Logical Methods in Computer Science
Vol. 2 (1:5) 2006, pp. 1–26
www.lmcs-online.org

Submitted Sep. 28, 2005
Published Mar. 8, 2006

ALGORITHMIC CORRESPONDENCE AND COMPLETENESS IN MODAL LOGIC. I. THE CORE ALGORITHM SQEMA

WILLEM CONRADIE[a], VALENTIN GORANKO[b], AND DIMITER VAKARELOV[c]

[a] Department of Mathematics, University of Johannesburg, and
School of Mathematics, University of the Witwatersrand, Johannesburg
e-mail address: wec@rau.ac.za

[b] School of Mathematics, University of the Witwatersrand, Johannesburg
e-mail address: goranko@maths.wits.ac.za

[c] Faculty of Mathematics and Computer Science, Sofia University
e-mail address: dvak@fmi.uni-sofia.bg

Abstract. Modal formulae express monadic second-order properties on Kripke frames, but in many important cases these have first-order equivalents. Computing such equivalents is important for both logical and computational reasons. On the other hand, canonicity of modal formulae is important, too, because it implies frame-completeness of logics axiomatized with canonical formulae.

Computing a first-order equivalent of a modal formula amounts to elimination of second-order quantifiers. Two algorithms have been developed for second-order quantifier elimination: SCAN, based on constraint resolution, and DLS, based on a logical equivalence established by Ackermann.

In this paper we introduce a new algorithm, SQEMA, for computing first-order equivalents (using a modal version of Ackermann's lemma) and, moreover, for proving canonicity of modal formulae. Unlike SCAN and DLS, it works directly on modal formulae, thus avoiding Skolemization and the subsequent problem of unskolemization. We present the core algorithm and illustrate it with some examples. We then prove its correctness and the canonicity of all formulae on which the algorithm succeeds. We show that it succeeds not only on all Sahlqvist formulae, but also on the larger class of inductive formulae, introduced in our earlier papers. Thus, we develop a purely algorithmic approach to proving canonical completeness in modal logic and, in particular, establish one of the most general completeness results in modal logic so far.

1991 *Mathematics Subject Classification:* F.4.1,I.2.4.
Key words and phrases: modal logic; second-order quantifier elimination; first-order correspondence, canonicity, completeness.

LOGICAL METHODS IN COMPUTER SCIENCE
DOI:10.2168/LMCS-2 (1:5) 2006

Introduction

The correspondence between modal logic and first-order logic has been studied extensively, see for example [29]. Every modal formula defines a (local or global) second-order condition on frames, but it is well known that many of these can in fact be equivalently reduced to first-order conditions. Modal formulae for which this is possible are called *elementary*. As Chagrova has shown in [3], the class of elementary modal formulae is undecidable. Hence, any attempt at an effective characterization of this class can be an approximation at best.

The best-known syntactic approximation is the class of *Sahlqvist formulae* [21], the first-order correspondents of which can be computed using the Sahlqvist-van Benthem method of substitutions, see e.g. [29] and [2].

The Sahlqvist formulae are properly included in the (still syntactically defined) *inductive formulae*, introduced in [15] and [17]; for a survey on these, see also [4]. The *complex formulae* ([26]) represent yet another extension of the Sahlqvist-class. The formulae in these three classes have the important property of being *canonical*, and hence of axiomatizing complete logics.

Computing first-order equivalents of modal formulae amounts to *elimination of the second-order quantifiers* in their semantic translation. Thus, algorithmic classes of elementary modal formulae can be generated by algorithms for elimination of second-order predicate quantifiers, such as SCAN (see [12, 10, 20]) and DLS (first introduced in [9] as an extension of an algorithm presented in [24], see also [9, 20, 25] and [19]). Each of them, applied to the standard translation into second-order logic of a negated modal formula, attempts to eliminate the existentially quantified predicate variables in it and thus to compute a first-order correspondent. Both SCAN and DLS have been applied to compute the first-order equivalents of standard translations of modal formulae, and can be regarded as the first purely algorithmic approaches to the first-order correspondence theory of modal formulae. [12] and [24] were soon followed by Simmons' algorithm presented in [23] as an extension of Sahlqvist-van Benthem method of substitutions (for the Simmons algorithm see also [8]). Simmons' algorithm is also applicable to some non-elementary modal formulae (e.g. modal reduction schemes such as McKinsey's formula), but the output formulae produced by it generally involve Skolem functions.

Briefly, SCAN accomplishes its task on an existential second-order input formula by removing the existential second-order quantifiers, skolemizing away the existential first-order quantifiers, and transforming the result into clausal form. It then attempts to obtain an equivalent formula in which the predicate variables do not occur anymore, by running a constraint resolution procedure to generate sufficiently many first-order consequences and by applying deletion rules. The introduced Skolem functions are then eliminated from the result, if possible. SCAN fails to find a first-order equivalent either if the constraint resolution phase does not terminate, or if it cannot unskolemize. It has been proved in [16] that SCAN can successfully compute the first-order equivalents of all Sahlqvist formulae.

The algorithm DLS is based on the following lemma by Ackermann, see [1].

Lemma 0.1 (Ackermann's Lemma)**.** *Let P be a predicate variable and $A(\overline{x},\overline{z})$ and $B(P)$ be first-order formulae such that there are no occurrences of P in $A(\overline{x},\overline{z})$. If P occurs only negatively in $B(P)$ then*

$$\exists P\,(\forall\overline{x}[A(\overline{x},\overline{z}) \rightarrow P(\overline{x})] \wedge B(P)) \equiv B(A(\overline{t},\overline{z})/P(\overline{t}))$$

and, respectively, if P occurs only positively in $B(P)$, then

$$\exists P\,(\forall \overline{x}[P(\overline{x}) \to A(\overline{x},\overline{z})] \wedge B(P)) \equiv B(A(\overline{t},\overline{z})/P(\overline{t}))$$

where $\overline{z}$ are parameters, and each occurrence of $P(\overline{t})$ in B on the right hand side of the equivalences, for terms $\overline{t}$, is substituted by $A(\overline{t},\overline{z})$.

The algorithm DLS performs various syntactic manipulations, including skolemization, on its input in order to transform it into a form suitable for the application of Ackermann's lemma. If that cannot be accomplished, the algorithm fails. When all predicate variables have been eliminated it may be necessary to unskolemize the resulting formula in order to obtain a first-order equivalent. If it cannot do that, the algorithm fails.

Both SCAN and DLS have been implemented and are available online. For in-depth discussion of the theory and implementation details of these algorithms see [10] (for SCAN) and [18] (for DLS).

In this paper we present a new algorithm called SQEMA (**S**econd-Order **Q**uantifier **E**limination for **M**odal formulae using **A**ckermann's lemma) for computing the first-order frame correspondents of modal formulae. It is influenced by both SCAN and DLS, but unlike them it works directly on modal formulae, rather than on their standard translation, and computes their (local) first-order equivalents by reducing them to (locally) equivalent pure formulae in a hybrid language (see e.g. [2] or [14]) with nominals and inverse modalities. This is done using a modal version of Ackermann's lemma. This approach originated from [27] (see also [28]), where Ackermann's lemma was rephrased in the framework of solving equations in Boolean and modal algebras. The fact that modal formulae are not translated has many advantages, e.g.:

- it avoids the introduction of Skolem functions, and hence the need for unskolemization, which is in general undecidable. Nominals play the role of Skolem constants, but disappear when translating the resulting pure hybrid formulae into first-order logic.
- it enables the use of a stronger version of Ackermann's lemma, where instead of polarity (positive or negative), modal formulae are tested for monotonicity (upwards or downwards) in propositional variables. The extension of SQEMA based on the stronger version of Ackermann's lemma is not presented in this paper, but only briefly discussed in the concluding remarks in Section 6 and developed in more detail in the sequel to the present paper [6].
- it shows that the (suitably extended) modal language is rich enough to enable the computation of the first-order equivalents of many modal formulae.
- it allows one to terminate the execution of the algorithm at any stage and return a (hybrid) modal formula with partly eliminated propositional variables, in an extension of the basic modal language with inverse modalities and nominals.
- it allows for a uniform proof of the correctness and canonicity of the formulae on which the algorithm succeeds.

The rest of the paper is organized as follows: In Section 1 we introduce the languages we will use, together with some basic notions, and present a modal version of Ackermann's lemma. In Section 2 we introduce the algorithm SQEMA, followed in Section 3 by some examples of its execution on various input formulae. The correctness of the algorithm and the canonicity of the formulae on which it succeeds are proved in a uniform way in section 4, and its completeness for the classes of Sahlqvist and inductive formulae is shown in section 5. We conclude in section 6 by discussing some extensions of SQEMA.

1. Preliminaries

1.1. **Syntax, semantics and standard translations.** The *basic modal language* ML is built from $\bot$ and a countably infinite set of propositional variables (or atomic propositions) $\mathsf{PROP} = \{p_0, p_1, \dots\}$ as usual:

$$\varphi ::= \bot \mid \top \mid p \mid \neg\varphi \mid \varphi \vee \psi \mid \varphi \wedge \psi \mid \Diamond\varphi \mid \Box\varphi.$$

An occurrence θ of a modal operator or a subformula in a formula φ has *positive polarity* (or, is *positive*) if it is in the scope of an even number of negations; respectively, it has *negative polarity* (or, is *negative*) if it is in the scope of an odd number of negations.

The connectives $\rightarrow$, and $\leftrightarrow$ are defined as usual. Note that we take more logical connectives as primitives than necessary, in order to avoid syntactic complications when introducing rules for each of them, but whenever suitable, we will consider some of them definable in terms of the others, as usual.

A *Kripke frame* is a pair $\mathfrak{F} = (W, R)$ where W is a non-empty set of *possible worlds* and $R \subseteq W^2$ is an *accessibility relation* between possible worlds. A *Kripke model based on a frame* $\mathfrak{F}$ is a pair $\mathcal{M} = (\mathfrak{F}, V)$, where $V : \mathsf{PROP} \rightarrow 2^W$ is a *valuation* which assigns to every atomic proposition the set of possible worlds where it is true.

The *truth of a formula* φ *at a possible world* u *of a Kripke model* $\mathcal{M} = (W, R, V)$, denoted by $\mathcal{M}, u \Vdash \varphi$, is defined recursively as follows:

$\mathcal{M}, u \Vdash \top$;
$\mathcal{M}, u \Vdash p$ iff $u \in V(p)$;
$\mathcal{M}, u \Vdash \neg\varphi$ iff $\mathcal{M}, u \nVdash \varphi$;
$\mathcal{M}, u \Vdash \varphi \vee \psi$ iff $\mathcal{M}, u \Vdash \varphi$ or $\mathcal{M}, u \Vdash \psi$;
$\mathcal{M}, u \Vdash \varphi \wedge \psi$ iff $\mathcal{M}, u \Vdash \varphi$ and $\mathcal{M}, u \Vdash \psi$;
$\mathcal{M}, u \Vdash \Diamond\varphi$ if $\mathcal{M}, w \Vdash \varphi$ for some $w \in W$ such that Ruw;
$\mathcal{M}, u \Vdash \Box\varphi$ if $\mathcal{M}, w \Vdash \varphi$ for every $w \in W$ such that Ruw.

A modal formula φ is:

- *valid in a model* $\mathcal{M}$, denoted $\mathcal{M} \Vdash \varphi$, if it is true at every world of $\mathcal{M}$;
- *valid at a world* u *in a frame* $\mathfrak{F}$, denoted $\mathfrak{F}, u \Vdash \varphi$, if it is true at u in every model on $\mathfrak{F}$;
- *valid on a frame* $\mathfrak{F}$, denoted $\mathfrak{F} \Vdash \varphi$, if it is valid in every model based on $\mathfrak{F}$;
- *valid* if it is valid on every frame;
- *globally satisfiable on a frame* $\mathfrak{F}$, if there exists a valuation V such that $(\mathfrak{F}, V) \Vdash \varphi$.

A *general frame* is a structure $\mathfrak{F} = (W, R, \mathbb{W})$ where (W, R) is a frame, and $\mathbb{W}$ is a Boolean algebra of subsets of 2^W, also closed under the modal operators. The elements of $\mathbb{W}$ are called *admissible sets* in $\mathfrak{F}$.

Given a general frame $\mathfrak{F} = (W, R, \mathbb{W})$, a *model over* $\mathfrak{F}$ is a model over (W, R) with the valuation of the variables ranging over $\mathbb{W}$.

