Verification within the KARO Agent Theory

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

[1] Department of Computer Science
University of Liverpool, UK
U.Hustadt@csc.liv.ac.uk
C.Dixon@csc.liv.ac.uk
M.Fisher@csc.liv.ac.uk
wiebe@csc.liv.ac.uk

[2] Department of Computer Science
University of Manchester, UK
schmidt@cs.man.ac.uk

[3] Institute of Information and Computing Sciences
University of Utrecht, The Netherlands
jj@cs.uu.nl
wiebe@cs.uu.nl

Summary. This chapter proposes two new methods for realising automated reasoning within agent-based systems. We concentrate on a core of the KARO framework, which is a specification framework for modelling intelligent agent behaviour. We discuss the advantages of each approach and suggest ways of extending each variant to cover more of the KARO framework.

1 Introduction

The use of *agents* is now seen as an essential tool in representing, understanding and implementing complex software systems. In particular, the characterisation of complex components as *intelligent* or *rational* agents allows the system designer to analyse applications at a much higher level of abstraction [13, 47]. In order to describe and reason about such agents, a number of theories of rational agency have been developed, for example the BDI [33] and KARO [45] frameworks. Usually, these frameworks are represented as complex multi-modal logics. These logics, in addition to their use in agent theories, where the basic representation of agency and rationality is explored, form the basis for agent-based formal methods. In both uses, (automated) theorem proving is of vital importance. In agent theories, automated theorem proving allows us to examine properties of the overall theory and, in some cases, to characterise computation within that theory. In agent-based formal methods, theorem proving is clearly important in developing verification techniques.

The leading agent theories and formal methods in this area all share similar logical properties. Usually, the agent theories have:

- an *informational* component, being able to represent an agent's beliefs (by the modal logic $\mathsf{KD45}$) or knowledge (by the modal logic $\mathsf{S5}$),
- a *dynamic* component, allowing the representation of dynamic activity (by temporal or dynamic logic), and,
- a *motivational* component, often representing the agent's desires, intentions or goals (by the modal logic KD).

Thus, the predominant approaches use particular combinations of modal logics.

(For definitions of the modal logics mentioned in this chapter we refer the reader to [4].)

The particular agent theory that we consider here is the KARO framework, KARO is short for Knowledge, Abilities, Results and Opportunities [28]. The KARO framework combines actions, knowledge, and wishes via propositional dynamic logic PDL, $\mathsf{S5}_{(m)}$, and $\mathsf{KD}_{(m)}$, respectively.

While proof methods have been developed for other agent theories like the BDI framework [34], no such methods exist for the KARO framework. Thus, our aim in this paper is to examine possible approaches to the development of automated proof methods for the KARO framework. We study two approaches to the problem of proof in this complex system:

1. Proof methods for the fusion of PDL and $\mathsf{S5}_{(m)}$ based upon translation to classical logic and first-order resolution;
2. 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.

These approaches both show how we can verify properties of agent-based systems represented in the KARO theory of rational agents, but there are fundamental differences in the techniques used. The first approach involves translating all modal and dynamic logic aspects into classical logic and then carrying out proof by defining specific orderings on classical resolution. The second approach retains the non-classical structure and develops appropriate resolution rules for the combined logic. In addition, branching time temporal logic, rather than propositional dynamic logic, is used to represent the agent's dynamic behaviour.

2 Basic KARO Elements

The KARO logic [45, 28] is a formal system that may be used to *specify*, *analyse* and *reason about* the behaviour of rational agents. Concerning the informational attitudes of agents, in the basic framework [45], it can be expressed that agent i *knows* a fact φ (written as $\mathbf{K}_i\varphi$). The modality $\mathbf{K}_i$ is a standard $\mathsf{S5}$ modality. Consequently, the informational component of KARO is a multi-modal $\mathsf{S5}_{(m)}$ logic. In the full system we also consider beliefs; these epistemic and doxastic attitudes were extensively studied in [27]. On an equal footing with these informational attitudes, the language encompasses a *dynamic* component. Starting with some atomic actions $\mathsf{Ac_{at}}$, KARO allows for composite actions such as sequential composition $(\alpha\,;\beta)$, testing $\varphi!$, conditionals ($\mathtt{if}$ φ $\mathtt{then}$ α $\mathtt{else}$ β), repetition ($\mathtt{while}$ φ $\mathtt{do}$ α). We also investigated several notions of choice $(\alpha + \beta)$ in [44]. The framework is especially fit to reason about the preconditions for such actions: one can express whether agent i is *able* to perform action α ($\mathbf{A}_i\alpha$) or has the *opportunity* to do α ($\mathbf{O}_i\alpha$), and also that φ is a *result* of doing α ($[\mathtt{do}_i(\alpha)]\varphi$). 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 logic we consider is an extended modal logic given by the fusion of a PDL-like logic and multi-modal $\mathsf{S5}$ and KD. Given two (or more) modal logics L_1 and L_2 formulated in languages $\mathcal{L}_1$ and $\mathcal{L}_2$ with disjoint sets of modal operators, but the same non-modal base language, the *fusion* $\mathsf{L}_1 \oplus \mathsf{L}_2$ of L_1 and L_2 is the smallest modal logic L containing $\mathsf{L}_1 \cup \mathsf{L}_2$. In other words, if L_1 is axiomatised by a set of axioms Ax_1 and L_2 is axiomatised by Ax_2, then $\mathsf{L}_1 \oplus \mathsf{L}_2$ is axiomatised by the union $\mathrm{Ax}_1 \cup \mathrm{Ax}_2$. This means, in particular, that the modal operators in L_1 and L_2 do not interact.

The base language of the KARO framework is defined over three primitive types:

- a countably infinite set P of *propositional variables*,
- a set Ag of *agent names* (a finite subset of the positive integers), and
- a countably infinite set $\mathsf{Ac_{at}}$ of *atomic actions*.

Formulaeare defined inductively as follows.

- $\top$ is an atomic (propositional) formula, and so is every propositional variable in P;
- $(\varphi \vee \psi)$ and $\neg\varphi$ are formulae provided φ and ψ are formulae;
- $\mathbf{K}_i\varphi$ (knowledge), $[\mathtt{do}_i(\alpha)]\varphi$ (achievement), $\mathbf{A}_i\alpha$ (ability), $\mathbf{O}_i\alpha$ (opportunity), $\mathbf{W}_i^s\varphi$ (selected wish), and $\Diamond_i\varphi$ (implementability) are formulae, provided i is an agent name, α is an action formula and φ is a formula;
- $\mathtt{id}$ (skip) is an atomic action formula;
- $(\alpha \vee \beta)$ (non-deterministic choice), $(\alpha\,;\beta)$ (sequencing), $\varphi!$ (confirmation or test), $\alpha^{(n)}$ (bounded repetition), and $\alpha^{\star}$ (unbounded repetition) are action formulae, provided α and β are action formulae, φ is a formula, and n is a natural number (in unary coding).

Implicit connectives include the usual connectives such as $\bot$, $\wedge$, $\rightarrow$, $\ldots$ for (propositional) formulae, the duals of $\mathbf{K}_i$, $\mathbf{O}_i$ and $[\mathtt{do}_i(\alpha)]$ (denoted by $\langle\mathtt{do}_i(\alpha)\rangle$), as well as

$$\begin{aligned}
\mathbf{PracPoss}_i(\alpha,\varphi) &= \langle \mathtt{do}_i(\alpha)\rangle\varphi \wedge \mathbf{A}_i\alpha \\
\mathbf{Can}_i(\alpha,\varphi) &= \mathbf{K}_i\mathbf{PracPoss}_i(\alpha,\varphi) \\
\mathbf{Cannot}_i(\alpha,\varphi) &= \mathbf{K}_i\neg\mathbf{PracPoss}_i(\alpha,\varphi) \\
\mathbf{Goal}_i\varphi &= \neg\varphi \wedge \mathbf{W}^s_i\varphi \wedge \Diamond_i\varphi \\
\mathbf{Intend}_i(\alpha,\varphi) &= \mathbf{Can}_i(\alpha,\varphi) \wedge \mathbf{K}_i\mathbf{Goal}_i\varphi
\end{aligned}$$

We use the following notational convention in this paper. We denote atomic actions, as well as first-order constants, by a, b, c, (non-)atomic actions by α, β, agents by i, j, propositional variables by p, q, formulae by φ, ϕ, ψ, ϑ, first-order variables by x, y, z, terms by s, t, u, functions by f, g, h, predicate symbols by P, Q, R, atoms by A, A_1, A_2, literals by L, and clauses by C, D.

Even though the logic defined above does not include all the features of the KARO framework, we refer to it as the *KARO logic*.

Intuitively, the semantics of KARO logic is a *possible worlds semantics* in which the elements of the primitive types and the operators of KARO logic are interpreted as follows: A *model* consists of a set of *possible worlds*. Each possible world has associated with it a *truth assignment* to the propositional variables in P, that is, each propositional variable is either true or false in a world. Using such a truth assignment, we can determine the truth of purely propositional formulae in the standard way: The atomic propositional formula $\top$ is always true in a world, $\neg\varphi$ is true in a world w iff φ is false in w, $(\varphi \vee \psi)$ is true in a world w iff φ is true in w or ψ is true in w. To understand the remaining operators of KARO logic, we have to take into account the second component of the possible worlds semantics, the *accessibility relations*. First, for each agent $i \in \mathsf{Ag}$, there is a binary equivalence relation K_i on the set of worlds in a model. If v and w are two worlds such that the pair (w, v) is in K_i, then we say that *v is reachable from w via K_i* (reachability for relations other than K_i is defined in the same way). This relation is used to determine the truth of formulae of the form $\mathbf{K}_i\varphi$ as follows: $\mathbf{K}_i\varphi$ is true in a world w iff φ is true in every world v reachable from w via K_i. Second, for each agent $i \in \mathsf{Ag}$, there is a binary serial relation W_i on the set of worlds in a model. This relation is used to define the truth of formulae of the form $\mathbf{W}^s_i\varphi$: $\mathbf{W}^s_i\varphi$ is true in a world w iff φ is true in every world v reachable from w via W_i.

At first sight, using accessibility relations to provide meaning to knowledge and wishes may not look intuitive. However, it turns out that our definition ensures that these operators have many of the properties that we would like them to have. For example, if an agent i knows φ, we would like φ to be true. This is ensured by the fact that the accessibility relation K_i is reflexive. Consequently, if agent i knows φ in world w, that is, $\mathbf{K}_i\varphi$ is true in w, then φ is true in w, since w is reachable from w by K_i. Similarly, we would like the wishes of an agent i to be consistent with each other, that is, if agent i wishes both φ and ψ to be true, then it should be possible for both φ and ψ to be true in a world. This is ensured by the fact that the accessibility relation W_i is serial, that is, for every world w there exists is a world v reachable from w via W_i. For example, if agent i in world w wishes φ to be true, but also wishes $\neg\varphi$ to be true, then both φ and $\neg\varphi$ have to be true in every world v reachable from w via W_i. Since there is at least one such world v, both φ and $\neg\varphi$ have to be true in v, which is impossible. Thus, an agent i in a world w could never wish both φ and its negation to be true.

Accessibility relations also provide meaning to achievement of results by actions $[\mathtt{do}_i(\alpha)]\varphi$ and opportunities $\mathbf{O}_i\alpha$. For each agent i and atomic action $a \in \mathsf{Ac}_{\mathsf{at}}$ a model contains an arbitrary binary relation $r_{(i,a)}$ (i.e., there is no requirement that the relation is reflexive or serial etc). Intuitively, each world v reachable from a world w via $r_{(i,a)}$ is a possible outcome of agent i performing action a in world w. Since $r_{(i,a)}$ is an arbitrary relation, it may well be that for a world w there is no world v reachable from w via $r_{(i,a)}$. Our interpretation of this situation is that agent i does not have the opportunity to perform action a in w. It is also possible that there is more than one world reachable from w via $r_{(i,a)}$. Our interpretation of this situation is that agent i cannot predict or influence which particular world reachable from w via $r_{(i,a)}$, will be the outcome of performing action a in w.

Atomic actions can be used to form more complex actions using action forming operators of KARO, and with each agent and each complex action α we again associate a binary relation $r_{(i,\alpha)}$. With the atomic action `id` we associate the identity relation on the set of worlds. With $(\alpha \vee \beta)$ we associate the union of $r_{(i,\alpha)}$ and $r_{(i,\beta)}$, which means, that the outcome of agent i executing $(\alpha \vee \beta)$ in world w will either be the outcome of executing α or the outcome of executing β, with agent i being incapable of predicting or influencing which one it will be. With $(\alpha\,;\beta)$ we associate the composition of $r_{(i,\alpha)}$ and $r_{(i,\beta)}$, that is, the outcome of $(\alpha\,;\beta)$ is the outcome of first performing α and then performing β. The bounded repetition $\alpha^{(n)}$ is inductively defined to be identical to `id` if $n = 0$ and identical to $(\alpha\,;\alpha^{(n-1)})$ if $n > 0$. With the unbounded repetition $\alpha^\star$ we associate the reflexive, transitive closure of $r_{(i,\alpha)}$, that is, a world v is reachable from w via $r_{(i,\alpha^\star)}$ iff there is a finite sequence of worlds $v = w_0, \ldots, w_n = w$ with $n \geq 0$ such that the pair (w_i, w_{i+1}) is in $r_{(i,\alpha)}$ for every i, $0 \leq i < n$. Finally, with the confirmation action $\varphi!$ we associate the relation consisting of all pairs of worlds (w, w) such that φ is true at w in our model. Note that the accessibility relations of the confirmation action as well as the skip action will be the same for every agent.

