Project no.:	**PIRSES-GA-2011-295261**
Project full title:	**Mobility between Europe and Argentina applying Logics to Systems**
Project Acronym:	**MEALS**
Deliverable no.:	**2.1 / 3**
Title of Deliverable:	**Symmetries in Modal Logics**

Contractual Date of Delivery to the CEC:	**30-Sep-2015**
Actual Date of Delivery to the CEC:	**30-Sep-2015**
Organisation name of lead contractor for this deliverable:	**UNC**
Author(s):	**Carlos Areces, Ezequiel Orbe**
Participants(s):	**UNC**
Work package contributing to the deliverable:	**WP2**
Nature:	**R+D**
Dissemination Level:	**Public**
Total number of pages:	**32**
Start date of project:	1 Oct. 2011 **Duration:** 48 month

Abstract:

In this paper we develop the theoretical foundations to exploit symmetries in modal logics. We generalize the notion of symmetries of propositional formulas in conjunctive normal form to modal formulas using the framework provided by coinductive modal models introduced in [1]. Hence, the results apply to a wide class of modal logics including, for example, hybrid logics. We present two graph constructions that enable the reduction of symmetry detection in modal formulas to the graph automorphism detection problem, and we evaluate the graph constructions on modal benchmarks.

Note:

This deliverable is based on material that will be published in the Bulletin of Symbolic Logic."

This project has received funding from the European Union Seventh Framework Programme (FP7 2007-2013) under Grant Agreement Nr. 295261.

Contents

1 Symmetries in Automated Theorem Proving **3**

2 Symmetries in the Basic Modal Logic **4**

3 Beyond Basic Modal Logic **9**
3.1 Coinductive Modal Models 10
3.2 A Generalized Theory of Symmetries 13
3.3 Layered Permutations 16

4 Symmetry Detection for Modal Logics **19**
4.1 Experimental Evaluation 25
4.1.1 Implementation 25
4.1.2 Results 26

5 Discussion **27**

Bibliography **28**

MEALS Partner Abbreviations **31**

1 Symmetries in Automated Theorem Proving

Symmetry is a familiar notion. Intuitively, we say that an object is symmetric if "under any kind of transformation at least one property of the object is left invariant" [2]. Symmetry has many uses. Not only can we study the symmetric properties of an object (geometric, mathematical, etc.) to understand its behavior, but we can also derive specific consequences regarding the object under study based on its symmetry properties, i.e., using a "symmetry-based argument" [3]. In automated reasoning, many problem classes, in particular those arising from real world applications, present symmetries, and their presence is usually a source of additional complexity since we might end up looking for solutions in symmetrical subspaces of the problem's search space. Ideally, if we can recognize that such symmetries exist, we might use them to direct a search algorithm to look for solutions only in non-symmetric parts of the search space, thus reducing the overall difficulty [4].

In this context, a symmetry can be defined as a permutation of the variables (or literals) of the input formula that preserves its structure and its solutions. Symmetries can be classified into *semantic* or *syntactic* [5]. Semantic symmetries are permutations of the formula that preserves its set of the models (or solutions) and, therefore, can be regarded as properties of the underlying Boolean function, independent of the particular syntactic representation. For example, consider the Boolean function $f(a, b, x, y) = axy + bxy$, where a, b, x, y take values over $\{0, 1\}$ and sum and product are binary. It is straightforward to verify that the permutation that interchanges the parameters a and b leaves the function unchanged, maintaining its set of solutions. Syntactic symmetries, on the other hand, correspond to the specific representation of the function. In particular, a symmetry of a given function's syntactic representation might not be a symmetry of another equivalent syntactic representation. For example, the formulas $axy + bxy$ and $(ax + bx)y$ represent the same Boolean function, but it is easy to see that the permutation that interchanges x and y is a syntactic symmetry of the former (modulo commutativity of the sum) and not of the latter. Every syntactic symmetry of a formula is also a semantic symmetry of the underlying Boolean function, while the converse does not always hold.

Syntactic symmetries have received much attention in the context of propositional SAT solving since they were first used in [6] as a mechanism to strengthen resolution-based proof systems for propositional logic. Since then, many articles discuss how to detect and exploit syntactic symmetries in SAT solving [7, 5, 8, 9, 10, 11, 12, 13, 14]. Symmetries have been also extensively investigated and successfully exploited in other domains besides SAT like Constraint Satisfaction Problem [15, 16], Integer Programming [17, 18], Planning [19, 20], Model Checking [21, 22, 23, 24], Quantified Boolean Formulas (QBF) [25, 26, 27], and Satisfiability Modulo Theories (SMT) [28, 29, 30].

In modal logics, research has been done on how to exploit symmetries in model checking for the temporal logic LTL [21, 31, 32, 33], and for some temporal-epistemic logics [34]. However, to the best of our knowledge, the use of symmetries in satisfiability and automated theorem proving for modal logics remains largely unexplored. In this work, we generalize the notion of symmetries from the propositional setting to modal formulas in conjunctive normal form for different modal logics. Our generalization works for the basic modal language over different model classes (e.g., reflexive, linear or transitive models), and for modal logics with additional

modal operators (e.g., universal and hybrid operators).

We also present a method to detect symmetries of modal formulas in conjunctive normal form. This method reduces the symmetry detection problem to the graph automorphism problem. A general graph construction algorithm, suitable for many modal logics, is presented. In order to tackle a broad range of modal languages we use the semantics provided by coinductive modal models [1] instead of the more familiar relational semantics. Coinductive modal models provide a homogeneous framework to investigate different modal languages at a greater level of abstraction.

The paper is organized as follows. In Section 2 we define modal symmetries for the basic modal logic, together with the appropriate notion of simulation. The main goal of this section is to introduce the fundamental ideas behind symmetries in modal logic as clearly as possible, avoiding the complexities of the coinductive framework. In Section 3, we present coinductive modal models, and generalize the notion of modal symmetries to work with it. We also introduce layered permutations and show that they can be used when the modal logic has the adequate notion of the tree model property. In Section 4 we present two graph constructions to detect symmetries in modal formulas, prove their correctness and show experimental results on modal benchmarks. In Section 5, we draw our conclusions and discuss future research.

2 Symmetries in the Basic Modal Logic

In what follows, we assume basic knowledge of classical modal logics and group theory, see [35, 36, 37] for details.

