6) $\quad x \rightarrow \mathbf{A}\bigcirc x$

(G_7) $\quad \mathbf{true} \rightarrow \neg x \vee \neg\mathrm{on}(b,c) \vee \neg\mathrm{on}(c,d) \vee \neg\mathrm{clear}(d)$

Firstly, we apply the rules SRES2 and SRES5 to (G_6), (G_7), and instantiations of (N_1), (N_2), (E_1), and (A_1) given below

(N_1) $\quad \mathrm{clear}(d) \rightarrow \mathbf{A}\bigcirc(\neg\mathrm{done}_E(\mathrm{put}(c,d)) \vee \mathrm{clear}(d))$

(N_2) $\quad \mathrm{on}(b,c) \rightarrow \mathbf{A}\bigcirc(\neg\mathrm{done}_E(\mathrm{put}(c,d)) \vee \mathrm{on}(b,c))$

(E_1) $\quad \mathbf{true} \rightarrow \mathbf{A}\bigcirc(\neg\mathrm{done}_E(\mathrm{put}(c,d)) \vee \mathrm{on}(c,d))$

(A_1) $\quad \mathrm{clear}(c) \wedge \mathrm{clear}(d) \rightarrow \mathbf{E}\bigcirc\mathrm{done}_E(\mathrm{put}(c,d))_{\langle \mathtt{c}_1 \rangle}$

obtaining

$$x \wedge \mathrm{clear}(d) \wedge \mathrm{on}(b,c) \wedge \mathrm{clear}(c) \rightarrow \mathbf{E}\bigcirc\mathbf{false}_{\langle \mathtt{c}_1 \rangle}.$$

An application of SRES4 to this step clause results in

(G_8) $\quad \mathbf{true} \rightarrow \neg x \vee \neg\mathrm{clear}(d) \vee \neg\mathrm{on}(b,c) \vee \neg\mathrm{clear}(c).$

Next we again apply the rules SRES2, and SRES5 to (G_6), (G_8), (G_3) and the following instantiations of (N_1) and (E_1)

(N_1) $\quad \mathrm{clear}(c) \rightarrow \mathbf{A}\bigcirc(\neg\mathrm{done}_A(\mathrm{put}(b,c)) \vee \mathrm{clear}(c))$

(N_1) $\quad \mathrm{clear}(d) \rightarrow \mathbf{A}\bigcirc(\neg\mathrm{done}_A(\mathrm{put}(b,c)) \vee \mathrm{clear}(d))$

(E_1) $\quad \mathbf{true} \rightarrow \mathbf{A}\bigcirc(\neg\mathrm{done}_A(\mathrm{put}(b,c)) \vee \mathrm{on}(b,c))$

obtaining

$$\mathrm{clear}(c) \wedge \mathrm{clear}(d) \wedge x \wedge y \rightarrow \mathbf{E}\bigcirc\mathbf{false}_{\langle \mathtt{c}_2 \rangle}.$$

With an application of SRES4 to this clause we obtain

(G_9) $\quad \mathbf{true} \rightarrow \neg x \vee \neg y \vee \neg\mathrm{clear}(c) \vee \neg\mathrm{clear}(d)$

and then resolving with (G_5) using KRES1 we derive the following.

(G_{10}) $\quad \mathbf{true} \rightarrow \neg z \vee \neg y \vee \neg\mathrm{clear}(c) \vee \neg\mathrm{clear}(d)$

(G_{10}) is then resolved with (G_4) using KRES4 to obtain

(G_{11}) $\quad \mathbf{true} \rightarrow \neg w \vee \neg\mathbf{K}_E y \vee \neg\mathbf{K}_E\mathrm{clear}(c) \vee \neg\mathbf{K}_E\mathrm{clear}(d)$

which can be resolved with the initial conditions (I_3), (I_4), and (G_2) using KRES1 to obtain

(G_{12}) $\quad \mathbf{true} \rightarrow \neg w \vee \neg v.$

Finally resolving (G_{12}) with (I_2) and (G_1) using IRES1 the contradiction

$$\mathbf{start} \rightarrow \mathbf{false}$$

is obtained.

6 Conclusion

In this paper we have considered a fragment of the KARO framework of rational agency called the core KARO logic. We presented two approaches to providing practical proof methods for the core KARO logic, namely, an approach based on a combination of a translation of formulae in the core KARO logic to first-order logic and theorem proving by an ordering refined resolution calculus, and an approach based on a combination of an embedding of the core KARO logic into the fusion of CTL and $\mathsf{S5}_{(m)}$, a normal form transformation for this logic, and theorem proving by a non-classical resolution calculus.

Both approaches are able to provide sound, complete, and terminating proof methods for the core KARO logic excluding the implementability operator. We have discussed the problems with the implementability operator with respect to both approaches and suggested a possible solution in one of the approaches.

A comparion of the two approaches shows that the translation approach allows to deal quite elegantly with the informational component of KARO while the clausal temporal resolution approach has a better potential to provide a complete calculus for the dynamic component of KARO, in particular, in the presence of unbounded repetition. We believe that a detailed analysis of both approaches will help us to overcome their respective limitations. We also hope that it will help us to improve the practicality of both methods, by pointing out redundancies present in the proof search of either method.

