SQEMA with Universal Modality

Dimiter Georgiev

Sofia University "St. Kliment Ohridski" dimitertg@yahoo.com

Abstract

SQEMA is an algorithm which, for a given formula of the basic modal language, in many cases is able to find its first-order correspondent. In this paper we augment SQEMA to the basic modal language extended with the universal modality.

1 Introduction

The problem with the existence of first-order correspondent formulas for modal formulas was proposed by van Benthem. This problem is not computable, as shown by Chagrova in her PhD thesis in 1989, see [2]. However, there have been solutions for some modal formulas. The most famous class of formulas for which there is a first-order correspondent is the Sahlqvist class, where one can use the Sahlqvist-van Benthem algorithm as described in [1] to obtain first-order correspondents.

There have been other algorithms to find first-order correspondents, for example, in [6] Gabbay and Ohlbach introduced the SCAN algorithm, and in [10], Szalas introduced DLS. SCAN is based on a resolution procedure applied on a Skolemized translation of the modal formula into the first-order logic, while DLS works on the same translation, but is based on a transformation procedure using a lemma by Ackermann. Both algorithms use a procedure of unskolemization, which is not always successful.

In [4] and [5] another algorithm for computing first-order correspondents in modal logic was introduced, called SQEMA, which is based on a modal version of the Ackermann Lemma. The algorithm works directly on the modal formulas without translating them into a first-order logic and without using Skolemization. SQEMA succeeds not only on all Sahlqvist formulas, but also on the extended class of *inductive formulas* introduced in [9]. There are examples of modal formulas on which SQEMA succeeds, while both SCAN and DLS fail, e.g.: $(\Box(\Box p \leftrightarrow q) \to p)$.

As proved in [4] and [5], SQEMA only succeeds on d-persistent (and hence, by, e.g. [1], canonical) formulas, i.e., whenever successful, it not only computes a local first-order correspondent of the input modal formula, but also proves its canonicity and therefore the canonical completeness of the modal logic axiomatized with that formula. This accordingly extends to any set of modal formulas on which SQEMA succeeds. Thus, SQEMA can also be used as an automated prover of canonical model completeness of modal logics.

An implementation of SQEMA in Java was given in [7]. Some additional simplifications were added to the implementation thanks to a suggestion by Renate Schmidt, which helps the implementation succeed on formulas such as $((\Box\Diamond p \to \Diamond\Box p) \vee (\Box p \to \Diamond p))$.

In this paper, we augment the SQEMA algorithm to $\mathrm{ML}(\Box, [U])$, the basic modal language extended by adding the universal modality. We have a proof that formulas on which the new SQEMA+U succeeds are strongly complete and that the resulting first-order formula is frame-correspondent of the input formula.

2 Premilinaries

In Section 1 we discussed [4] where the SQEMA algorithm is described in detail and its properties are proved. This article is an extension of the previous work in [4] and will rely on the definitions and proofs given there. We now introduce SQEMA for the language

ML($\Box$, $[U]$), or the basic modal language extended by adding the universal modality, and we will refer to this variant of SQEMA as *SQEMA+U*.