We can now state the meaning of achievement of results by actions $[\mathtt{do}_i(\alpha)]\varphi$ and opportunities $\mathbf{O}_i\alpha$: $[\mathtt{do}_i(\alpha)]\varphi$ is true in a world w iff for every world v reachable from w via the relation $r_{(i,\alpha)}$, the formula φ is true in v, that is, φ is true in every world that agent i might be in after executing the action α. On the other hand, $\mathbf{O}_i\alpha$ is true in a world w iff there exists a world v reachable from w via the relation $r_{(i,\alpha)}$ such that φ is true in v, that is, we say that agent i has the opportunity to perform action α in world w iff there is a world v which is a possible outcome of executing α in w. Note that $\mathbf{O}_i\alpha$ is equivalent to $\langle\mathtt{do}_i(\alpha)\rangle\top$, which in turn is equivalent to $\neg[\mathtt{do}_i(\alpha)]\neg\top$. It is an interesting consequence of this definition that every agent i has the opportunity to perform an unbounded repetition $\alpha^\star$ in any world for arbitrary actions α.

To define the meaning of ability $\mathbf{A}_i\alpha$, that is, to define what it means for an agent i to be able to perform an action α we take a slightly different approach. For an atomic action a we simply take note of all the worlds in which agent i is able to perform a. We do so by associating a set $c_{(i,a)}$ of worlds with every agent i and atomic action a. For the atomic action $\mathtt{id}$ we postulate that every agent is able to perform it in every world. We assume that an agent i is able to perform $(\alpha \vee \beta)$ in a world w iff agent i can perform α or β in w. We also assume that an agent i is able to perform a sequence $\alpha\,;\beta$ in w iff agent i is able to perform α in w and for every world v reachable from w via $r_{(i,\alpha)}$, agent i is able to perform β in v. Note that this is a cautious view of an agent's abilities. Performing action α can have more than one outcome; our definition of agent i being able to perform $\alpha\,;\beta$ makes sure that whatever world agent i will end up in after performing α it will have the ability to perform β. The ability to perform a bounded repetition $\alpha^{(n)}$ is again defined inductively: for $n = 0$ it is identical to being able to perform $\mathtt{id}$, and for $n > 0$ it is identical to being able to perform the sequence $(\alpha\,;\alpha^{(n-1)})$. For unbounded repetitions $\alpha^\star$ we say that agent i is able to perform $\alpha^\star$ in w iff agent i is able to perform $\alpha^{(n)}$ in w for every $n \geq 0$.

Finally, the formula $\Diamond_i\varphi$ is true in world w, that is, an agent i can implement φ in world w, iff there exists a natural number $k \geq 0$, and atomic actions $a_1, \ldots, a_k$ such that $\mathbf{PracPoss}_i(a_1;\ldots;a_k,\varphi)$ is true in w. By definition of **PracPoss**, this is the case, iff agent i has the ability to perform the sequence $a_1;\ldots;a_k$ in w and there is a world v reachable from w via $r_{(i,a_1;\ldots;a_k)}$ in which φ is true.

In this paper we make the following simplifying assumptions: (i) we assume $\mathbf{A}_i\alpha = \mathbf{O}_i\alpha = \langle \mathtt{do}_i(\alpha)\rangle\top$, (ii) we exclude the unbounded repetition operator $\alpha^\star$, wishes $\mathbf{W}_i^s\varphi$, and implementability $\Diamond_i\varphi$ 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*. In Section 5 we will discuss in how far these simplifying assumptions can be relaxed.

Example: Eve in a blocks world

To illustrate the ideas presented in this paper, we specify a blocks world problem in KARO logic and later show how the two proof methods which will be presented in Sections 3 and 4 can be used to solve the problem. To make the example more interesting, the specification makes use of the implementability operator which has been excluded from the core KARO logic.

Consider two agents, Adam and Eve, living in a blocks world containing four blocks a, b, c, and d. We use $\mathrm{is\_on}(X,Y)$, $\mathrm{is\_clear}(X)$, $\mathrm{on\_floor}(X)$ to describe that a block Y is on top of a block X, that no block is on top of X, and that X is on the floor, respectively. 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 $\text{put}(X, Y)$ 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 $\text{put}(X, Y)$ the block Y is on X and X is no longer clear. The axioms (N_1) to (N_4) 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 remain true irrespective of the actions which are performed.

$$(A_1) \qquad \mathbf{A}_E\text{put}(X, Y) \equiv (\text{is\_clear}(X) \wedge \text{is\_clear}(Y) \wedge X \neq Y \wedge Y \neq c)$$

$$(E_1) \qquad [\mathtt{do}_i(\text{put}(X, Y))](\text{is\_on}(X, Y) \wedge \neg\text{is\_clear}(X))$$

$$(N_1) \qquad (\text{is\_clear}(Z) \wedge Z \neq X) \rightarrow [\mathtt{do}_i(\text{put}(X, Y))](\text{is\_clear}(Z))$$

$$(N_2) \qquad (\text{is\_on}(V, Z) \wedge Z \neq Y) \rightarrow [\mathtt{do}_i(\text{put}(X, Y)))](\text{is\_on}(V, Z))$$

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

$$(N_4) \qquad (\text{on\_floor}(Z) \wedge Z \neq Y) \rightarrow [\mathtt{do}_i(\text{put}(X, Y))]\text{on\_floor}(Z)$$

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

$$(A_1') \qquad \langle\mathtt{do}_E(\text{put}(X, Y))\rangle\top \equiv (\text{is\_clear}(X) \wedge \text{is\_clear}(Y) \wedge X \neq Y \wedge Y \neq c)$$

A tower is defined as follows.

$$(C_1) \qquad \begin{aligned} \text{tower}(X_1, X_2, X_3, X_4) \equiv {} & \textstyle\bigwedge_{i\neq j}(X_i \neq X_j) \wedge \text{on\_floor}(X_1) \\ & \wedge \text{is\_on}(X_1, X_2) \wedge \text{is\_on}(X_2, X_3) \\ & \wedge \text{is\_on}(X_3, X_4) \wedge \text{is\_clear}(X_4) \end{aligned}$$

That is, a tower consists of four distinct blocks such that the first block is on the floor, the second block is on top of the first block, the third block on top of the second block, the fourth block on top of the third block, and the fourth block is clear.

We are given the initial condition

$$(I) \qquad \begin{aligned} & \mathbf{K}_E\text{is\_on}(a, b) \wedge \mathbf{K}_E\text{is\_clear}(b) \wedge \mathbf{K}_E\text{is\_clear}(c) \\ & \wedge \mathbf{K}_E\text{is\_clear}(d) \wedge \mathbf{K}_E\text{on\_floor}(a) \end{aligned}$$

Axiom (I) states that Eve knows that block b is on block a, block b, c, and d are clear, and block a is on the floor.

In Sections 3 and 4 we will prove, using our two proof methods, that the axioms (A_1) to (C_1) together with (I) imply that if Eve knows that Adam puts block c on block b, then she knows that she can implement the tower (a, b, c, d), that is, we show that the assumption

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

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 'is_on(a, b)' denotes a propositional variable uniquely associated with the atom is_on(a, b) 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.

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 techniques. The basic idea underlying the translation approach is that first-order logic is sufficiently expressive to encode the semantics of a wide range of modal logics. For example, take the informal semantics given in Section 2: In defining the semantics of the various operators of KARO logic we have used the boolean connectives (negation, conjunction, and disjunction) and the universal and existential quantifier. This is exactly how our translation in Table 2 is devised. The only exception is unbounded repetition (excluded from the core KARO logic) which has been defined using the reflexive, transitive closure of a relation. The reflexive, transitive closure of a relation cannot be defined in first-order logic.

Various translation morphisms exist and their 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, see e.g. [8, 14, 21, 23, 36, 37].

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, (iii) a particular transformation of this fragment of first-order logic into the clausal class DL^*, and (iv) a resolution-based decision procedure for DL^*. The first two steps of the procedure could be merged into one. It is perfectly possible to devise a translation morphism for arbitrary formulae of core KARO logic without normalising them first. However, the definition of this translation morphism would be rather tedious and difficult to understand. Introducing normalisation as a first step, allows a concise and straightforward definition of the translation morphism.

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

Table 1. Transformation rules for the core KARO logic

Reduction of complex actions

Using the rewrite rules given in Table 1 and similar rules for $\mathbf{O}_i\alpha$ and $\mathbf{A}_i\alpha$, the normalisation function maps any formula φ of the core KARO logic to a normal form $\varphi\!\downarrow$. The basic idea underlying these rewrite rules is to 'break down' any non-atomic actions occurring in formulae of core KARO logic until only atomic actions remain. It is straightforward to see that the rewrite relation defined by these rules is confluent and terminating, that is, the order in which we apply the rewrite rules will not alter the final result and we can be sure that after a finite number of applications of the rewrite rules no further rule applications will be possible. 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. *Let φ be a formula in the core KARO logic. Then $\varphi\!\downarrow$ is logically equivalent to φ, and $\varphi\!\downarrow$ does not contain any non-atomic action formulae.*

Translation to first-order logic

The idea underlying the particular translation we use was first proposed in [7] and has been developed further in [38], where it has been called *axiomatic translation* and can be seen as a special case of the T-encoding introduced in [31]. It allows for conceptually simple decision procedures for extensions of $\mathsf{K4}$ by ordered resolution. As compared to tableaux-based procedures a feature of this approach is the absence of loop checking mechanisms for transitive modal logics.

Without loss of generality we assume that the modal formulae under consideration are normalised and in negation normal form. We define the translation function π as given in Table 2. 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

$$\begin{aligned}
\pi([\mathtt{do}_i(a)]\varphi, x) &= \forall y\,(R_{(i,a)}(x,y) \to \pi(\varphi,y)) & \pi(\top, x) &= \top\\
\pi(\langle\mathtt{do}_i(a)\rangle\varphi, x) &= \exists y\,(R_{(i,a)}(x,y) \wedge \pi(\varphi,y)) & \pi(p, x) &= Q_p(x)\\
\pi(\mathbf{O}_i\alpha, x) &= \pi(\langle\mathtt{do}_i(\alpha)\rangle\top, x) & \pi(\neg\varphi, x) &= \neg\pi(\varphi, x)\\
\pi(\mathbf{A}_i\alpha, x) &= \pi(\langle\mathtt{do}_i(\alpha)\rangle\top, x) & \pi(\varphi \vee \psi, x) &= \pi(\varphi, x) \vee \pi(\psi, x)\\
\pi(\mathbf{K}_i\varphi, x) &= Q_{\mathbf{K}_i\varphi}(x)
\end{aligned}$$

where 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 $R_{(i,a)}$ is a binary predicate symbol uniquely associated with a and i, which represents the relation $r_{(i,a)}$ in the semantics.

Table 2. Translation morphism π

$$\begin{aligned}
&\forall x\,(Q_{\mathbf{K}_i\varphi}(x) \leftrightarrow \forall y\,(R_{(i,\mathbf{K})}(x,y) \to \pi(\varphi,y)))\\
&\wedge \forall x,y\,(Q_{\mathbf{K}_i\varphi}(x) \wedge R_{(i,\mathbf{K})}(x,y) \to Q_{\mathbf{K}_i\varphi}(y))\\
&\wedge \forall x,y\,(Q_{\mathbf{K}_i\varphi}(y) \wedge R_{(i,\mathbf{K})}(x,y) \to Q_{\mathbf{K}_i\varphi}(x))\\
&\wedge \forall x\; R_{(i,\mathbf{K})}(x,x).
\end{aligned}$$

Here $R_{(i,\mathbf{K})}$ is a binary predicate symbol uniquely associated with the modal operator $\mathbf{K}_i$. No additional definition of $R_{(i,\mathbf{K})}$ is required, in particular, Π does not state the symmetry or transitivity of $R_{(i,\mathbf{K})}$. Note that the translation Π preserves the structure of the core KARO formula, that is, with every subformula occurrence ψ in a core KARO formula φ we can associate a particular subformula occurrence ϑ in $\Pi(\varphi)$ such that $\vartheta = \pi(\psi)$. 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 1. *A formula φ of the core KARO logic is satisfiable iff $\Pi(\varphi)$ is first-order satisfiable.*

One of the advantages of using the axiomatic translation morphism is the fact that for any formula φ of the core KARO logic $\Pi(\varphi)$ can easily be seen to belong to a number of well-known solvable first-order classes, including the two-variable fragment of first-order logic [30], the guarded fragment [1], or the clausal class DL* [8].

