

# Idempotent Transductions for Modal Logics

Tim French

School of Computer Science & Software Engineering  
The University of Western Australia  
tim@csse.uwa.edu.au

**Abstract.** We investigate the extension of modal logics by bisimulation quantifiers and present a class of modal logics which is decidable when augmented with bisimulation quantifiers. These logics are referred to as the idempotent transduction logics and are defined using the programs of propositional dynamic logic including converse and tests. This is a nontrivial extension of the decidability of the positive idempotent transduction logics which do not use converse operators in the programs (French, 2006). This extension allows us to apply bisimulation quantifiers to, for example, logics of knowledge, logics of belief and tense logics. We show the idempotent transduction logics preserve the axioms of propositional quantification and are decidable. The definition of idempotent transduction logics allows us to apply these results to a number of combined modal logics with a variety of interactions between modalities.

## 1 Introduction

Bisimulation quantifiers were introduced by Visser [1] and Ghilardi [2] as a semantic characterization of uniform interpolants in modal logics such as **K**, **Grz** and **GL**. More recently, bisimulation quantifiers have been investigated as an expressive extension of many modal logics such as **PDL** [3], and **S4** [4].

Syntactically, bisimulation quantifiers are the same as propositional quantifiers. Semantically, they quantify over all interpretations of the specified propositional atom in *all bisimilar models*. Thus bisimulation quantified modal logics are typically less expressive than standard propositional quantified modal logics (which are generally undecidable [5]).

The motivation for considering bisimulation quantifiers is to extend modal logics with the ability to reason about *hypothetical assignments of atoms*. That is we may express the property, “there could be an interpretation of atoms that agrees with the present interpretation on all atoms except  $x$ , that makes  $\alpha$  true.” To evaluate this statement we must decide what it means for two interpretations to *agree*. In our case we suppose two interpretations agree if they are structures that are bisimilar up to the proposition  $x$ . This is a very natural standard to apply. Bisimulations were defined by Milner [6], Park [7] and (as  $p$ -relations) van Benthem [8]. Stirling [9] surveys the relationships between bisimulations and modal logic. In [8] it is shown that properties definable in modal logic are exactly the first-order properties that are bisimulation invariant. Therefore modal formulas are unable to distinguish between bisimilar structures, and this degree of abstraction has allowed modal logics to be easily applied in a range of contexts (such as reasoning about time [10] or knowledge [11]). We consider quantification

modulo bisimulation because we would like to preserve this degree of abstraction whilst considering hypothetical assignments of atoms.

Bisimulation quantifiers were originally investigated in [1] and [2] as abbreviations for uniform interpolants in some modal logics. In [3] this characterization was extended to abbreviations for uniform interpolants in the modal  $\mu$ -calculus. More recently, bisimulation quantifiers have been investigated independently of uniform interpolants. We briefly survey some of these results here: In [4] axiomatizations are given for the bisimulation quantified extension of **K** and the bisimulation quantified extension of **PDL**. In [12] bisimulation quantifiers are considered in the context of arbitrary modal logics. It is shown that there are cases where extension by bisimulation quantifiers does not preserve decidability, or the axioms for propositional quantification. The paper [13] explores the relationship between fixed-point operators and bisimulation quantifiers, and it is shown that there are cases where bisimulation quantified logics are more expressive than the corresponding fixed-point logic. The paper [14] introduces the *positive idempotent transduction logics*. This is a decidable class of bisimulation quantified logics which includes many of the well-known and applied modal logics (such as **K4**, **GL** and a fragment of **CTL**).

In this paper we enhance the results of [14]. We extend the definition of idempotent transduction logics in such a way as to capture the logics **S5**, **KD45** and many more besides<sup>1</sup>. The logic **S5** is applied for reasoning about knowledge and security [11] and **KD45** is used for reasoning about belief in agent systems, so there are practical reasons to extend the class of idempotent transduction logics to include such logics as these. We show that the idempotent transduction logics are decidable, and preserve the axioms of propositional quantification. This work is a non-trivial extension of the results of [14], which presented the class of positive idempotent transduction logics. Positive idempotent transductions specify modalities as  $\mu$ -programs without converse, whereas idempotent transduction specify modalities as  $\mu$ -programs with converse. While this significantly complicates the proofs, it allows us to extend our generalized results to combinations of logics including knowledge and belief.

These results have been presented in [15], although the proofs are improved here.

## 2 Preliminaries

In this section we give the basic notation, definitions and lemmas required throughout this paper.

### 2.1 Modal logic

**Definition 1.** Let  $\Lambda$  be a non-empty set. A  $\Lambda$ -frame is a tuple  $(S, R)$  where  $S$  is some set and  $R : \Lambda \longrightarrow \wp(S \times S)$ .

The set  $S$  corresponds to the set of possible worlds, and for each  $a \in \Lambda$ ,  $R(a)$  refers to some relation between the possible worlds. A model adds a valuation to a  $\Lambda$ -frame

---

<sup>1</sup> Using the language of [15] we will refer to the idempotent transduction logics of [14] as *positive idempotent transduction logics*

which defines which atomic propositions are true at each element of  $S$ , and identifies the *current world*.

**Definition 2.** A  $\Lambda$ - $\mathcal{V}$ -model is a tuple  $(S, R, \rho, s)$  where  $(S, R)$  is a  $\Lambda$ -frame,  $\rho : S \rightarrow \wp(\mathcal{V})$ , and  $s \in S$ .

Here  $\mathcal{V}$  is a countable set of atomic propositions,  $\rho$  assigns those propositions to states (so that  $x \in \rho(t)$  if and only if  $x$  is true at  $t$ ), and  $s$  is the *current state of the model*. We will refer to  $\Lambda$ - $\mathcal{V}$ -models as  $\Lambda$ -models when there is no ambiguity about  $\mathcal{V}$ . The use of the term *model* rather than *structure* or *system* is quite deliberate. A structure implies some concrete entity, whereas we consider the models to be abstractions of some real-world entity (but importantly, an abstraction that is equally as good as any other bisimilar model).

The syntax of the modal logic  $(L_\Lambda)$  is given with respect to the sets  $\Lambda$  (the set of modalities) and  $\mathcal{V}$  (the set of atomic propositions).

$$\alpha ::= x \mid \neg\alpha \mid \alpha_1 \vee \alpha_2 \mid \diamond_a \alpha \quad (1)$$

where  $x \in \mathcal{V}$  and  $a \in \Lambda$ . Correspondingly, we will refer to  $\Lambda$ - $\mathcal{V}$ -models as  $\Lambda$ -models when there is no ambiguity about  $\mathcal{V}$ . We let the abbreviations  $\wedge$ ,  $\rightarrow$ ,  $\leftrightarrow$ ,  $\top$  and  $\perp$  be defined as usual and let  $\square_a \alpha$  abbreviate  $\neg \diamond_a \neg \alpha$ . We let  $var(\alpha)$  be the set of propositional atoms appearing in  $\alpha$ .