Definition 1 (Syntax)**.** A *modal signature* is a pair $\mathcal{S} = \langle \mathsf{PROP}, \mathsf{MOD} \rangle$, where PROP and MOD are two countable disjoint sets and PROP is infinite. The well-formed formulas over the signature $\mathcal{S}$ are defined by the rule

$$\varphi, \psi ::= p \mid \neg\varphi \mid \varphi \vee \psi \mid [m]\varphi,$$

where $p \in \mathsf{PROP}$ and $m \in \mathsf{MOD}$. FORM denotes the set of all well-formed formulas over a given signature $\mathcal{S}$. $\top$ and $\bot$ stand for an arbitrary tautology and contradiction, respectively. $\langle m \rangle \varphi$ is defined as $\neg[m]\neg\varphi$. Other connectives such as $\wedge, \rightarrow$, etc. are defined in the usual way. $[m]$ is called a *box* operator, and $\langle m \rangle$ is called a *diamond* operator. When MOD is a singleton, i.e., in the mono-modal case, we simply write $\Box$ and $\Diamond$ for the box and diamond operators.

Definition 2 (Propositional literals)**.** A *propositional literal* l is either a propositional variable $p \in \mathsf{PROP}$ or its negation $\neg p$. The set of propositional literals over PROP is $\mathsf{PLIT} = \mathsf{PROP} \cup \{\neg p \mid p \in \mathsf{PROP}\}$. $L \subseteq \mathsf{PLIT}$ is *complete* if for each $p \in \mathsf{PROP}$ either $p \in L$ or $\neg p \in L$. It is *consistent* if for each $p \in \mathsf{PROP}$ either $p \notin L$ or $\neg p \notin L$. Any complete and consistent set of literals L defines a unique propositional valuation $v \subseteq \mathsf{PROP}$ which is defined as $p \in v$ iff $p \in L$.

Definition 3 (Modal literals, clauses and modal CNF)**.** A formula is in *modal conjunctive normal form (modal CNF)* if it is a conjunction of clauses. A *clause* is a disjunction of propositional and modal literals. A *modal literal* is a formula of the form $[m]C$ or $\neg[m]C$ where C is a clause.

Every modal formula can be transformed into an equivalent formula in modal CNF at the risk of an exponential blowup in the size of the formula; it can be transformed into an equisatisfiable formula in polynomial time, using additional propositional variables (see [38] for details).

To consider formulas modulo commutativity and idempotency of conjunction and disjunction, we represent a modal CNF formula as a *set* of clauses (interpreted conjunctively), and each clause as a *set* of propositional and modal literals (interpreted disjunctively). This set representation disregards order and multiplicity of clauses and literals in a formula. We will assume that modal CNF formulas are represented using set notation, even though we will often write them using the familiar notation and, we will write $l \in C$ to denote that the literal l occurs in the clause C, and $C \in \varphi$ to denote that the clause C is a clause of the formula φ.

Example 1. The formula $\neg\Box(\neg p \vee \Box q \vee \Box\neg q \vee \neg p) \wedge \neg\Box(\neg q \vee \Box p \vee \Box\neg p)$ is written, using the set notation, as $\{\neg\Box\{\neg p, \Box\{q\}, \Box\{\neg q\}\}, \neg\Box\{\neg q, \Box\{p\}, \Box\{\neg p\}\}\}$.

Definition 4 (clauses function)**.** The function clauses returns the multiset of clauses in a formula φ. Let $\uplus$ be the operation of union with repetition between multisets. Let φ be a modal CNF formula, C a clause, $p \in \mathsf{PROP}$, $l \in \mathsf{PLIT}$ and $m \in \mathsf{MOD}$, we define clauses as follows,

$$\begin{array}{lcl} \mathsf{clauses}(p) & = & \{\} \\ \mathsf{clauses}(\neg p) & = & \{\} \\ \mathsf{clauses}([m]C) & = & \mathsf{clauses}(C) \\ \mathsf{clauses}(\neg[m]C) & = & \mathsf{clauses}(C) \\ \mathsf{clauses}(C) & = & \{C\} \uplus \biguplus_{l \in C} \mathsf{clauses}(l) \\ \mathsf{clauses}(\varphi) & = & \biguplus_{C \in \varphi} \mathsf{clauses}(C). \end{array}$$

Example 2. The formula $\varphi = \neg\Box(\neg p \vee \Box q \vee \Box\neg q \vee \neg p) \wedge \neg\Box(\neg q \vee \Box p \vee \Box\neg p)$ is in modal CNF, and, $\mathsf{clauses}(\varphi) = \{\neg\Box(\neg p \vee \Box q \vee \Box\neg q \vee \neg p), \neg\Box(\neg q \vee \Box p \vee \Box\neg p), \neg p \vee \Box q \vee \Box\neg q \vee \neg p, \neg q \vee \Box p \vee \Box\neg p, q, \neg q, p, \neg p\}$.

It is worth mentioning that, even though we focus on modal CNF formulas, the forthcoming results apply to arbitrary modal formulas. We could have used arbitrary modal formulas and work modulo equivalence to their CNFs.

Definition 5 (Models)**.** A *(pointed) model* is a tuple $\langle w, W, \{R_m\}_{m \in \mathsf{MOD}}, V\rangle$, where W is a non-empty set, $w \in W$, $R_m \subseteq W \times W$ for all $m \in \mathsf{MOD}$ and $V(v) \subseteq \mathsf{PROP}$ for all $v \in W$. Given a model $\mathcal{M} = \langle w, W, \{R_m\}_{m \in \mathsf{MOD}}, V\rangle$ and $w' \in W$, we define $\mathcal{M}, w' = \langle w', W, \{R_m\}_{m \in \mathsf{MOD}}, V\rangle$.

Definition 6 (Semantics)**.** Let $\mathcal{M} = \langle w, W, \{R_m\}_{m \in \mathsf{MOD}}, V\rangle$ be a model and φ a modal formula. We define the satisfiability relation $\models$, in $\mathcal{M}$ at point w, as follows,

$$\begin{array}{lll} \mathcal{M} \models p & \text{iff} & p \in V(w) \\ \mathcal{M} \models \neg\varphi & \text{iff} & \mathcal{M} \not\models \varphi \\ \mathcal{M} \models \varphi \vee \psi & \text{iff} & \mathcal{M} \models \varphi \text{ or } \mathcal{M} \models \psi \\ \mathcal{M} \models [m]\varphi & \text{iff} & \mathcal{M}, w' \models \varphi, \text{ for all } w' \text{ s.t. } wR_m w'. \end{array}$$

We say that a formula φ is *satisfiable* if for some pointed model $\mathcal{M}$ we have that $\mathcal{M} \models \varphi$. Otherwise, it is unsatisfiable. $\mathsf{Mods}(\varphi) = \{\mathcal{M} \mid \mathcal{M} \models \varphi\}$ denotes the class of all models that satisfy φ.