A clause C is a DL*-*clause* iff (i) all literals are unary, or binary, (ii) there is no nesting of function symbols, (iii) every functional term in C contains all variables of C, and (iv) every binary literal (even if it has no functional terms) contains all variables of C. A set of clauses N belongs to the class DL* iff all clauses in N are DL*-clauses.

Transformation into DL*

We will now present a structural transformation of first-order formulae into clausal form which will transform translated formulae of the core KARO logic into sets of first-order clauses belonging to the class DL*.

Let $\mathrm{Pos}(\varphi)$ be the set of positions of a first-order formula φ. If λ is a position in φ, then $\varphi|_\lambda$ denotes the subformula of φ at position λ and $\varphi[\psi \mapsto \lambda]$ is the result of replacing $\varphi|_\lambda$ at position λ by ψ. The polarity of (occurrences of) first-order subformulae is defined as usual: Any occurrence of a subformula of an equivalence has *zero polarity*. For occurrences of subformulae not below a '$\leftrightarrow$' symbol, an occurrence of a subformula has *positive polarity* if it is one inside the scope of an even number of (explicit or implicit) negations, and it has *negative polarity* if it is one inside the scope of an odd number of negations.

Structural transformation, also referred to as renaming, associates with each element λ of a set $\Lambda \subseteq \mathrm{Pos}(\varphi)$ a predicate symbol Q_λ and a literal $Q_\lambda(x_1, \ldots, x_n)$, where $x_1, \ldots, x_n$ are the free variables of $\varphi|_\lambda$, the symbol Q_λ does not occur in φ and two symbols Q_λ and $Q_{\lambda'}$ are equal only if $\varphi|_\lambda$ and $\varphi|_{\lambda'}$ are equivalent formulae. Let

$$\begin{aligned}\mathrm{Def}^+_\lambda(\varphi) &= \forall x_1 \ldots x_n.\, Q_\lambda(x_1, \ldots, x_n) \rightarrow \varphi|_\lambda \quad \text{and}\\ \mathrm{Def}^-_\lambda(\varphi) &= \forall x_1 \ldots x_n.\, \varphi|_\lambda \rightarrow Q_\lambda(x_1, \ldots, x_n).\end{aligned}$$

The *definition* of Q_λ is the formula

$$\mathrm{Def}_\lambda(\varphi) = \begin{cases}\mathrm{Def}^+_\lambda(\varphi) & \text{if } \varphi|_\lambda \text{ has positive polarity}\\ \mathrm{Def}^-_\lambda(\varphi) & \text{if } \varphi|_\lambda \text{ has negative polarity}\\ \mathrm{Def}^+_\lambda(\varphi) \wedge \mathrm{Def}^-_\lambda(\varphi) & \text{otherwise.}\end{cases}$$

Based on Def_λ we can inductively define $\mathrm{Def}_\Lambda(\varphi)$, where $\Lambda \subseteq \mathrm{Pos}(\varphi)$, by:

$$\begin{aligned}\mathrm{Def}_\emptyset(\varphi) &= \varphi \quad \text{and}\\ \mathrm{Def}_{\Lambda\cup\{\lambda\}}(\varphi) &= \mathrm{Def}_\Lambda(\varphi[Q_\lambda(x_1, \ldots, x_n) \mapsto \lambda]) \wedge \mathrm{Def}_\lambda(\varphi).\end{aligned}$$

Here λ is maximal in $\Lambda \cup \{\lambda\}$ with respect to the prefix ordering on positions. A *definitional form* of φ is $\mathrm{Def}_\Lambda(\varphi)$, where Λ is a subset of all positions of subformulae (usually, non-atomic or non-literal subformulae).

Theorem 2. *Let φ be a first-order formula.*

1. *φ is satisfiable iff $\mathrm{Def}_\Lambda(\varphi)$ is satisfiable, for any $\Lambda \subseteq \mathrm{Pos}(\varphi)$.*
2. *$\mathrm{Def}_\Lambda(\varphi)$ can be computed in linear time, provided Λ includes all positions of non-literal subformula occurrences and φ is linearised.*

Recall that with every subformula occurrence ψ in a core KARO formula φ we can associate a particular subformula occurrence ϑ in $\Pi(\varphi)$ such that $\vartheta = \pi(\psi)$. So, for every core KARO formula φ we can define a set of $\Lambda(\varphi)$ of positions in $\Pi(\varphi)$ by

$$\begin{aligned}\Lambda(\varphi) = \{\lambda \mid\ & \text{there is a non-literal subformula } \varphi|_{\lambda'} \text{ of } \varphi \text{ and}\\ & \Pi(\varphi)|_\lambda = \pi(\varphi|_{\lambda'})\}.\end{aligned}$$

Then we can show the following.

Lemma 2. *Let φ be a formula of the core KARO logic. Then every clause in the clausal form of $\mathrm{Def}_{\Lambda(\varphi)}(\Pi(\varphi))$ is a DL^*-clause.*

Deduce: $$\frac{N}{N \cup \{\mathrm{Cond}(C)\}}$$

if C is either a resolvent or a factor of clauses in N, and $\mathrm{Cond}(C)$ is the condensation of C.

Delete: $$\frac{N \cup \{C\}}{N}$$

if C is a tautology or N contains a clause which is a variant of C.

Split: $$\frac{N \cup \{C \vee D\}}{N \cup \{C\} \mid N \cup \{D\}}$$

if C and D are variable-disjoint.

Resolvents and factors are derived by the following rules.

Resolution: $$\frac{C \vee A_1 \quad \neg A_2 \vee D}{(C \vee D)\sigma}$$

where (i) σ is a most general unifier of A_1 and A_2, (ii) no literal in C is selected, and $A_1\sigma$ is strictly $\succ$-maximal with respect to $C\sigma$, and (iii) $\neg A_2$ is either selected, or $\neg A_2\sigma$ is maximal with respect to $D\sigma$ and no literal in D is selected.

$C \vee A_1$ is called the *positive premise* and $\neg A_2 \vee D$ the *negative premise*. We implicitly assume that the premises have no common variables.

Factoring: $$\frac{C \vee A_1 \vee A_2}{(C \vee A_1)\sigma}$$

where (i) σ is a most general unifier of A_1 and A_2, and (ii) no literal in C is selected and $A_1\sigma$ is $\succ$-maximal with respect to $C\sigma$.

Table 3. Expansion and inference rules

A first-order resolution calculus

For the clausal class DL* a decision procedure can be formulated in the resolution framework of Bachmair and Ganzinger [2]. In this framework, the resolution calculus is parameterised by two parameters: an admissible ordering $\succ$ and a selection function S. Essentially, an *admissible ordering* is a total (well-founded) strict ordering on the ground level such that for literals $\ldots \succ \neg A_n \succ A_n \succ \ldots \succ \neg A_1 \succ A_1$ holds. This is extended to the non-ground level in a canonical manner. A *selection function* assigns to each clause a possibly empty set of occurrences of negative literals and no restrictions are imposed on the selection function.

The calculus itself consists of general *expansion rules* of the form:

$$\frac{N}{N_1 \mid \cdots \mid N_n}$$

Each represents a finite derivation of alternatives $N_1, \dots, N_n$ from N. The rules given Table 3 describe how derivation trees can be expanded at the leaves. A *derivation* from a set of clauses N is a finitely branching, ordered tree T with root N and nodes being sets of clauses. The tree is constructed by applications of the expansion rules to the leaves so that factoring, splitting and resolution are applied in this order. We assume that no resolution or factoring inference is computed twice on the same branch of the derivation. Any path $N(= N_0), N_1, \dots$ in a derivation T is called a *closed branch* in T iff the clause set $\bigcup_j N_j$ contains the empty clause, otherwise it is called an *open branch.* A derivation T is a *refutation* iff every path $N(= N_0), N_1, \dots$ in it is a closed branch. A derivation T from N is called *fair* iff for any path $N(= N_0), N_1, \dots$ in T, with *limit* $N_\infty = \bigcup_j \bigcap_{k \geq j} N_k$, it is the case that each clause C that can be deduced from non-redundant premises in N_∞ is contained in some N_j. Note that for a finite path $N(= N_0), N_1, \dots N_n$, the limit N_∞ is equal to N_n.

The calculus is refutationally complete and compatible with a general notion of *redundancy* for clauses and inferences, with which additional don't-care non-deterministic simplification and deletion rules can be applied [2]. For our purposes it is sufficient that tautological clauses and variant clauses are eliminated from the clause set during a derivation.

Theorem 3 ([3]). *Let T be a fair derivation from a set N of clauses. Then,*

1. *if $N(= N_0), N_1, \dots$ is a path with limit N_∞, N_∞ is saturated up to redundancy.*
2. *N is satisfiable if and only if there exists a path in T with limit N_∞ such that N_∞ is satisfiable.*
3. *N is unsatisfiable if and only if for every path $N(= N_0), N_1, \dots$ the clause set $\bigcup_j N_j$ contains the empty clause.*

A decision procedure for DL*

A decision procedure for DL* can be obtained using an ordering $\succ$ defined as follows. Let $>_d$ be an ordering on terms which is defined by $s >_d t$ if s is deeper than t, and every variable that occurs in t, occurs deeper in s. Then define $P(s_1, \dots, s_n) \succ_A Q(t_1, \dots, t_n)$ by $\{s_1, \dots, s_n\} >_d^{mul} \{t_1, \dots, t_n\}$, where $>_d^{mul}$ is the multiset extension of $>_d$. Finally, for a negative literal $\neg A$ let $\mathrm{ms}(A)$ denote the multiset $\{A, A\}$, while for a positive literal A, $\mathrm{ms}(A)$ denotes $\{A\}$. We define an ordering $\succ$ on literals by $L_1 \succ L_2$ iff $\mathrm{ms}(L_1) \succ_A \mathrm{ms}(L_2)$.

Theorem 4 ([8, Theorem 5.4]). *Let N be a set of* DL**-clauses and $\succ$ be the ordering of literals defined above. Then,*

1. *$\succ$ is an admissible ordering;*
2. *any derivation from N based on $\succ$ terminates in time double exponential in the size of the signature of N.*

A decision procedure for core KARO logic

We can now put together the four components of our first decision procedure for core KARO logic. Given a formulae φ in the core KARO logic, we proceed by normalising φ using the rules in Table 1, translating the result to first-order logic using the translation morphism Π, transforming the resulting first-order formula to clausal form using Def_Λ and a standard clause form transformation, and finally applying the resolution calculus with the ordering $\succ$ specified above to the set of clauses we obtain.

Theorem 5 (Soundness, completeness, and termination). *Let φ be a formula of core KARO logic and let N be the clausal form of $\mathrm{Def}_{\Lambda(\varphi)}(\Pi(\varphi))$. Then,*

1. *any derivation from N based on $\succ$ terminates in time exponential in the size of the signature of N;*
2. *φ is unsatisfiable iff all branches in any fair derivation with root N are closed.*

The computation of $\varphi\!\downarrow$ using the transformation rules given in Table 1 may require exponential time and the size of $\varphi\!\downarrow$ can be exponential in the size of φ. The translation of $\varphi\!\downarrow$ to first-order logic and the transformation to clausal form requires only linear time. The size of the signature of the resulting clause set N is linear in the size of $\varphi\!\downarrow$. By Theorem 5 the saturation of N requires exponential time in the size of the signature of N. Overall this gives a decision procedure which requires time double exponential in the size of φ.

A variation of the approach can be used to show that the satisfiability problem of core KARO logic is actually PSPACE-complete and to obtain a decision procedure for core KARO logic which requires only polynomial space. Basically, two modifications are required. First, we have to ensure applications of the normalisation function to formulae of the form $\langle \mathtt{do}_i(\alpha \vee \beta)\rangle\psi$ and $[\mathtt{do}_i(\alpha \vee \beta)]\psi$ do not result in formulae in which the subformula ψ occurs twice. This can be achieved by replacing ψ with a new propositional variable q and adding a definition $\forall(q \leftrightarrow \psi)$ for q to the formula, where $\forall$ is the universal modality (see Section 5 for a definition of its semantics). Second, we reduce formulae of $\mathsf{S5}_{(m)}$ to $\mathsf{K}_{(m)}$. Let φ be the result of the first transformation and let n be the number of subformula occurrences of the form $\mathbf{K}_i\psi$ in φ for arbitrary $i \in \mathsf{Ag}$. Let k be a new atomic action not occurring in φ. For each subformula occurrence $\mathbf{K}_i\psi$ we introduce two new propositional variables q_ψ and $q_{\mathbf{K}_i\psi}$, and we let

$$\begin{aligned}\Gamma(\mathbf{K}_i\psi) = {} & \forall(q_\psi \leftrightarrow \psi) \wedge \\ & \forall(q_{\mathbf{K}_i\psi} \leftrightarrow [\mathtt{do}_i(k)]q_\psi) \wedge \\ & (q_{\mathbf{K}_i\psi} \rightarrow (q_{\mathbf{K}_i\psi} \wedge [\mathtt{do}_i(k)]q_{\mathbf{K}_i\psi} \wedge \ldots \wedge [\mathtt{do}_i(k)]^n q_{\mathbf{K}_i\psi})) \wedge \\ & (\neg q_{\mathbf{K}_i\psi} \rightarrow (\neg q_{\mathbf{K}_i\psi} \wedge [\mathtt{do}_i(k)]\neg q_{\mathbf{K}_i\psi} \wedge \ldots \wedge [\mathtt{do}_i(k)]^n \neg q_{\mathbf{K}_i\psi}))\end{aligned}$$

where $[\mathtt{do}_i(k)]^n$ is an abbreviation for $[\mathtt{do}_i(k)]$ repeated n times. Then the second transformation consists of a series of rewrite steps

$$\varphi[\mathbf{K}_i\psi] \Rightarrow \varphi[q_{\mathbf{K}_i\psi}] \wedge \Gamma(\mathbf{K}_i\psi),$$

where ψ itself does not contain any occurrence of a modal operator $\mathbf{K}_j$, until a normal form has been computed. The result $\varphi\Downarrow$ of the two transformations is satisfiability equivalent to the original formula φ. It can be computed in time polynomial in the size of φ, and is of size quadratic in the size of φ. The target logic of the translation can be seen as a notational variant of $\mathcal{ALC}$ with acyclic TBoxes whose satisfiability problem is PSPACE-complete [25]. Therefore:

Theorem 6. *The satisfiability problem of the core KARO logic is PSPACE-complete.*

A computationally space optimal decision procedure for $\mathcal{ALC}$ with acyclic TBoxes, based on translation and a refinement of the resolution calculus using a particular selection function instead of an ordering refinement, can be developed along the lines of [16]. This alternative decision procedure uses only polynomial space in the worst case.