Given a modal formula ψ, a general frame $\mathfrak{F}$, and $w \in W$, we say that ψ is *(locally) valid at* w *in* $\mathfrak{F}$, denoted $\mathfrak{F}, w \Vdash \psi$, if ψ is true at w in every model over $\mathfrak{F}$.

Following [29] we define L_0 to be the first-order language with $=$, a binary predicate R, and individual variables $\mathsf{VAR} = \{x_0, x_1, \dots\}$. Also, let L_1 be the extension of L_0 with a set of unary predicates $\{P_0, P_1, \dots\}$, corresponding to the atomic propositions $\{p_0, p_1, \dots\}$.

The formulae of ML are translated into L_1 by means of the following *standard translation* function, $\mathrm{ST}(\cdot,\cdot)$, which takes as arguments an ML-formula together with a variable from VAR:

$\mathrm{ST}(p_i, x) := P_i(x)$ for every $p_i \in \mathsf{PROP}$; $\mathrm{ST}(\neg\varphi, x) := \neg\mathrm{ST}(\varphi, x)$;
$\mathrm{ST}(\varphi \vee \psi, x) := \mathrm{ST}(\varphi, x) \vee \mathrm{ST}(\psi, x)$; $\mathrm{ST}(\varphi \wedge \psi, x) := \mathrm{ST}(\varphi, x) \wedge \mathrm{ST}(\psi, x)$;
$ST(\Diamond\varphi, x) := \exists y(Rxy \wedge \mathrm{ST}(\varphi, y))$, $ST(\Box\varphi, x) := \forall y(Rxy \rightarrow \mathrm{ST}(\varphi, y))$,
where y is the first variable in VAR not appearing in $ST(\varphi, x)$.

Now, for every Kripke model $\mathcal{M}$, $w \in \mathcal{M}$ and $\varphi \in \mathrm{ML}$:

$$\mathcal{M}, w \Vdash \varphi \ \text{ iff } \ \mathcal{M} \models \mathrm{ST}(\varphi, x)(x := w),$$

and

$$\mathcal{M} \Vdash \varphi \ \text{ iff } \ \mathcal{M} \models \forall x \mathrm{ST}(\varphi, x).$$

Thus, on Kripke models the modal language is a fragment of L_1.

Further, for every frame $\mathfrak{F}$, $w \in \mathfrak{F}$ and $\varphi \in \mathrm{ML}$

$$\mathfrak{F}, w \Vdash \varphi \ \text{ iff } \ \mathfrak{F} \models \forall \overline{P} \mathrm{ST}(\varphi, x)(x := w),$$

and

$$\mathfrak{F} \Vdash \varphi \ \text{ iff } \ \mathfrak{F} \models \forall \overline{P} \forall x \mathrm{ST}(\varphi, x),$$

where $\overline{P}$ is the tuple of all unary predicate symbols occurring in $\mathrm{ST}(\varphi, x)$. Thus modal formulae express universal monadic second-order conditions on frames.

A modal formula $\varphi \in \mathrm{ML}$ is:

- *locally first-order definable*, if there is a first-order formula $\alpha(x)$ such that for every frame $\mathfrak{F}$ and $w \in \mathfrak{F}$ it is the case that $\mathfrak{F}, w \Vdash \varphi$ iff $\mathfrak{F} \models \alpha(x := w)$;
- *(globally) first-order definable*, if there is a first-order sentence α such that for every frame $\mathfrak{F}$ it is the case that $\mathfrak{F} \Vdash \varphi$ iff $\mathfrak{F} \models \alpha$.

Two modal formulae are:

- *semantically equivalent* if they are true at exactly the same states in the same models;
- *locally frame equivalent* if they are valid at exactly the same states in the same frames;
- *frame equivalent* if they are valid on the same frames.

For the execution of the algorithm we enrich the language ML by adding:

- the *inverse modality* $\Box^{-1}$ with semantics $\mathcal{M}, u \Vdash \Box^{-1}\varphi$ iff $\mathcal{M}, w \Vdash \varphi$ for every $w \in \mathcal{M}$ such that $R^{-1}uw$, i.e. Rwu. We extend the standard translation for the inverse modality in the obvious way: $\mathrm{ST}(\Box^{-1}\varphi, x) := \forall y(Ryx \rightarrow \mathrm{ST}(\varphi, y))$, and define $\Diamond^{-1}$ as the dual of $\Box^{-1}$.
- *nominals* (see [13],[2]), a special sort of propositional variables $\mathsf{Nom} = \{\mathbf{i}_1, \mathbf{i}_2, \dots\}$ with admissible valuations restricted to *singletons*. The truth definition of nominals is: $(W, V), u \Vdash \mathbf{i}$ iff $V(\mathbf{i}) = \{u\}$. The standard translation: $\mathrm{ST}(\mathbf{i}_i, x) := x = y_i$, where $y_0, y_1, \dots$ are reserved variables associated with the nominals $\mathbf{i}_0, \mathbf{i}_1, \dots$.

The so extended modal language will be denoted by ML^+.

Let $\mathcal{M}$ be a model and φ a formula from ML^+. We denote $[\![\varphi]\!]_{\mathcal{M}} = \{m \in \mathcal{M} : \mathcal{M}, m \Vdash \varphi\}$, i.e., $[\![\varphi]\!]_{\mathcal{M}}$ is the *extension* of the formula φ in the model $\mathcal{M}$.

A *pure* formula in ML^+ is a formula which contains no propositional variables, but which may contain nominals. Every pure formula γ, is locally first-order definable by the formula $\forall \overline{y} \mathrm{ST}(\gamma, x)$, where $\overline{y}$ is the tuple of all variables y_i corresponding to nominals $\mathbf{i}_i$ occurring in γ.

Let $A, B(p)$ be formulae in ML^+. Then $B(A/p)$, hereafter also written as $B(A)$, is the formula obtained from $B(p)$ by uniform substitution of A for all occurrences of p.

Although formally ML^+ is sufficient for the execution of our algorithm, it is sometimes useful to further enrich the language with the *universal modality* $[\mathsf{U}]$, with semantics $\mathcal{M}, u \Vdash [\mathsf{U}]\varphi$ iff $\mathcal{M}, w \Vdash \varphi$ for every $w \in \mathcal{M}$, i.e. iff $\mathcal{M} \Vdash \varphi$, and define $\langle \mathsf{U} \rangle$ as the dual of $[\mathsf{U}]$: $\mathcal{M}, u \Vdash \langle \mathsf{U} \rangle \varphi$ iff $\mathcal{M}, w \Vdash \varphi$ for some $w \in \mathcal{M}$ iff $\mathcal{M}, u \Vdash \neg[\mathsf{U}]\neg\varphi$. The standard translation is extended accordingly: $\mathrm{ST}([\mathsf{U}]\varphi, x) := \forall x \mathrm{ST}(\varphi, x)$ and $\mathrm{ST}(\langle \mathsf{U} \rangle \varphi, x) := \exists x \mathrm{ST}(\varphi, x)$. In particular, we use the universal modality in the next sub-section to re-phrase the modal version of Ackermann's Lemma and compare it with the method of substitutions.

1.2. **Modal version of Ackermann's Lemma.** From now on we will work in ML^+, unless otherwise specified. Let $\mathsf{AT} = \mathsf{PROP} \cup \mathsf{NOM}$.

A formula φ is *positive (negative) in a propositional variable* p if all occurrences of p in φ are in the scope of an even (odd) number of negations. φ is *positive (negative)* if it is positive (negative) in all propositional variables. Note that nominals are not taken in consideration here, and a pure formula is both positive and negative.

Informally, a formula φ is *upwards monotone* in a propositional variable p if its truth is preserved under extensions of the interpretation of p. Similarly, φ is said to be *downwards monotone* in p if its truth is preserved under shrinkings of the interpretation of p. Here are the formal definitions.

Definition 1.1. (Monotonicity of a formula) A formula φ is *upwards monotone* in a propositional variable p if for every frame $\mathfrak{F}$, all states $w \in \mathfrak{F}$, and all valuations V and V' on $\mathfrak{F}$ such that $V(p) \subseteq V'(p)$ and $V(q) = V'(q)$ for all $q \in \mathsf{AT}$, $q \neq p$, if $(\mathfrak{F}, V), w \Vdash \varphi$ then $(\mathfrak{F}, V'), w \Vdash \varphi$, or, equivalently $[\![\varphi]\!]_{(\mathfrak{F},V)} \subseteq [\![\varphi]\!]_{(\mathfrak{F},V')}$, where $[\![\varphi]\!]_{(\mathfrak{F},V)}$ denotes the extension of φ in the Kripke model $(\mathfrak{F}, V)$.

Similarly, a formula φ is *downwards monotone* in a propositional variable p if for every frame $\mathfrak{F}$, all states $w \in \mathfrak{F}$, and all valuations V and V' on $\mathfrak{F}$ such that $V'(p) \subseteq V(p)$ and $V(q) = V'(q)$ for all $q \in \mathsf{AT}$, $q \neq p$, if $\mathfrak{F}, V \Vdash \varphi$ then $\mathfrak{F}, V' \Vdash \varphi$, or, equivalently $[\![\varphi]\!]_{(\mathfrak{F},V)} \subseteq [\![\varphi]\!]_{(\mathfrak{F},V')}$.

Note that:

- the negation of a positive (resp. upwards monotone) formula is a negative (resp. downwards monotone) formula.
- every formula positive in p is upwards monotone in p, and respectively, every formula negative in p is downwards monotone in p.
- the result of substitution of upwards (resp. downwards) monotone formulae for the variables in any upwards monotone (in particular, positive) formula is upwards (resp. downwards) monotone. In particular, $\varphi(p)$ is upwards (resp. downwards) monotone in p iff $\varphi(\neg p)$ is downwards (resp. upwards) monotone in p.

Lemma 1.2 (Modal Ackermann Lemma)**.** *Let $A, B(p)$ be formulae in* ML^+ *such that the propositional variable p does not occur in A and $B(p)$ is negative in p. Then for any model $\mathcal{M}$, $\mathcal{M} \Vdash B(A)$ iff $\mathcal{M}' \Vdash (A \to p) \land B(p)$ for some model $\mathcal{M}'$ which may only differ from $\mathcal{M}$ on the valuation of p.*

Proof. If $\mathcal{M} \Vdash B(A)$, then $\mathcal{M}' \Vdash (A \to p) \wedge B(p)$ for a model $\mathcal{M}'$ such that $[\![p]\!]_{\mathcal{M}'} = [\![A]\!]_{\mathcal{M}}$. Conversely, if $\mathcal{M}' \Vdash (A \to p) \wedge B(p)$ for some model $\mathcal{M}'$ then $\mathcal{M}' \Vdash B(A/p)$ since $B(p)$ is downwards monotone. Therefore, $\mathcal{M} \Vdash B(A/p)$. □

An analogue of this lemma can be formulated for positive formulae B, too. We note that a somewhat different version of Ackermann's lemma has been proved for modal first-order formulae in [25], where it is also applied to some modal formulae.

1.3. **Ackermann's Lemma and the substitution method.** Note that the contrapositive form of the modal Ackermann's lemma (after replacing $\neg B$ with B) is equivalent to:

$$\forall p([\mathsf{U}](A \to p) \to B(p)) \;\equiv\; B(A/p),$$

for any modal formula A not containing p, and a modal formula B which is positive with respect to p. To see that equivalence, note that $[\mathsf{U}](A \to p)$ is true in a model $\mathcal{M}$ iff $[\![A]\!]_{\mathcal{M}} \subseteq [\![p]\!]_{\mathcal{M}}$.

The statement above can be interpreted as follows: $[\mathsf{U}](A \to p) \to B(p)$ is valid in a given frame iff $B(p)$ is true for the 'minimal' valuation satisfying the antecedent, viz. A. *This is precisely the technical idea at the heart of the substitution method!*

2. The algorithm SQEMA

In this section we formally introduce the core algorithm SQEMA. To reduce technical overhead in the exposition, the algorithm will be presented in the basic modal language. In [5] SQEMA is extended to arbitrary polyadic and hybrid multi-modal languages.

2.1. **The core algorithm.** Given a modal formula φ as input, we transform $\neg\varphi$ into negation normal form. We then apply the algorithm's transformation rules (all of which preserve local frame-equivalence), to formulae of the form $\alpha \to \beta$, with α and β both in negation normal form, which we call 'equations'[1]. All equations are interpreted as global statements (i.e., valid in the whole model). The algorithm works with systems of equations. Each transformation rule transforms an equation into one or more new equations. The Ackermann-rule, based on Ackermann's lemma, is applicable to a whole system of equations. The idea is to choose a propositional variable and transform the current system of equations, using the transformation rules presented further, into a system to which the Ackermann-rule can be applied with respect to the chosen variable, thus eliminating it. This process is then repeated until all propositional variables are eliminated. As will be seen (see example 3.3), the success of the algorithm may in general depend upon the order in which this elimination of propositional variables is attempted. The algorithm accordingly makes provision for trying different orders of elimination.