We take a fixed countable set of *propositional variables* PROP. Formulas of ML($\Box$, $[U]$) are defined inductively as $\phi = p|\top|\bot|\neg\psi_1|(\psi_1 \vee \psi_2)|(\psi_1 \wedge \psi_2)|(\psi_1 \to \psi_2)|\Diamond\psi_1|\Box\psi_1|\langle U\rangle\psi_1$ $|[U]\psi_1$, where $p \in$ PROP, ψ_1 and ψ_2 are already defined formulas. A Kripke frame is a tuple $\mathcal{F} = \langle W, R\rangle$, where W is a non-empty set and $R \subseteq W \times W$ is the *accessibility relation*. A *valuation* V is a mapping from the propositional variables to subsets of W, and a *model* $\mathcal{M}$ is a tuple $\langle \mathcal{F}, V\rangle$ of a Kripke frame and a valuation on it. The *extension* of ϕ in $\mathcal{M}$, $[\![\phi]\!]_{\mathcal{M}}$, is: $[\![p]\!]_{\mathcal{M}} = V(p)$, $[\![\top]\!]_{\mathcal{M}} = W$, $[\![\bot]\!]_{\mathcal{M}} = \emptyset$, $[\![\neg\psi_1]\!]_{\mathcal{M}} = W \setminus [\![\psi_1]\!]_{\mathcal{M}}$, $[\![(\psi_1 \vee \psi_2)]\!]_{\mathcal{M}} = [\![\psi_1]\!]_{\mathcal{M}} \cup [\![\psi_2]\!]_{\mathcal{M}}$, $[\![(\psi_1 \wedge \psi_2)]\!]_{\mathcal{M}} = [\![\psi_1]\!]_{\mathcal{M}} \cap [\![\psi_2]\!]_{\mathcal{M}}$, $[\![(\psi_1 \to \psi_2)]\!]_{\mathcal{M}} = (W \setminus [\![\psi_1]\!]_{\mathcal{M}}) \cup [\![\psi_2]\!]_{\mathcal{M}}$, $[\![\Box\psi_1]\!]_{\mathcal{M}} = \{w \in W|\ (\forall y \in W)(wRy \Rightarrow y \in [\![\psi_1]\!]_{\mathcal{M}})\}$, $[\![\Diamond\psi_1]\!]_{\mathcal{M}} = \{w \in W|\ (\exists y \in W)(wRy\ \&\ y \in [\![\psi_1]\!]_{\mathcal{M}})\}$, $[\![[U]\psi_1]\!]_{\mathcal{M}} = \{w \in W|\ (\forall y \in W)(y \in [\![\psi_1]\!]_{\mathcal{M}})\}$, $[\![\langle U\rangle\psi_1]\!]_{\mathcal{M}} = \{w \in W|\ (\exists y \in W)(y \in [\![\psi_1]\!]_{\mathcal{M}})\}$. If $w \in [\![\phi]\!]_{\mathcal{M}}$ for some $w \in W$ and all models $\mathcal{M}$ over a Kripke frame $\mathcal{F} = \langle W, R\rangle$, then ϕ is *valid* in $\mathcal{F}, w$; $\mathcal{F}, w \Vdash \phi$. Two formulas are *locally equivalent* ($\phi \equiv \psi$), iff, in any model, their extensions are equal to one another.

As in [4], we extend ML($\Box$, $[U]$) to ML$^+$($\Box$, $[U]$), adding *nominals* and *reversed modalities* ($\Box^{-1}$, $\Diamond^{-1}$). Nominals are a new countable set of symbols, evaluated with singletons. Reversed modalities use R^{-1} instead of R for a given Kripke frame.

A *pure formula* is a formula which does not contain propositional variables but may contain nominals. A formula, which does not contain implications and where negations only occur immediately before propositional variables, is a formula in *negation normal form*. A formula, which is an implication whose right-hand side and left-hand side are both in negation normal form, is called an *equation* due to an allusion to the Gaussian elimination method. *Syntactically closed* is a formula where all occurrences of nominals and $\Diamond^{-1}$are positive, and all occurrences of $\Box^{-1}$are negative. *Syntactically open* is a formula where all occurrences of $\Diamond^{-1}$and nominals are negative, and all occurrences of $\Box^{-1}$are positive.