An alternative approach is to consider the combination of both approaches taking their respective strength into account. In [14] we present a combination of clausal temporal resolution (restricted to a linear time temporal logic) and the translation approach plus first-order resolution (restricted to extensions of the multi-modal logic $\mathsf{K}_{(m)}$), and we were able to show soundness, completeness, and termination of this combination for a range of combined logics. We are confident that we can extend this combination to cover the fusion of CTL and various modal logics plus additional operators like implementability, which would bring us closer to our goal of providing proof methods for the core KARO logic and beyond.

References

1. L. Bachmair and H. Ganzinger. Resolution theorem proving. To appear in J. A. Robinson and A. Voronkov, editors, *Handbook of Automated Reasoning*.
2. M. Benerecetti, F. Giunchiglia, and L. Serafini. Model checking multiagent systems (extended abstract). In *Proc. ATAL-98*, volume 1555 of *LNAI*. Springer, 1999.
3. A. Bolotov and M. Fisher. A clausal resolution method for ctl branching time temporal logic. *Journal of Experimental and Theoretical Artificial Intelligence*, 11:77–93, 1999.
4. E. M. Clarke, E. A. Emerson, and A. P. Sistla. Automatic verification of finite-state concurrent systems using temporal logic specifications. *ACM Trans. on Programming Languages and Systems*, 8(2):244–263, 1986.
5. H. de Nivelle. Translation of S4 into GF and 2VAR. Manuscript, May 1999.

6. H. De Nivelle, R. A. Schmidt, and U. Hustadt. Resolution-based methods for modal logics. *Logic Journal of the IGPL*, 8(3):265–292, 2000.
7. C. Dixon, M. Fisher, and A. Bolotov. Resolution in a Logic of Rational Agency. In *Proc. ECAI 2000*, pages 358–362. IOS Press, 2000.
8. C. Dixon, M. Fisher, and M. Wooldridge. Resolution for temporal logics of knowledge. *Journal of Logic and Computation*, 8(3):345–372, 1998.
9. C. Fermüller, A. Leitsch, T. Tammet, and N. Zamov. *Resolution Method for the Decicion Problem*, volume 679 of *LNCS*. Springer, 1993.
10. M. Fisher, C. Dixon, and M. Peim. Clausal Temporal Resolution. ACM Transactions on Computational Logic, 2(1), 2001.
11. R. Goré. Tableau methods for modal and temporal logics. In M. D'Agostino, D. Gabbay, R. Hähnle, and J. Posegga, editors, *Handbook of Tableau Methods*, pages 297–396. Kluwer, 1999.
12. J. Y. Halpern and M. Y. Vardi. The complexity of reasoning about knowledge and time I: Lower bounds. *Journal of Computer and System Sciences*, 38:195–237, 1989.
13. U. Hustadt. *Resolution-Based Decision Procedures for Subclasses of First-Order Logic*. PhD thesis, Universität des Saarlandes, Saarbrücken, Germany, 1999.
14. U. Hustadt, C. Dixon, R. A. Schmidt, and M. Fisher. Normal forms and proofs in combined modal and temporal logics. In *Proc. FroCoS'2000*, volume 1794 of *LNAI*, pages 73–87. Springer, 2000.
15. U. Hustadt and R. A. Schmidt. Using resolution for testing modal satisfiability and building models. To appear in the *Journal of Automated Reasoning*, 2001.
16. B. van Linder, W. van der Hoek, and J.-J. Ch. Meyer. Formalizing abilities and opportunities of agents. *Fundamenta Informaticae*, 34(1,2):53–101, 1998.
17. G. Mints. Gentzen-type systems and resolution rules. Part I: Propositional logic. In *Proc. COLOG-88*, volume 417 of *LNCS*, pages 198–231. Springer, 1990.
18. H. J. Ohlbach. Combining Hilbert style and semantic reasoning in a resolution framework. In *Proc. CADE-15*, volume 1421 of *LNAI*, pages 205–219. Springer, 1998.
19. D. A. Plaisted and S. Greenbaum. A structure-preserving clause form translation. *Journal of Symbolic Computation*, 2:293–304, 1986.
20. A. S. Rao and M. P. Georgeff. Modeling agents withing a BDI-architecture. In *Proc. KR-91*, pages 473–484. Morgan Kaufmann, 1991.
21. R. A. Schmidt. Decidability by resolution for propositional modal logics. *Journal of Automated Reasoning*, 22(4):379–396, 1999.
22. B. van Linder, W. van der Hoek, and J.-J. Ch. Meyer. Communicating rational agents. In *Proc. KI-94*, volume 861 of *LNAI*, pages 202–213. Springer, 1994.
23. B. van Linder, W. van der Hoek, and J.-J. Ch. Meyer. How to motivate your agents. In *Intelligent Agents II*, volume 1037 of *LNAI*. Springer, 1996.
24. C. Weidenbach et al. System description: SPASS version 1.0.0. In *Proc. CADE-16*, volume 1632 of *LNAI*, pages 378–382. Springer, 1999.
25. G. Weiß, editor. *Multiagent systems: A modern approach to distributed artificial intelligence*. MIT Press, 1999.