The modal logic  $L_\Lambda$  has the following semantics. Given some  $\Lambda$ -model  $M = (S, R, \rho, s)$  for each  $t \in S$  let  $M_t = (S, R, \rho, t)$ . We then define:

$$\begin{aligned} M \models x &\iff x \in \rho(s) \\ M \models \diamond_a \alpha &\iff \text{for some } (s, t) \in R(a), M_t \models \alpha \end{aligned}$$

where  $\neg$  and  $\vee$  have their usual interpretation.

From a purely semantic view of logic, we can define a variety of modal logics by restricting the class of models over which formulas are evaluated.

**Definition 3.** Given some set  $\Lambda$ , a  $\Lambda$ -class  $\mathcal{C}$  is a class of  $\Lambda$ -frames. The logic  $L_{\mathcal{C}}$  is the set of all  $L_\Lambda$  formulas which are true on all  $\Lambda$ - $\mathcal{V}$ -models,  $(S, R, \rho, s)$ , where  $(S, R) \in \mathcal{C}$ .

In an abuse of notation we may refer to  $M \in \mathcal{C}$ , where  $M = (S, R, \rho, s)$  and  $(S, R) \in \mathcal{C}$ .

## 2.2 Bisimulation Quantifiers

The *bisimulation quantified logic of  $\mathcal{C}$*  ( $BQL_{\mathcal{C}}$ ) is the extension of  $L_{\mathcal{C}}$  by bisimulation quantifiers. The syntax is given by:

$$\alpha ::= x \mid \neg\alpha \mid \alpha_1 \vee \alpha_2 \mid \diamond_a \alpha \mid \exists x \alpha \quad (2)$$

where  $x \in \mathcal{V}$  and  $a \in \Lambda$ . To give the formal semantics we require bisimulations. These are well-known and we extend this definition to  $\Theta$ -bisimulations.

**Definition 4.** Let  $M = (S, R, \rho, s_0)$  and  $N = (T, P, \lambda, t_0)$  be  $\Lambda$ - $\mathcal{V}$ -models and suppose  $\Theta \subseteq \mathcal{V}$ . We say the models  $M$  and  $N$  are  $\Theta$ -bisimilar (written  $M \cong_{\Theta} N$ ) if there is some relation  $B \subseteq S \times T$  such that:

1.  $(s_0, t_0) \in B$ ;
2. for all  $(s, t) \in B$ ,  $\rho(s) \setminus \Theta = \lambda(t) \setminus \Theta$ ;<sup>2</sup>
3. for all  $(s, t) \in B$ , for all  $a \in \Lambda$ , for all  $u \in S$  such that  $(s, u) \in R(a)$  there is some  $v \in T$  such that  $(u, v) \in B$  and  $(t, v) \in P(a)$ ;
4. for all  $(s, t) \in B$ , for all  $a \in \Lambda$ , for all  $v \in T$  such that  $(t, v) \in P(a)$  there is some  $u \in S$  such that  $(u, v) \in B$  and  $(s, u) \in R(a)$ .

We call such a relation  $B$  a  $\Theta$ -bisimulation from  $M$  to  $N$  and if  $M \cong_{\{x\}} N$  we say  $M$  and  $N$  are  $x$ -bisimilar (written  $M \cong_x N$ ).

Given  $M = (S, R, \rho, s)$  is a  $\mathcal{C}$ -model the interpretation of  $\exists x \alpha$  in  $\text{BQL}_{\mathcal{C}}$  is  $M \models_{\mathcal{C}} \exists x \alpha$  if and only if there is some  $\mathcal{C}$ -model  $N$  where  $N \cong_x M$  and  $N \models_{\mathcal{C}} \alpha$ .

We note that the class  $\mathcal{C}$  appears explicitly in the semantics, and thus has a direct effect on the meaning of formulas. For example, we might have some  $\Lambda$ -model  $M$  and some formula  $\alpha$  such that for two classes,  $\mathcal{C}$  and  $\mathcal{D}$   $M \models_{\mathcal{C}} \alpha$  and  $M \not\models_{\mathcal{D}} \alpha$ . Consider, for example, the formula  $\alpha = \forall x(x \rightarrow \diamond_a x)$ , and suppose  $M$  is a  $\{a\}$ -model consisting of a single world related to itself by  $R(a)$ .

- In the class  $\mathcal{C}$  of all reflexive  $\{a\}$ -frames we would have  $M \models_{\mathcal{C}} \alpha$ .
- However, in the class of all  $\{a\}$ -frames,  $\mathcal{D}$ , we have  $M \not\models_{\mathcal{D}} \alpha$ , since  $M$  is  $\{x\}$ -bisimilar to some model  $(T, P, \tau, t)$  where  $x \in \tau(t)$  and  $(t, t) \notin P(a)$ .

This is in contrast to the case of pure modal logic, where there is one semantics for all logics, so that  $L_{\mathcal{C}}$  and  $L_{\mathcal{D}}$  will always agree on the meaning of a formula (i.e. whether a given formula is satisfied by a given model), but they may differ on whether a formula is valid.

In the remainder of this paper we will let  $\text{BQL}_{\Lambda}$  refer to the logic  $\text{BQL}_{\mathcal{C}}$  where  $\mathcal{C}$  is the class of all  $\Lambda$ -frames, and write  $M \models_{\Lambda} \alpha$  to denote satisfaction in this class.

### 2.3 Amalgamation

It was shown in [12] that bisimulation quantifiers are not well behaved for all classes of frames. Particularly we can define a class of frames  $\mathcal{C}$  such that  $\text{BQL}_{\mathcal{C}}$  does not validate the standard rules of propositional quantification:

1. *Existential elimination:* If  $\phi \rightarrow \psi$  is a validity and  $\psi$  does not contain free occurrences of the variable  $x$ , then  $\exists x \phi \rightarrow \psi$  should also be a validity.
2. *Existential introduction:* Suppose  $\alpha$  is a formula such that  $\beta$  is free for  $x$  in  $\alpha$ . Then  $\alpha[x \setminus \beta] \rightarrow \exists x \alpha$  is a validity.