We consider a first-order language with formal equality, a binary predicate symbol R, and no other predicate symbols, no functional symbols and no constants. We denote this language as L_0 and we call L_0 formulas *FOL* formulas. A Kripke frame $\mathcal{F} = \langle W, R\rangle$ is also a structure for L_0, interpreting the symbol R with the relation R. We inductively define the *standard translation* (ST) for pure formulas into first-order formulas of L_0, where the resulting FOL formula will always have $n + 1$ free variables, where n is the number of distinct nominals in the pure formula. In this ST, we always allocate a countably infinite subset YVAR $= \{y_1, \dots\}$ of the FOL individual variables VAR and designate them to each of the nominals of ML$^+$($\Box$, $[U]$), so our translation is unambiguous. A modal formula ϕ and an L_0 formula ψ with at most one free variable x are *locally correspondent* iff for every Kripke frame $\mathcal{F} = \langle W, R\rangle$, and for every $w \in W$: $\mathcal{F}, w \Vdash \phi$ iff $\mathcal{F} \vDash \psi_x[w]$.

3 The Algorithm SQEMA+U

Following [4], we define SQEMA+U in the following way:
INPUT: $\phi \in$ ML($\Box$, $[U]$)
OUTPUT: $\langle$*success*, *fol*$(\phi)\rangle$ or $\langle$*failure*$\rangle$

STEP 1: Negate ϕ and rewrite $\neg\phi$ in negation normal form. Then, distribute the diamonds $\Diamond$ and $\langle U\rangle$ and conjunctions over disjunctions as much as possible, using the local equivalences:
Rule 1.1: $\Diamond_0(\gamma_1 \vee \gamma_2) \equiv (\Diamond_0\gamma_1 \vee \Diamond_0\gamma_2)$ for $\Diamond_0$ among $\Diamond$ and $\langle U\rangle$
Rule 1.2: $(\gamma_1 \vee \gamma_2) \wedge \gamma_3 \equiv (\gamma_1 \wedge \gamma_3) \vee (\gamma_2 \wedge \gamma_3)$

Thus, obtain $\neg\phi \equiv \bigvee_k \psi_k$ where no further applications of rules 1.1 or 1.2 are possible on any ψ_k. The algorithm now reserves the nominal $i_0 \in$ NOM, it does not occur in any ψ_k (which is trivially true because in our case, $\psi_k \in$ ML($\Box$, $[U]$)) and it will be used throughout the steps. The algorithm now proceeds with STEP 2, applied separately on each of the subformulas ψ_k, and if it succeeds for all ψ_k, it will proceed to STEP 6. Otherwise, if even one of the branches for a single k fails, the algorithm returns $\langle failure \rangle$ as output and stops.

STEP 2: Let ψ be one of the disjuncts from STEP 1. Now, consider the equation $i_0 \to \psi$, where $i_0 \in$ NOM is the reserved in STEP 1 nominal. Start solving a system of equations that only contains $(i_0 \to \psi)$ by proceeding to STEP 3.

STEP 3: From the current system, eliminate every propositional variable that occurs only positively or negatively throughout the system, by replacing it with $\top$ or $\bot$, respectively.

STEP 4: Non-deterministically pick an elimination order for the remaining propositional variables in the current system. Try eliminating each variable in order, by proceeding to STEP 5. If any elimination order succeeds, and thus, all propositional variables have been eliminated from the current system, proceed to STEP 6. If all elimination orders fail, report failure for the current system and return to STEP 1.

STEP 5: Take the propositional variable p that has to be eliminated as input from STEP 4. Apply rules for converting the current system, as to eliminate any occurrences of p. The rules to use will be listed below. If p has been eliminated, report success and return the current system to STEP 4 to try eliminating the remaining variables. If elimination fails, backtrack, as will be described in detail below, and try STEP 5 again. If all attempts fail, report failure to eliminate p and resume executing STEP 4.

STEP 6: If this step is reached by all branches of the execution, then all propositional variables have been eliminated from all systems resulting from the negation of the input formula. In each system, take the conjunction of all equations, to obtain a pure formula *pure*. Then, form the formula $(\forall y_1 \ldots \forall y_n \exists x_0)\mathrm{ST}(\neg pure, x_0)$, where $y_1, \ldots, y_n$ are all occurring variables corresponding to the nominals in *pure*, except for y_0, corresponding to the designated current state nominal i_0. y_0 is left free because we are computing the local FOL correspondent. Then take the conjunction of all FOL correspondents to form $fol(\phi)$. Return $\langle success, fol(\phi)\rangle$.