Now for a more formal description. The algorithm takes as input a modal formula φ and proceeds as follows:

Step 1: Negate φ and rewrite $\neg\varphi$ in negation normal form by eliminating the connectives '$\to$' and '$\leftrightarrow$', and by driving all negation signs inwards until they appear only directly in front of propositional variables.

Then distribute diamonds and conjunctions over disjunctions as much as possible, using the equivalences $\Diamond(\varphi \vee \psi) \equiv (\Diamond\varphi \vee \Diamond\psi)$ and $(\varphi \vee \psi) \wedge \theta \equiv (\varphi \wedge \theta) \vee (\psi \wedge \theta)$,

[1] Because the algorithm resembles a procedure of solving systems of linear algebraic equations by Gaussian elimination.

in order to obtain $\bigvee \alpha_k$, where no further distribution of diamonds and conjunctions over disjunctions is possible in any α_k.

The algorithm now proceeds on each of the disjuncts, α_k separately, as follows:

Step 2: Rewrite α_k as $\mathbf{i} \to \alpha_k$, where $\mathbf{i}$ is a fixed, reserved nominal, not occurring in any α_k, and only used to denote the initial current state. This is the only equation in the initial system.

Step 3: Eliminate every propositional variable in which the system is positive or negative, by replacing it with $\top$ or $\bot$, respectively.

Step 4: If there are propositional variables remaining in equations of the system, choose one to eliminate, say p, the elimination of which has not been attempted yet. Proceed to step 5. (So far this choice is made non-deterministically; some heuristics will be suggested in a sequel paper.) If all remaining variables have been attempted and Step 5 has failed, backtrack and attempt another order of elimination.

If all orders of elimination and all remaining variables have been attempted and step 5 has failed, report failure.

If all propositional variables have been eliminated from the system, proceed to Step 6.

Step 5: The goal now is, by applying the transformation rules listed below, to rewrite the system of equations so that the Ackermann-rule becomes applicable with respect to the chosen variable p in order to eliminate it. Thus, the current goal is to transform the system into one in which every equation is either negative in p, or of the form $\alpha \to p$, with p not occurring in α, i.e. to 'extract' p and 'solve' for it.

If this fails, backtrack, change the polarity of p by substituting $\neg p$ for it everywhere, and attempt again to prepare for the Ackermann-rule.

If this fails again, or after the completion of this step, return to Step 4.

Step 6: If this step is reached by all the branches of the execution it means that all propositional variables have been successfully eliminated from all systems resulting from the input formula. What remains now is to return the desired first-order equivalent. In each system, take the conjunction of all equations to obtain a formula pure, and form the formula $\forall \overline{y} \exists x_0 \mathrm{ST}(\neg \mathsf{pure}, x_0)$, where $\overline{y}$ is the tuple of all occurring variables corresponding to nominals, but with y_i (corresponding to the designated current state nominal $\mathbf{i}$) left free if the local correspondent is to be computed. Then take the conjunction of these translations over the systems on all disjunctive branches. For motivation of the correctness of this translation the reader is referred to the examples in the following section as well as the correctness proof in section 4.

Return the result, which is the (local) first-order condition corresponding to the input formula.

Note that, if a right order of the elimination of the variables is guessed (i.e. chosen non-deterministically) rather than all possible orders explored sequentially, this algorithm runs in nondeterministic polynomial time. However, we believe that additional rules which determine the right order of elimination (if any) can reduce the complexity to PTIME.

2.2. **Transformation Rules.** The transformation rules used by SQEMA are listed below. Note that these are *rewriting rules*, i.e. they replace the equations to which they apply.[2]

I. Rules for the logical connectives:

[2]Rajeev Gore has remarked that these transformation rules are reminiscent of the rules in display logics.

$\wedge$-rule:
$$\frac{\beta \to \gamma \wedge \delta}{\beta \to \gamma, \beta \to \delta}$$

Left-shift $\vee$-rule:
$$\frac{\beta \to \gamma \vee \delta}{(\beta \wedge \neg\gamma) \to \delta}$$

Right-shift $\vee$-rule:
$$\frac{(\beta \wedge \neg\gamma) \to \delta}{\beta \to \gamma \vee \delta}$$

Left-shift $\Box$-rule:
$$\frac{\gamma \to \Box\delta}{\Diamond^{-1}\gamma \to \delta}$$

Right-shift $\Box$-rule:
$$\frac{\Diamond^{-1}\gamma \to \delta}{\gamma \to \Box\delta}$$

$\Diamond$-rule:
$$\frac{\mathbf{j} \to \Diamond\gamma}{\mathbf{j} \to \Diamond\mathbf{k},\ \mathbf{k} \to \gamma}$$
where $\mathbf{j}$ is any nominal and $\mathbf{k}$ is a new nominal.

Sometimes we will write $R\mathbf{jk}$ as an abbreviation for $\mathbf{j} \to \Diamond\mathbf{k}$.

II. Ackermann-Rule: This rule is based on the equivalence given in Ackermann's Lemma. It works, not on a single equation, but by transforming a whole set of equations as follows:

$$\left\|\begin{array}{l} \alpha_1 \to p, \\ \ldots \\ \alpha_n \to p, \\ \\ \beta_1(p), \\ \ldots \\ \beta_m(p), \end{array}\right. \quad \Rightarrow \quad \left\|\begin{array}{l} \beta_1[(\alpha_1 \vee \ldots \vee \alpha_n)/p], \\ \ldots \\ \beta_m[(\alpha_1 \vee \ldots \vee \alpha_n)/p]. \end{array}\right.$$

where:

1. p does not occur in $\alpha_1, \ldots, \alpha_n$;
2. each of $\beta_1, \ldots, \beta_m$ is negative in p;
3. no other equations in the system contain p.

Hereafter, we will refer to the formulae $\alpha_1, \ldots, \alpha_n$ in an application of the Ackermann-rule as *α-formulae*, and to $\beta_1, \ldots, \beta_m$ as *β-formulae*.

III. Polarity switching rule: Switch the polarity of every occurrence of a chosen variable p within the current system, i.e. replace $\neg p$ by p and p by $\neg p$ for every occurrence of p not prefixed by $\neg$.

IV. Auxiliary Rules: These rules are intended to provide the algorithm with some propositional reasoning capabilities and to effect the duality between the modal operators.

1. Commutativity and associativity of $\wedge$ and $\vee$ (tacitly used).
2. Replace $\gamma \vee \neg\gamma$ with $\top$, and $\gamma \wedge \neg\gamma$ with $\bot$.
3. Replace $\gamma \vee \top$ with $\top$, and $\gamma \vee \bot$ with γ.
4. Replace $\gamma \wedge \top$ with γ, and $\gamma \wedge \bot$ with $\bot$.
5. Replace $\gamma \to \bot$ with $\neg\gamma$ and $\gamma \to \top$ with $\top$.

6. Replace $\bot \to \gamma$ with $\top$ and $\top \to \gamma$ with γ.
7. Replace $\neg\Diamond\neg$ with $\Box$ and $\neg\Box\neg$ with $\Diamond$.

Note that, apart from the polarity switching rule, no transformation rule changes the polarity of any occurrence of a propositional variable.

3. Some examples of the execution of SQEMA

In this section we illustrate the execution of SQEMA on several formulae and discuss some of its features.

Example 3.1. Consider the formula $\Diamond\Box p \to \Box\Diamond p$.

Step 1: Negating, we obtain $\Diamond\Box p \wedge \Diamond\Box\neg p$.

Step 2: The initial system of equations:

$$\|\mathbf{i} \to (\Diamond\Box p \wedge \Diamond\Box\neg p)\ .$$

Step 3: The formula is neither positive nor negative in p, the only occurring propositional variable.

Step 4: We choose p to eliminate — our only option.

Step 5: We will try to transform the system, using the rules, so that the Ackermann-rule becomes applicable.

Applying the $\wedge$-rule gives

$$\left\| \begin{array}{l} \mathbf{i} \to \Diamond\Box p \\ \mathbf{i} \to \Diamond\Box\neg p \end{array} \right. ,$$

Applying first the $\Diamond$-rule and then the $\Box$-rule to the first equation yields:

$$\left\| \begin{array}{l} R\mathbf{ij} \\ \mathbf{j} \to \Box p \\ \mathbf{i} \to \Diamond\Box\neg p \end{array} \right. ,$$

and then

$$\left\| \begin{array}{l} R\mathbf{ij} \\ \Diamond^{-1}\mathbf{j} \to p \\ \mathbf{i} \to \Diamond\Box\neg p \end{array} \right. .$$

The Ackermann-rule is now applicable, yielding the system

$$\left\| \begin{array}{l} R\mathbf{ij} \\ \mathbf{i} \to \Diamond\Box\neg(\Diamond^{-1}\mathbf{j}) \end{array} \right.$$

All propositional variables have been successfully eliminated, so we proceed to step 6.

Step 6: Taking the conjunction of the equations gives

$$R\mathbf{ij} \wedge (\mathbf{i} \to \Diamond\Box\neg(\Diamond^{-1}\mathbf{j})).$$

Negating we obtain

$$R\mathbf{ij} \to (\mathbf{i} \wedge \Box\Diamond\Diamond^{-1}\mathbf{j}),$$

which, translated, becomes

$$\forall y_j \exists x_0 [R y_i y_j \to (x_0 = y_i) \wedge \forall y (R x_0 y \to \exists u (R y u \wedge \exists v (R v u \wedge v = y_j)))],$$

and simplifies to

$$\forall y_j[Ry_iy_j \to \forall y(Ry_iy \to \exists u(Ryu \wedge Ry_ju))]$$

defining the Church-Rosser property, as expected. Note that the variable y_i occurs free and corresponds to the nominal $\mathbf{i}$, which we interpret as the current state. Hence we obtain a *local* property. Also, we directly translate the abbreviation $R\mathbf{ij}$ as Ry_iy_j, since $\mathrm{ST}(\mathbf{i} \to \Diamond\mathbf{j}, x_0)$ is $x_0 = y_i \to \exists z(Rx_0z \wedge z = y_j)$, which clearly simplifies.

Example 3.2. Consider the formula $p \wedge \Box(\Diamond p \to \Box q) \to \Diamond\Box\Box q$. Note that it is *not equivalent to a Sahlqvist formula* (see [17]).

Step 1: Negating we obtain $p \wedge \Box(\Box\neg p \vee \Box q) \wedge \Box\Diamond\Diamond\neg q$.
Step 2: $\mathbf{i} \to [p \wedge \Box(\Box\neg p \vee \Box q) \wedge \Box\Diamond\Diamond\neg q]$.
Step 3: Nothing to do, as the system is neither positive nor negative in p or q.
Step 4: Choose p to eliminate.
Step 5: Applying the $\wedge$-rule twice, we get

$$\left\| \begin{array}{l} \mathbf{i} \to p \\ \mathbf{i} \to \Box(\Box\neg p \vee \Box q)) \\ \mathbf{i} \to \Box\Diamond\Diamond\neg q \end{array} \right. .$$

The system is now ready for the application of the Ackermann-rule, as p has been successfully isolated, and $\mathbf{i} \to \Box(\Box\neg p \vee \Box q))$ is negative in p:

$$\left\| \begin{array}{l} \mathbf{i} \to \Box(\Box\neg\mathbf{i} \vee \Box q)) \\ \mathbf{i} \to \Box\Diamond\Diamond\neg q \end{array} \right. .$$

Step 4: Choose q, the only remaining option, to eliminate next.
Step 5: Successively applying the $\Box$-rule, Left-shift $\vee$-rule and $\Box$-rule again, we get

$$\left\| \begin{array}{l} \Diamond^{-1}(\Diamond^{-1}\mathbf{i} \wedge \Diamond\mathbf{i}) \to q \\ \mathbf{i} \to \Box\Diamond\Diamond\neg q \end{array} \right. .$$

Applying the Ackermann-rule to eliminate q, and rewriting in negation normal form yields:

$$\| \ \mathbf{i} \to \Box\Diamond\Diamond[\Box^{-1}(\Box^{-1}\neg\mathbf{i} \vee \Box\neg\mathbf{i})] \ .$$

Step 6: $(\mathbf{i} \to \Box\Diamond\Diamond[\Box^{-1}(\Box^{-1}\neg\mathbf{i} \vee \Box\neg\mathbf{i}))$, when negated becomes $\mathbf{i} \wedge \Diamond\Box\Box[\Diamond^{-1}(\Diamond^{-1}\mathbf{i} \wedge \Diamond\mathbf{i})$. Translating into first-order logic we obtain

$$\exists x_0[x_0 = y_i \wedge \exists z_1(Rx_0z_1 \wedge \forall z_2(Rz_1z_2 \to \forall z_3(Rz_2z_3 \to \exists u_1[Ru_1z_3 \\ \wedge \exists u_2(Ru_2u_1 \wedge u_2 = y_i) \wedge \exists u_3(Ru_1u_3 \wedge u_3 = y_i)]))))].$$

Note that, in this example, the order of elimination of the propositional variables is inessential, as eliminating q first and then p works equally well.