Here  $\alpha[x \setminus \beta]$  is the formula  $\alpha$  with every free occurrence of the variable  $x$  replaced by the formula  $\beta$ , and  $\beta$  is *free for  $x$  in  $\alpha$*  if and only if for every free variable,  $y$ , of  $\beta$  the

<sup>2</sup> Here,  $\setminus$  is used to indicate set subtraction.

variable  $x$  is not in the scope of a quantifier,  $\exists y$ , in  $\alpha$ . In [15], it is shown that these laws, along with the tautologies of propositional logic, are sufficient to give a sound and complete axiomatization of the extension of propositional logic by propositional quantification.

Several classes of frames that do not validate these rules are presented in [12]. These rules are sound for all logics,  $\text{BQL}_{\mathcal{C}}$ , where  $\mathcal{C}$  has the *amalgamation property*.

**Definition 5.** We say the class of frames  $\mathcal{C}$  has the amalgamation property (or  $\mathcal{C}$  is amalgamative) if and only if for any  $\Theta_1, \Theta_2 \subset \mathcal{V}$ , for any  $\mathcal{C}$ -models  $M$  and  $N$  such that  $M \cong_{\Theta_1 \cup \Theta_2} N$ , there is some  $\mathcal{C}$ -model,  $K$  such that  $M \cong_{\Theta_1} K$  and  $N \cong_{\Theta_2} K$ .

The following was shown in [12] (where amalgamation was referred to as safety).

**Lemma 1.** If  $\mathcal{C}$  enjoys amalgamation, then the rules existential elimination and existential introduction are sound for  $\text{BQL}_{\mathcal{C}}$ .

Note that the soundness of existential elimination rule implies bisimulation invariance, so as a corollary we have, if  $\mathcal{C}$  is amalgamative, then  $\text{BQL}_{\mathcal{C}}$  is bisimulation invariant.

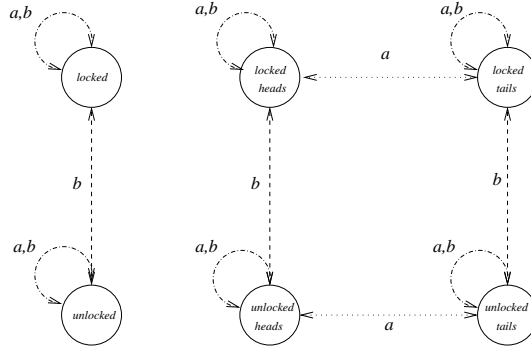
### 3 Motivation and examples

We will present some examples of bisimulation quantified modal logics, and then consider some of the applications for bisimulation quantifiers. The first two examples demonstrate the use of bisimulation quantifiers in the context of knowledge and belief. As the modalities of knowledge and belief are, respectively, symmetric and Euclidean [16], their interpretation relies on the converse relations. As such they cannot be presented using the positive idempotent transductions of [14].

#### 3.1 Knowledge

We consider a very simple example in the context of epistemic logic. Suppose that  $A = \{a, b\}$  (for *Alice* and *Bob*). We let  $\mathcal{C}$  be the  $A$ -class of frames  $(S, R)$  where both  $R(a)$  and  $R(b)$  are equivalence relations. This corresponds to a very simple logic of knowledge where two worlds  $u$  and  $v$  are related by  $R(a)$  if and only if Alice cannot distinguish between those two worlds, and thus  $\Box_a \alpha$  is true at the current world if and only if  $\alpha$  is true at every world that Alice cannot distinguish from the current world (i.e. Alice knows  $\alpha$ ). See [11] for a good introduction to epistemic logics.

Consider a simple model which consists of only two worlds, *locked* and *unlocked*, that Alice can distinguish, but which Bob cannot. That is, in all worlds the formula  $\text{locked} \leftrightarrow \Box_a \text{locked}$  is true. Even though Bob does not know whether the door is locked or not, he does know that Alice knows. Now suppose that (hypothetically) Bob announces he is going to flip a coin and keep the result secret. We can consider the hypothetical proposition *heads*, where  $\Box_b \text{heads} \wedge \neg \Box_a \text{heads}$  is true. As a bisimulation quantified formula this is expressed as  $\exists \text{heads}(\Box_b \text{heads} \wedge \neg \Box_a \text{heads})$  and its semantic interpretation is illustrated in Figure 1. The model on the right is  $\{\text{heads}\}$ -bisimilar to the one on the left, and satisfies  $\Box_b \text{heads} \wedge \neg \Box_a \text{heads}$  at its bottom left world.



**Fig. 1.** Two models demonstrating the interpretation of the formula  $\exists heads(\Box_b heads \wedge \neg\Box_a heads)$  in bisimulation quantified epistemic logic.

This is a very simple example, but we can see how bisimulation quantifiers can be applied in different contexts. In this example we supposed it was possible for Bob to be able to flip a coin and keep the result secret from Alice. However, if we restricted the class  $\mathcal{C}$  so that the equivalence classes of  $R(a)$  were always subsets of equivalence classes that belong to  $R(b)$  we would have a semantic where Alice *always* knows more than Bob (for example Alice might be a super-user who has access to all of Bob's files). In this case  $\exists heads(\Box_b heads \wedge \neg\Box_a heads)$  would not be satisfied. Such hierarchical logics of knowledge (without bisimulation quantifiers) are considered in [17]. In [18] a bisimulation quantified logic of knowledge is presented, although several of the proofs are flawed. The results of this paper redress some of those errors.

### 3.2 Belief

Another interesting example of the application of bisimulation quantifiers is in the logic **KD45**, which is Kripke complete for the class of serial, transitive, Euclidean frames (a frame is Euclidean if any two successors of a world are related to each other). In  $BQL_{KD45}$ ,

$$\Diamond(\exists x(x \wedge \Box\neg x)) \quad (3)$$

is a validity, but  $\exists x\Diamond(x \wedge \Box\neg x)$  is a contradiction. The unsatisfiability of the second follows from the fact that  $\Box(x \rightarrow \Diamond x)$  is a validity of **KD45**. Figure 2 demonstrates how formula (3) is satisfied. Here,  $M$  is a **KD45** model, and  $M_t \cong_x N_{t'}$ , but there is no **KD45**-model  $K = (U, Q, \gamma, u)$  where  $K \cong_x M$  and  $K_v \cong N_{v'}$  for some  $v$  where  $(u, v) \in Q$ . That is, the bisimulation quantifier does not necessarily commute with modalities, so bisimulation quantified **KD45** does not satisfy the Barcan formula,  $\forall x\Box\alpha \rightarrow \Box\forall x\alpha$ . This may appear to be a flaw in the interpretation of bisimulation quantifiers. However, **KD45** is the logic of belief and in this context it makes sense. An agent *believes* that some set of worlds are possible, and refuses to believe any other world is possible. Suppose that we are dealing with an informed agent that knows the difference between belief and knowledge. The agent would *believe* that in principle it

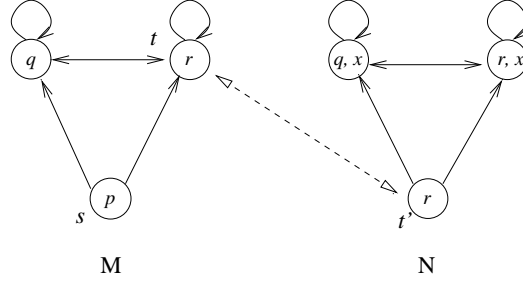


Fig. 2. A depiction of the validity (3) with respect to the class of **KD45** frames.

is *possible* that he believes some (hypothetical) property to be true, even though that property is not satisfied at the current world. That is,  $\Diamond \exists x (\neg x \wedge \Box x)$  is valid. However the agent would still not concede that any of the real, non-hypothetical properties (i.e. unquantified propositions) that he believes are false.