Now, we will discuss the rules that the algorithm will use in STEP 5 above.

Transformation Rules: This is the detailed description of STEP 5's rules. Denote the propositional variable that needs to be eliminated with p. Apply rules for converting the current system, so that in the end we get a current system with only three classes of formulas: 1. alphas, 2. betas, and 3. thetas for p. Alphas will be of the form $\alpha_j \to p$ for $1 \leq j \leq n$, betas will be of the form $\beta_k(p)$ for $1 \leq k \leq m$, and thetas will be of the form θ_l for $1 \leq l \leq q$, such that:
1. p does not occur in any α_j part of any alpha;
2. all betas are negative in p;
3. there are no occurrences of p in any theta.
If the current system is successfully split into three types of formulas as described above, then the current system is replaced with a new current system $\{\beta_1(p/(\alpha_1 \vee \cdots \vee \alpha_n)), \ldots, \beta_m(p/(\alpha_1 \vee \cdots \vee \alpha_n)), \theta_1, \ldots, \theta_q\}$, thus p is eliminated. Note that, if n or m were 0, then rule 8, described below, or STEP 3 would have eliminated p, so we assume non-zero n and m. We call this elimination rule the Ackermann rule. If one of the newly obtained formulas above, ψ, is not in equation form, rewrite it to negation normal form as ψ', then replace it with $\top \to \psi'$.

The rules that the algorithm can use to obtain a split system as described above, are:

1. $\wedge$-rule. Replace $\beta \to (\gamma \wedge \delta)$ with $\beta \to \gamma$, $\beta \to \delta$.

2. Left-shift $\vee$-rule. Replace $\beta \to (\gamma \vee \delta)$ with $(\beta \wedge \neg\gamma) \to \delta$. This includes a non-deterministic choice between γ and δ, and backtracking if the first choice fails.

3. Right-shift $\vee$-rule. Replace $(\beta \wedge \neg\gamma) \to \delta$ with $\beta \to (\gamma \vee \delta)$.

4. Left-shift $\Box$-rule. For any $\Box_0 \in \{\Box, [U], \Box^{-1}\}$, replace $\gamma \to \Box_0\delta$ with $\Diamond_0^{-1}\gamma \to \delta$, where $\Diamond_0^{-1}$ is the corresponding to $\Box_0$ reversed diamond.

5. Right-shift $\Box$-rule. For any $\Diamond_0^{-1} \in \{\Diamond^{-1}, \langle U\rangle, \Diamond\}$, replace $\Diamond_0^{-1}\gamma \to \delta$ with $\gamma \to \Box_0\delta$, where $\Box_0$ is the corresponding to $\Diamond_0^{-1}$ un-reversed box.

6. $\Diamond$-rule. For any $\Diamond_0 \in \{\Diamond, \langle U\rangle, \Diamond^{-1}\}$, replace $j \to \Diamond_0\gamma$ with $j{\to}\Diamond_0 k$, $k \to \gamma$, where k is a new nominal that hasn't been used anywhere in any system in any branch of this run of SQEMA+U.

7. Reasoning rules. Freely replace any equation with a locally equivalent equation, if:

7.1. The result is an equation and is locally equivalent to the replaced one.

7.2. If before the replacement, the left-hand side of the equation was syntactically closed and the right-hand side was syntactically open, the same holds for the replacement.

Example rules of type 7:

— Commutativity and associativity of $\wedge$ and $\vee$

— Convert a formula into negation normal form

— Replace $\gamma \vee \neg\gamma$ with $\top$, and $\gamma \wedge \neg\gamma$ with $\bot$

— Replace $\gamma \vee \top$ with $\top$, and $\gamma \vee \bot$ with γ

— Replace $\gamma \wedge \top$ with γ, and $\gamma \wedge \bot$ with $\bot$

— Replace $\gamma \to \bot$ with $\neg\gamma$ and $\gamma \to \top$ with $\top$

— Replace $\bot \to \gamma$ with $\top$ and $\top \to \gamma$ with γ and vice versa, when it is needed to obtain an equation after a variable elimination

— Replace $\neg\Diamond_0\neg\gamma$ with $\Box_0\gamma$ and $\neg\Box_0\neg\gamma$ with $\Diamond_0\gamma$ for corresponding modalities.