Solving the Eve example by translation

We will now show how we obtain a refutation for the specification of Eve's blocks world using the translation approach. To deal with implementability in the translation approach, we extend the translation morphism π by $\pi(\Diamond_i\varphi, x) = \exists y \,.\, \pi(\varphi, y)$. We will discuss the appropriateness of both definitions in Section 5.

Let ψ be the conjunction of the axioms (A_1) to (C_1), (I), and (K_1). Then $\mathrm{CL}_{\mathrm{DL}^*}(\Pi(\psi))$ contains among 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 'is_on(a, b, x)' instead of '$Q_{\mathrm{is\_on}(a,b)}(x)$'. Note that the translation of the axiom (A_1') 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 (17) contain unary Skolem functions g_b^c and g_c^d, respectively. These Skolem functions are associated with particular actions, namely, put(b, c) and put(c, d), respectively. In addition, the Skolem constant ϵ is introduced by Π itself.

$[A_1']$	(1)	$\neg\mathrm{is\_clear}(c, y) \vee \neg\mathrm{is\_clear}(d, y) \vee R_{(E,\mathrm{put}(c,d))}(x, g_c^d(x))_*$
$[E_1]$	(2)	$\neg R_{(A,\mathrm{put}(b,c))}(x, y)_* \vee \mathrm{is\_on}(b, c, y)$
$[E_1]$	(3)	$\neg R_{(E,\mathrm{put}(c,d))}(x, y)_* \vee \mathrm{is\_on}(c, d, y)$
$[N_1]$	(4)	$\neg\mathrm{is\_clear}(c, x) \vee \neg R_{(A,\mathrm{put}(b,c))}(x, y)_* \vee \mathrm{is\_clear}(c, y)$
$[N_1]$	(5)	$\neg\mathrm{is\_clear}(d, x) \vee \neg R_{(A,\mathrm{put}(b,c))}(x, y)_* \vee \mathrm{is\_clear}(d, y)$
$[N_1]$	(6)	$\neg\mathrm{is\_clear}(d, x) \vee \neg R_{(E,\mathrm{put}(c,d))}(x, y)_* \vee \mathrm{is\_clear}(d, y)$
$[N_2]$	(7)	$\neg\mathrm{is\_on}(a, b, x) \vee \neg R_{(A,\mathrm{put}(b,c))}(x, y)_* \vee \mathrm{is\_on}(a, b, y)$
$[N_2]$	(8)	$\neg\mathrm{is\_on}(a, b, x) \vee \neg R_{(E,\mathrm{put}(c,d))}(x, y)_* \vee \mathrm{is\_on}(a, b, y)$

[N_2] (9) $\neg\text{is\_on}(b,c,x) \vee \neg R_{(E,\text{put}(c,d))}(x,y)_* \vee \text{is\_on}(b,c,y)$
[N_3] (10) $\neg\text{on\_floor}(a,x) \vee \neg R_{(A,\text{put}(b,c))}(x,y)_* \vee \text{on\_floor}(a,y)$
[N_3] (11) $\neg\text{on\_floor}(a,x) \vee \neg R_{(E,\text{put}(c,d))}(x,y)_* \vee \text{on\_floor}(a,y)$
[C_1] (12) $\neg\text{on\_floor}(a,y) \vee \neg\text{is\_on}(a,b,y) \vee \neg\text{is\_on}(b,c,y)$
$\vee \neg\text{is\_on}(c,d,y) \vee \neg\text{is\_clear}(d,y) \vee \text{tower}(a,b,c,d,y)_*$
[K_1] (13) $Q_{\mathbf{K}_E\langle\mathtt{do}_E(\text{put}(b,c))\rangle\top}(\epsilon)$
[K_1] (14) $\neg Q_{\mathbf{K}_E\text{tower}(a,b,c,d)}(\epsilon)$
[K_1] (15) $Q_{\mathbf{K}_E\text{tower}(a,b,c,d)}(x) \vee R_{(E,\mathbf{K})}(x, h_{\mathbf{K}_E}(x))_*$
[K_1] (16) $Q_{\mathbf{K}_E\text{tower}(a,b,c,d)}(x) \vee \neg\text{tower}(a,b,c,d,y)_*$
[Ax] (17) $\neg Q_{\mathbf{K}_E\langle\mathtt{do}_E(\text{put}(b,c))\rangle\top}(x) \vee \neg R_{(E,\mathbf{K})}(x,y)_* \vee Q_{\langle\mathtt{do}_E(\text{put}(b,c))\rangle\top}(y)$
[Ax] (18) $\neg Q_{\langle\mathtt{do}_E(\text{put}(b,c))\rangle\top}(x) \vee R_{(A,\text{put}(b,c))}(x, g^c_b(x))_*$
[Ax] (19) $\neg Q_{\mathbf{K}_E\text{is\_on}(a,b)}(x) \vee \neg R_{(E,\mathbf{K})}(x,y)_* \vee \text{is\_on}(a,b,y)$
[Ax] (20) $\neg Q_{\mathbf{K}_E\text{is\_clear}(b)}(x) \vee \neg R_{(E,\mathbf{K})}(x,y)_* \vee \text{is\_clear}(b,y)$
[Ax] (21) $\neg Q_{\mathbf{K}_E\text{is\_clear}(c)}(x) \vee \neg R_{(E,\mathbf{K})}(x,y)_* \vee \text{is\_clear}(c,y)$
[Ax] (22) $\neg Q_{\mathbf{K}_E\text{is\_clear}(d)}(x) \vee \neg R_{(E,\mathbf{K})}(x,y)_* \vee \text{is\_clear}(d,y)$
[Ax] (23) $\neg Q_{\mathbf{K}_E\text{on\_floor}(a)}(x) \vee \neg R_{(E,\mathbf{K})}(x,y)_* \vee \text{on\_floor}(a,y)$
[I] (24) $Q_{\mathbf{K}_E\text{is\_on}(a,b)}(\epsilon)$
[I] (25) $Q_{\mathbf{K}_E\text{is\_clear}(b)}(\epsilon)$
[I] (26) $Q_{\mathbf{K}_E\text{is\_clear}(c)}(\epsilon)$
[I] (27) $Q_{\mathbf{K}_E\text{is\_clear}(d)}(\epsilon)$
[I] (28) $Q_{\mathbf{K}_E\text{on\_floor}(a)}(\epsilon)$

We have obtained the refutation of $\text{CL}_{\text{DL}^*}(\Pi(\psi))$ by using the first-order theorem prover SPASS 1.0.0 [46] which implements the resolution framework of [2]. 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 5. 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. Finding the refutation takes SPASS less than 0.01 seconds.

We observe that clause (16) 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 $Q_{\mathbf{K}_E\text{tower}(a,b,c,d)}(x)$ is an element of the clause set and one on which the unit clause $\neg\text{tower}(a,b,c,d,y)$ is an element of the clause set instead. For the first set of clauses we directly obtain a contradiction using clause (14). For the second set of clauses

[16.2] (29) $\neg\text{tower}(a,b,c,d,y)_*$

replaces clause (16). We see that among the clause (1) to (16), only (1), (12), (18), and (15) 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.

[15.2, 17.7] (30) $\neg Q_{\mathbf{K}_E\langle\mathtt{do}_E(\text{put}(b,c))\rangle\top}(x) \vee Q_{\langle\mathtt{do}_E(\text{put}(b,c))\rangle\top}(h_{\mathbf{K}_E}(x))_*$
[18.2, 2.2] (31) $\neg Q_{\langle\mathtt{do}_E(\text{put}(b,c))\rangle\top}(x) \vee \text{is\_on}(b,c,g^c_b(x))_*$
[18.2, 4.2] (32) $\neg\text{is\_clear}(c,x) \vee \neg Q_{\langle\mathtt{do}_E(\text{put}(b,c))\rangle\top}(x) \vee \text{is\_clear}(c, g^c_b(x))_*$
[18.2, 5.2] (33) $\neg\text{is\_clear}(d,x) \vee \neg Q_{\langle\mathtt{do}_E(\text{put}(b,c))\rangle\top}(x) \vee \text{is\_clear}(d, g^c_b(x))_*$
[18.2, 7.2] (34) $\neg\text{is\_on}(a,b,x) \vee \neg Q_{\langle\mathtt{do}_E(\text{put}(b,c))\rangle\top}(x) \vee \text{is\_on}(a,b,g^c_b(x))_*$
[18.2, 10.2] (35) $\neg\text{on\_floor}(a,x) \vee \neg Q_{\langle\mathtt{do}_E(\text{put}(b,c))\rangle\top}(x) \vee \text{on\_floor}(a, g^c_b(x))_*$

[1.3, 3.1] (36) $\neg\text{is\_clear}(c,x) \vee \neg\text{is\_clear}(d,x) \vee \text{is\_on}(c,d,g_c^d(x))_*$

[1.3, 6.2] (37) $\neg\text{is\_clear}(c,x) \vee \neg\text{is\_clear}(d,x) \vee \text{is\_clear}(d,g_c^d(x))_*$

[1.3, 8.2] (38) $\neg\text{is\_clear}(c,x) \vee \neg\text{is\_clear}(d,x) \vee \neg\text{is\_on}(a,b,x) \vee \text{is\_on}(a,b,g_c^d(x))_*$

[1.3, 9.2] (39) $\neg\text{is\_clear}(c,x) \vee \neg\text{is\_clear}(d,x) \vee \neg\text{is\_on}(b,c,x) \vee \text{is\_on}(b,c,g_c^d(x))_*$

[1.3, 11.2] (40) $\neg\text{is\_clear}(c,x) \vee \neg\text{is\_clear}(d,x) \vee \neg\text{on\_floor}(a,x) \vee \text{on\_floor}(a,g_c^d(x))_*$

[12.6, 29.1] (41) $\neg\text{on\_floor}(a,x) \vee \neg\text{is\_clear}(d,x) \vee \neg\text{is\_on}(b,c,x) \vee \neg\text{is\_on}(c,d,x) \vee \neg\text{is\_on}(a,b,x)_*$

Intuitively, clause (41) says that there is no situation x in which the blocks a, b, c, and d form a tower. The remainder of the derivation shows that this assumption leads to a contradiction. We choose clause (38) to derive the following clause.

[38.4, 41.2] (42) $\neg\text{is\_clear}(d,x) \vee \neg\text{is\_clear}(c,x) \vee \neg\text{is\_on}(a,b,x) \vee \neg\text{is\_clear}(d,g_c^d(x)) \vee \neg\text{on\_floor}(a,g_c^d(x)) \vee \neg\text{is\_on}(c,d,g_c^d(x)) \vee \neg\text{is\_on}(b,c,g_c^d(x))_*$

Note that in clause (42) all literals containing a Skolem term originate from the negative premise (41) while all the remaining literals originate from the positive premise (38). Intuitively, literals containing the Skolem term $g_c^d(x)$ impose constraints on the situation we are in after performing a $\text{put}(c,d)$ 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 (36), (37), (39), and (40) and obtain

(43) $\neg\text{is\_clear}(d,x) \vee \neg\text{is\_clear}(c,x) \vee \neg\text{is\_on}(a,b,x)_* \vee \neg\text{on\_floor}(a,x) \vee \neg\text{is\_on}(b,c,x)$

Here again the literal $\neg\text{is\_on}(a,b,x)$ is maximal. This time we choose clause (34) which is related to a $\text{put}(b,c)$ action as positive premise.