Definition 7 (Modal depth)**.** The modal depth of a modal formula φ (notation: $\mathsf{md}(\varphi)$) is the maximum nesting of modal operators that occurs in φ:

$$\begin{array}{lcl} \mathsf{md}(p) & = & 0 \\ \mathsf{md}(\neg\varphi) & = & \mathsf{md}(\varphi) \\ \mathsf{md}(\varphi \vee \psi) & = & \max\{\mathsf{md}(\varphi), \mathsf{md}(\psi)\} \\ \mathsf{md}([m]\varphi) & = & 1 + \mathsf{md}(\varphi). \end{array}$$

Definition 8 (Permutations)**.** A *permutation* ρ of a set A is a bijection $\rho : A \mapsto A$. ρ^0 denotes the identity permutation and ρ^{-1} the inverse of ρ. For $n \in \mathbb{Z}$, ρ^n denotes the n-times composition of ρ with itself if $n \geq 1$, and the n-times composition of ρ^{-1} with itself if $n \leq -1$.

Each permutation ρ of a set A determines a natural partition of A into equivalence classes, also called the *orbits of* ρ, with the property that $a, b \in A$ are in the same equivalence class if and only if $b = \rho^n(a)$ for some $n \in \mathbb{Z}$.

A permutation ρ *has finite support* if for only a finite number of elements $a \in A$ we have that $\rho(a) \neq a$. Permutations with finite support can be represented using *cyclic notation* [37]. A *cycle* is a finite sequence of different elements $(a_1\ a_2\ \ldots\ a_k)$, that defines the permutation

$$\begin{array}{lcll} \rho(a) & = & a & \text{if } a \text{ does not appear in the sequence} \\ \rho(a_i) & = & a_{i+1} & \text{if } i < k \\ \rho(a_k) & = & a_1. & \end{array}$$

A cycle with only two elements is called a *transposition*. A permutation is represented as a finite sequence of cycles, e.g., $(a_{11}\ \ldots\ a_{1k_1}) \ldots (a_{i1}\ \ldots\ a_{ik_i})$. The associated permutation is obtained as the composition of the permutations associated to each cycle.

Definition 9 (Permutation over PLIT)**.** A *permutation of propositional literals* is a bijection $\rho : \mathsf{PLIT} \mapsto \mathsf{PLIT}$. If L is a set of propositional literals then $\rho(L) = \{\rho(l) \mid l \in L\}$.

Notice that Definition 9 defines permutations over the infinite set PLIT. However, in practice we only deal with permutations over a finite subset A of PLIT, i.e., the set of propositional literals occurring in the formula at hand, which enables us to use all the available results on permutations over finite sets and with finite support.

Definition 10 (Consistent Permutation)**.** For $p \in \mathsf{PROP}$, let $\sim p = \neg p$ and $\sim\neg p = p$. We say that a permutation ρ is *consistent* if for every propositional literal l, $\rho(\sim l) = \sim\rho(l)$.

Consistency guarantees that a permutation interacts nicely when applied to a set of literals, e.g., if we have a consistent set of literals it will remain consistent after applying a consistent permutation to it. From a group theoretic perspective, consistent permutations form a subgroup of the group of all permutations over a given set, as the composition of two or more consistent permutations is again a consistent permutation. From now on, we only consider consistent permutations of propositional literals.

Definition 11 (Permutation of a formula)**.** Let φ be a modal formula. We define the application of a permutation ρ to φ as follows

$$\begin{array}{lcl} \rho(\neg\varphi) & = & \neg\rho(\varphi) \\ \rho(\varphi \vee \psi) & = & \rho(\varphi) \vee \rho(\psi) \\ \rho([m]\varphi) & = & [m]\rho(\varphi). \end{array}$$

Definition 12 (Symmetry)**.** Let φ be a modal CNF formula and ρ a consistent permutation. We say that ρ is a *symmetry* of φ if $\varphi = \rho(\varphi)$, when φ is represented using set notation.

Example 3. $\rho = (p\ \neg q)(\neg p\ q)$ is the permutation that makes $\rho(p) = \neg q$, $\rho(\neg q) = p$, $\rho(\neg p) = q$, $\rho(q) = \neg p$ and leaves unchanged all other literals. ρ is a consistent permutation, and it is a symmetry of the formula $\varphi = (\neg p \vee r) \wedge (q \vee r) \wedge \Box(\neg p \vee q)$ since φ and $\rho(\varphi)$ are identical using set notation.

Definition 13 (Generated set of propositional literals)**.**
Let $\mathcal{M} = \langle w, W, \{R_m\}_{m\in\mathsf{MOD}}, V\rangle$ be a model. For every $v \in W$, we define its *generated set of propositional literals* (notation: $L_{V(v)}$) as $V(v) \cup \{\neg p \mid p \in \mathsf{PROP}\backslash V(v)\}$. $L_{V(v)}$ is a consistent and complete set of propositional literals.

The key concept to talk about symmetries in modal logics is that of ρ-simulations.

Definition 14 (ρ-simulation)**.** Let ρ be a permutation. A *ρ-simulation* between models $\mathcal{M} = \langle w, W, \{R_m\}_{m\in\mathsf{MOD}}, V\rangle$ and $\mathcal{M}' = \langle w', W', \{R'_m\}_{m\in\mathsf{MOD}}, V'\rangle$ is a relation $Z \subseteq W \times W'$ that satisfies the following conditions:

Root: wZw'.

ρ-Harmony: wZw' implies $l \in L_{V(w)}$ iff $\rho(l) \in L_{V'(w')}$.

Zig: If wZw' and wR_mv then there exists v' such that $w'R'_mv'$ and vZv'.

Zag: If wZw' and $w'R'_mv'$ then there exists v such that wR_mv and vZv'.

We say that two pointed models $\mathcal{M}$ and $\mathcal{M}'$ are *ρ-similar* (notation: $\mathcal{M} \underline{\rightarrow}_\rho \mathcal{M}'$) if there is a ρ-simulation Z between them.

Notice that the relation $\underline{\rightarrow}_\rho$ is not symmetric (hence, it is not a bisimulation): $\mathcal{M} \underline{\rightarrow}_\rho \mathcal{M}'$ does not imply $\mathcal{M}' \underline{\rightarrow}_\rho \mathcal{M}$. On the other hand, it is not difficult to prove that $\mathcal{M} \underline{\rightarrow}_\rho \mathcal{M}'$ implies $\mathcal{M}' \underline{\rightarrow}_{\rho^{-1}} \mathcal{M}$. We can think of a ρ-simulation as relaxing the atomic harmony condition of bisimulations to incorporate the effect of permutations.