### 3.3 Expressing refinement

We consider the applications of modal logic to the verification and modeling of computational systems. Such systems can be specified at many levels of abstraction, ranging from the individual transistors that make up a circuit, to objects of high level programming languages.

Abadi and Lamport [19] examine the problem of specifying and reasoning about systems at multiple levels of abstraction. They divide the properties at any level of abstraction into external properties and internal properties, where the external properties describe the visible behavior of a system. At a higher level of abstraction less properties are externally visible. Abadi and Lamport use finite state machines to represent the behavior of systems, and apply the notion of a *refinement mapping* [20] to verify if one specification implements another specification (or agrees with the external properties of that specification).

In the context of general modal logics, such refinements can be modeled by bisimulation quantifiers. For example two security protocols could be specified by the formulas of epistemic logic,  $S_1(\bar{x}, \bar{y})$  and  $S_2(\bar{x}, \bar{z})$ , respectively. We are able to express that the specification  $S_1$  always agrees with the specification  $S_2$  with respect to the propositional atoms (or properties),  $\bar{x}$  using the bisimulation quantified epistemic formula

$$\forall \bar{x} \forall \bar{y} (S_1(\bar{x}, \bar{y}) \rightarrow \exists \bar{z} S_2(\bar{x}, \bar{z})). \quad (4)$$

We note that the use of bisimulation quantifiers rather than naive propositional quantifiers is significant here. For example, it is possible that the specification  $S_1(\bar{x}, \bar{y})$  is satisfiable in a model containing only three worlds, whilst  $S_2(\bar{x}, \bar{z})$ , can only be satisfied by models with more than three worlds. In such a case the formula (4) would not be valid.

We do not consider any particular specifications, or any specification languages here. Rather we simply note that bisimulation quantifiers seem to naturally capture the intuitive meaning of one specification *implementing another*. A general aim of this paper and other works [14, 15, 13] is to develop a formal framework for the iterative refinement of specifications in different ontologies.

## 4 The modal $\mu$ -calculus with converse

In this section we examine the *modal  $\mu$ -calculus with converse*[21]. The converse operators invert the modal accessibility relation allowing us to “see” backwards along a relation. Including converse programs is important to allow us to capture the symmetric nature of the modal relations in **S5** and other similar logics. However, the converse operators also significantly increase the complexity of the decidability proof.

### 4.1 Syntax and semantics

The *full  $\mu$ -calculus*,  $\mu L_A$  is an extension of modal logic that includes the *least fixed point operator*  $\mu$ , and converse modalities:

$$\alpha ::= x \mid \neg\alpha \mid \alpha \vee \alpha \mid \langle a \rangle \alpha \mid \langle a^- \rangle \alpha \mid \mu x \alpha \quad (5)$$

where  $x \in \mathcal{V}$ ,  $a \in A$  and in the recursion for  $\mu x \alpha$  we require that  $x$  only occurs in the scope of an even number of negations in  $\alpha$ . We let  $\nu x \alpha$  be an abbreviation for  $\neg \mu x \neg \alpha[x \setminus \neg x]$ , (where  $\alpha[x \setminus \neg x]$  is  $\alpha$  with every free occurrence of  $x$  replaced by  $\neg x$ ).

The interpretation of  $\langle a \rangle \alpha$  is the same as the interpretation of  $\diamond_a \alpha$  and the interpretation of  $\langle a^- \rangle \alpha$  is given as:

$$M_t \models \langle a^- \rangle \alpha \iff \text{for some } (s, t) \in R(a), M_s \models \alpha.$$

The interpretation of  $\mu x \alpha$  is the least of all assignments of  $x$  such that  $x$  is true exactly where  $\alpha$  is true, and the operator  $\nu$  refers to the greatest such assignment[22].

### 4.2 Results for the modal $\mu$ -calculus with converse

The modal  $\mu$ -calculus without converse was introduced by Kozen in [22]. In [23] it was shown that  $\mu L_A$  corresponds to the bisimulation invariant fragment of monadic second-order logic. Emerson and Jutla presented an exponential time decision procedure for the modal  $\mu$ -calculus in [24]. In [3] it was shown that the  $\mu$ -calculus was closed under bisimulation quantifiers. That is, for all formulas of the  $\mu$ -calculus  $\alpha$ , for all atomic propositions  $x$ , there is some formula of the  $\mu$ -calculus,  $\tilde{\exists}x\alpha$ , such that for all models,  $M$ ,  $M \models \tilde{\exists}x\alpha$  if and only if there is some model  $N \cong_x M$  such that  $N \models \alpha$ .

The  $\mu$ -calculus with converse (or the *full  $\mu$ -calculus*) was shown to be decidable in [21]. The proof of decidability defined *two-way alternating automata* to accept models of full  $\mu$ -calculus formulas, and then reduced the two-way alternating automata to (one-way) non-deterministic automata.

It is well-known that the  $\mu$ -calculus can express *dynamic modalities* (or *programs*).

**Definition 6.** Let  $\Lambda$  be a set of atomic programs. The  $\mu$ -programs  $\pi$  over  $\Lambda$  ( $Prog_\Lambda$ ) are:

$$\pi := a \mid \pi; \pi \mid \pi \cup \pi \mid \pi^* \mid \pi^- \mid \alpha?;$$

where  $a \in \Lambda$  and

$$\alpha := P \mid \neg\phi \mid \alpha \vee \alpha \mid \langle \pi \rangle \phi \mid \mu P \alpha,$$

provided  $P$  is under an even number of negations in  $\alpha$ .