Example 3.3. Consider the formula $\Box(\Box p \leftrightarrow q) \to p$. The current implementations of both SCAN (`http://www.mpi-inf.mpg.de/~ohlbach/scan/corr_form.html`) and DLS (`http://www.ida.liu.se/labs/kplab/projects/dls`) fail on it. Let's see what SQEMA does with it:

Step 1: $\Box((\Diamond\neg p \vee q) \wedge (\neg q \vee \Box p)) \wedge \neg p$.
Step 2:

$$\| \ \mathbf{i} \to \Box((\Diamond\neg p \vee q) \wedge (\neg q \vee \Box p)) \wedge \neg p \ .$$

Step 3: All variables occur both positively and negatively, so we go on to step 4.

Step 4: Choose q to eliminate.
Step 5: Applying the $\wedge$-rule and the $\Box$-rule we transform the system into

$$\left\|\begin{array}{l}\Diamond^{-1}\mathbf{i} \to (\Diamond\neg p \vee q) \\ \Diamond^{-1}\mathbf{i} \to (\neg q \vee \Box p) \\ \mathbf{i} \to \neg p\end{array}\right. .$$

Applying the Left Shift $\vee$-rule to the first equation yields

$$\left\|\begin{array}{l}(\Diamond^{-1}\mathbf{i} \wedge \Box p) \to q \\ \Diamond^{-1}\mathbf{i} \to (\neg q \vee \Box p) \\ \mathbf{i} \to \neg p\end{array}\right. .$$

to which the Ackermann-rule is applicable with respect to q. This gives

$$\left\|\begin{array}{l}\Diamond^{-1}\mathbf{i} \to (\neg\Diamond^{-1}\mathbf{i} \vee \neg\Box p \vee \Box p) \\ \mathbf{i} \to \neg p\end{array}\right. .$$

The first equation in the system is now a tautology and may be removed (we interpret systems of equations conjunctively), yielding the system

$$\|\ \mathbf{i} \to \neg p \quad ,$$

in which p may be replaced by $\bot$ since it occurs only negatively, resulting in the system

$$\|\ \top$$

Step 6: Negating we obtain $\bot$.

Some remarks are in order here. Firstly, note that the success of the algorithm may depend essentially upon the ability to do some propositional reasoning. Particularly, it had to recognize $\Diamond^{-1}\mathbf{i} \to (\neg\Diamond^{-1}\mathbf{i} \vee \neg\Box p \vee \Box p)$ as a tautology. If this had been written in a slightly different form, however, say as $\Diamond^{-1}\mathbf{i} \to (\neg\Diamond^{-1}\mathbf{i} \vee \Diamond\neg p \vee \Box p)$, its tautological status may not be recognized so easily. This is where a stronger version of Ackermann-rule (see Section 6), which involves testing for *monotonicity*, rather than polarity would prove to be useful. For, testing the consequent for (upward) monotonicity would give the answer 'yes' — this is, post hoc, easy to see, since we already know that the consequent is a tautology. This would allow us, after changing the polarity of p, to apply the stronger, monotonicity based, version of the Ackermann-rule to the system

$$\left\|\begin{array}{l}\Diamond^{-1}\mathbf{i} \to (\neg\Diamond^{-1}\mathbf{i} \vee \Diamond p \vee \Box\neg p) \\ \mathbf{i} \to p\end{array}\right. ,$$

and obtain

$$\|\ \Diamond^{-1}\mathbf{i} \to (\Box^{-1}\neg\mathbf{i} \vee \Diamond\mathbf{i} \vee \Box\neg\mathbf{i}) \quad ,$$

which simplifies to

$$\|\ \Diamond^{-1}\mathbf{i} \to (\Box^{-1}\neg\mathbf{i} \vee \Diamond\mathbf{i} \vee \neg\Diamond\mathbf{i}) \quad ,$$

which, as before, is a tautology, yielding, in step 6, the first order equivalent $\bot$.

Secondly, suppose that we had tried to eliminate p first. Note that we will gain nothing by changing the polarity of p for we cannot get the occurrence of p 'out' under the diamond in $\Diamond^{-1}\mathbf{i} \to (\Diamond p \vee q)$. Indeed, the system may be transformed to become

$$\left\|\begin{array}{l}\Diamond^{-1}\mathbf{i} \to (\Diamond\neg p \vee q) \\ \Diamond^{-1}(\Diamond^{-1}\mathbf{i} \wedge q) \to p \\ \mathbf{i} \to \neg p\end{array}\right. ,$$

to which we may apply the Ackermann-rule with respect to p and obtain

$$\left\| \begin{array}{l} \Diamond^{-1}\mathbf{i} \rightarrow (\Diamond\neg(\Diamond^{-1}(\Diamond^{-1}\mathbf{i} \wedge q)) \vee q) \\ \mathbf{i} \rightarrow \neg(\Diamond^{-1}(\Diamond^{-1}\mathbf{i} \wedge q)) \end{array} \right. .$$

It is now not difficult to see that SQEMA gets stuck on this system.

Moral: the order of elimination does matter sometimes, that is why the algorithm incorporates the backtrack option. Of course, theoretically this may lead to an exponential increase of the number of steps, but in practice this apparently does not happen if suitable heuristics or additional rules can be applied to determine the right order of elimination.

4. Correctness and canonicity of SQEMA

4.1. **Modal formulae as operators on general frames.** For what follows, it is useful to think of modal formulae as set-theoretic operators. If $\psi \in ML^+$, let $\mathsf{PROP}(\psi) = \{q_1, \dots, q_m\}$ be the propositional variables occurring in ψ, $\mathsf{NOM}(\psi) = \{\mathbf{i}_1, \dots, \mathbf{i}_n\}$ the nominals occurring in ψ, and $\mathsf{AT}(\psi) = \mathsf{PROP}(\psi) \cup \mathsf{NOM}(\psi)$. Recall that with every model $\mathcal{M} = \langle W, R, V\rangle$ and a modal formula ψ in ML we associate the set $[\![\psi]\!]_{\mathcal{M}}$ denoting the extension of the formula ψ in the model $\mathcal{M}$. Clearly, $[\![\psi]\!]_{\mathcal{M}}$ only depends on the valuation of $\mathsf{PROP}(\psi)$ in $\mathcal{M}$, and therefore $[\![\psi]\!]$ defines an operator from $\mathcal{P}(W)^m$ to $\mathcal{P}(W)$. Likewise for formulae of the extended language ML^+, with the amendment that if $\mathsf{PROP}(\psi) = \{q_1, \dots, q_m\}$ and $\mathsf{NOM}(\psi) = \{\mathbf{i}_1, \dots, \mathbf{i}_n\}$, then $[\![\psi]\!] : \mathcal{P}(W)^m \times W^n \rightarrow \mathcal{P}(W)$. Whenever appropriate, we will simply identify formulae with the operators they define. For instance, if $\varphi = \Box^{-1}p \vee \Diamond\mathbf{j}$, $x \in W$ and $A \subseteq W$, we will write $\Box^{-1}A \vee \Diamond\{x\}$ or $\Box^{-1}A \cup \Diamond\{x\}$ to denote the extension of φ when the valuation of p is A and that of $\mathbf{j}$ is $\{x\}$. If we have to specify the relation, we will write $\langle R\rangle$ and $[R]$ for $\Diamond A$ and $\Box A$ respectively. Also $R(A)$ (respectively $R^{-1}(A)$) will be used to denote the set of successors (respectively predecessors) of points in A. We will be sloppy and write $R(x)$ (respectively $R^{-1}(x)$) for $R(\{x\})$ (respectively $R^{-1}(\{x\})$).

We will now extend (*ad hoc*) the notion of local validity at a state w of a general frame $\mathfrak{F} = \langle W, R, \mathbb{W}\rangle$ for the basic language ML, to formulae of the extended language ML^+ as follows: For a formula $\varphi \in ML^+$, it is the case that $\mathfrak{F}, w \Vdash \psi$ if $\mathcal{M}, w \Vdash \psi$ for every model $\mathcal{M} \in M(\mathfrak{F})$ extended in a standard way to the inverse modalities, and where all nominals can take as values any $v \in W$, *except for the special nominal* $\mathbf{i}$ *which is only interpreted in the current state* w. Note that, in doing so, we lose the property that the extension of any formula in a model based on a general frame will itself be an admissible set (i.e. a member of $\mathbb{W}$).

4.2. **Topology on Descriptive Frames.** Hereafter, unless otherwise specified, we only deal with general (and in particular, descriptive) frames in ML. For the basic topological concepts used in this section, the reader may consult any standard reference on topology, e.g. [30].

With every general frame $\mathfrak{F} = \langle W, R, \mathbb{W}\rangle$, we associate a topological space $(W, T(\mathfrak{F}))$, where $\mathbb{W}$ is taken as a base of clopen sets for the topology $T(\mathfrak{F})$. Let $\mathbf{C}(\mathbb{W})$ denote the set of sets closed with respect to $T(\mathfrak{F})$.

A general frame $\mathfrak{F} = \langle W, R, \mathbb{W}\rangle$ is said to be *differentiated* if for every $x, y \in W$, $x \neq y$, there exists $X \in \mathbb{W}$ such that $x \in X$ and $y \notin X$ (equivalently, if $T(\mathfrak{F})$ is Hausdorff); *tight* if for all $x, y \in W$ it is the case that Rxy iff $x \in \bigcap\{\langle R\rangle(Y) : Y \in \mathbb{W} \text{ and } y \in Y\}$ (equivalently, if R is point-closed, i.e. R({x}) is closed for every $x \in W$); *compact* if every family of admissible sets from $\mathbb{W}$ with the finite intersection property (FIP) has non empty intersection (equivalently, if $T(\mathfrak{F})$ is compact). $\mathfrak{F}$ is called *descriptive* it is differentiated, tight and compact.

Definition 4.1. A formula $\psi \in$ ML is *locally d-persistent* if for every descriptive frame $\mathfrak{F} = \langle W, R, \mathbb{W} \rangle$ and $w \in W$,

$$\mathfrak{F}, w \Vdash \psi \text{ iff } F, w \Vdash \psi,$$

where $F = \langle W, R \rangle$.

Note that we only talk about local d-persistence of formulae from the basic language ML. It is well known (see e.g. [2]) that (local) d-persistence implies canonicity of formulae in ML because the canonical general frame for every normal modal logic is descriptive, and hence every d-persistent axiom is valid in its canonical frame.

Definition 4.2. A formula $\gamma = \gamma(p_1, \dots, p_n)$ from ML^+ is a *closed operator in* ML, if for every descriptive frame $\mathfrak{F} = \langle W, R, \mathbb{W} \rangle$, if $P_1, \dots, P_n \in \mathbf{C}(\mathbb{W})$ and any singletons are assigned to the nominals in γ, then $\gamma(P_1, \dots, P_n) \in \mathbf{C}(\mathbb{W})$, i.e. when applied to closed sets in a descriptive frame it produces a closed set; γ is a *closed formula in* ML if whenever applied to *admissible* sets in any descriptive frame it produces a closed set. Thus, if a formula is a closed operator, then it is a closed formula, but not necessarily vice versa.

Similarly, a formula from ML^+ is an *open operator in* ML if whenever applied to open sets in a descriptive frame it produces an open set; it is an *open formula in* ML if whenever applied to admissible sets it produces an open set.