Example 4. Consider the models $\mathcal{M} = \langle w, \{w\}, \emptyset, V\rangle$ and $\mathcal{M}' = \langle w', \{w'\}, \emptyset, V'\rangle$ where $V(w) = \{p, s\}$ and $V'(w') = \{q, s\}$. Let $\rho = (p\ q\ r)(\neg p\ \neg q\ \neg r)$ be a consistent permutation. That $\mathcal{M} \underline{\rightarrow}_\rho \mathcal{M}'$ is straightforward: just consider the set $L_{V(w)} = \{p, \neg q, \neg r, s\}$, then we have that $\rho(L_{V(w)}) = L_{V'(w')}$ and the ρ-harmony condition holds. However it is not the case that $\mathcal{M}' \underline{\rightarrow}_\rho \mathcal{M}$. To see this, consider the set $L_{V'(w')} = \{\neg p, q, \neg r, s\}$, then $\rho(L_{V'(w')}) = \{\neg q, r, \neg p, s\} \neq L_{V(w)}$ and the ρ-harmony condition fails.

If we restrict ourselves to permutations that can be defined as the product of disjoint transpositions then the ρ-simulation relation is indeed symmetric.

From the definition of ρ-simulations it follows that while they do not preserve satisfiability of modal formulas (as is the case with bisimulations) they do preserve satisfiability of *permutations* of formulas.

Proposition 1. *Let ρ be a consistent permutation, φ a modal formula and $\mathcal{M} = \langle w, W, \{R_m\}_{m\in\mathsf{MOD}}, V\rangle$, $\mathcal{M}' = \langle w', W', \{R'_m\}_{m\in\mathsf{MOD}}, V'\rangle$ models such that $\mathcal{M} \underline{\rightarrow}_\rho \mathcal{M}'$. Then $\mathcal{M} \models \varphi$ iff $\mathcal{M}' \models \rho(\varphi)$.*

Proof. The proof is by induction on the syntactic structure of φ.

Base Case:

[$\varphi = p$]: $\mathcal{M} \models p$ iff $p \in V(w)$ iff $p \in L_{V(w)}$ iff, by **ρ-Harmony**, $\rho(p) \in L_{V'(w')}$ iff $\mathcal{M}' \models \rho(p)$.

Inductive Step:

[$\varphi = [m]\psi$]: $\mathcal{M} \models [m]\psi$ iff $\langle v, W, \{R_m\}_{m\in\mathsf{MOD}}, V\rangle \models \psi$ for all $v \in W$ s.t. wR_mv. Since $\mathcal{M} \underline{\rightarrow}_\rho \mathcal{M}'$, by **Zig**, for all v exists $v' \in W'$ s.t. $w'R'_mv'$ and vZv', and, by inductive hypothesis, $\langle v', W', \{R'_m\}_{m\in\mathsf{MOD}}, V'\rangle \models \rho(\psi)$. Hence, $\mathcal{M}' \models [m]\rho(\psi) = \rho([m]\psi)$. The converse follows using **Zag** and the inductive hypothesis.

The remaining cases follow by induction. □

We also need to consider the effect of applying permutations to models.

Definition 15 (Permutation of a model)**.** Let ρ be a permutation and $\mathcal{M} = \langle w, W, \{R_m\}_{m\in\mathsf{MOD}}, V\rangle$ a model. Then $\rho(\mathcal{M}) = \langle w, W, \{R_m\}_{m\in\mathsf{MOD}}, V'\rangle$, where $V'(v) = \rho(L_{V(v)}) \cap \mathsf{PROP}$ for all $v \in W$. For M a class of models, $\rho(M) = \{\rho(\mathcal{M}) \mid \mathcal{M} \in M\}$.

It follows that $\mathcal{M}$ and $\rho(\mathcal{M})$ are always ρ-similar.

Proposition 2. *Let ρ be a consistent permutation and $\mathcal{M}$ a model. Then $\mathcal{M} \underline{\rightarrow}_\rho \rho(\mathcal{M})$.*

Proof. We show that the identity relation is a ρ-simulation.

[ρ-Harmony]: From the definition of $\rho(\mathcal{M})$, $L_{V'(v)} = \rho(L_{V(v)})$ for all $v \in W$, hence if $l \in L_{V(v)}$ then $\rho(l) \in \rho(L_{V(v)})$. Moreover, $\rho(L_{V(v)})$ is a complete set of literals since $L_{V(v)}$ is a complete set of literals and ρ is a consistent permutation, and hence the converse also follows.

[Root, Zig and Zag]: Trivial as the relation in both models is the same.

□

If φ is true in a model $\mathcal{M}$, we intuitively want $\rho(\varphi)$ to be true in $\rho(\mathcal{M})$.

Proposition 3. *Let ρ be a consistent permutation, φ a modal formula and $\mathcal{M} = \langle w, W, \{R_m\}_{m\in\mathsf{MOD}}, V\rangle$ a model. Then $\mathcal{M} \models \varphi$ iff $\rho(\mathcal{M}) \models \rho(\varphi)$.*

Proof. It follows directly from Proposition 1 and Proposition 2. □

If ρ is a symmetry of φ, a model $\mathcal{M}$ satisfies φ if and only if $\rho(\mathcal{M})$ also does.

Theorem 1. *Let φ be a modal CNF formula and ρ a symmetry of φ. Then $\mathcal{M} \models \varphi$ iff $\rho(\mathcal{M}) \models \varphi$.*

Proof. It follows from Proposition 3 in the particular case when φ is a modal CNF formula and ρ is a symmetry of φ and hence $\rho(\varphi) = \varphi$. □

A consequence of Theorem 1 is that given a set of models and the symmetries of a formula φ, the latter partition the former in such a way that the resulting equivalence classes (orbits) contain only models satisfying φ or models not satisfying φ. From a practical point of view, this implies that, when searching for a satisfying model for a given formula, we can focus just on the representatives from each equivalence class, provided that we can compute them.

Symmetries also provide us with an inference mechanism.

Theorem 2. *Let φ and ψ be modal CNF formulas and let ρ be a symmetry of φ. Then $\varphi \models \psi$ iff $\varphi \models \rho(\psi)$.*

Proof. We first show that under the hypothesis of the theorem the following claim holds

Claim: $\mathsf{Mods}(\varphi) = \rho(\mathsf{Mods}(\varphi))$.