The modalities,  $\langle \pi \rangle$ , corresponding to programs,  $\pi$  are defined as usual by extending the function  $R$  to map all programs,  $\pi$ , to a binary relation over  $S$ ,  $R(\pi)$ . This is done inductively where: the base case is given by  $R(a)$  for  $a \in \Lambda$ ;  $R(\pi_1; \pi_2)$  is the composition of the relations  $R(\pi_1)$  and  $R(\pi_2)$ ;  $R(\pi_1 \cup \pi_2)$  is the union of  $R(\pi_1)$  and  $R(\pi_2)$ ;  $R(\pi^*)$  is the reflexive transitive closure of  $R(\pi)$ ;  $R(\pi^-)$  is the converse of the relation  $R(\pi)$  (so that  $(u, v) \in R(\pi^-)$  if and only if  $(v, u) \in R(\pi)$ ); and  $\alpha?$  is a *test* program defined by  $R(\alpha?) = \{(u, u) \mid M_u \models_C \alpha\}$ .

We use the  $\mu$ -programs defined here to describe a large class of bisimulation quantified modal logics. Whilst these classes of logics may be defined using converse programs, the logics themselves only use forward modalities. To address this asymmetry we give the following definitions.

We first focus on translating the class of  $\mathcal{C}_\Lambda$ -models to a class of models where the converse of each atomic modality is explicitly represented by another atomic modality.

- Definition 7.** 1. Given the set  $\Lambda = \{a_1, a_2, \dots\}$ , let  $\Lambda'$  be a set  $\{b_1, b_2, \dots\}$  such that  $\Lambda \cap \Lambda' = \emptyset$ . The converse closure of  $\Lambda$ , with respect to  $\Lambda'$ , is the set  $\bar{\Lambda} = \{a_1, b_1, a_2, b_2, \dots\}$ , and for all  $i$ , we let  $\bar{a}_i = b_i$  and  $\bar{b}_i = a_i$ .
2. Given the  $\Lambda$ -class of frames  $\mathcal{C}$ , let the converse extension of  $\mathcal{C}$  be the  $\bar{\Lambda}$ -class  $\bar{\mathcal{C}} = \{(S, \bar{R}) \mid (S, R) \in \mathcal{C}\}$  where  $\bar{R} : \bar{\Lambda} \rightarrow S \times S$  is defined such that for all  $a \in \Lambda$ ,  $\bar{R}(a) = R(a)$  and  $(u, v) \in \bar{R}(\bar{a})$  if and only if  $(v, u) \in R(a)$ .
3. Given the  $\mathcal{C}$ -model  $M = (S, R, \rho, s)$  let the converse extension of  $M$  be the  $\bar{\mathcal{C}}$ -model  $\bar{M} = (S, \bar{R}, \rho, s)$ .
4. Given any  $\mathcal{C}_\Lambda$ -model  $M = (S, R, \rho, s)$  we define the converse closure of  $M$  to be the  $\bar{\mathcal{C}}_\Lambda$ -model  $M^+ = (S, R^+, \rho, s)$  where for  $o \in \bar{\Lambda}$ ,  $R^+(o) = R(o) \cup \{(u, v) \mid (v, u) \in R(\bar{o})\}$ .

The following definition introduces *tree-like* models. Tree-like models are useful since they require that every world in the model is reachable from the current world. This allows us to interpret converse modalities in terms of atomic modalities.

**Definition 8.** For any  $\Lambda$ , we say a  $\Lambda$ -model  $M = (S, R, \rho, s)$  is tree-like if

1. for all  $u \in S \setminus \{s\}$ , there is exactly one pair  $(o, v) \in \Lambda \times S$  such that  $(v, u) \in R(o)$ ; and
2. for all  $u \in S$ , for all  $o \in \Lambda$ ,  $(u, s) \notin R(o)$
3. for all  $u \in S \setminus \{s\}$ , there is some finite sequence  $u_0, u_1, \dots, u_n$  such that  $u_0 = s$ ,  $u_n = u$  and for all  $i < n$ , there is some  $o \in \Lambda$  such that  $(u_i, u_{i+1}) \in R(o)$ .

**Lemma 2.** For every  $\mathcal{C}_\Lambda$ -model  $M$ , there is some tree-like  $\mathcal{C}_\Lambda$ -model  $N$  such that  $M \cong N$ .

The proof is given in [15] (Lemma 5.89).

**Lemma 3.** *For every  $\overline{\mathcal{C}}_\Lambda$  model,  $M$ , for every tree-like  $\mathcal{C}_{\overline{\Lambda}}$  model  $N$  where  $N \cong_x M$ , we have  $N^+ \cong_x M$ .*

The proof is given in [15] (Lemma 5.93).

Using these definitions, we can describe a translation from the modal  $\mu$ -calculus with converse, to the modal  $\mu$ -calculus without converse.

**Theorem 1.** *There exists a translation  $(\cdot)'$  from  $\mu L_\Lambda$  with converse to  $\mu L_{\overline{\Lambda}}$  without converse, such that for all tree-like  $\overline{\Lambda}$ -models  $M$ , for all  $\mu L_\Lambda$  formulas  $\alpha$ ,  $M^+ \models \alpha$  if and only if  $M \models \alpha'$ .*

We sketch the proof of this theorem via alternating automata and  $\mu$ -automata here (see [25] for an overview of these formalisms). As the language of alternating automata is equivalent to the modal  $\mu$ -calculus we can define an alternating automaton,  $A$  (over the set of programs  $\{a, a^- \mid a \in \Lambda\}$ ) such that  $A$  accepts  $M^+$  if and only if  $M^+ \models \alpha$ . From  $A$ , we can define a  $\mu$ -automaton  $A^2$ , such that  $A^2$  accepts tree-like  $\overline{\Lambda}$ -model  $M$  if and only if  $A$  accepts  $M^+$ . This is based on Vardi's construction [21] and a complete description is given in [15], Appendix C.

Finally, as there is an effective translation from  $\mu$ -automata to the  $\mu$ -calculus [23], we can convert  $A^2$  to the  $\mu L_{\overline{\Lambda}}$  formula  $\alpha'$ , so that  $M^+ \models \alpha$  if and only if  $M \models \alpha'$ .

## 5 Idempotent transductions

Here we consider a class of decidable bisimulation quantified modal logics which includes the bisimulation quantified extensions of **S4**, **GL**, **S5** and **KD45**. We refer to this class of logics as the *idempotent transduction logics*. Aside from the well-known logics above there are many other idempotent transduction logics. These logics are defined by encoding modalities as  $\mu$ -programs, so potentially we can rapidly provide expressive and decidable logics for different ontologies “on the fly”.

Recall the definition of  $Prog_\Lambda$  (Definition 6).