Note that the operators $\Diamond$ and $\Diamond^{-1}$ distribute over arbitrary unions, and $\Box$ and $\Box^{-1}$ distribute over arbitrary intersections. Since every closed set can be obtained as the intersection admissible sets and each open set as the union of admissible sets, and $\Diamond$ and $\Box$ applied to admissible sets yield admissible sets, it follows that $\Diamond p$ is an open operator and $\Box p$ is a closed operator, and this holds even for arbitrary general frames.

The proof of the next lemma, originally from [11], can also be found in [22] or [2].

Lemma 4.3 (Esakia's Lemma for $\Diamond$)**.** *Let $\mathfrak{F}$ be a descriptive frame. Then for any downward directed family of nonempty closed sets $\{C_i : i \in I\}$ from $T(\mathfrak{F})$, it is the case that $\Diamond \bigcap_{i\in I} C_i = \bigcap_{i\in I} \Diamond C_i$.*

Corollary 4.4. *$\Diamond p$ is a closed operator in ML.*

Lemma 4.5 (Esakia's Lemma for $\Diamond^{-1}$ in ML)**.** *Let $\mathfrak{F}$ be a descriptive frame. Then $\Diamond^{-1} \bigcap_{i\in I} C_i = \bigcap_{i\in I} \Diamond^{-1} C_i$ for any downwards directed family of nonempty closed sets $\{C_i : i \in I\}$ from $T(\mathfrak{F})$.*

Proof. The inclusion $\Diamond^{-1} \bigcap_{i\in I} C_i \subseteq \bigcap_{i\in I} \Diamond^{-1} C_i$ is trivial, so suppose that $x_0 \notin \Diamond^{-1} \bigcap_{i\in I} C_i$, i.e. $\langle R \rangle(x_0) \cap \bigcap_{i\in I} C_i = \emptyset$. Now $\langle R \rangle(x_0)$ is closed by corollary 4.4 and the fact that singletons are closed in descriptive frames. Hence $\{\langle R \rangle(x_0)\} \cup \{C_i : i \in I\}$ is a family of closed subsets with empty intersection which, by compactness, cannot have the FIP. Thus there is a finite subfamily $\{C_1, \dots, C_n\} \subseteq \{C_i : i \in I\}$ such that $\langle R \rangle(x_0) \cap C_1 \cap \dots \cap C_n = \emptyset$. Since $\{C_i : i \in I\}$ is downwards directed, it follows that there that there exists a $C \in \{C_i : i \in I\}$ such that $C \subseteq \bigcap\{C_1, \dots, C_n\}$ and $\langle R \rangle(x_0) \cap C = \emptyset$. But then $x_0 \notin \Diamond^{-1} C$, and hence $x_0 \notin \bigcap_{i\in I} \Diamond^{-1} C_i$. □

Lemma 4.6. *$\Diamond^{-1}p$ is a closed operator in ML.*

Proof. The proof is adapted from [17]. Let $\mathfrak{F} = \langle W, R, \mathbb{W}\rangle$ be a descriptive frame in ML. We have to show that for any closed $A \subseteq W$ it is the case that $\Diamond^{-1}(A) = \bigcap\{B \in \mathbb{W} : \Diamond^{-1}A \subseteq B\}$. Note that $\Diamond^{-1}(A) = R(A)$. The inclusion from left to right is trivial. In order to prove the right-to-left inclusion, suppose that $x_0 \notin \Diamond^{-1}(A)$, i.e. for all $y \in A$ it is not the case that Ryx_0. By the point-closedness of R we have that $R(y) = \bigcap\{B \in \mathbb{W} : y \in \Box B\}$. Then for each $y \in A$ there must exist a $B^y \in \mathbb{W}$ such that $y \in \Box B^y$ and $x_0 \notin B^y$, and hence $A \subseteq \bigcup\{\Box B^y : y \in A\}$. Therefore $\{\Box B^y : y \in A\}$ is an open cover of the closed set A, so by compactness there exists a finite subcover $\Box B_1, \dots, \Box B_n$. Then $A \subseteq \Box B_1 \cup \dots \cup \Box B_n$ and $x_0 \notin B_i, 1 \leq i \leq n$. Since $\Diamond^{-1}$ distributes over arbitrary unions, we then have $\Diamond^{-1}A \subseteq \Diamond^{-1}\Box B_1 \cup \dots \cup \Diamond^{-1}\Box B_n$. And, since for any $X \subseteq W$, $\Diamond^{-1}\Box X \subseteq X$, we have $\Diamond^{-1}A \subseteq B_1 \cup \dots \cup B_n$. So we have found an admissible set containing $\Diamond^{-1}A$, not containing x_0, and hence $x_0 \notin \bigcap\{B \in \mathbb{W} : \Diamond^{-1}A \subseteq B\}$, proving the inclusion and the lemma. □

Corollary 4.7. *$\Box^{-1}$ is an open operator in ML.*

Proof. By the duality of $\Diamond^{-1}$ and $\Box^{-1}$. □

Definition 4.8. A formula $\varphi \in ML^+$ is *syntactically closed* if all occurrences of nominals and $\Diamond^{-1}$ in φ are positive, and all occurrences of $\Box^{-1}$ in φ are negative; φ is *syntactically open* if all occurrences of $\Diamond^{-1}$ and nominals in φ are negative, and all occurrences of $\Box^{-1}$ in φ are positive. Clearly $\neg$ maps syntactically open formulae to syntactically closed formulae, and vice versa.

Lemma 4.9. *Every syntactically closed formula from ML^+ is a closed formula, and every syntactically open formula is an open formula.*

Proof. By structural induction on syntactically open / closed formulae, written in negation normal form, using the facts that $\Diamond$ and $\Box$ are open and closed operators, $\Diamond^{-1}$ is a closed operator, $\Box^{-1}$ and open operator and the fact that singletons are closed in descriptive frames. □

Lemma 4.10. *Let $\varphi(q_1, \dots, q_n, p, \mathbf{i}_1, \dots \mathbf{i}_m)$ be a syntactically closed formula which is positive in p and with $\mathsf{PROP}(\varphi) = \{q_1, \dots, q_n, p\}$ and $\mathsf{NOM}(\varphi) = \{\mathbf{i}_1, \dots \mathbf{i}_m\}$. Let $\mathfrak{F} = \langle W, R, \mathbb{W}\rangle$ be a descriptive frame in ML. Then for all $Q_1, \dots, Q_n \in \mathbb{W}$, $x_1, \dots, x_m \in W$, and $C \in \mathbf{C}(\mathbb{W})$, it is the case that $\varphi(Q_1, \dots, Q_n, C, \{x_1\}, \dots, \{x_m\})$ is closed in $T(\mathfrak{F})$.*

Proof. We assume that φ is written in negation normal form, hence we may also assume that $\Box^{-1}$ does not occur, as all occurrences have to be negative, and rewriting in negation normal form will change these into $\Diamond^{-1}$. We proceed by induction on φ. If φ is $\top$, $\bot$ or one of $q_1, \dots, q_n, p, i_1, \dots i_m$ it is clear that $\varphi(Q_1, \dots, Q_n, \{x_1\}, \dots, \{x_m\}, C)$ is a closed set. This is also the case if φ is the negation of a propositional variable from $q_1, \dots, q_n$. The case when φ is $\neg p$ does not occur.

The cases for $\wedge$ and $\vee$ follow since the finite unions and intersections of closed sets are closed. The cases for $\Diamond$ and $\Diamond^{-1}$ follow from corollary 4.4 and 4.6, respectively. The case for $\Box$ follows is immediate from the fact that $\Box$ is a closed operator, as was noted earlier. □

Lemma 4.11 (Esakia's Lemma for Syntactically Closed Formulae)**.** *Let* $\varphi(q_1, \dots, q_n, p, \mathbf{i}_1, \dots \mathbf{i}_m)$ *be a syntactically closed formula with* $\mathsf{PROP}(\varphi) = \{q_1, \dots, q_n, p\}$ *and* $\mathsf{NOM}(\varphi) = \{\mathbf{i}_1, \dots \mathbf{i}_m\}$ *which is positive in* p*. Let* $\mathfrak{F} = \langle W, R, \mathbb{W} \rangle$ *be a descriptive frame in* ML*. Then for all* $Q_1, \dots, Q_n \in \mathbb{W}$*,* $x_1, \dots, x_m \in W$ *and downwards directed family of closed sets* $\{C_i : i \in I\}$ *it is the case that*

$$\varphi(Q_1, \dots, Q_n, \bigcap_{i \in I} C_i, \{x_1\}, \dots, \{x_m\}) = \bigcap_{i \in I} \varphi(Q_1, \dots, Q_n, C_i, \{x_1\}, \dots, \{x_m\}).$$

Proof. The proof is by induction on φ. For brevity we will omit the parameters $Q_1, \dots, Q_n$, $x_1, \dots, x_m$ when writing (sub)formulae. We assume that formulae are written in negation normal form, i.e we may also assume that $\Box^{-1}$ does not occur, as all occurrences have to be negative, and rewriting in negation normal form will change these into $\Diamond^{-1}$. The cases when φ is $\bot$, $\top$ or among $q_1, \dots, q_n, p, i_1, \dots i_m$ are trivial, as are the cases when φ is the negation of a propositional variable among $q_1, \dots, q_n$. The inductive step in the case when φ is of the form $\gamma_1 \wedge \gamma_2$ is also trivial.

Suppose φ is of the form $\gamma_1 \vee \gamma_2$. We have to show that $\gamma_1(\bigcap_{i \in I} C_i) \cup \gamma_2(\bigcap_{i \in I} C_i) = \bigcap_{i \in I}(\gamma_1(C_i) \cup \gamma_2(C_i))$. The interesting inclusion is from right to left, so assume that $x_0 \notin \gamma_1(\bigcap_{i \in I} C_i) \cup \gamma_2(\bigcap_{i \in I} C_i)$, i.e $x_0 \notin \bigcap_{i \in I} \gamma_1(C_i) \cup \bigcap_{i \in I} \gamma_2(C_i)$, by the induction hypothesis. Thus there exists $C_1, C_2 \in \{C_i : i \in I\}$ such that $x_0 \notin \gamma_1(C_1)$ and $x_0 \notin \gamma_2(C_2)$. By the downward directedness of $\{C_i : i \in I\}$ there is a $C \in \{C_i : i \in I\}$ such that $C \subseteq C_1 \cap C_2$. Thus, since γ_1 and γ_2 are positive and hence upwards monotone in p, it follows that $x_0 \notin \gamma_1(C)$ and $x_0 \notin \gamma_2(C)$, and hence that $x_0 \notin \bigcap_{i \in I}(\gamma_1(C_i) \cup \gamma_2(C_i))$.

Suppose φ is of the form $\Diamond\gamma$. We have to show that $\Diamond\gamma(\bigcap_{i \in I} C_i) = \bigcap_{i \in I} \Diamond\gamma(C_i)$. By the inductive hypothesis we have $\Diamond\gamma(\bigcap_{i \in I} C_i) = \Diamond \bigcap_{i \in I} \gamma(C_i)$. If $\gamma(C_i) = \emptyset$ for some C_i, then $\Diamond \bigcap_{i \in I} \gamma(C_i) = \emptyset = \bigcap_{i \in I} \Diamond\gamma(C_i)$, so we may assume that $\gamma(C_i) \neq \emptyset$ for all $i \in I$. Then, by Lemma 4.10, $\{\gamma(C_i) : i \in I\}$ is a family of non-empty closed sets. Moreover, $\{\gamma(C_i) : i \in I\}$ is downwards directed. For, consider any finite number of members of $\{\gamma(C_i) : i \in I\}$, $\gamma(C_1), \dots, \gamma(C_n)$, say. Then there is a $C \in \{C_i : i \in I\}$ such that $C \subseteq \bigcap_{i=1}^{n} C_i$. But then $\gamma(C) \in \{\gamma(C_i) : i \in I\}$ and $\gamma(C) \subseteq \bigcap_{i=1}^{n} \gamma(C_i)$ by the upwards monotonicity of γ in p. Now we may apply Lemma 4.3 and conclude that $\Diamond \bigcap_{i \in I} \gamma(C_i) = \bigcap_{i \in I} \Diamond\gamma C_i)$.

The case when φ is of the form $\Diamond^{-1}\gamma$ is verbatim the same the previous case, except that we appeal to Lemma 4.5 rather than Lemma 4.3 in the last step.