[$\supseteq$]: Let $\mathcal{X} \in \rho(\mathsf{Mods}(\varphi))$ and $\mathcal{M} \in \mathsf{Mods}(\varphi)$ be models such that $\mathcal{X} = \rho(\mathcal{M})$. Then $\mathcal{M} \models \varphi$ and, by Theorem 1, $\rho(\mathcal{M}) \models \varphi$ and hence $\rho(\mathcal{M}) = \mathcal{X} \in \mathsf{Mods}(\varphi)$. [$\subseteq$]: Let $\mathcal{M} \in \mathsf{Mods}(\varphi)$, then $\mathcal{M} \models \varphi$. By Theorem 1, $\rho(\mathcal{M}) \models \varphi$, therefore, $\rho(\mathcal{M}) \in \mathsf{Mods}(\varphi)$. Since ρ is arbitrary, the results holds also for ρ^k, $k \in \mathbb{Z}$. Since ρ is a finite permutation, there exists n such that ρ^n is the identity permutation. Now consider $\rho^{n-1}(\mathcal{M})$, we know $\rho^{n-1}(\mathcal{M}) \in \mathsf{Mods}(\varphi)$. Hence $\rho^n(\mathcal{M}) = \mathcal{M} \in \rho(\mathsf{Mods}(\varphi))$.

Now, we have to prove that $\varphi \models \psi$ if and only if $\varphi \models \rho(\psi)$. $\varphi \models \psi$ iff $\mathsf{Mods}(\varphi) \models \psi$ iff, using Proposition 3, $\rho(\mathsf{Mods}(\varphi)) \models \rho(\psi)$. Since ρ is a symmetry of φ, by the Claim above $\rho(\mathsf{Mods}(\varphi)) \models \rho(\psi)$ iff $\mathsf{Mods}(\varphi) \models \rho(\psi)$ and hence $\varphi \models \rho(\psi)$. □

It follows from Theorem 2 that we can use the symmetries of a formula as an inexpensive inference mechanism in every situation where entailment is involved during modal automated reasoning. Indeed, applying a permutation on a formula is a calculation that is, arguably, computationally cheaper than. e.g., a tableaux expansion or a resolution step. Therefore, inferences obtained by this mechanism may reduce the total running time of an inference algorithm. In the case of propositional logic, strengthening clause learning with symmetric inferences has been proved efficient [14].

Theorem 1 and Theorem 2 are the core results that enable the exploitation of symmetries in the basic modal logic.

3 Beyond Basic Modal Logic

We now generalize the obtained results to richer modal logics. To do so we use the framework of *coinductive modal models* introduced in [1] to investigate normal forms for modal logics. This framework allows the representation of different modal logics homogeneously: results obtained can be extended to concrete modal languages by just selecting the appropriate model classes.

3.1 Coinductive Modal Models

We now introduce the basic definitions concerning coinductive modal models (see [1] for more details).

Definition 16 (Syntax)**.** A *modal signature* is a pair $\mathcal{S} = \langle \mathsf{ATOM}, \mathsf{MOD} \rangle$, where ATOM and MOD are two countable disjoint sets and ATOM is infinite. The well-formed formulas over the signature $\mathcal{S}$ are defined by the rule

$$\varphi, \psi ::= a \mid \neg\varphi \mid \varphi \vee \psi \mid [m]\varphi,$$

where $a \in \mathsf{ATOM}$ and $m \in \mathsf{MOD}$. FORM denotes the set of all well-formed formulas over the signature $\mathcal{S}$. $\top$ and $\bot$ stand for an arbitrary tautology and contradiction, respectively. Classical connectives such as $\wedge, \rightarrow$ and $\langle m \rangle$ are defined in the usual way.

As for the basic modal logic case, we are going to work with formulas in modal CNF as defined previously, but now using *atom literals* instead of propositional literals, where $\mathsf{ALIT} = \mathsf{ATOM} \cup \{\neg a \mid \text{ for all } a \in \mathsf{ATOM}\}$.

Notice that the language we just defined is the language of the basic multi-modal logic introduced in Definition 1 but, as we will now see, we will be able to cast other modal logics including, for example, hybrid operators right into this same syntax in a natural way. How we do this will become clear once we provide our definition of models.

Definition 17 (Models)**.** Let $\mathcal{S} = \langle \mathsf{ATOM}, \mathsf{MOD} \rangle$ be a signature and W be a fixed, non-empty set. Mods_W, the class of all models with domain W, for the signature $\mathcal{S}$, is the class of all tuples $\langle w, W, R, V \rangle$ such that $w \in W$ is the *point of evaluation*, W is the *domain*, $R(m, v) \subseteq \mathsf{Mods}_W$ for $m \in \mathsf{MOD}$ and $v \in W$ is the *accessibility relation*, and $V(v) \subseteq \mathsf{ATOM}$ for $v \in W$ is the *valuation*.

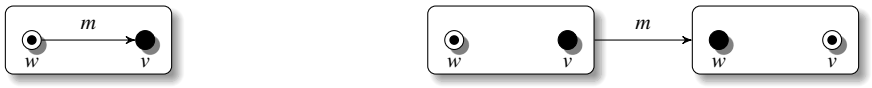
Let Mods be the class of all models over all domains, i.e., $\mathsf{Mods} = \bigcup_W \mathsf{Mods}_W$.

Observe that for each W, Mods_W is well-defined (coinductively), and so is Mods, the class of all models.

The main difference between Definition 5 and Definition 17 lies in how we handle m-successors. In the latter, for each modality m and each state v in a model, we define $R(m, v)$, the successors of v through the m modality, as a set of *models* (therefore, this definition of models is coinductive), while in the former we define it as a set of points in the domain.

Example 5. Consider the relational model in Figure 1a, and its equivalent coinductive model in Figure 1b. The point of evaluation in each model is circled. Notice that the relation of a coinductive model leads to another coinductive model, whereas in a relational model the relation leads to another point of the same model.

Definition 18 (Domain, point of evaluation, valuation and accessibility relation of an arbitrary model)**.** Let $\mathcal{M}$ be a coinductive model, we will write $|\mathcal{M}|$ for its domain, $w^{\mathcal{M}}$ for its point of evaluation, $V^{\mathcal{M}}$ for its valuation, and $R^{\mathcal{M}}$ for its accessibility relation.



(a) Relational. (b) Coinductive.

Figure 1: A relational model and its equivalent coinductive model.

Example 6. Let us consider the following coinductive model: $\mathcal{N} = \langle 0, \mathbb{N}, \{\}, \{n \mapsto \{\} \mid n \in \mathbb{N}\}\rangle$, where 0 is the point of evaluation, $\mathbb{N}$ is the set of natural numbers, the accessibility relation is the empty relation, and the valuation assigns the empty set to all elements in the domain. Then $|\mathcal{N}| = \mathbb{N}$, $w^{\mathcal{N}} = 0$, $R^{\mathcal{N}} = \{\}$ and $V^{\mathcal{N}} = \{n \mapsto \{\} \mid n \in \mathbb{N}\}$.