Rules that affect $\langle U\rangle$ and $[U]$ are listed in the table below. For $j \in \{1,2\}$, we abbreviate U_j for either $[U]$ or $\langle U\rangle$, we abbreviate $\overset{\wedge}{\vee}$ for either $\vee$ or $\wedge$, we abbreviate $\Box^\varepsilon$ for either $\Box$ or $\Box^{-1}$, and we abbreviate $\Diamond^\varepsilon$ for either $\Diamond$ or $\Diamond^{-1}$.

Replace	with	Replace	with
$(i_1 \to \langle U\rangle i_2)$, $i_1, i_2 \in$ NOM	$\top$	$\langle U\rangle i$, where $i \in$ NOM	$\top$
$U_1U_2\gamma$	$U_2\gamma$	$\langle U\rangle\gamma_1 \vee \gamma_2$, for $\gamma_2 \equiv \neg\gamma_1$	$\top$
$\Box^\varepsilon U_1\gamma$	$U_1\gamma \vee \Box^\varepsilon\bot$	$\langle U\rangle\gamma \vee \Diamond^\varepsilon\gamma$	$\langle U\rangle\gamma$
$[U](U_1\gamma_1 \overset{\wedge}{\vee} U_2\gamma_2)$	$U_1\gamma_1 \overset{\wedge}{\vee} U_2\gamma_2$	$[U]\gamma \vee \Box^\varepsilon\gamma$	$\Box^\varepsilon\gamma$
$[U](U_1\gamma_1 \overset{\wedge}{\vee} \gamma_2)$	$U_1\gamma_1 \overset{\wedge}{\vee} [U]\gamma_2$	$[U]\gamma_1 \wedge \gamma_2$, for $\gamma_2 \equiv \neg\gamma_1$	$\bot$
$[U]\neg i$, where $i \in$ NOM	$\bot$	$\langle U\rangle\gamma \wedge \Diamond^\varepsilon\gamma$	$\Diamond^\varepsilon\gamma$
$\Diamond^\varepsilon U_1\gamma$	$U_1\gamma \wedge \Diamond^\varepsilon\top$	$[U]\gamma \wedge \Box^\varepsilon\gamma$	$[U]\gamma$
$\langle U\rangle(U_1\gamma_1 \wedge U_2\gamma_2)$	$U_1\gamma_1 \wedge U_2\gamma_2$	$[U]\gamma \wedge \gamma$	$[U]\gamma$
$\langle U\rangle(U_1\gamma_1 \wedge \gamma_2)$	$U_1\gamma_1 \wedge \langle U\rangle\gamma_2$	$\langle U\rangle\gamma \wedge \gamma$	γ

8. Positive and negative variables rule: As in STEP 3, replace any variable that occurs only positively in the system with $\top$, and replace any variable that occurs only negatively in the system with $\bot$.

If STEP 5 fails to eliminate p, backtrack, try the other choice for any Left-shift $\vee$-rule that is still pending. If this fails, backtrack, change the polarity of p by substituting $\neg p$ for p in the starting system, convert all formulas to equations as described above, and try STEP 5 again. If this also fails, report failure to eliminate p and resume STEP 4. If p has been eliminated, report success and return the current system to STEP 4 to try eliminating the remaining variables.