Lastly, consider the case when φ is of the form $\Box\gamma$. This follows by the inductive hypothesis and the fact that $\Box$ distributes over arbitrary intersections of subsets of W. $\Box$

The next lemma is needed for the following reason: for the proof of canonicity of all formulae on which SQEMA succeeds we will need a version of Ackermann's lemma that is true of formulae in the extended language ML^+, when interpreted over descriptive frames. As already noted, the extension of an ML^+-formula need in general not be an admissible set in such general frames. This creates and obvious impediment for the proof of one direction of the equivalence in the modal Ackermann's lemma, as the interpretations of propositional variables *must* be admissible sets. We can push it through, however, at the price of additional restrictions on formulae in terms of syntactic openness and closedness.

Lemma 4.12 (Restricted Version of Ackermann's Lemma for Descriptive Frames)**.** *Let:*

- $\mathfrak{F} = \langle W, R, \mathbb{W} \rangle$ *be a descriptive or Kripke frame in* ML*;*

- *$A(q_1, ..., q_n, \mathbf{i}_1, ..., \mathbf{i}_m)$ be a syntactically closed formula with $\mathsf{PROP}(A) \subseteq \{q_1, ..., q_n\}$ and $\mathsf{NOM}(A) \subseteq \{\mathbf{i}_1, ..., \mathbf{i}_m\}$;*
- *$B(q_1, ..., q_n, p, \mathbf{i}_1, ..., \mathbf{i}_m)$ be a syntactically open formula with $\mathsf{PROP}(B) \subseteq \{q_1, ..., q_n, p\}$ and $\mathsf{NOM}(B) \subseteq \{\mathbf{i}_1, ..., \mathbf{i}_m\}$, which is downwards monotone in p.*

Then for all $Q_1, \dots, Q_n \in \mathbb{W}$ and $x_1, \dots, x_m \in W$:

$$B(Q_1, \dots, Q_n, A(Q_1, \dots, Q_n, \{x_1\}, \dots, \{x_m\}), \{x_1\}, \dots, \{x_m\}) = W$$

if and only if there is a $P \in \mathbb{W}$ such that

$$A(Q_1, \dots, Q_n, \{x_1\}, \dots, \{x_m\}) \subseteq P \text{ and } B(Q_1, \dots, Q_n, P, \{x_1\}, \dots, \{x_m\}) = W.$$

Proof. For the sake of brevity we will omit the parameters $Q_1, \dots, Q_n, \{x_1\}, \dots, \{x_m\}$ in what follows, and simply write A, $B(P)$ etc. The implication from right to left follows by the downwards monotonicity of B in p. For the converse, if $\mathfrak{F}$ is a Kripke frame, i.e $\mathbb{W} = 2^W$, we can simply take $P = A(Q_1, \dots, Q_n, \{x_1\}, \dots, \{x_m\})$, since all subsets of W are admissible.

Now, assume that $\mathfrak{F}$ is any descriptive frame and $B(A) = W$. Let $B'(p)$ be the negation of $B(p)$ written in negation normal form. Then $B'(p)$ is a syntactically closed formula, and $B'(A) = \emptyset$. We need to find an admissible set $P \in \mathbb{W}$ such that $A \subseteq P$ and $B'(P) = \emptyset$. Since A is a syntactically closed formula, it follows by Lemma 4.9 that A is a closed subset of W and hence that $A = \bigcap\{C \in \mathbb{W} : A \subseteq C\}$. Hence $\emptyset = B'(A) = B'(\bigcap\{C \in \mathbb{W} : A \subseteq C\}) = \bigcap\{B'(C) : C \in \mathbb{W} \text{ and } A \subseteq C\}$, by Lemma 4.11. Again by Lemma 4.9, $\{B'(C) : C \in \mathbb{W}, A \subseteq C\}$ is a family of closed sets with empty intersection. Hence, by compactness, there must be a finite subfamily, $C_1, \dots, C_n$ say, such that $\bigcap_{i=1}^{n} B'(C_i) = \emptyset$. But then $C = \bigcap_{i=1}^{n} C_i$ is an admissible set containing A, and $B'(C) = \emptyset$, i.e $B(C) = W$. Hence we can choose $P = C$. ☐

We will refer to the equations which are pure formulae (i.e. do not contain propositional variables) as *pure equations*, and to the rest, as *non-pure equations*.

Lemma 4.13. *During the entire (successful or unsuccessful) execution of* SQEMA *on any input formula from ML, all formulae on the left-hand side of all non-pure* SQEMA*-equations are syntactically closed, and all formulae on the right-hand side of the non-pure equations are syntactically open.*

Proof. We follow any one branch of the execution, proceeding by induction. The starting system is of the form $\|\mathbf{i} \rightarrow \psi$, where $\psi \in ML$. For this system the conditions of the lemma hold, since $\mathbf{i}$ is syntactically closed and all ML-formulae are both syntactically closed and open. Now suppose that in the process of the execution we have reached a system satisfying the conditions of the lemma, i.e. all formulae on the left-hand side of all non-pure SQEMA-equations are syntactically closed, and all formulae on the right-hand side of the non-pure equations are syntactically open. It is straightforward to check that the application of any transformation rule to this system will preserve these conditions. In the particular case when the Ackermann-rule is applied, we note the following: (i) the pure equations in the system contain no propositional variables, and are hence disregarded in any application of the Ackermann-rule; (ii) by the inductive hypothesis the disjunction of left-hand sides which is substituted for the variable being eliminated are syntactically closed; (iii) substituting a syntactically closed formula for positive (resp., negative) occurrences of a variable in a syntactically closed (resp., open) formula yields a syntactically closed (resp., open) formula. □

The content of the next lemma is essentially that the global satisfiability of systems of SQEMA-equations is invariant under all SQEMA transformation rules, subject to the constraint that $\mathbf{i}$ is evaluated to the current state.

Lemma 4.14. *Let $E_0, \dots, E_r$ be the sequence of systems of equations produced on one branch by SQEMA when executed on a certain input formula $\varphi(q_1, \dots, q_n) \in ML$ with $\mathsf{PROP}(\varphi) = \{q_1, \dots, q_n\}$, and let $\mathbf{i}, \mathbf{i}_1, \dots, \mathbf{i}_m$ be the nominals introduced during the execution. Further, let $\varphi_i(q_1, \dots, q_n, \mathbf{i}, \mathbf{i}_1, \dots, \mathbf{i}_m)$ be the formula obtained by taking the conjunction of the equations in E_i for $0 \leq i \leq r$. Then for any descriptive or Kripke frame $\mathfrak{F} = \langle W, R, \mathbb{W} \rangle$ and current state $w \in W$, there are $Q_1, \dots, Q_n \in \mathbb{W}$ and $x_1, \dots, x_m \in W$ such that*

$$\varphi_i(Q_1, \dots, Q_n, \{w\}, \{x_1\}, \dots, \{x_m\}) = W$$

if and only of there are $Q_1, \dots, Q_n \in \mathbb{W}$ and $x_1, \dots, x_m \in W$ such that

$$\varphi_{i+1}(Q_1, \dots, Q_n, \{w\}, \{x_1\}, \dots, \{x_m\}) = W,$$

for $0 \leq i < r$.

Proof. Note that $\varphi_1 = \mathbf{i} \rightarrow \neg\varphi$. For each i, the system E_{i+1} is obtained from the system E_i by the application of some transformation rule. Let $\mathfrak{F}$ be a descriptive or Kripke frame, and w a state in it. We need to verify that, whichever transformation rule was applied, φ_i is globally satisfiable on $\mathfrak{F}$ if and only if φ_{i+1} is, subject to the constraint that $\mathbf{i}$ be always interpreted as w. This is immediate to see for all the transformation rules, except the Ackermann-rule, since they are based on simple propositional and (modal and hybrid) semantic equivalences. The only interesting case then, is the one for the Ackermann-rule. Suppose E_{i+1} is obtained from E_i by application of this rule. Then $\varphi_i = \bigwedge_{j=1}^{l}(\alpha_j \rightarrow p) \wedge \bigwedge_{j=1}^{m}(\beta_j) \wedge \bigwedge_{j=1}^{o}\gamma_j$, where no α_j contains p, each β_j is negative in p, and no γ_j contains any occurrence of p. Note that all pure equations in the system E_i will be among the γ_i. Now all the α_j's are left-hand sides of non-pure SQEMA equations and hence, by lemma 4.13, $\bigvee_{j=1}^{l}\alpha_j$ is syntactically closed. Further, if we eliminate the implication sign from the β_j's, each becomes a disjunction of the negation of a left-hand side of a non-pure SQEMA-equation with the right-hand side of such an equation, which, again by lemma 4.13, is syntactically open.

Then $\varphi_{i+1} = \bigwedge_{j=1}^{m}(\beta'_j) \wedge \bigwedge_{j=1}^{o}\gamma_j$, where each β'_j is obtained from β_j by substituting $\bigvee_{j=1}^{l}\alpha_j$ for all occurrences of p. The proof is complete once we appeal to lemma 4.12. □

Theorem 4.15. *If SQEMA succeeds on a formula $\varphi \in ML$, then φ is locally d-persistent and hence canonical, and moreover the first-order formula returned by SQEMA is a local equivalent of φ.*

Proof. Suppose that SQEMA succeeds on $\varphi \in ML$, and that $\mathsf{pure}(\varphi)$ is the pure formula obtained as the conjunction of the final system of SQEMA-equations in the execution. Further, for simplicity and without loss of generality, assume that the execution does not branch because of disjunctions. We may make this assumption since a conjunction of d-persistent formulae is d-persistent, and the conjunction of local first-order correspondents of modal formulae is a local first-order correspondent for the conjunction of those formulae.

For the canonicity, let $\mathfrak{F} = \langle W, R, \mathbb{W}\rangle$ be a descriptive frame and $w \in W$. Then:

$\mathfrak{F}, w \not\Vdash \varphi$ iff

for some model $\mathcal{M}$ based on $\mathfrak{F}$ with $\mathbf{i}$ denoting w, it is the case that $\mathcal{M} \Vdash \mathbf{i} \to \neg\varphi$.

Note that $\|\mathbf{i} \to \neg\varphi$ is exactly the first system of SQEMA-equations obtained when the algorithm is run on φ. Hence $\mathsf{pure}(\varphi)$ is obtained from $\mathbf{i} \to \neg\varphi$ by the application of transformation rules.

Hence, $\mathcal{M} \Vdash \mathbf{i} \to \neg\varphi$, for some model $\mathcal{M}$, based on $\mathfrak{F}$ with $\mathbf{i}$ denoting w, iff

(by lemma 4.14 for descriptive frames) for some model $\mathcal{M}$, based on $\mathfrak{F}$ with $\mathbf{i}$ denoting w, it is the case that $\mathcal{M} \Vdash \mathsf{pure}(\varphi)$, iff

for some model $\mathcal{M}$ based on the underlying Kripke frame F with $\mathbf{i}$ denoting w, it is the case that $\mathcal{M} \Vdash \mathsf{pure}(\varphi)$, since we allow nominals to range over all singletons both in Kripke and general frames.

Now by the Kripke frames version of Lemma 4.14, and the fact that $\mathsf{pure}(\varphi)$ is obtained from $\mathbf{i} \to \neg\varphi$ by the application of transformation rules, this is the case iff for some model $\mathcal{M}$ based on the underlying Kripke frame F with $\mathbf{i}$ denoting w, we have $\mathcal{M} \Vdash \mathbf{i} \to \neg\varphi$.

This, in turn, will be so iff $F, w \not\Vdash \varphi$. This proves the local d-persistence of φ.

Now, for the local first-order equivalence let $\mathfrak{F} = \langle W, R\rangle$ be a Kripke frame and $w \in W$. As in the formulation of lemma 4.14, let $E_0, \ldots, E_r$ be the sequence of systems of equations produced by SQEMA when executed on φ and let φ_j be the formula obtained by taking the conjunction of the equations in E_j. We define the *translation of a system* E_j, $\mathrm{TR}(E_j)$, to be the second order formula $\exists\overline{P}\exists\overline{y}\forall x_0 \mathrm{ST}(\varphi_j, x_0)$, where $\overline{P}$ is the tuple of all predicate variables and $\overline{y}$ the tuple of all variables corresponding to nominals *other than* $\mathbf{i}$, occurring in φ_j. Note that y_i, corresponding to $\mathbf{i}$, is the only free variable in $\mathrm{TR}(E_j)$, and that $\mathrm{TR}(E_r)$ is $\exists\overline{y}\forall x_0 \mathrm{ST}(\mathsf{pure}(\varphi), x_0)$. Then:

$\mathfrak{F}, w \Vdash \varphi$ iff

$\mathfrak{F} \models \forall\overline{P} ST(\varphi, x_0)[x_0 := w]$ iff

$\mathfrak{F} \models \forall\overline{P}\exists x_0 ST(\mathbf{i} \wedge \varphi, x_0)[y_i := w]$ iff

$\mathfrak{F} \not\models \exists\overline{P}\forall x_0 ST(\mathbf{i} \to \neg\varphi, x_0)[y_i := w]$, i.e. iff

$\mathfrak{F} \not\models \mathrm{TR}(E_1)[y_i := w]$.