[34.4, 43.3] (44) $\neg Q_{\langle \mathtt{do}_E(\text{put}(b,c))\rangle\top}(x) \vee \neg\text{is\_on}(a,b,x) \vee \neg\text{is\_clear}(d,g_b^c(x)) \vee \neg\text{is\_clear}(c,g_b^c(x)) \vee \neg\text{on\_floor}(a,g_b^c(x)) \vee \neg\text{is\_on}(b,c,g_b^c(x))_*$

By inference steps with the clauses (31), (32), (33), and (35) we eliminate all literals containing Skolem terms and obtain

(45) $\neg Q_{\langle \mathtt{do}_E(\text{put}(b,c))\rangle\top}(x) \vee \neg\text{is\_on}(a,b,x)_* \vee \neg\text{is\_clear}(d,x) \vee \neg\text{is\_clear}(c,x) \vee \neg\text{on\_floor}(a,x)$

Intuitively, this part of the derivation has established that in any situation x where clause (43) is false, it is possible to perform a $\text{put}(b,c)$ action which results in a situation x' where $\text{is\_on}(b,c,x')$ is true.

Using clause (15), and (19) to (23) we derive

[15.2, 19.2] (46) $\neg Q_{\mathbf{K}_E \mathrm{is\_on}(a,b)}(x) \lor \mathrm{is\_on}(a, b, h_{\mathbf{K}_E}(x))_*$

which is then used to derive

(47) $\neg Q_{\langle \mathsf{do}_E(\mathrm{put}(b,c))\rangle\top}(x) \lor \neg Q_{\mathbf{K}_E \mathrm{is\_on}(a,b)}(x) \lor \neg Q_{\mathbf{K}_E \mathrm{is\_clear}(d)}(x)$
$\lor \neg Q_{\mathbf{K}_E \mathrm{is\_clear}(c)}(x) \lor \neg Q_{\mathbf{K}_E \mathrm{on\_floor}(a)}(x)$

from clause (45). Using clauses (24) to (28) we derive from (47):

(48) $\neg Q_{\langle \mathsf{do}_E(\mathrm{put}(b,c))\rangle\top}(h_{K_E}(x))$
[30.2, 48.1] (49) $\neg Q_{\mathbf{K}_E \langle \mathsf{do}_E(\mathrm{put}(b,c))\rangle\top}(x)$
[13.1, 49.1] (50) $\square$

4 Proof by Clausal Temporal Resolution

Here we use the simple observation that the use of 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, while the epistemic component of core KARO logic remains unchanged. Clausal resolution-based theorem proving is then applied to this branching time temporal logic of knowledge, that is, the fusion of CTL and $\mathsf{S5}_{(m)}$. Resolution-based proof methods for the combination of linear time temporal logic with the modal logic of knowledge S5 are given in [11] and the branching time logic CTL and the modal logic of belief KD45 are presented in [10].

In the subsequent pages we give (i) a translation from the core of KARO to the fusion of CTL and $\mathsf{S5}_{(m)}$, (ii) a translation of formulae in $\mathsf{CTL} \oplus \mathsf{S5}_{(m)}$ into a normal form for this logic, and (iii) a resolution decision procedure for these clauses.

Translation into $\mathsf{CTL} \oplus \mathsf{S5}_{(m)}$

We begin by presenting the syntax and semantics for $\mathsf{CTL} \oplus \mathsf{S5}_{(m)}$. Given a countably infinite set P of *propositional variables* and a set Ag of *agent names*, formulae of $\mathsf{CTL} \oplus \mathsf{S5}_{(m)}$ are defined inductively as follows: $\top$ is a $\mathsf{CTL} \oplus \mathsf{S5}_{(m)}$ formula, every propositional variable in P is a $\mathsf{CTL} \oplus \mathsf{S5}_{(m)}$ formula, if φ and ψ are $\mathsf{CTL} \oplus \mathsf{S5}_{(m)}$ formulae, then $\neg\varphi$, $\varphi \lor \psi$, $\mathbf{A}\Diamond\varphi$, $\mathbf{A}\Box\varphi$, $\mathbf{A}(\varphi\mathcal{U}\psi)$, $\mathbf{A}(\varphi\mathcal{W}\psi)$, $\mathbf{A}\bigcirc\varphi$, $\mathbf{E}\Diamond\varphi$, $\mathbf{E}\Box\varphi$, $\mathbf{E}(\varphi\mathcal{U}\psi)$, $\mathbf{E}(\varphi\mathcal{W}\psi)$, and $\mathbf{E}\bigcirc\varphi$ are $\mathsf{CTL} \oplus \mathsf{S5}_{(m)}$ formulae, if φ is a $\mathsf{CTL} \oplus \mathsf{S5}_{(m)}$ formula and i is an agent name in Ag, then $\mathbf{K}_i\varphi$ is a $\mathsf{CTL} \oplus \mathsf{S5}_{(m)}$ formula.

The semantics of $\mathsf{CTL} \oplus \mathsf{S5}_{(m)}$ formulae is as follows. Let S be a set of *states*. A *tree* is a structure (S, η), where S is the set of states and $\eta \subseteq S \times S$ is a relation between states such that (i) $s_0 \in S$ is a unique root node (i.e. $\neg\exists s_i \in S$ such that $(s_i, s_0) \in \eta$), (ii) for each $s_i \in S$ there exists $s_j \in S$ such that $(s_i, s_j) \in \eta$, and (iii) for all $s_i, s_j, s_k \in S$ if $(s_j, s_i) \in \eta$ and $(s_k, s_i) \in \eta$ then $s_j = s_k$. A *timeline*, t, is an infinitely long, linear, discrete sequence of states, indexed by the natural numbers. Note that timelines correspond to the *runs*

$\mathcal{M},(t,u) \models \top$	
$\mathcal{M},(t,u) \models p$	iff $(t,u) \in V(p)$ where $p \in \mathsf{P}$
$\mathcal{M},(t,u) \models \neg\varphi$	iff $\mathcal{M},(t,u) \not\models \varphi$
$\mathcal{M},(t,u) \models \varphi \vee \psi$	iff $\mathcal{M},(t,u) \models \varphi$ or $\mathcal{M},(t,u) \models \psi$
$\mathcal{M},(t,u) \models \mathbf{A}\varphi$	iff $\mathcal{M},(t',u) \models \varphi$ for all timelines t' extending (t,u)
$\mathcal{M},(t,u) \models \mathbf{E}\varphi$	iff $\mathcal{M},(t',u) \models \varphi$ for some timeline t' extending (t,u)
$\mathcal{M},(t,u) \models \bigcirc\varphi$	iff $\mathcal{M},(t,u+1) \models \varphi$
$\mathcal{M},(t,u) \models \Box\varphi$	iff for all $u' \in \mathbb{N}$ if $(u \leq u')$ then $\mathcal{M},(t,u') \models \varphi$
$\mathcal{M},(t,u) \models \Diamond\varphi$	iff there exists $u' \in \mathbb{N}$ such that $(u \leq u')$ and $\mathcal{M},(t,u') \models \varphi$
$\mathcal{M},(t,u) \models \varphi\mathcal{U}\psi$	iff there exists $u' \in \mathbb{N}$ such that $(u' \geq u)$ and $\mathcal{M},(t,u') \models \psi$ and for all $u'' \in \mathbb{N}$ if $(u \leq u'' < u')$ then $\mathcal{M},(t,u'') \models \varphi$
$\mathcal{M},(t,u) \models \varphi\mathcal{W}\psi$	iff $\mathcal{M},(t,u) \models \varphi\mathcal{U}\psi$ or $\mathcal{M},(t,u) \models \Box\varphi$
$\mathcal{M},(t,u) \models \mathbf{K}_i\varphi$	iff for all timelines t' and for all $u' \in \mathbb{N}$ if $((t,u),(t',u')) \in R_i$ then $\mathcal{M},(t',u') \models \varphi$

Table 4. Semantics of $\mathsf{CTL} \oplus \mathsf{S5}_{(m)}$

of Halpern and Vardi [18, 19]. Given a set of trees T, the set of timelines can be extracted by taking the union of the infinite branches that start at the root node of each tree in T. Let TL_T be the set of all timelines in T. A point, p, is a pair $p = (t,u)$, where $t \in TL_T$ is a timeline and $u \in \mathbb{N}$ is a temporal index into t. Given T, a set of trees, let $TLines$ be the set of timelines constructed from T. Two timelines t and t' *coincide up to point* (t,n) if, and only if, $(t,n') = (t',n')$ for all $n' \leq n$. A timeline t' *extends* (t,n) if, and only if, t and t' coincide up to (t,n). Let P_T be the set of all points.

An *interpretation* $\mathcal{M}$ for $\mathsf{CTL} \oplus \mathsf{S5}_{(m)}$ is a structure $\mathcal{M} = (T,\mathcal{R},V)$ where (i) T is a set of infinite trees, with a distinguished tree r_0, (ii) for every $i \in \mathsf{Ag}$, $\mathcal{R}$ contains an equivalence relation $R_i \subseteq P_T \times P_T$, and (iii) V maps P to subsets of P_T.

The semantics of $\mathsf{CTL} \oplus \mathsf{S5}_{(m)}$ formula is defined in Table 4. For any formula φ, if there is some interpretation $\mathcal{M}$ such that $\mathcal{M},(t_0,0) \models \varphi$, for any timeline t_0 extracted from the distinguished tree r_0, then φ is said to be *satisfiable* and $\mathcal{M}$ is a *model of* φ. If $\mathcal{M},(t_0,0) \models \varphi$ for all interpretations $\mathcal{M}$, for any timeline t_0 extracted from the distinguished tree r_0, then φ is said to be valid.

We assume that formulae of the core KARO logic are normalised using the rewrite rules of Table 1. We define a translation τ from core KARO logic into the fusion of CTL and $\mathsf{S5}_{(m)}$ as given in Table 5.

Theorem 7. *Let φ be formula of the core KARO logic. Then φ is satisfiable iff $\tau(\varphi{\downarrow})$ is.*

Transformation to separated normal form (SNF$_{karo}$)

We intend to use a resolution-based calculus to prove or disprove the satisfiability of formulae in $\mathsf{CTL} \oplus \mathsf{S5}_{(m)}$. As most resolution-based calculi, the inference rules of the calculus only deal with formulae in a particular clausal

$$
\begin{array}{rclrcl}
\tau([\mathtt{do}_i(a)]\varphi) &=& \mathbf{A}\bigcirc(done_i^a \rightarrow \tau(\varphi)) & \tau(\top) &=& \top \\
\tau(\langle\mathtt{do}_i(a)\rangle\varphi) &=& \mathbf{E}\bigcirc(done_i^a \wedge \tau(\varphi)) & \tau(p) &=& p \\
\tau(\mathbf{O}_i\alpha) &=& \tau(\langle\mathtt{do}_i(\alpha)\rangle\top) & \tau(\neg\varphi) &=& \neg\tau(\varphi) \\
\tau(\mathbf{A}_i\alpha) &=& \tau(\langle\mathtt{do}_i(\alpha)\rangle\top) & \tau(\varphi \vee \psi) &=& \tau(\varphi) \vee \tau(\psi) \\
\tau(\mathbf{K}_i\varphi) &=& \mathbf{K}_i\tau(\varphi) & & &
\end{array}
$$

where a is an atomic action, p is a propositional variable, and $done_i^a$ is a propositional variable uniquely associated with a and i.

Table 5. Translation morphism τ

normal form. So, as an intermediate step we have to rewrite formulae in the fusion of CTL and $\mathsf{S5}_{(m)}$ into such a normal form, called SNF_{karo} here. A particular property of this normal form is that it separates temporal and modal aspects (as is done in [11]). Formulae in SNF_{karo} are of the general form

$$\mathbf{A}\square^* \bigwedge_i T_i$$

where each T_i is known as a *clause* and must be one of the following forms and $\mathbf{A}\square^*$ is the universal relation (which can be defined in terms of the operators 'everyone knows' and 'common knowledge')

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

where L'_j, L_k, and L are literals and M_k^i 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 $\mathbf{E}$-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 $\mathbf{E}$ step clauses and an $\mathbf{E}$ 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 '$\mathbf{A}\square^*$' 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 transformation of formulae in $\mathsf{CTL} \oplus \mathsf{S5}_{(m)}$ into SNF_{karo} by SNF.

Theorem 8. *Let φ be a formula in $\mathsf{CTL} \oplus \mathsf{S5}_{(m)}$. Then,*

1. *φ is satisfiable iff* $\mathrm{SNF}(\varphi)$ *is satisfiable.*
2. $\mathrm{SNF}(\varphi)$ *can be computed in polynomial time.*

A resolution calculus for SNF$_{karo}$

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 underlying idea is to extend a resolution calculus for propositional logic to temporal and modal logic. The basic insight underlying propositional resolution is that a propositional variable p and its negation $\neg p$ cannot be both true. So, if two propositional clauses $C \vee p$ and $D \vee \neg p$ are both supposed to be true, then also $C \vee D$ must be true. We now simply have to extend this insight to the logic we are dealing with here. For example, it cannot be both true that an agent i knows φ and also knows $\neg\varphi$. So, $\mathbf{K}_i\varphi$ and $\mathbf{K}_i\neg\varphi$ cannot both be true, and if two clauses $C \vee \mathbf{K}_i\varphi$ and $D \vee \mathbf{K}_i\neg\varphi$ are both supposed to be true, then so must be $C \vee D$. This is the basis for rule KRES2 below. However, there are many more cases that need to be considered and instead of just one resolution rule as for propositional logic, we need 17 inference rules to cover all of them.