Definition 19 (Extension of a model)**.** Given $\mathcal{M} \in \mathsf{Mods}_W$, let Ext($\mathcal{M}$), the *extension of* $\mathcal{M}$, be the smallest subset of Mods_W that contains $\mathcal{M}$ and is such that if $\mathcal{N} \in \text{Ext}(\mathcal{M})$, then $R^{\mathcal{N}}(m, v) \subseteq \text{Ext}(\mathcal{M})$ for all $m \in \mathsf{MOD}$, $v \in W$.

In words, the *extension of a model* is the class of models reachable via the reflexive and transitive closure of the union of its accessibility relations.

In what follows, we are interested in working with classes of coinductive models that are *closed under accessibility relations*.

Definition 20 (Closed class)**.** A non-empty class of models C is *closed under accessibility relations* (we will say that C is a *closed class*, for short) whenever $\mathcal{M} \in C$ implies $\text{Ext}(\mathcal{M}) \subseteq C$.

Definition 21 (Semantics)**.** Let φ be a modal formula and $\mathcal{M} = \langle w, W, R, V\rangle$ a model in Mods. We define the satisfiability relation $\models$ as follows

$$\begin{array}{lll} \mathcal{M} \models a & \text{iff} & a \in V(w) \text{ for } a \in \mathsf{ATOM} \\ \mathcal{M} \models \neg\varphi & \text{iff} & \mathcal{M} \not\models \varphi \\ \mathcal{M} \models \varphi \vee \psi & \text{iff} & \mathcal{M} \models \varphi \text{ or } \mathcal{M} \models \psi \\ \mathcal{M} \models [m]\varphi & \text{iff} & \mathcal{M}' \models \varphi, \text{ for all } \mathcal{M}' \in R(m, w). \end{array}$$

If C is a closed class, we write $C \models \varphi$ whenever $\mathcal{M} \models \varphi$ for every $\mathcal{M}$ in C. We say that $\Gamma_C = \{\varphi \mid C \models \varphi\}$ is the *logic* defined by C, and we write $\models_C$ to indicate entailment modulo the closed class C. $\mathsf{Mods}(\varphi) = \{\mathcal{M} \mid \mathcal{M} \models \varphi\}$ denotes the class of all models that satisfy φ and $\mathsf{Mods}_C(\varphi) = \{\mathcal{M} \mid \mathcal{M} \models \varphi \text{ and } \mathcal{M} \in C\}$ denotes the class of all models in C that satisfy φ.

The logic Γ_{Mods} (generated by the class of all possible models) coincides with the basic multimodal logic K [1]. Inspecting the definition above, we can see that the semantic clause for $[m]$ is the classical condition defining a box operator. However, if we restrict ourselves to the appropriate class of models, we can actually ensure that $[m]$ behaves in different ways and capture different modal logics.

Let us call a predicate P on models a *defining condition* for a class C whenever C is such that $\mathcal{M} \in C$ if and only if $P(\mathcal{M})$ holds. Consider the signature $\mathcal{S} = \langle \mathsf{ATOM}, \mathsf{MOD}\rangle$ where $\mathsf{ATOM} = \mathsf{PROP} \cup \mathsf{NOM}$, $\mathsf{MOD} = \mathsf{REL} \cup \{\mathsf{A}\} \cup \{@_i \mid i \in \mathsf{NOM}\}$; and $\mathsf{PROP} = \{p_1, p_2, \ldots\}$,

Class	Defining condition
$\mathcal{C}^{\mathsf{K}}_m$	$\mathcal{P}^{\mathsf{K}}_m(\mathcal{M})$ iff $\forall m, w \,.\, R^{\mathcal{M}}(m, w) \subseteq \{\langle v, \lvert\mathcal{M}\rvert, R^{\mathcal{M}}, V^{\mathcal{M}}\rangle \mid v \in \lvert\mathcal{M}\rvert\}$
$\mathcal{C}_{\mathsf{A}}$	$\mathcal{P}_{\mathsf{A}}(\mathcal{M})$ iff $\forall w \,.\, R^{\mathcal{M}}(\mathsf{A}, w) = \{\langle v, \lvert\mathcal{M}\rvert, R^{\mathcal{M}}, V^{\mathcal{M}}\rangle \mid v \in \lvert\mathcal{M}\rvert\}$
$\mathcal{C}_{@_i}$	$\mathcal{P}_{@_i}(\mathcal{M})$ iff $\forall i, w \,.\, R^{\mathcal{M}}(@_i, w) = \{\langle v, \lvert\mathcal{M}\rvert, R^{\mathcal{M}}, V^{\mathcal{M}}\rangle \mid i \in V(v)\}$
$\mathcal{C}_{\mathsf{NOM}}$	$\mathcal{P}_{\mathsf{NOM}}(\mathcal{M})$ iff $\forall i, w \,.\, \{w \mid i \in V^{\mathcal{M}}(w)\}$ is a singleton

Table 1: Defining conditions for different modal logics.

$\mathsf{NOM} = \{n_1, n_2, \ldots\}$ and $\mathsf{REL} = \{r_1, r_2, \ldots\}$ are mutually disjoint, countable infinite sets. Table 1 introduces a number of closed model classes by means of their defining conditions.

Observe that $\mathcal{P}^{\mathsf{K}}_m$ is true for a model $\mathcal{M}$ if every successor of $w^{\mathcal{M}}$ is identical to $\mathcal{M}$ except perhaps on its point of evaluation. We call m a *relational modality* when it is interpreted in $\mathcal{C}^{\mathsf{K}}_m$, because over this class they behave as classical relational modalities [1].

We can capture different modal operators, like the ones from hybrid logics [39], by choosing the proper class of models. Predicates $\mathcal{P}_{\mathsf{A}}$ and $\mathcal{P}_{@_i}$, for instance, impose conditions on the point of evaluation of the accessible models restricting the evaluation to the class of models where the relation is, respectively, the total relation ($\forall xy.R(x, y)$) and the 'point to all i' relation ($\forall xy.R(x, y) \leftrightarrow i(y)$). Observe that whenever the atom i is interpreted as a singleton set, the 'point to all i' relation becomes the usual 'point to i' relation ($\forall xy.R(x, y) \leftrightarrow y = i$) of hybrid logics. Finally, predicate $\mathcal{P}_{\mathsf{NOM}}$ turns elements of NOM into nominals, i.e., true at a unique element of the domain of the model.