**Definition 9.** *Given a finite set of modalities  $\Lambda$ , a  $\Lambda$ -transduction is a function  $\Pi : \Lambda \rightarrow Prog_\Lambda$ .*

We can apply  $\Lambda$ -transductions to frames as described in the following definition:

**Definition 10.** *Given a  $\Lambda$ -transduction,  $\Pi$ , and a  $\Lambda$ -frame  $F = (S, R)$  we define  $F^\Pi = (S, R^\pi)$  where for all  $u, v \in S$ ,  $(u, v) \in R^\pi(a)$  if and only if for all models  $M = (S, R, \rho, s)$ ,  $(u, v) \in R(\Pi(a))$ . We define the transduction class of  $\Pi$  (written  $\mathcal{C}_\Pi$ ) to be the class of  $\Lambda$ -frames  $F^\Pi$  where  $F$  is any  $\Lambda$ -frame.*

**Definition 11.** *A  $\Lambda$ -transduction  $\Pi$  is an idempotent transduction if for all  $a \in \Lambda$ ,  $\Pi(a)$  contains no free atomic propositions and for all  $\Lambda$ -models  $M$ , for all tree-like  $\Lambda$ -models  $N$  such that  $N \cong M^\Pi$  we have  $M^\Pi \cong N^\Pi$ . If  $\Pi$  is a idempotent transduction, then  $\mathcal{C}_\Pi$  is an idempotent transduction class of frames and the logic  $BQL_{\mathcal{C}_\Pi}$  is an idempotent transduction logic.*

Positive Idempotent transductions were identified for transductions that did not use the converse operator [14]. Positive idempotent transductions are truly idempotent (where idempotent means  $M^{\Pi} = (M^{\Pi})^{\Pi}$ ). To accommodate converse operators in the transductions the definition has been generalized.

### 5.1 Amalgamation for idempotent transduction logics

Amalgamation is an important property as it allows us to use the tautologies of propositional quantification in a logic. We can show that all idempotent transduction logics enjoy amalgamation.

**Lemma 4.** *Given any two tree-like  $\Lambda$ -models  $M$  and  $N$ , and some  $\Lambda$ -transduction  $\Pi$  such that for all  $a \in \Lambda$ ,  $\Pi(a)$  contains no free atomic propositions from  $\Theta$ , if  $M \cong_{\Theta} N$  then  $M^{\Pi} \cong_{\Theta} N^{\Pi}$*

The proof is given in [15] (Lemma 6.133).

Given any idempotent transduction,  $\Pi$ , we can show that  $\mathcal{C}_{\Pi}$  enjoys the amalgamation property (Definition 5).

**Lemma 5.** *Every idempotent transduction class is amalgamative.*

*Proof.* Suppose that  $\Pi$  is an idempotent  $\Lambda$ -transduction,  $M = (S, R, \rho, s)$  and  $N = (T, P, \tau, t)$  are  $\mathcal{C}_{\Pi}$ -models and  $M \cong_{\Theta \cup \Gamma} N$ . By Lemma 2 there is some tree-like  $\Lambda$ -model  $M' = (S', R', \rho', s')$  and some tree like  $\Lambda$ -model  $N' = (T', P', \tau', t')$  such that  $M' \cong M$  and  $N' \cong N$ . Therefore  $M' \cong_{\Theta \cup \Gamma} N'$  via some bisimulation  $B \subseteq S' \times T'$ . Let  $L = (B, Q, \eta, (s', t'))$  be a  $\Lambda$ -model where

1. for all  $a \in \Lambda$ ,  $((u, v), (w, z)) \in Q(a)$  if and only if  $(u, w) \in R(a)$  and  $(v, z) \in P(a)$ , and
2.  $\eta(u, v) = (\rho'(u) \setminus \Theta) \cup (\tau'(v) \setminus \Gamma)$ .

Clearly,  $L \cong_{\Theta} M'$  and  $L \cong_{\Gamma} N'$ . Again by Lemma 2 there is some tree-like  $\Lambda$ -model  $K$  such that  $K \cong L$  and thus  $K \cong_{\Theta} M'$  and  $K \cong_{\Gamma} N'$ . By Lemma 4 and the fact that  $\Pi$  is idempotent,

$$M \cong (M')^{\Pi} \cong_{\Theta} K^{\Pi} \cong_{\Gamma} (N')^{\Pi} \cong N,$$

so  $\mathcal{C}_{\Pi}$  is amalgamative.

The following lemma can be shown by a simple induction over the complexity of programs.

**Lemma 6.** *Given any  $\Lambda$ -models,  $M$  and  $N$  such that  $M \cong_{\Theta}^2 N$ , and a  $\Lambda$ -transduction  $\Pi$  such that for all  $a \in \Theta$ ,  $\Pi(a)$  does not contain any free propositional atoms from  $\Theta$ , we have  $M^{\Pi} \cong_{\Theta}^2 N^{\Pi}$ .*

## 5.2 Examples

Before we proceed with the proof of decidability, let us consider some examples. The logic **S4** has been shown to be equivalent to the pure modal logic of all reflexive, transitive frames. The class of all reflexive transitive  $\{a\}$ -frames (where  $a$  is some arbitrary label on a modality) is equivalent to the class  $\mathcal{C}_\Pi$  where  $\Pi$  is the  $\{a\}$ -transduction defined by  $\Pi(a) = a^*$ . Therefore the bisimulation quantified extension of **S4** is an idempotent transduction logic. Similarly:

1. **S5** is the pure modal logic of all reflexive, symmetric and transitive frames. It is defined by the transduction  $\Pi(a) = (a \cup a^-)^*$ .
2. **GL** is the pure modal logic of all transitive, well-founded frames. It is defined by the transduction

$$\Pi(a) = (\mu x[a]x)?; a; a^*.$$

The test ensures the for all  $(u, v) \in \Pi(a)$ , every  $a$ -path starting at  $u$  is finite. The correctness of this transduction (using a different notation) is presented in [15], Lemma 6.139.

3. **KD45** is the pure modal logic of all serial, transitive, Euclidean frames (where a frame is Euclidean if every successor of any given world is related to one another, see Figure 2). It is defined by the transduction  $\Pi(a) = ((a \cup a^-)^*; a) \cup ([a]\perp)?$ .

Idempotent transductions also allow us to define the interactions between different modalities. Consider the following logic of hierarchical knowledge. Let **S5**<sub>⊆n</sub> be the class of  $\Lambda$ -frames,  $(S, R)$  where

- $\Lambda = \{1, 2, \dots, n\}$ ,
- for all  $i \in \Lambda$ ,  $R(i)$  is reflexive, transitive and symmetric, and
- for all  $i \in \Lambda$ , for all  $j < i$ ,  $R(j) \subseteq R(i)$ .