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.

In the following, if L is a literal, then $\sim L$ denotes A if $L = \neg A$ and it denotes $\neg L$, otherwise. A literal clause may be resolved with an initial clause (IRES1) or two initial clauses may be resolved together (IRES2) as follows where C and D are disjunctions of literals.

$$\textbf{IRES1:}\ \frac{\begin{array}{l}\textbf{true} \rightarrow (C \vee L)\\ \textbf{start} \rightarrow (D \vee \sim L)\end{array}}{\textbf{start} \rightarrow (C \vee D)} \qquad \textbf{IRES2:}\ \frac{\begin{array}{l}\textbf{start} \rightarrow (C \vee L)\\ \textbf{start} \rightarrow (D \vee \sim L)\end{array}}{\textbf{start} \rightarrow (C \vee D)}$$

During knowledge resolution we apply the following rules which are based on the modal resolution system introduced by Mints [29]. 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.

$$\textbf{KRES1:}\ \frac{\begin{array}{l}\textbf{true} \rightarrow C \vee M\\ \textbf{true} \rightarrow D \vee \sim M\end{array}}{\textbf{true} \rightarrow C \vee D} \qquad \textbf{KRES2:}\ \frac{\begin{array}{l}\textbf{true} \rightarrow C \vee \mathbf{K}_i L\\ \textbf{true} \rightarrow D \vee \mathbf{K}_i \sim L\end{array}}{\textbf{true} \rightarrow C \vee D}$$

$$\textbf{KRES3:}\ \frac{\begin{array}{l}\textbf{true} \rightarrow C \vee \mathbf{K}_i L\\ \textbf{true} \rightarrow D \vee \sim L\end{array}}{\textbf{true} \rightarrow C \vee D} \qquad \textbf{KRES4:}\ \frac{\begin{array}{l}\textbf{true} \rightarrow C \vee \neg\mathbf{K}_i L\\ \textbf{true} \rightarrow D \vee L\end{array}}{\textbf{true} \rightarrow C \vee \mathrm{mod}_i(D)}$$

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

$$\begin{aligned}
\mathrm{mod}_i(A \vee B) &= \mathrm{mod}_i(A) \vee \mathrm{mod}_i(B)\\
\mathrm{mod}_i(\mathbf{K}_i L) &= \mathbf{K}_i L\\
\mathrm{mod}_i(\neg\mathbf{K}_i L) &= \neg\mathbf{K}_i L\\
\mathrm{mod}_i(L) &= \neg\mathbf{K}_i {\sim} L
\end{aligned}$$

The last resolution rule requires explanation. Take KRES4 and distribute in the external $\mathbf{K}_i$ operator from the surrounding $\mathbf{A}\Box^*$ operator into the second premise obtaining $\mathbf{true} \rightarrow \neg\mathbf{K}_i\neg D \vee \mathbf{K}_i L$ where D is a disjunction of literals or modal literals. Since, in $\mathsf{S5}$, from axioms $\mathsf{4}$, $\mathsf{5}$ and D we have

$$\begin{aligned}
&\vdash \quad \neg\mathbf{K}_i\mathbf{K}_i\varphi \iff \neg\mathbf{K}_i\varphi\\
&\vdash \neg\mathbf{K}_i\neg\mathbf{K}_i\neg\varphi \iff \mathbf{K}_i\neg\varphi.
\end{aligned}$$

so we can delete $\neg\mathbf{K}_i\neg$ from any of the disjuncts in D that are modal literals and obtain the required resolvent.

Finally we require the following rewrite rule to allow us to obtain the most comprehensive set of literal clauses for use during step and temporal resolution

$$\textbf{KRES5:}\ \frac{\mathbf{true} \rightarrow D \vee \mathbf{K}_i L_1 \vee \mathbf{K}_i L_2 \vee \ldots}{\mathbf{true} \rightarrow D \vee L_1 \vee L_2 \vee \ldots}$$

where D is a disjunction of literals.

'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.

Pairs of step clauses may be resolved using the (step resolution) rules SRES1, SRES2, and SRES3.

$$\textbf{SRES1:}\ \begin{array}{r} P \rightarrow \mathbf{A}\bigcirc(F \vee L)\\ Q \rightarrow \mathbf{A}\bigcirc(G \vee {\sim}L)\\ \hline (P \wedge Q) \rightarrow \mathbf{A}\bigcirc(F \vee G)\end{array} \qquad \textbf{SRES2:}\ \begin{array}{r} P \rightarrow \mathbf{E}\bigcirc(F \vee L)_{\langle \mathsf{c}_i \rangle}\\ Q \rightarrow \mathbf{A}\bigcirc(G \vee {\sim}L)\\ \hline (P \wedge Q) \rightarrow \mathbf{E}\bigcirc(F \vee G)_{\langle \mathsf{c}_i \rangle}\end{array}$$

$$\textbf{SRES3:}\ \begin{array}{r} P \rightarrow \mathbf{E}\bigcirc(F \vee L)_{\langle \mathsf{c}_i \rangle}\\ Q \rightarrow \mathbf{E}\bigcirc(G \vee {\sim}L)_{\langle \mathsf{c}_i \rangle}\\ \hline (P \wedge Q) \rightarrow \mathbf{E}\bigcirc(F \vee G)_{\langle \mathsf{c}_i \rangle}\end{array}$$

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 resolution rules SRES4 and SRES5.

$$\textbf{SRES4:}\ \begin{array}{r} P \rightarrow \mathbf{A}\bigcirc(F \vee L)\\ \mathbf{true} \rightarrow (G \vee {\sim}L)\\ \hline P \rightarrow \mathbf{A}\bigcirc(F \vee G)\end{array} \qquad \textbf{SRES5:}\ \begin{array}{r} P \rightarrow \mathbf{E}\bigcirc(F \vee L)_{\langle \mathsf{c}_i \rangle}\\ \mathbf{true} \rightarrow (G \vee {\sim}L)\\ \hline P \rightarrow \mathbf{E}\bigcirc(F \vee G)_{\langle \mathsf{c}_i \rangle}\end{array}$$

Once a contradiction within a state is found, the following rule can be used to generate extra global constraints.

$$\textbf{SRES6:}\ \frac{Q \rightarrow \mathbf{P}\bigcirc\mathbf{false}}{\mathbf{true} \rightarrow {\sim}Q}$$

where $\mathbf{P}$ is either path operator.

Rule SRES6 states that if, by satisfying Q in the last moment in time a contradiction is produced, then P must never be satisfied in *any* moment in time. The new constraint therefore represents $\mathbf{A}\Box^*{\sim}Q$.

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.

TRES1: $$\begin{array}{l} P \rightarrow \mathbf{A}\bigcirc\mathbf{A}\Box L \\ Q \rightarrow \mathbf{A}\Diamond{\sim}L \\ \hline Q \rightarrow \mathbf{A}({\sim}P\mathcal{W}{\sim}L) \end{array}$$

TRES2: $$\begin{array}{l} P \rightarrow \mathbf{A}\bigcirc\mathbf{A}\Box L \\ Q \rightarrow \mathbf{E}\Diamond{\sim}L_{\langle \mathsf{c}_i \rangle} \\ \hline Q \rightarrow \mathbf{E}({\sim}P\mathcal{W}{\sim}L)_{\langle \mathsf{c}_i \rangle} \end{array}$$

TRES3: $$\begin{array}{l} P \rightarrow \mathbf{E}\bigcirc\mathbf{E}\Box L_{\langle \mathsf{c}_i \rangle} \\ Q \rightarrow \mathbf{A}\Diamond{\sim}L \\ \hline Q \rightarrow \mathbf{A}({\sim}P\mathcal{W}{\sim}L) \end{array}$$

TRES4: $$\begin{array}{l} P \rightarrow \mathbf{E}\bigcirc\mathbf{E}\Box L_{\langle \mathsf{c}_i \rangle} \\ Q \rightarrow \mathbf{E}\Diamond{\sim}L_{\langle \mathsf{c}_i \rangle} \\ \hline Q \rightarrow \mathbf{E}({\sim}P\mathcal{W}{\sim}L)_{\langle \mathsf{c}_i \rangle} \end{array}$$

In each case the resolvent ensures that once Q has been satisfied, meaning that the eventuality $\Diamond{\sim}L$ must be satisfied on some or all paths, the conditions for triggering a $\Box$-formula are not allowed to occur, that is, P must be false, until the eventuality (${\sim}L$) has been satisfied. It may be surprising that resolving a **A**-formula with a **E**-formula in TRES3 results in a **A**-formula. This is because the eventuality ${\sim}L$ must appear on *all* paths so similarly the resolvent will also hold on all paths

Given a set N of SNF_{karo} clauses to be to be tested for satisfiability, the following steps are performed.

1. Perform initial, knowledge and step resolution (including simplification and subsumption) on N until either
   a) false is derived: terminate noting that N is unsatisfiable; or
   b) no new resolvents are generated: continue to step (2).
2. Select an eventuality from the right-hand side of a sometime clause within N. Search for a set of clauses with which one of the temporal resolution rules can be applied.
3. If the resolvent is new (i.e. is not subsumed by previously detected resolvents) translate into SNF_{karo} and go to step (1). Otherwise if no new resolvents have been found for any eventuality, terminate declaring N satisfiable, else go to step (2).

Theorem 9. *Let N be a set of SNF_{karo} clauses. Then,*

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

A decision procedure for core KARO logic

We can now put the four components of our second decision procedure for core KARO logic together. Given a formula φ in the core KARO logic, we proceed by normalising φ using the rules in Table 1, translate the result into the fusion of CTL and $\mathsf{S5}_{(m)}$, transforming the resulting formula to SNF_{karo}, and finally applying the temporal resolution procedure for SNF_{karo} to the set of SNF_{karo} clauses we obtain.

Theorem 10 (Soundness, completeness, and termination). *Let φ be a formula of the core KARO logic and let $N = \mathrm{SNF}(\tau(\varphi\downarrow))$. Then,*

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

Solving the Eve example by temporal resolution

We will now show how to obtain a refutation for the specification of Eve's blocks world using clausal temporal resolution. To deal with implementability, we extend the translation morphism τ by $\tau(\Diamond_i \varphi) = \mathbf{E}\Diamond\tau(\varphi)$. We will discuss the appropriateness of both definitions in the following section.

The specification of the problem can then be written as formulae in the normal form as follows. For example (E_1) instantiated where $X = a$ and $Y = b$ can be written as the following two rules.

$$\begin{aligned}\mathbf{true} &\rightarrow \mathbf{A}\bigcirc(\neg\mathrm{done}_E^{\mathrm{put}(a,b)} \vee \mathrm{is\_on}(a,b))\\ \mathbf{true} &\rightarrow \mathbf{A}\bigcirc(\neg\mathrm{done}_E^{\mathrm{put}(a,b)} \vee \neg\mathrm{is\_clear}(a))\end{aligned}$$

The conjunction of initial conditions is rewritten by a new proposition v and each conjunct, e.g. $\mathbf{K}_E\mathrm{is\_on}(a,b)$ can be can be written as follows

(I_0) $\quad\mathbf{start} \rightarrow v$

(I_1) $\quad\mathbf{true} \rightarrow \neg v \vee \mathbf{K}_E\mathrm{is\_on}(a,b)$

and similarly with the conjuncts $\mathbf{K}_E\mathrm{is\_clear}(b)$, $\mathbf{K}_E\mathrm{is\_clear}(c)$, $\mathbf{K}_E\mathrm{is\_clear}(d)$ and $\mathbf{K}_E\mathrm{on\_floor}(a)$ (giving I_0–I_5). We try to prove

$$\mathbf{K}_E\langle\mathtt{do}_A(\mathrm{put}(b,c))\rangle\top \rightarrow \mathbf{K}_E\Diamond_E\mathrm{tower}(a,b,c,d)$$

Firstly we translate as follows.

$$\mathbf{K}_E\mathbf{E}\bigcirc(\mathrm{done}_A^{\mathrm{put}(b,c)}) \rightarrow \mathbf{K}_E\mathbf{E}\Diamond\mathrm{tower}(a,b,c,d)$$

Next we must negate and look for a contradiction with the specification above, i.e.

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

Next we rewrite into the normal form introducing new variables w, x, y, z and replacing $\mathrm{tower}(a,b,c,d)$ with its definition.