We can express the combination of modalities as the intersection of their respective classes. For example, $\mathcal{C}_{\mathcal{H}(@)}$, the class of models for the hybrid logic $\mathcal{H}(@)$, can be defined as follows

$$\mathcal{C}_{\mathcal{H}(@)} = \mathcal{C}_{\mathsf{Nom}} \cap \mathcal{C}_{@} \cap \mathcal{C}_{\mathsf{Rel}}, \text{ where}$$
$$\mathcal{C}_{@} = \bigcap_{i \in \mathsf{Nom}} \mathcal{C}_{@_i}, \text{ and } \mathcal{C}_{\mathsf{Rel}} = \bigcap_{m \in \mathsf{Rel}} \mathcal{C}^{\mathsf{K}}_m.$$

The crucial characteristic of the coinductive framework is that these different modal operators are captured using the same semantic condition introduced in Definition 21. All the details defining each particular operator are now introduced as properties of the accessibility relation. As a result, a unique notion of bisimulation is sufficient to cover all of them.

Definition 22 (Bisimulations)**.** A *bisimulation* between models $\mathcal{M}$ and $\mathcal{M}'$ is a relation $Z \subseteq \mathrm{Ext}(\mathcal{M}) \times \mathrm{Ext}(\mathcal{M}')$ that satisfies the following conditions:

Root: $\mathcal{M} Z \mathcal{M}'$.

Atomic Harmony: $\mathcal{S} Z \mathcal{S}'$ implies $V^{\mathcal{S}}(w^{\mathcal{S}}) = V^{\mathcal{S}'}(w^{\mathcal{S}'})$, for all $a \in \mathsf{ATOM}$.

Zig: If $\mathcal{S} Z \mathcal{S}'$, then $\mathcal{T} \in R^{\mathcal{S}}(m, w^{\mathcal{S}})$ implies $\mathcal{T} Z \mathcal{T}'$ for some $\mathcal{T}' \in R^{\mathcal{S}'}(m, w^{\mathcal{S}'})$.

Zag: If $\mathcal{S} Z \mathcal{S}'$, then $\mathcal{T}' \in R^{\mathcal{S}'}(m, w^{\mathcal{S}'})$ implies $\mathcal{T} Z \mathcal{T}'$ for some $\mathcal{T} \in R^{\mathcal{S}}(m, w^{\mathcal{S}})$.

We say that two models $\mathcal{M}$ and $\mathcal{M}'$ are bisimilar (notation: $\mathcal{M} \underline{\leftrightarrow} \mathcal{M}'$) if there is a bisimulation Z between them.

The classic result of invariance of modal formulas under bisimulation can easily be proved [1].

Theorem 3. *If $\mathcal{M} \underline{\leftrightarrow} \mathcal{M}'$, then $\mathcal{M} \models \varphi$ iff $\mathcal{M}' \models \varphi$, for all φ.*

This general notion of bisimulation works for every modal logic definable as a closed subclass of Mods. Many well known bisimulations (e.g., for temporal logics, global modalities, hybrid logics, etc.) can be seen as specializations of Definition 22.

3.2 A Generalized Theory of Symmetries

We now extend the theory presented in Section 2 for the basic modal logic to the coinductive framework. We start by adapting the needed definitions.

Definition 23 (Permutation over ALIT)**.** A *permutation of atom literals* is a bijective function $\rho : \mathsf{ALIT} \mapsto \mathsf{ALIT}$. For L a set of atom literals, $\rho(L) = \{\rho(l) \mid l \in L\}$.

As for the basic modal logic case, we will restrict ourselves to work with permutations over a finite set of atom literals, i.e., permutations over the set of atom literals occurring in the formula at hand.

Definition 24 (Consistent Permutation)**.** For $a \in \mathsf{ATOM}$, let $\sim\neg a = a$ and $\sim a = \neg a$. We say that a permutation ρ is *consistent* if for every atom literal l, $\rho(\sim l) = \sim\rho(l)$.

From now on we only consider consistent permutations of atom literals.

Since, in our language, atoms may occur within modalities, e.g., $@_i$, we should take some care when applying permutations to modal formulas. We say that a modality is *indexed by atoms* if its definition depends on the value of an atom. If m is indexed by an atom a we write $m(a)$.

Definition 25 (Permutation of a formula)**.** Let φ be a modal formula. We define the application of a permutation ρ to φ as follows

$$\begin{array}{lcl} \rho(\neg\varphi) & = & \neg\rho(\varphi) \\ \rho(\varphi \vee \psi) & = & \rho(\varphi) \vee \rho(\psi) \\ \rho([m]\varphi) & = & [\rho(m)]\rho(\varphi). \end{array}$$

Where $\rho(m) = \rho(m(a)) = m(\rho(a))$ if m is indexed by a, and $\rho(m) = m$ otherwise.

Definition 26 (Symmetry)**.** Let φ be a modal CNF formula and ρ a consistent permutation. We say that ρ is a *symmetry* of φ if $\varphi = \rho(\varphi)$, when φ is represented using set notation.

Definition 27 (Generated set of atom literals)**.** Let $\mathcal{M} = \langle w, W, R, V\rangle$ be a model. For every $v \in W$, we define its *generated set of atom literals* (notation: $L_{V(v)}$) as $V(v) \cup \{\neg a \mid a \in \mathsf{ATOM}\backslash V(v)\}$. $L_{V(v)}$ is a consistent and complete set of atom literals.

Now we are ready to generalize the results obtained for the basic modal logic to the coinductive framework. We begin with the notion of ρ-simulation.

Definition 28 (ρ-simulation)**.** Let ρ be a permutation. A *ρ-simulation* between models $\mathcal{M}$ and $\mathcal{M}'$ is a relation $Z \subseteq \mathrm{Ext}(\mathcal{M}) \times \mathrm{Ext}(\mathcal{M}')$ that satisfies the following conditions:

Root: $\mathcal{M} Z \mathcal{M}'$.

ρ-Harmony: $\mathcal{S} Z \mathcal{S}'$ implies $l \in L_{V^{\mathcal{S}}(w^{\mathcal{S}})}$ iff $\rho(l) \in L_{V^{\mathcal{S}'}(w^{\mathcal{S}'})}$.

Zig: If $\mathcal{S} Z \mathcal{S}'$, then $\mathcal{T} \in R^{\mathcal{S}}(m, w^{\mathcal{S}})$ implies $\mathcal{T} Z \mathcal{T}'$ for some $\mathcal{T}' \in R^{\mathcal{S}'}(\rho(m), w^{\mathcal{S}'})$.

Zag: If $\mathcal{S} Z \mathcal{S}'$, then $\mathcal{T}' \in R^{\mathcal{S}'}(m, w^{\mathcal{S}'})$ implies $\mathcal{T} Z \mathcal{T}'$ for some $\mathcal{T} \in R^{\mathcal{S}}(\rho^{-1}(m), w^{\mathcal{S}})$.