This describes a system of hierarchical knowledge so that  $\Box_i$  is a knowledge operator for agent  $i$ , and if  $j < i$ , agent  $j$  knows at least as much as agent  $i$ . See [17] for a discussion of logics of hierarchical knowledge. For each  $i$  in  $\Lambda$  we define  $\Pi(i) = (1 \cup 1^- \cup \dots \cup i \cup i^-)^*$ .

An advantage of idempotent transduction logics is that they allow us to easily define powerful combinations of logics. For example, to reason about a system of belief and knowledge, we may suppose the standard interpretations for belief and knowledge hold, and also that knowledge implies belief, but not necessarily vice-versa. This situation can be described using the idempotent transduction,  $\Pi$ :

$$\begin{aligned} \Pi(k) &= (k \cup k^- \cup b \cup b^-)^* \\ \Pi(b) &= ((b \cup b^-)^*; b) \cup ([b]\perp)? \end{aligned}$$

where  $[k]\phi$  means the agent knows  $\phi$ , and  $[b]\phi$  means the agent believes  $\phi$ .

## 5.3 The decidability of idempotent transduction logics

We can now show that all idempotent transduction logics are decidable by reducing their satisfiability problem to the satisfiability problem for the full  $\mu$ -calculus.

**Theorem 2.** For every idempotent  $\Lambda$ -transduction,  $\Pi$ , there is a translation  $(\cdot)^\Pi$  from  $\text{BQL}_{\mathcal{C}_\Pi}$  to  $\mu L_\Lambda$  such that for all  $\Lambda$ -models,  $M$

$$M \models_{\mathcal{C}_\Pi} \alpha \iff M \models \alpha^\Pi. \quad (6)$$

*Proof.* The proof follows from the inductive construction of  $\alpha^\Pi$  which we describe here. As the base of this induction we may suppose that  $\alpha$  contains no bisimulation quantifiers at all. In this case we let  $\alpha^\Pi = \alpha[a \setminus \Pi(a)]_{a \in \Lambda}$ , where  $\alpha[a \setminus \Pi(a)]_{a \in \Lambda}$  is the formula  $\alpha$  with every modality,  $\diamond_a$ , replaced by  $\langle \Pi(a) \rangle$ . By comparing the definition of  $M^\Pi$  with the semantics for  $\mu L_\Lambda$  extended with dynamic modalities (Definition 6), we can see

$$M^\Pi \models_{\mathcal{C}_\Pi} \alpha \iff M \models \alpha[a \setminus \Pi(a)]_{a \in \Lambda}. \quad (7)$$

Now we proceed by induction. For all operators except the bisimulation quantifier the constructions are trivial:  $(\alpha_1 \vee \alpha_2)^\Pi = \alpha_1^\Pi \vee \alpha_2^\Pi$ ;  $(\neg\alpha)^\Pi = \neg\alpha^\Pi$ ; and  $\diamond_a \alpha = \langle \Pi(a) \rangle \alpha$ . The proofs follow directly from the semantic definitions of the operators.

We are left to define the translation for the bisimulation quantifier. Suppose that for all  $\Lambda$ -models  $N$ ,  $N^\Pi \models_{\mathcal{C}_\Pi} \alpha$  if and only if  $N \models \alpha^\Pi$ , and suppose that  $M^\Pi \models_{\mathcal{C}_\Pi} \exists x \alpha$ . By the definition,

$$M^\Pi \models_{\mathcal{C}_\Pi} \exists x \alpha \iff \exists N^\Pi \in \mathcal{C}_\Pi, N^\Pi \cong_x M^\Pi, \text{ and } N^\Pi \models_{\mathcal{C}_\Pi} \alpha. \quad (8)$$

Since  $\Pi$  is an idempotent transduction, if  $T$  is a tree-like  $\Lambda$ -model such that  $T \cong N^\Pi$ , then  $T^\Pi \cong N^\Pi$ . Let  $\mathcal{T}_\Lambda$  the set of tree-like  $\Lambda$ -models. By Lemma 2, the transitivity of bisimulation and Lemma 1

$$M^\Pi \models_{\mathcal{C}_\Pi} \exists x \alpha \iff \exists T \in \mathcal{T}_\Lambda, T \cong_x M^\Pi, \text{ and } T^\Pi \models_{\mathcal{C}_\Pi} \alpha. \quad (9)$$

Now, by applying the induction hypothesis it follows:

$$M^\Pi \models_{\mathcal{C}_\Pi} \exists x \alpha \iff \exists T \in \mathcal{T}_\Lambda, T \cong_x M^\Pi, \text{ and } T \models \alpha^\Pi. \quad (10)$$

The full  $\mu$ -calculus is not closed under bisimulation quantifiers (since bisimulations only respect forward modalities). As  $\alpha^\Pi$  is a formula of the full  $\mu$ -calculus, by Theorem 1 there is a  $\mu L_{\overline{\Lambda}}$  formula without converse,  $(\alpha^\Pi)'$ , such that  $T \models (\alpha^\Pi)'$  if and only if  $T^+ \models \alpha^\Pi$ . As  $T$  is a  $\Lambda$ -model (and thus a  $\overline{\Lambda}$ -model), we have  $T \models \alpha^\Pi$  if and only if  $T^+ \models \alpha^\Pi$ . Furthermore, we may suppose that  $(\alpha^\Pi)'$  does not contain converse modalities  $\overline{a}$ , (since  $\langle \overline{a} \rangle \phi$  will always be false). Therefore  $(\alpha^\Pi)'$  is a formula of  $\mu L_\Lambda$  without converse, which is closed under bisimulation quantifiers [3]. That is, there is a computable  $\mu L_\Lambda$  formula,  $\tilde{\exists} x (\alpha^\Pi)'$  such that for all  $\Lambda$ -models  $N$ ,  $N \models \tilde{\exists} x (\alpha^\Pi)'$  if and only if, for some model  $K \cong_x N$ ,  $K \models (\alpha^\Pi)'$ . Therefore

$$M^\Pi \models_{\mathcal{C}_\Pi} \exists x \alpha \iff M^\Pi \models \tilde{\exists} x (\alpha^\Pi)'. \quad (11)$$

Finally, from the semantic interpretation of the full modal  $\mu$ -calculus we can see

$$M^\Pi \models_{\mathcal{C}_\Pi} \exists x \alpha \iff M \models (\tilde{\exists} x (\alpha^\Pi)')[a \setminus \Pi(a)]_{a \in \Lambda}, \quad (12)$$

where  $\alpha[a \setminus \Pi(a)]_{a \in \Lambda}$  is the formula  $\alpha$  with every atomic program,  $a$ , replaced by  $\Pi(a)$ . To complete the proof we let  $(\exists x \alpha)^\Pi = (\tilde{\exists} x (\alpha^\Pi)')[a \setminus \Pi(a)]_{a \in \Lambda}$ .