(G_1) $\quad \textbf{start} \rightarrow w$

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

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

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

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

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

(G_7) $\quad \textbf{true} \rightarrow \neg x \vee \neg \mathrm{on\_floor}(a) \vee \neg \mathrm{is\_on}(a,b) \vee \neg \mathrm{is\_on}(b,c) \vee \neg \mathrm{is\_on}(c,d) \vee \neg \mathrm{is\_clear}(d)$

Firstly, we apply the rules SRES1, SRES2 and SRES4 to (G_6), (G_7), and the instantiations of (N_1), (N_2), (N_4), (E_1), and (A_1) given below

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

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

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

(N_4) $\quad \mathrm{on\_floor}(a) \rightarrow \mathbf{A}\bigcirc(\neg \mathrm{done}_E^{\mathrm{put}(c,d)} \vee \mathrm{on\_floor}(a))$

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

(A_1) $\quad \mathrm{is\_clear}(c) \rightarrow \mathbf{E}\bigcirc \mathrm{done}_E^{\mathrm{put}(c,d)}{}_{\langle \mathsf{c}_1 \rangle}$

obtaining

$$x \wedge \mathrm{is\_clear}(d) \wedge \mathrm{is\_on}(a,b) \wedge \mathrm{is\_on}(b,c) \wedge \mathrm{on\_floor}(a) \wedge \mathrm{is\_clear}(c) \rightarrow \mathbf{E}\bigcirc \textbf{false}_{\langle \mathsf{c}_1 \rangle}.$$

An application of SRES6 to this step clause results in

(G_8) $\quad \textbf{true} \rightarrow \neg x \vee \neg \mathrm{is\_clear}(d) \vee \neg \mathrm{is\_on}(a,b) \vee \neg \mathrm{is\_on}(b,c) \vee \neg \mathrm{on\_floor}(a) \vee \neg \mathrm{is\_clear}(c)$

Next we again apply the rules SRES1, SRES2, and SRES4 to (G_6), (G_8), and instantiations of (N_1), (N_2), (N_4), (E_1), and (G_3) obtaining the following

$$\mathrm{is\_clear}(c) \wedge \mathrm{is\_clear}(d) \wedge \mathrm{is\_on}(a,b) \wedge \mathrm{on\_floor}(a) \wedge x \wedge y \rightarrow \mathbf{E}\bigcirc \textbf{false}_{\langle \mathsf{c}_2 \rangle}.$$

With an application of SRES6 to this clause we obtain

(G_9) $\quad \textbf{true} \rightarrow \neg x \vee \neg y \vee \neg \mathrm{is\_clear}(c) \vee \neg \mathrm{is\_clear}(d) \vee \neg \mathrm{is\_on}(a,b) \vee \neg \mathrm{on\_floor}(a)$

Resolving (G_9) with (G_5) using KRES1 and then with (G_4) using KRES4 we obtain

(G_{10}) $\quad \textbf{true} \rightarrow \neg z \vee \neg \mathbf{K}_E y \vee \neg \mathbf{K}_E \mathrm{is\_clear}(c) \vee \neg \mathbf{K}_E \mathrm{is\_clear}(d) \vee \neg \mathbf{K}_E \mathrm{is\_on}(a,b) \vee \neg \mathbf{K}_E \mathrm{on\_floor}(a)$

which can be resolved with the initial conditions (I_1), (I_3), (I_4), (I_5), and (G_2) using KRES1 to obtain

(G_{11}) $\quad \textbf{true} \rightarrow \neg w \vee \neg v.$

Finally, resolving (G_{11}) with (I_0) and (G_1) using IRES1 and IRES2 the contradiction

$$\textbf{start} \rightarrow \textbf{false}$$

is obtained.

5 Beyond the Core KARO Logic

In Sections 3 and 4 we have presented two methods for modal reasoning in a restricted core of the KARO logic. We will now consider whether and how each method can be extended to cover a larger fragment of the KARO logic, and then indicate how KARO can be put to work in more complex environments than the blocks world.

In the full framework opportunities $\mathbf{O}_i\alpha$ and abilities $\mathbf{A}_i\alpha$ are not the same. There $\mathbf{O}_i\alpha{=}\langle \mathsf{do}_i(\alpha)\rangle\top$, and $\mathbf{A}_i\alpha$ is defined as in Section 2. Consequently, we can extend the normalisation function defined by the rewrite rules in Table 1 to reduce any formula φ with occurrences of $\mathbf{O}_i\alpha$, $\mathbf{A}_i\alpha$, or $[\mathsf{do}_i(\alpha)]\psi$ where α is a non-atomic action formula to a formula $\varphi\!\downarrow$ which is logically equivalent to φ and in the absence of the unbounded repetition operator $\varphi\!\downarrow$ contains no non-atomic action formulae.

In the translation approach the translation function π has to be modified such that $\pi(\mathbf{A}_i a, x) = c_i^a(x)$ where a is an atomic action, and c_i^a represents the relation $c_{(i,a)}$ in our semantics. In the clausal temporal resolution approach $\mathbf{A}_i\alpha$ is simply represented by propositional variables c_i^α uniquely associated with i and α. It seems an alternative for both approaches that would incorporate also a commitment operator could exploit the ideas of [39].

We have also excluded wishes in our presentation. In the full KARO framework, $\mathbf{W}_i^s$ is a KD modality. The incorporation of wishes into the translation approach presents no difficulties. The translation function π is extended by $\pi(\mathbf{W}_i^s\varphi, x) = \forall y \,.\, R_{(i,\mathbf{W})}(x,y) \rightarrow \pi(\varphi, y)$, where $R_{(i,\mathbf{W})}$ is a binary predicate symbol uniquely associated with the modal operator $\mathbf{W}_i^s$, and $\Pi(\psi)$ contains additional conjuncts $\forall x\, \exists y\, R_{(i,\mathbf{W})}(x,y)$ for every agent i, ensuring that the binary relations $R_{(i,\mathbf{W})}$ are serial. For the clausal temporal resolution approach the addition of wishes to the core of KARO requires (i) an extension of the normal form which allows for clauses for the wishes of each agent, and (ii) additional sound and complete resolution rules for the KD modalities $\mathbf{W}_i^s$.

The implementability operator $\Diamond_i$ excluded from core KARO logic is one of the most interesting operators of KARO logic. The formula $\Diamond_i\varphi$ is true in world w, that is, an agent i can implement φ in world w, iff there exists a natural number $k \geq 0$, and atomic actions $a_1, \ldots, a_k$ such that $\mathbf{PracPoss}_i(a_1; \ldots; a_k, \varphi)$ is true in w. Formally, the semantics of $\Diamond_i$ is defined by

$$\begin{aligned} \mathcal{M}, w \models \Diamond_i\varphi \text{ iff } & \exists k \in \mathbb{N}\, \exists a_1, \ldots, a_k \in \mathsf{Ac_{at}}. \\ & \mathcal{M}, w \models \mathbf{PracPoss}_i(a_1; \ldots; a_k, \varphi) \end{aligned}$$

where $\mathbf{PracPoss}_i(\alpha, \varphi)$ is an abbreviation for $\langle \mathsf{do}_i(\alpha)\rangle\varphi \wedge \mathbf{A}_i\alpha$. So, $\Diamond_i\varphi$ holds if we can find atomic actions $a_1, \ldots, a_k$ such that agent i is able to perform the sequence $a_1; \ldots; a_k$ and performing this sequence possibly leads to a situation in which φ is true. Intuitively, proving $\Diamond_i\varphi$ requires that we find a *plan* which might bring about a situation in which φ is true. In other words, the intention for including the implementability operator into KARO logic is to internalise the *planning problem* in the logic.

However, it turns out that this intuition is slightly misleading. To give a precise analysis of the implementability operator, let us add modal operators $\forall$ and $\exists$ to our language with the following semantics.

$$\mathcal{M}, w \models \forall\varphi \text{ iff } \forall v \in W.\, \mathcal{M}, v \models \varphi$$
$$\mathcal{M}, w \models \exists\varphi \text{ iff } \exists v \in W.\, \mathcal{M}, v \models \varphi$$

The modal operator $\forall$ is the *universal modality* while $\exists$ is the *dual universal modality.*

Furthermore, if $\varphi[\psi_1]$ is a formula containing a subformula occurrence of ψ_1, then by $\varphi[\psi_1']$ we denote the formula obtained by replacing in φ the subformula occurrence of ψ_1 by the formulae ψ_1'.

Lemma 3. *1. Let $\varphi[\Diamond_i\psi]$ be a formula of KARO logic with a positive subformula occurrence of $\Diamond_i\psi$ and no negative subformula occurrences of the form $\Diamond_j\vartheta$. Then $\varphi[\Diamond_i\psi]$ is satisfiable iff $\varphi[\exists\psi]$ is satisfiable.*
2. Let $\varphi[\Diamond_i\vartheta]$ be a formula of KARO logic with a negative subformula occurrence of $\Diamond_i\vartheta$. Then the unsatisfiability of $\varphi[\Diamond_i\vartheta]$ implies the unsatisfiability of $\varphi[\exists\vartheta]$, but not vice versa.

Thus, positive occurrences of $\Diamond_i$ give little indication of the existence of a plan. The mapping of $\Diamond_i\varphi$ to $\exists y\, \pi(\varphi, y)$ by the translation morphism π as defined in Section 3 is only correct for positive occurrences of $\Diamond_i\varphi$, but not for negative occurrences. There is no straightforward way to translate negative occurrences of $\Diamond_i$ that correctly reflects its semantics.

Although the language of SNF$_{karo}$ contains with $\mathbf{A}\Box^*$ a combination of operators corresponding to the master modality, $\mathbf{A}\Box^*$ can only occur at one particular position, that is, surrounding a conjunction of clauses. For positive occurrences of $\Diamond_i$ we can show that $\mathbf{E}\Diamond\tau(\varphi)$ is a correct translation of $\Diamond_i\varphi$ by extending the model transformation sketched in the proof of Theorem 7. Again, there is no straightforward way to translate negative occurrences of $\Diamond_i$.

However, it is clear that the current semantical definition of $\Diamond_i$ fails to correspond to our intuitive understanding of implementability. A more accurate semantical definition restricts the choice of atomic actions a_1, …, a_k, which an agent i performs to bring about a situation where φ holds, to a particular finite set of actions, for example, the set of atomic actions occurring in the formula under consideration. So, if $\mathsf{Ac_{at}}\psi$ denotes the finite set of atomic actions occurring in a formula ψ, then the modified semantical definition could be as follows,

$$\mathcal{M}, w \models \Diamond_i\varphi \text{ iff } \exists k \in \mathbb{N}\, \exists a_1, \ldots, a_k \in \mathsf{Ac_{at}}\psi.$$
$$\mathcal{M}, w \models \mathbf{PracPoss}_i(a_1; \ldots; a_k, \varphi)$$

where ψ is a specific KARO formula. In this case the existential quantifier in the definition of $\Diamond_i\varphi$ can be replaced by a disjunction over all actions in $\mathsf{Ac_{at}}\psi$.

Then $\Diamond_i\varphi$ can be embedded into CTL^* as $\varphi \vee \mathbf{E}(\bigvee_{a\in\mathsf{Ac_{at}}\psi}(c_i^a \wedge \bigcirc done_i^a)))\mathcal{U}\varphi)$. Although this formula is not in CTL, it can be rewritten into a satisfiability equivalent set of SNF_{karo} clauses making use of the additional expressiveness of SNF_{karo} clauses due to the index labels we can attach to step clauses.

Also the use of the unbounded repetition operation on actions is excluded from the core KARO logic we have considered. This operation is not first-order definable and there can be no translation into first-order logic based solely on the semantics of the unbounded repetition operation. Unbounded repetition also presents problems for the clausal temporal resolution approach as we require that only atomic actions a occur in $[\mathsf{do}_i(a)]\varphi$ and $\mathbf{A}_i a$. In the presence of unbounded repetition we are not able to remove occurrences of $\alpha^\star$ or non-atomic action below unbounded repetition using the rules of Table 1 or similar rewrite rules. However, one possibility which may be fruitful is to translate formulae such as $\langle\mathsf{do}_i(a^\star)\rangle\varphi$, where a is an atomic action, directly into CTL as $\varphi \vee \mathbf{E}\bigcirc(\mathbf{E}(done_i^a\mathcal{U}(\varphi \wedge done_i^a)))$. This could be further rewritten into the normal form SNF_{karo}.

It is important to note that embeddings of the extension of core KARO logic by unbounded repetition into first-order logic and $\mathsf{CTL} \oplus \mathsf{S5}_{(m)}$ do exist. There are polynomial time computable, satisfiability equivalence preserving embeddings of $\mathsf{S5}_{(m)}$ into Converse PDL [43] and of Converse PDL into PDL [6]. The combination of these two embeddings allows us to reduce the satisfiability problem of the extension of core KARO logic by unbounded repetition to the satisfiability problem of PDL. The satisfiability problem of PDL is EXPTIME-complete [15, 32] and so are the satisfiability problem of the guarded fragment with relations of bounded arity GF_k [17] and CTL [12]. Thus, there are again polynomial time computable embeddings τ_{GF_k} and τ_{PDL} mapping formulae of PDL to satisfiability equivalent formulae in GF_k and CTL, respectively. However, these embeddings are based on the fact that any polynomial space alternating Turing machine T and its input I can be embedded into GF_k and PDL in such a way that the resulting formula $\varphi_{(T,I)}$ is satisfiable iff the original Turing machine T halts on I in an accepting state. So, given a decision procedure for PDL as a polynomial space alternating Turing machine M_{PDL}, τ_{GF_k} and τ_{PDL} can be used to translate M_{PDL} together with a PDL formula ψ into a formula $\varphi_{(M_{\mathsf{PDL}},\psi)}$ of the target logic which satisfies the property stated above. Thus, these embeddings together with the appropriate decision procedures for the target logics provide us with decision procedures for the extension of core KARO logic by unbounded repetition.