Now, phrased in second-order logic, lemma 4.14 says that

$\mathfrak{F} \models \mathrm{TR}(E_j)[y_i := w]$ if and only if $\mathfrak{F} \models \mathrm{TR}(E_{j+1})[y_i := w]$, for all $1 \leq j < r$.

Hence we get that $\mathfrak{F}, w \Vdash \varphi$ iff $\mathfrak{F} \not\models \exists\overline{y}\forall x_0 \mathrm{ST}(\mathsf{pure}(\varphi), x_0)[y_i := w]$,

i.e that $\mathfrak{F}, w \Vdash \varphi$ iff $\mathfrak{F} \models \forall\overline{y}\exists x_0 \neg\mathrm{ST}(\mathsf{pure}(\varphi), x_0)[y_i := w]$.

Hence $\forall\overline{y}\exists x_0 \neg\mathrm{ST}(\mathsf{pure}(\varphi), x_0)$ is a local first-order correspondent for φ, and exactly what SQEMA returns. Accordingly, $\forall y_i \forall\overline{y}\exists x_0 \neg\mathrm{ST}(\mathsf{pure}(\varphi), x_0)$ is a global first-order correspondent of φ. □

5. Completeness results

In this section we show that SQEMA succeeds in computing the local first-order frame equivalent of every Sahlqvist formulae, and then we extend that to every monadic inductive formula. While the latter result subsumes the former, we nevertheless present both proofs in order to take the reader first through the simpler case.

5.1. **Completeness of SQEMA on Sahlqvist Formulae.** Let us briefly recall what Sahlqvist formulae are (see [2, 29, 21]):

- a *boxed atom* is a propositional variable, prefixed with finitely many (possibly none) boxes.
- a *Sahlqvist antecedent* is a formula built up from $\top$, $\bot$, boxed atoms, and negative formulae, using $\wedge$, $\vee$ and diamonds.
- a *Sahlqvist implication* is a formula of the form $\varphi \rightarrow \mathsf{Pos}$ where φ is a Sahlqvist antecedent and Pos is a positive formula. In particular, note that any negative formula is a Sahlqvist antecedent.
- a *Sahlqvist formula* is built up from Sahlqvist implications by applying conjunctions, disjunctions, and boxes.[3]

The following lemma is useful:

Lemma 5.1. *Let φ be a Sahlqvist formula, and φ' the formula obtained from $\neg\varphi$ by importing the negation over all connectives. Then φ' is a Sahlqvist antecedent.*

Proof. Induction on the construction of φ from Sahlqvist implications. If φ is a Sahlqvist implication $\alpha \rightarrow \mathsf{Pos}$, negating and rewriting it as $\alpha \wedge \neg\mathsf{Pos}$ already turns it into a Sahlqvist antecedent.

If $\varphi = \Box\psi$, where ψ satisfies the claim, then $\neg\varphi \equiv \Diamond\neg\psi$ hence the claim follows for φ, because Sahlqvist antecedents are closed under diamonds.

Likewise, if $\varphi = \psi_1 \wedge \psi_2$, where ψ_1 and ψ_2 satisfy the claim, then $\neg\varphi \equiv \neg\psi_1 \vee \neg\psi_2$ hence the claim follows for φ, because Sahlqvist antecedents are closed under disjunctions.

The case of $\varphi = \psi_1 \vee \psi_2$ is completely analogous. □

Corollary 5.2. *Every Sahlqvist formula is semantically equivalent to a negated Sahlqvist antecedent, and hence to a Sahlqvist implication.*

Sahlqvist's theorem [21] states that all Sahlqvist formulae are elementary and canonical. For a thorough discussion on Sahlqvist formulae and proof of Sahlqvist's theorem, see e.g. [2].

Lemma 5.3. *Let E be a system of SQEMA equations of the form $\mathbf{j} \rightarrow \beta$, with $\mathbf{j}$ a nominal and β a Sahlqvist antecedent built up without using $\vee$, except possibly inside negative formulae. Let p be any propositional variable occurring both negatively and positively in E. Then E can be transformed, using only the $\wedge$-rule, $\Diamond$-rule, $\Box$-rule and Ackermann-rule, into a system E', again with all equations of the form $\mathbf{j} \rightarrow \beta$, which does not contain p.*

Proof. All positive occurrences of p are in boxed atoms, occurring at most in the scope of conjunctions and diamonds. We first have to separate p, i.e. transform E into a system in which the all equations in which p occurs positively are of the form $\gamma \rightarrow p$, with p not occurring in γ. Start by applying the $\Diamond$-rule and $\wedge$-rule to separate the boxed atoms of p. The system then obtained is still one in which all equations are of the form $\mathbf{j} \rightarrow \beta$, with $\mathbf{j}$ a nominal and β a Sahlqvist antecedent — recall that pure formulae are regarded as both positive and negative. All equations in which p occurs positively are now of the form $\mathbf{j} \rightarrow \Box^n p$. By applying the $\Box$-rule these are transformed into equations of the form $(\Diamond^{-1})^n\mathbf{j} \rightarrow p$. Note that the system is now no longer in the desired form of implications

[3]The definition in [2] requires that disjunctions are only applied to formulae not sharing variables. Apparently, this requirement is unnecessary.

from nominals to Sahlqvist antecedents. This is remedied by applying the Ackermann-rule to eliminate p, together with all equations of the form $(\Diamond^{-1})^n\mathbf{j} \to p$. Note that pure formulae are substituted for negative occurrences of p, hence the substitution leaves negative formulae negative, and hence Sahlqvist antecedents as Sahlqvist antecedents. The system now obtained is once again of the desired form, and p has been eliminated. □

Theorem 5.4. *SQEMA succeeds on every Sahlqvist formula.*

Proof. Let φ be a Sahlqvist formula. Let us see what SQEMA does with it. In step one, $\neg\varphi$ is formed and the negation imported over all connectives. Call the formula so obtained φ'. By Lemma 5.1 φ' is a Sahlqvist antecedent. φ' is now transformed into the form $\bigvee_{j=1}^{n} \alpha_j$ with each α_j a Sahlqvist antecedent built up from $\top$, $\bot$, boxed atoms and negative formulae without using disjunctions by distributing conjunctions and diamonds over disjunctions. Note that it is possible to "extract" all disjunctions, except possibly some within negative formulae, since in φ' none of these disjunctions occur within the scope of boxes.

From here the algorithm proceeds on each disjunct separately; we will follow one of them, α_j. Let $p_1, \ldots, p_m$ be an arbitrary ordering of the variables occurring in α_j. The initial system of equations is

$$\| \ \mathbf{i} \to \alpha_j.$$

Note that this is a system of the form required by Lemma 5.3. If p_1 occurs only positively (negatively) SQEMA eliminates it by substituting it with $\top$ ($\bot$). If p_1 occurs both negatively and positively, it follows from Lemma 5.3 that SQEMA will eliminate it. In both cases the resulting system is again one of the form required by Lemma 5.3. Propositional variables $p_2, \ldots, p_m$ remain to be eliminated. Proceeding inductively, noting that after each elimination Lemma 5.3 remains applicable, it follows that SQEMA will succeed in doing this. Negating and translating the resulting pure formula, SQEMA obtains the first-order (local) frame equivalent for α_j. Taking the conjunction of these equivalents for $1 \leq j \leq n$ yields the first-order (local) frame equivalent for the Sahlqvist formula φ given as input. □

5.2. **Completeness of SQEMA on Monadic Inductive Formulae.**

Definition 5.5. Let $\sharp$ be a symbol not belonging to ML^+. Then a *box-form of* $\sharp$ in ML^+ is defined recursively as follows:

1. $\sharp$ is a box-form of $\sharp$;
2. If $\mathbf{B}(\sharp)$ is a box-form of $\sharp$ and $\Box$ is a box-modality then $\Box\mathbf{B}(\sharp)$ is a box-form of $\sharp$.
3. If $\mathbf{B}(\sharp)$ is a box-form of $\sharp$ and A is a positive formula then $A \to \mathbf{B}(\sharp)$ is a box-form of $\sharp$.

Thus, box-forms of # are, up to semantic equivalence, of the type

$$A_0 \to \Box_1(A_1 \to \ldots \Box_n(A_n \to \#) \ldots)$$

where $\Box_1, \ldots, \Box_n$ are compositions of boxes in ML^+ and $A_1, \ldots, A_n$ are positive formulae (possibly, just $\top$).

Definition 5.6. By substituting a propositional variable p for $\sharp$ in a box-form $\mathbf{B}(\sharp)$ we obtain a *box-formula of* p, namely $\mathbf{B}(p)$. The last occurrence of the variable p is the *head* of $\mathbf{B}(p)$ and every other occurrence of a variable in $\mathbf{B}(p)$ is *inessential* there.

Definition 5.7 ([17, 4])**.** A *monadic regular formula* is a modal formula built up from $\top$, $\bot$, positive formulae and negated box-formulae by applying conjunctions, disjunctions, and boxes.

Definition 5.8. The *dependency digraph* of a set of box-formulae $\mathcal{B} = \{\mathbf{B}_1(p_1), \ldots, \mathbf{B}_n(p_n)\}$ is a digraph $G_{\mathcal{B}} = \langle V, E\rangle$ where $V = \{p_1, \ldots, p_n\}$ is the set of heads in $\mathcal{B}$, and edge set E, such that $p_i E p_j$ iff p_i occurs as an inessential variable in a box from $\mathcal{B}$ with a head p_j. A digraph is *acyclic* if it does not contain oriented cycles (including loops).

The dependency digraph of a formula is the dependency digraph of the set of box-formulae that occur as subformulae of that formula.

Definition 5.9 ([17, 4])**.** A *monadic inductive formula (MIF)* is a monadic regular formula with an acyclic dependency digraph.

Example 5.10. An example of a monadic inductive formula, which is not a Sahlqvist formula (see [15, 17]), is:

$$D = p \wedge \Box(\Diamond p \to \Box q) \to \Diamond\Box\Box q \equiv \neg p \vee \neg\Box(\Diamond p \to \Box q) \vee \Diamond\Box\Box q,$$

obtained as a disjunction of the negated box-formulae $\neg p$ and $\neg\Box(\Diamond p \to \Box q)$, and the positive formula $\Diamond\Box\Box q$. The dependency digraph of D over the set of heads $\{p, q\}$ has only one edge, from p to q.

An example of a regular, but non-inductive formula is $\Box((\neg\Box p \vee q) \vee (\neg q \vee \Box p)) \vee \neg p$, because the heads p and q depend on each other.

The abbreviation NegMIF will be used for the negation of a monadic inductive formula in negation normal form. Note that the class of NegMIFs in a language are precisely those formulae built from $\top$, $\bot$, negative formulae and box-formulae using conjunction, disjunction and diamonds, which have acyclic dependency digraphs. The abbreviation NegMIF^* will be used for NegMIFs built up without the use of disjunction, i.e. the class of all formulae of the language built from $\top$, $\bot$, negative formulae and box-formulae using conjunctions and diamonds, which have acyclic dependency digraphs.

We will call a system of SQEMA equations a NegMIF-*system* (NegMIF^*-*system*) if it has the form

$$\left\| \begin{array}{l} \mathbf{i}_1 \to \alpha_1 \\ \vdots \\ \mathbf{i}_n \to \alpha_n \end{array} \right. ,$$

where each $\mathbf{i}_i$ is a nominal, and the formula $\alpha_1 \wedge \ldots \wedge \alpha_n$, obtained by taking the conjunction of all consequents of equations in the system, is a NegMIF (NegMIF^*) in ML^+.

Lemma 5.11. *Let E be a NegMIF^*-system of SQEMA-equations, and p any propositional variable occurring in E. Then p can be eliminated from E, either by substitution with $\top$ or $\bot$, or by application of the SQEMA transformation rules. Moreover, the system of equations obtained after the elimination of p will again be a NegMIF^*-system.*

Proof. Suppose p occurs only positively (negatively) in the system, then we substitute $\top$ ($\bot$) for all its occurrences, eliminating it, and yielding a NegMIF^*-system.