In [14] a more simple reduction that did not consider converse modalities was used to prove decidability for positive idempotent transduction logics. A more complex decidability proof for idempotent transduction logics is given in [15] (Theorem 6.136).

We proof in [15] relied on the following theorem for  $BQDL_{\Lambda}$ , (which is *propositional dynamic logic with converse programs* extended by bisimulation quantifiers).

**Theorem 3.** *Given any  $\Lambda$ , for all formulas  $\alpha$  of  $BQDL_{\Lambda}$  there is some computable  $BQDL_{\Lambda}$  formula  $\alpha^*$  such that for any  $\Lambda$ -class  $\mathcal{C}$ , for any  $\mathcal{C}$ -model  $M$ ,  $M \models_{\mathcal{C}} \alpha^*$  if and only if  $M \models_{\Lambda} \alpha$ .*

The proof is given in [15] (Theorem 5.108) and follows the same strategy as the proof of Theorem 2, where  $\alpha^*$  is inductively defined to be  $\mu$ -calculus formula equivalent to  $\alpha$ . As the interpretation of the  $\mu$ -calculus is independent of the class of frames, the Theorem follows.

## 6 Conclusion

This paper presents a general class of bisimulation quantified modal logics, the *idempotent transduction logics*, that are amalgamative and decidable. This class is defined by taking the fixed points (modulo bisimulation) of transductions defined by programs of propositional dynamic logic with converse.

The benefits of this approach are as follows:

1. This definition provides a powerful methodology for extending and combining bisimulation quantified modal logics. A number of logics can be combined, where the modalities interaction can be expressed using programs of propositional dynamic logic.
2. Including bisimulation quantifiers in a logic can greatly increase the expressivity of the logic, allowing us to represent operators such as the *until* operator of temporal logic or fixed point operators. Furthermore in [15] it is shown that  $BQDL_{\Lambda}$  is non-elementarily more succinct than the modal  $\mu$ -calculus.
3. Bisimulation quantifiers have a natural interpretation as a *hypothetical assignment of atoms*. This allows them to be applied directly to refinement and simulation problems in a variety of ontologies.

Future work will examine further generalizations of the class of idempotent transduction logics. However, we note that this great generality comes at the cost of complexity. Therefore, we will also consider sub-classes of the idempotent transduction logics and look for efficient model-checking procedures, decision procedures, and axiomatizations.

*Acknowledgements* The author would like to thank the anonymous reviewers for their helpful suggestions and comments.

## References

1. Visser, A.: Uniform interpolation and layered bisimulation. In: Godel '96. Volume 6 of Lecture Notes Logic. (1996) 139–164
2. Ghilardi, S., Zawadowski, M.: Undefinability of propositional quantifiers in the modal system S4. *Studia Logica* **55** (1995) 259–271
3. D'Agostino, G., Hollenberg, M.: Logical questions concerning the  $\mu$ -calculus: interpolation, Lyndon and Los-Tarski. *J. Symb. Log.* **65**(1) (2000) 310–332
4. D'Agostino, G., Lenzi, G.: An axiomatization of bisimulation quantifiers via the  $\mu$ -calculus. *Theor. Comput. Sci.* **338** (2005) 64–95
5. Fine, K.: Propositional quantifiers in modal logic. *Theoria* **36** (1970) 336–346
6. Milner, R.: A calculus of communicating systems. LNCS **92** (1980)
7. Park, D.: Concurrency and automata on infinite sequences. LNCS **104** (1981) 167–183
8. van Benthem, J.: Correspondence theory. *Handbook of Philosophical Logic* **2** (1984) 167–247
9. Stirling, C.: The joys of bisimulation. In: MFCS 1998. Volume 1450 of LNCS. (1998) 142–151
10. Pnueli, A.: The temporal logic of programs. In: Proceedings of the Eighteenth Symposium on Foundations of Computer Science. (1977) 46–57
11. Fagin, R., Halpern, J., Moses, Y., Vardi, M.: Reasoning about knowledge. MIT Press (1995)
12. French, T.: Bisimulation quantified logics: undecidability. In: Proceedings of FSTTCS '05. Volume 3821 of LNCS. (2005) 396–407
13. D'Agostino, G., Lenzi, G., French, T.:  $\mu$ -programs, uniform interpolation and bisimulation quantifiers. *Journal of Applied Non-classical Logics* **16** (2006) 297–310
14. French, T.: Bisimulation quantified modal logics: decidability. *Advances in Modal Logic* **6** (2006) 147–166
15. French, T.: Bisimulation quantifiers for modal logic. PhD thesis, The University of Western Australia (2006) Available from <http://people.csse.uwa.edu.au/tim/>.
16. Gabbay, D., Kurucz, A., Wolter, F., Zakharayashv, M.: Many Dimensional Modal Logics: Theory and Applications. Elsevier (2003)
17. Engelhardt, K., van der Meyden, R., Su, K.: Modal logics with a linear hierarchy of local propositional quantifiers. *Advances in Modal Logic* **4** (2003) 9–30
18. French, T.: Decidability of propositionally quantified logics of knowledge. In: Australian Conference on Artificial Intelligence. (2003) 352–363
19. Abadi, M., Lamport, L.: The existence of refinement mappings. *Theoretical Computer Science* **82**(2) (1991) 253–284
20. Lamport, L.: Specifying concurrent program modules. *ACM Trans. Program. Lang. Syst.* **5**(2) (1983) 190–222
21. Vardi, M.: Reasoning about the past with two-way automata. In: Proceedings of ICALP '98. Volume 1443 of LNCS. (1998) 628–641
22. Kozen, D.: Results on the propositional  $\mu$ -calculus. *Theor. Comp. Sci.* **27** (1983)
23. Janin, D., Walukiewicz, I.: Automata for the modal  $\mu$ -calculus and related results. In: MFCS '95: Proceedings of the 20th International Symposium on Mathematical Foundations of Computer Science, London, UK, Springer-Verlag (1995) 552–562
24. Emerson, E.A., Jutla, C.S.: Tree automata,  $\mu$ -calculus and determinacy. In: Proceedings of the 32nd annual symposium on Foundations of computer science, IEEE Computer Society Press (1991) 368–377
25. Gradel, E., Thomas, W., Wilke, T., eds.: Automata, Logics, and Infinite Games. Springer (2002)