While this approach is an appropriate way to establish the complexity of a class of problems, it is doubtful whether it can be used to obtain practical proof methods. The embeddings τ_{GF_k} and τ_{PDL} induce mappings from computations of a decision procedure M_{PDL} for the source logic PDL to interpretations of the target logic. So, we can only expect to be as efficient as the decision procedure M_{PDL}. In contrast, the embeddings Π and τ described in Sections 3 and 4, respectively, constitute mappings from interpretations of the source logic to interpretations of the target logic. The embeddings do not impose any constraints on the way we solve the satisfiability problem in the target logic. This means, we can take advantage of the sophisticated techniques available for the target logics.

In the full KARO framework interaction between the dynamic logic and epistemic logic components of KARO logic is allowed and various additional properties of the modal operators have been investigated [45]. Of particular interest is *accordance*, formalised by the axiom schema $\mathbf{K}_i[\mathsf{do}_i(\alpha)]\varphi \rightarrow [\mathsf{do}_i(\alpha)]\mathbf{K}_i\varphi$. This is similar to the interaction axiom between linear time temporal logic and $\mathsf{S5}_{(m)}$, $\mathbf{K}_i\bigcirc\varphi \rightarrow \bigcirc\mathbf{K}_i\varphi$, given in [13], known as synchrony and perfect recall and is known to make the validity problem much more complex. For example in the single agent case allowing this interaction between propositional linear time temporal logic and $\mathsf{S5}$ turns the satisfiability problem from a PSPACE-complete problem into a double exponential time complete problem [18]. However, in many cases the addition of such interactions even leads to undecidability [18] so care is needed here. Resolution-based proof methods for the single agent case of linear time temporal logics of knowledge allowing synchrony and perfect recall in are given in [9]. Also, some recent positive results have been shown in [40, 41, 42].

Further it is interesting to consider what fragment of the fusion of CTL and $\mathsf{S5}_{(m)}$ we obtain when translating from KARO specifications in this way. For example is it ever possible to obtain $\mathbf{A}\diamond L$ from translating from the core of KARO? Our conjecture is it is not possible and therefore we do not require the temporal resolution rules TRES1 and TRES3.

Although the blocks world is a well accepted test-bed for planning and AI, we are also aiming at applying KARO in other areas. Breunesse [5] used a subset of KARO to reason about soccer players in the simulation league of RoboCup [35], where, as in the blocks world, the number of atomic actions is limited, but, unlike the blocks world, the result of these actions is not precise. Thus, in [5], besides knowledge, probabilities are added to the framework. His work shows that to overcome the accumulating uncertainties after a sequence of actions, there is a need to incorporate some notion of *sensing* to KARO, which, together with the notions of updating an agent's belief in a KARO setting, gives the agents a richer and dynamic epistemic attitude.

Another KARO issue still in research is the question how to *realise* agents that are specified in KARO. A first step toward this end was taken in [26], where we try to link KARO to agent programming languages. In essence, an agent programming language enables the programmer to program (the dynamics of) mental states. Thus, the semantics of such a program can be conceived of as 'mental state transformers'. KARO should be a suitable verification language for such a programming language. In [26], we analysed the language 3APL [20] of which the semantics is given in terms of a goal-base (KARO: commitments) and a belief-base (KARO: knowledge) of the agent, and were able to identify a number of 3APL-specific properties about them. In particular, we gave a number of properties that the practical reasoning rule of 3APL satisfies. Explaining this in detail would require too much additional definitions here. For further details the reader is referred to [26].

6 Conclusion

Although there exist a number of theories of rational agency which are formulated in the framework of combinations of modal logics, the work on practical proof methods for the expressive logics involved in these theories has been sparse. Examples are the tableaux-based proof methods developed by Rao and Georgeff for propositional BDI logics [34], and the resolution-based proof methods developed by Dixon, Fisher, and Wooldridge for temporal logics of knowledge [11]. In this paper we presented the current state of our attempt to provide proof methods for the logics of the KARO framework, whose expressiveness exceeds those of previous theories of rational agency.

The presentation of the proof methods in Sections 3 and 4, and the discussion in Section 5, shows that although our proof methods already cover an interesting core fragment of the KARO framework, there are still essential gaps. We believe that this is not a sign that our approach is insufficient, but due to the fact that combinations of interacting logic inherently pose difficult proof theoretical problems, which have not received the necessary attention. Recent experiments support the view that even for rather simple classes of temporal and dynamic logic formulae the performance of various theorem provers varies greatly [24]. This indicates that the theoretical and practical problems of theorem proving in temporal and dynamic logic, and their extensions, is not yet well investigated.

One of the motivations for pursuing two different approaches at the same time is the fact that the strength of the approaches lies within different areas of the KARO framework. The translation approach allows a quite elegant treatment of the informational component of KARO. On the other hand, 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.

A promising approach is the possibility of combining both proof methods. In [22] 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 extension of the multi-modal logic $\mathsf{K}_{(m)}$), and we were able to show soundness, completeness, and termination of this combination of logics.

Acknowledgements

This research was supported by a grant of the Netherlands Organization for Scientific Research (NWO) and the British Council under the UK-Dutch Joint Scientific Research Programme JRP 442, and EPSRC grant no. GR/M88761.

References

1. H. Andréka, J. van Benthem, and I. Németi. Modal languages and bounded fragments of predicate logic. *J. Philos. Logic*, 27(3):217–274, 1998.
2. L. Bachmair and H. Ganzinger. Resolution theorem proving. In A. Robinson and A. Voronkov, editors, *Handbook of Automated Reasoning*, chapter 2, pages 19–99. Elsevier, 2001.
3. L. Bachmair, H. Ganzinger, and U. Waldmann. Superposition with simplification as a decision procedure for the monadic class with equality. In *Proc. KGC'93*, volume 713 of *LNCS*, pages 83–96. Springer, 1993.
4. P. Blackburn, M. de Rijke, and V. Venema. *Modal Logic.* Cambridge University Press, 2001.
5. C. B. Breunesse. The logic of soccer. Master's thesis, ICS, University of Utrecht, The Netherlands, 2000.
6. G. De Giacomo. Eliminating "converse" from converse PDL. *J. Logic, Language and Inform.*, 5(2):193–208, 1996.
7. H. De Nivelle. Translation of S4 into GF and 2VAR. Unpublished manuscript, May 1999.
8. H. De Nivelle, R. A. Schmidt, and U. Hustadt. Resolution-based methods for modal logics. *Logic J. IGPL*, 8(3):265–292, 2000.
9. C. Dixon and M. Fisher. Clausal Resolution for Logics of Time and Knowledge with Synchrony and Perfect Recall. In *Proc. ICTL 2000*, pages 43–52, 2000.
10. C. Dixon, M. Fisher, and A. Bolotov. Clausal Resolution in a Logic of Rational Agency. *Artificial Intelligence*, 139(1):47–89, July 2002.
11. C. Dixon, M. Fisher, and M. Wooldridge. Resolution for temporal logics of knowledge. *J. Logic Computat.*, 8(3):345–372, 1998.
12. E. A. Emerson. Temporal and modal logic. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, pages 997–1072. Elsevier, 1990.
13. R. Fagin, J. Y. Halpern, Y. Moses, and M. Y. Vardi. *Reasoning About Knowledge.* MIT Press, 1996.
14. C. Fermüller, A. Leitsch, U. Hustadt, and T. Tammet. Resolution decision procedures. In Alan Robinson and Andrei Voronkov, editors, *Handbook of Automated Reasoning*, chapter 25, pages 1791–1850. Elsevier, 2001.
15. M. J. Fischer and R. Ladner. Propositional dynamic logic of regular programs. *J. Computer and System Sci.*, 18:194–211, 1979.
16. L. Georgieva, U. Hustadt, and R. A. Schmidt. Computational space efficiency and minimal model generation for guarded formulae. In *Proc. LPAR'01*, volume 2250 of *LNAI*, pages 85–99. Springer, 2001.
17. E. Grädel. On the restraining power of guards. *J. Symbolic Logic*, 64:1719–1742, 1999.
18. J. Y. Halpern and M. Y. Vardi. The complexity of reasoning about knowledge and time. I lower bounds. *J. Computer and System Sci.*, 38:195–237, 1989.
19. J. Y. Halpern and M. Y. Vardi. The complexity of reasoning about knowledge and time: Extended abstract. In *Proc. STOC'86*, pages 304–315, 1996.
20. K. V. Hindriks, F. S. de Boer, W. van der Hoek, and J.-J. Ch. Meyer. Agent programming in 3APL. *Internat. J. Autonomous Agents and Multi-Agent Systems*, 2(3):357–401, 1999.
21. U. Hustadt. *Resolution-Based Decision Procedures for Subclasses of First-Order Logic.* PhD thesis, Saarland University, Saarbrücken, Germany, 1999.

22. 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.
23. U. Hustadt and R. A. Schmidt. Using resolution for testing modal satisfiability and building models. In I. Gent, H. van Maaren, and T. Walsh, editors, *SAT2000: Highlights of Satisfiability Research in the Year 2000*, pages 459–483. IOS Press, 2000.
24. U. Hustadt and R. A. Schmidt. Formulae which highlight differences between temporal logic and dynamic logic provers. In *Issues in the Design and Experimental Evaluation of Systems for Modal and Temporal Logics*, volume DII 14/01 of *Technical Report*, pages 68–76. Department of Informatics, University of Siena, 2001.
25. C. Lutz. Complexity of terminological reasoning revisited. In *Proc. LPAR'99*, volume 1705 of *LNAI*, pages 181–200. Springer, 1999.
26. J.-J. Ch. Meyer, F. S. de Boer, R. M. van Eijk, K. V. Hindriks, and W. van der Hoek. On programming KARO agents. In *Proc. Int. Conf. on Formal and Applied Practical Reasoning (FAPR 2000)*, pages 93–103. Imperial College, London, 2000.
27. J.-J. Ch. Meyer and W. van der Hoek. *Epistemic Logic for AI and Computer Science*. Cambridge University Press, 1995.
28. J.-J. Ch. Meyer, W. van der Hoek, and B. van Linder. A logical approach to the dynamics of commitments. *Artificial Intelligence*, 113(1–2):1–40, 1999.
29. 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.
30. M. Mortimer. On languages with two variables. *Z. Math. Logik Grundlagen Math.*, 21:135–140, 1975.
31. 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.
32. V. R. Pratt. Models of program logics. In *Proc. 20th Symp. Found. Comput. Sci.*, pages 115–122. IEEE Computer Society Press, 1979.
33. A. S. Rao and M. P. Georgeff. Modeling agents within a BDI-architecture. In *Proc. KR-91*, pages 473–484. Morgan Kaufmann, 1991.
34. A. S. Rao and M. P. Georgeff. Decision procedures for BDI logics. *J. Logic Computat.*, 8(3):293–343, 1998.
35. Robocup web site, 1998–2001. `http://www.robocup.org`.
36. R. A. Schmidt. Decidability by resolution for propositional modal logics. *J. Automated Reasoning*, 22(4):379–396, 1999.
37. R. A. Schmidt and U. Hustadt. Mechanised reasoning and model generation for extended modal logics. In *Theory and Applications of Relational Structures as Knowledge Instruments. COST Action 274, TARSKI. Revised Papers*, volume 2929 of *LNCS*, pages 38–67. Springer, 2003.
38. R. A. Schmidt and U. Hustadt. A principle for incorporating axioms into the first-order translation of modal formulae. In *Proc. CADE-19*, volume 2741 of *LNAI*, pages 412–426. Springer, 2003.
39. R. A. Schmidt and D. Tishkovsky. On calculi and Kripke semantics for multiagent systems within the KARO framework. In *IJCAR 2001: Short Papers*, pages 150–159. Department of Informatics, University of Siena, 2001.

40. R. A. Schmidt and D. Tishkovsky. Combining dynamic logic with doxastic modal logics. In *Advances in Modal Logic, Volume 4*, chapter 18, pages 371–391. King's College London Publications, 2003.
41. R. A. Schmidt and D. Tishkovsky. Multi-agent dynamic logics with informational test. *Annals of Mathematics and Artificial Intelligence*, 2003. Special issue on *Computational Logic in Multi-Agent Systems*. Accepted for publication.
42. R. A. Schmidt, D. Tishkovsky, and U. Hustadt. Interaction between knowledge, action and commitment within agent dynamic logic. Preprint CSPP-27, University of Manchester, UK, 2003. Short version to appear in *Studia Logica*.
43. H. Tuominen. Dynamic logic as a uniform framework for theorem proving in intensional logic. In *Proc. CADE-10*, volume 449 of *LNAI*, pages 514–527. Springer, 1990.
44. W. van der Hoek, B. van Linder, and J.-J. Ch. Meyer. On agents that have the ability to choose. *Studia Logica*, 65:79–119, 2000.
45. 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.
46. Ch. Weidenbach et al. System description: SPASS version 1.0.0. In *Proc. CADE-16*, volume 1632 of *LNAI*, pages 378–382. Springer, 1999.
47. M. Wooldridge. *Reasoning about Rational Agents*. MIT Press, 2000.