So suppose p occurs both positively and negatively. We will separate out all positive occurrences to prepare for the application of the Ackermann-rule. The only positive occurrences of p are heads of (possibly trivial) box-formulae in the consequents of the equations of the system. Let $\mathbf{i}_i \to \alpha$ be such an equation, where α contains positive occurrences of p. Exhaustive application of the $\wedge$-rule and the $\Diamond$-rule splits $\mathbf{i}_i \to \alpha$ into equations of the forms $\mathbf{j} \to \Diamond\mathbf{k}$, $\mathbf{l} \to \mathsf{Box}$ and $\mathbf{m} \to \mathsf{Neg}$, with $\mathbf{j}, \mathbf{k}, \mathbf{l}, \mathbf{m}$ nominals, Box a box-formula and Neg a negative formula. As all box-formulae have been left intact, the dependency digraph is unchanged. Hence, after the application of these rules we still have a NegMIF^*-system.

Now it remains to separate the positive occurrences of p out of the equations of the form $\mathbf{l} \to \mathsf{Box}$, where Box is of the form $\Box_1(A_1 \vee \Box_2(\dots \Box_n(A_n \vee p)\dots))$ where each A_i is a negative formula and each $\Box_i$ a finite, possibly empty, sequence of box modalities. Successive alternative applications of the Left shift $\vee$-rule and Left shift $\Box$-rule transforms the equation

$$\mathbf{l} \to A_0 \vee \Box_1(A_1 \vee \dots \Box_n(A_n \vee p)\dots))$$

into

$$\neg A_n \wedge \Diamond_n^{-1}(\dots \Diamond_1^{-1}(\mathbf{l} \wedge \neg A_0)\dots) \to p.$$

The antecedent in the above equation is a positive formula, not containing p, because the dependency graph is loopless. Hence all positive occurrences of p in the system now occur as the consequents in equations of the form $\mathsf{Pos} \to p$, where Pos is a positive formula not containing p. Let ρ be the disjunction of all the antecedents of the equations of the form $\mathsf{Pos} \to p$. Then, by applying the Ackermann-rule, all equations of this form are deleted and p is eliminated by substitution of the positive formula ρ for every negative occurrence of p. Thus, the resulting system does not contain p, all antecedents of equations are nominals, and all consequents of equations are built up from negative formulae and box-formulae, by using only conjunctions and diamonds.

To show that the resulting system is again NegMIF^*, it remains to show that the dependency digraph is acyclic. We will do so by showing that whenever a new arc (q, u) was introduced by the application of the Ackermann-rule, there was already a directed path from vertex q to vertex u in the digraph *before* the substitution. Indeed, for the only way (q, u) could have been introduced was by the substitution of ρ for an inessential occurrence of p in some box-formula with u as head. But then q must occur in ρ, hence, by the construction of ρ, it must have occurred inessentially in some box-formula headed by p. But then (q, p) and (p, u) were arcs in the dependency digraph before the application of the Ackermann-rule, giving the desired path. Thus, the application of the Ackermann-rule cannot introduce cycles in a previously acyclic dependency graph, which completes the argument. □

Theorem 5.12. *SQEMA succeeds on all conjunctions of monadic inductive formulae.*

Proof. An easy inductive argument, noting that the initial SQEMA-systems for every conjunction of monadic inductive formulae are NegMIF^*-systems, and using Lemma 5.11. □

Remark 5.13. It has been proved in [15] (see also [17]) that every Sahlqvist formula is tautologically equivalent to a conjunction of inductive formulae, which now provides another proof of completeness of SQEMA for Sahlqvist formulae.

Corollary 5.14 (Sahlqvist theorem for inductive formulae, see [17])**.** *All monadic inductive formulae are elementary and canonical.*

Conjecture 5.15. All modal formulae on which SQEMA succeeds are locally equivalent to inductive formulae.

6. Concluding remarks and further work

The algorithm SQEMA is presented here in its core version, which is apparently already quite powerful. In particular, we are currently not aware of examples whereby SQEMA fails on a modal formula on which the current implementations of either SCAN or DLS succeed.

Furthermore, SQEMA is amenable to various extensions, some of which are being developed in forthcoming sequels [5, 6, 7] to this paper. In particular:

Monotonicity-based version: While the condition on the formula $B(P)$ in the formulation of Ackermann's lemma is stated in syntactic terms, the proof of that lemma uses a *semantic* property of $B(P)$, viz *upwards, resp. downwards monotonicity* with respect to P. Thus, the lemma can be strengthened accordingly to a *strong*, or *semantic* Ackermann's lemma, which accordingly yields a stronger, semantic version of the respective Ackermann rule used by SQEMA. Moreover, testing a modal formula for monotonicity in a propositional variable occurring in it is decidable, which enables the algorithmic use of such rule in a stronger version of SQEMA, albeit at the price of possibly higher complexity.

Polyadic languages: The algorithm can be accordingly extended to work in arbitrary polyadic modal languages (presented as *purely modal languages* in [15, 17]) and to succeed on all polyadic inductive formulae (ibid.).

Complex formulae: Furthermore, it can be extended with suitable rules to succeed on the so called *complex formulae* introduced in [26]. These can be converted to inductive formulae by using rather non-trivial substitutions.

Least fixed points: As shown in [17], all regular formulae (in arbitrary polyadic languages) have equivalents in first-order logic extended with least fixed points FO+LFP. These equivalents can be computed by a suitable extension of SQEMA with a recursive version of the Ackermann-rule. For instance, that extension succeeds on the Gödel-Löb formula $\Box(\Box p \to p) \to \Box p$ and on Segerberg's induction axiom $IND = [2](q \to [1]q) \to (q \to [2]q)$. Note that these formulae are not canonical anymore.

Theory reasoning: The algorithm SQEMA is amenable to further extensions with rules capturing *theory reasoning*, which enables computing first-order correspondents relativized over classes of frames, and thus succeeding on notorious cases of elementary canonical formulae, such as modal reduction principles over the class of transitive frames.

Acknowledgements

This work was mostly done while V. Goranko was on a sabbatical leave from Rand Afrikaans University. He would also like to acknowledge the financial support for his research from the National Research Foundation of South Africa. Some of the work on this paper was carried out during the visit of V. Goranko and W. Conradie to the University of Manchester, organized by Renate Schmidt and funded by the British Research Council, which we gratefully acknowledge. The cooperation of Dimiter Vakarelov was supported by EC COST Action 274 "Theory and Applications of Relational Structures as Knowledge

Instruments" (TARSKI), `http:\\www.tarski.org`, and by the project No 1510 of the Bulgarian Ministry of Science and Education. We would also like to thank Renate Schmidt and Rajeev Gore for useful discussions and remarks, as well as the referees for some corrections and constructive criticism.

References

[1] W. Ackermann. Untersuchung über das Eliminationsproblem der mathematischen Logic. *Mathematische Annalen,* 110:390-413, 1935.

[2] P. Blackburn, M. de Rijke, and Y. Venema. **Modal Logic**. Cambridge University Press, 2001.

[3] L. A. Chagrova. An undecidable problem in correspondence theory. *Journal of Symbolic Logic*, 56:1261-1272, 1991.

[4] W. Conradie, V. Goranko, and D. Vakarelov. Elementary canonical formulae: a survey on syntactic, algorithmic, and model-theoretic aspects. In: R. Schmidt, I. Pratt-Hartmann, M. Reynolds, and H. Wansing (editors). **Advances in Modal Logic, vol. 5**, Kings College London, 2005, pages 17-51.

[5] W. Conradie, V. Goranko, and D. Vakarelov. Algorithmic correspondence and completeness in modal logic II: Polyadic and hybrid extensions of the algorithm SQEMA. 2006, submitted.

[6] W. Conradie and V. Goranko. Algorithmic correspondence and completeness in modal logic III : Semantic extensions of the algorithm SQEMA. 2006, under submission.

[7] W. Conradie, V. Goranko, and D. Vakarelov. Algorithmic correspondence and completeness in modal logic IV : Recursive and complex extensions of the algorithm SQEMA. In preparation.

[8] M. de Rijke. **Extending Modal Logic**. PhD thesis, ILLC, University of Amsterdam, ILLC Dissertation Series 1993-4, 1993.

[9] P. Doherty, W. Łukaszewicz, and A. Szalas. Computing circumscription revisited. *Journal of Automated Reasoning*, 18(3):297–336, 1997.

[10] T. Engel. Quantifier Elimination in Second-Order Predicate Logic. Diploma Thesis, MPI, Saarbrücken, 1996.

[11] L. L. Esakia. Topological Kripke Models. *Soviet Mathematics Doklady*, 15(1):147–151, 1974.

[12] D. Gabbay, and H.-J. Ohlbach. Quantifier elimination in second-order predicate logic. *South African Computer Journal,* 7:35-43, 1992.

[13] G. Gargov, and V. Goranko. Modal Logic with Names. *Journal of Philosophical Logic*, 22(6):607-636, 1993.

[14] V. Goranko, and D. Vakarelov. Sahlqvist Formulae in Hybrid Polyadic Modal Languages. *Journal of Logic and Computation,* 11(5):737-254, 2001.

[15] V. Goranko, and D. Vakarelov. Sahlqvist Formulas Unleashed in Polyadic Modal Languages. In: F. Wolter et al (eds.). **Advances in Modal Logic, vol. 3**, World Scientific, Singapore, 2002, pp. 221-240.

[16] V. Goranko, U. Hustadt, R. Schmidt, and D. Vakarelov. SCAN is complete for all Sahlqvist formulae. In: R. Berghammer, B. Möller, and G. Struth (editors). **Relational and Kleene-Algebraic Methods in Computer Science (Proc. of RelMiCS 7)**, LNCS 3051, Springer, 2004, pages 149-162.

[17] V. Goranko, and D. Vakarelov. Elementary Canonical Formulae: Extending Sahlqvist Theorem, to appear in *Annals of Pure and Applied Logic*.

[18] J. Gustafsson. An Implementation and Optimization of an Algorithm for Reducing Formulae in Second-Order Logic. Technical Report LiTH-MAT-R-96-04. Dept. of Mathematics, Linkoping University, Sweden, 1996.

[19] A. Nonnengart, and A. Szalas. A fixpoint approach to second-order quantifier elimination with applications to correspondence theory. In: E. Orlowska (editor). **Logic at Work, Essays dedicated to the memory of Helena Rasiowa**. Springer Physica-Verlag, 1998, pages 89-108.

[20] A. Nonnengart, H.-J. Ohlbach, and A. Szalas. Elimination of Predicate Quantifiers. In: H.-J. Ohlbach and U. Reyle (editors). **Logic, Language and Reasoning: Essays in honour of Dov Gabbay, Part I**. Kluwer, 1999, pages 159–181.

[21] H. Sahlqvist. Correspondence and completeness in the first and second-order semantics for modal logic. In: S. Kanger (editor). **Proc. of the 3rd Scandinavian Logic Symposium, Uppsala 1973**, North-Holland, Amsterdam, 1975, pages 110-143.

[22] G. Sambin, and V. Vaccaro. A New Proof of Sahlqvist's Theorem on Modal Definability and Completeness. *Journal of Symbolic Logic*, 54:992-999, 1989.
[23] H. Simmons. The Monotonous Elimination of Predicate Variables. *Journal of Logic and Computation*, 4(1):23-68, 1994.
[24] A. Szalas. On the correspondence between modal and classical logic: An automated approach. *Journal of Logic and Computation*, 3:605–620, 1993.
[25] A. Szalas. Second-order quantifier elimination in modal contexts. In: S. Flesca and G. Ianni (editors). **JELIA'02**, LNAI 2424, Springer-Verlag, 2002, pp. 223-232.
[26] D. Vakarelov. Modal definability in languages with a finite number of propositional variables, and a new extension of the Sahlqvist class. In: P. Balbiani, N.-Y. Suzuki, F. Wolter, and M. Zakharyaschev (editors). **Advances in Modal Logic, vol. 4**, King's College Publications, London, 2003, pp. 495-518.
[27] D. Vakarelov. On a generalization of the Ackermann's Lemma for computing first-order equivalents of modal formulae, abstract. TARSKI Workshop, March 11 - 13, 2003, Toulouse, France.
[28] D. Vakarelov. Modal definability, solving equations in modal algebras and generalizations of the Ackermann lemma. In the Proc. of 5th Panhellenic Logic Symposium, July 25-28, 2005, Athens.
[29] J. F. A. K. van Benthem. **Modal Logic and Classical Logic**. Bibliopolis, Napoli, 1983.
[30] S. Willard. *General Topology*. Dover Publications, 2004.