Theorem 1. *Let ϕ be an* ML$(\Box, [U])$ *formula and let SQEMA+U return $\langle success, fol(\phi)\rangle$ on ϕ. Then, the following hold: (1) ϕ and $fol(\phi)$ are locally correspondent; (2) The normal modal logic $K_U + \phi$ is strongly complete with respect to the class of frames validating $fol(\phi)$.*

Proof. We use a special case of descriptive general frames and d-persistence for ML$(\Box, [U])$ similarly to [1]. The proof of (1) and that ϕ is d-persistent is analogous to the proof given in [4]. The axioms and completeness of K_U are given in [8]. Similarly to [1], with some additional work because the canonical relation for $[U]$ is an S_5 relation, we prove that there is a descriptive general canonical sub-frame for $K_U + \phi$ and any MCS, thus proving (2).

4 Implementation Notes

The implementation of SQEMA+U is based on the existing servlet-based SQEMA implementation from 2006. It is written in Java. The program runs deterministically and always terminates. SQEMA+U is compiled with the Google GWT compiler to Javascript, runs directly in the browser, and displays the result if SQEMA+U succeeds. The user can see the proof of the result if the run was successful, and can also see the full log of the actions taken by the program. The website URL is `http://www.fmi.uni-sofia.bg/fmi/logic/sqema`. SQEMA+U is vNext, while SQEMA is 0.9.2.2. A complete version of the proofs for SQEMA+U for ML($\Box, [U]$) can be obtained from the author in PDF format.

4.1 Future Work

— Prove that SQEMA+U succeeds on all ML($\Box, [U]$) Sahlqvist formulas, similarly to [4].
— Extend SQEMA+U to ML$^+$($\Box, [U]$) inputs (with proofs).
— Implement the polyadic version of SQEMA.

Acknowledgemens

Thanks to Dimiter Vakarelov for his guidance. Thanks to the anonymous referee for the suggestions. This paper is partially supported by the Science Fund of Sofia University, contract 5/2015, and the Bulgarian Science Fund, programme Rila 2014, contract DRILA01/2.

References

[1] Blackburn, P., de Rijke, M., Venema, Y.: *Modal Logic (Cambridge Tracts in Theoretical Computer Science)*, Cambridge University Press, 2002.

[2] Chagrova, L.: *An undecidable problem in correspondance theory.* The Journal of Symbolic Logic **56** (1991) 1261–1272.

[3] Conradie, W., Goranko, V., Vakarelov, D.: *Elementary Canonical Formulae: a survey on syntactic, algorithmic, and model-theoretic aspects.* In: Advances in Modal Logic, **5**, Kings College London Publ., 2005, 17–51.

[4] Conradie, W., Goranko, V., Vakarelov, D.: *Algorithmic correspondence and completeness in modal logic. I. The core algorithm SQEMA*. Logical Methods in Computer Science, **2** (1:5) 2006, 1–26.

[5] Conradie, W., Goranko, V., Vakarelov, D.: *Algorithmic correspondence and completeness in modal logic II. Polyadic and hybrid extensions of the algorithm SQEMA*. J Logic Computation (2006) 16 (5): 579–612.

[6] Gabbay, D., Ohlbach, H. J.: *Quantifier Elimination in Second-Order Predicate Logic.* South African Computer Journal, **7**, (1992), 35–43.

[7] Georgiev, D.: *An implementation of the algorithm SQEMA for computing first-order correspondences of modal formulas.* Master thesis. Sofia University, FMI, 2006, in Bulgarian. http://www.fmi.uni-sofia.bg/fmi/logic/theses/Dimiter_Georgiev.pdf

[8] Goranko, V., Passy S.: *Using the Universal Modality: Gains and Questions.* In: J Logic Computation (1992) 2 (1): 5–30

[9] Goranko, V., Vakarelov, D.: *Elementary Canonical Formulae: Extending Sahlqvist Theorem.* Annals of Pure and Applied Logics, 2006, **141**, 1-2, pp. 180–217.

[10] Szalas, A.: *On the correspondence between Modal and Classical Logic: an Automated Approach.* J. Logic and Comp. Vol 3 No 6 (1993), 605–620.