We say that two models $\mathcal{M}$ and $\mathcal{M}'$ are ρ-similar (notation: $\mathcal{M} \underline{\rightarrow}_{\rho} \mathcal{M}'$) if there is a ρ-simulation Z between them.

Notice that the definition of ρ-simulation takes into account the fact that there exist modalities that are indexed by atoms by considering the permutation when accessing successors, e.g., $R^{\mathcal{S}'}(\rho(m), w^{\mathcal{S}'})$. Also, as for the basic modal logic case, the relation $\underline{\rightarrow}_{\rho}$ is not symmetric.

As expected, ρ-simulations preserve validity of permutations of formulas.

Proposition 4. *Let ρ be a consistent permutation, φ a modal formula and $\mathcal{M} = \langle w, W, R, V\rangle$, $\mathcal{M}' = \langle w', W', R', V'\rangle$ models such that $\mathcal{M} \underline{\rightarrow}_{\rho} \mathcal{M}'$. Then $\mathcal{M} \models \varphi$ iff $\mathcal{M}' \models \rho(\varphi)$.*

Proof. The proof is by induction on the syntactic structure of φ.

Base Case:

$[\varphi = a]$: $\mathcal{M} \models a$ iff $a \in V(w)$ iff $a \in L_{V(w)}$ iff, by **ρ-Harmony**, $\rho(a) \in L_{V'(w')}$ iff $\mathcal{M}' \models \rho(a)$.

Inductive Step:

$[\varphi = [m]\psi]$: $\mathcal{M} \models [m]\psi$ iff $\mathcal{N} \models \psi$ for all $\mathcal{N} \in R(m, w)$. Since $\mathcal{M} \underline{\rightarrow}_{\rho} \mathcal{M}'$, by **Zig**, for all $\mathcal{N}$ exist $\mathcal{N}'$ such that $\mathcal{N} \underline{\rightarrow}_{\rho} \mathcal{N}'$ and $\mathcal{N}' \in R'(\rho(m), w')$. By the inductive hypothesis, $\mathcal{N}' \models \rho(\psi)$ for all $\mathcal{N}' \in R'(\rho(m), w')$ iff $\mathcal{M}' \models [\rho(m)]\rho(\psi)$ iff $\mathcal{M}' \models \rho([m]\psi)$. The converse follows by using **Zag** and the inductive hypothesis.

The remaining cases follow by induction directly. □

Now we define how to apply permutations to coinductive models.

Definition 29 (Permutation of a model)**.** Let ρ be a permutation and $\mathcal{M} = \langle w, W, R, V\rangle$ a model. Then $\rho(\mathcal{M}) = \langle w, W, R', V'\rangle$, where, $V'(v) = \rho(L_{V(v)}) \cap \mathsf{ATOM}$ for all $v \in W$, and $R'(m, v) = \{\rho(\mathcal{N}) \mid \mathcal{N} \in R(\rho(m), v)\}$ for all $m \in \mathsf{MOD}$ and $v \in W$. For M a class of models, $\rho(M) = \{\rho(\mathcal{M}) \mid \mathcal{M} \in M\}$.

Besides modifying the valuation, permuting a coinductive modal model involves propagating the permutation to all accessible models.

As before, $\mathcal{M}$ and $\rho(\mathcal{M})$ are ρ-similar.

Proposition 5. *Let ρ be a consistent permutation and $\mathcal{M}$ a model. Then $\mathcal{M} \underline{\rightarrow}_{\rho} \rho(\mathcal{M})$.*

Proof. We show that the relation $Z = \{(\mathcal{N}, \rho(\mathcal{N})) \mid \mathcal{N} \in \mathrm{Ext}(\mathcal{M})\}$ is a ρ-simulation between $\mathcal{M}$ and $\rho(\mathcal{M})$.

[ρ-Harmony:] From the definition of $\rho(\mathcal{M})$, $L_{V'(v)} = \rho(L_{V(v)})$ for all $v \in W$, hence if $l \in L_{V(v)}$ then $\rho(l) \in \rho(L_{V(v)})$. Moreover, $\rho(L_{V(v)})$ is a complete set of literals since $L_{V(v)}$ is a complete set of literals and ρ is a consistent permutation, and hence the converse also follows.

[Root, Zig and Zag:] Straightforward. □

Proposition 6. *Let ρ be a consistent permutation, $\mathcal{M}$ a model and φ a modal formula. Then $\mathcal{M} \models \varphi$ iff $\rho(\mathcal{M}) \models \rho(\varphi)$.*

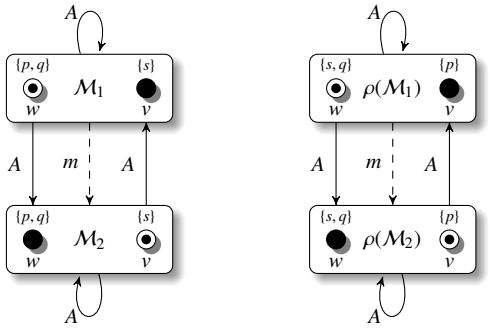
Proof. It follows directly from Proposition 5 and Proposition 4. □

Theorem 4. *Let φ be a modal CNF formula and ρ a symmetry of φ. Then $\mathcal{M} \models \varphi$ iff $\rho(\mathcal{M}) \models \varphi$.*

Proof. It follows from Proposition 6 in the particular case when φ is a modal CNF formula and ρ is a symmetry of φ and hence $\rho(\varphi) = \varphi$. □

Theorem 4 tells us that the symmetries of a formula φ have the same effect on the coinductive framework as for the basic modal logic, i.e., they partition the space of models into equivalence classes in such a way that every equivalence class contains either models satisfying φ or models not satisfying φ. The result applies to any modal logic that can be cast into the coinductive framework. To clarify the implications of Theorem 4, consider the following example.

Example 7. Given $\varphi = (p \vee q \vee r) \wedge (s \vee q \vee r) \wedge (\neg p \vee \neg s) \wedge \langle m \rangle (p \vee s) \wedge [\mathsf{A}] \neg r$ and $\rho = (p\ s)(\neg p\ \neg s)$ a symmetry of φ. We have that $\mathcal{M}_1 \models \varphi$ (Figure 2a), and, by Theorem 4, $\rho(\mathcal{M}_1) \models \varphi$ (Figure 2b).



(a) $\mathcal{M}_1$. (b) $\rho(\mathcal{M}_1)$.

Figure 2: A coinductive model and its permutation.

Now we show that in the coinductive framework we can also use symmetries as an inference mechanism.

Theorem 5. *Let φ and ψ be modal CNF formulas and let ρ be a symmetry of φ. Then $\varphi \models \psi$ iff $\varphi \models \rho(\psi)$.*

Proof. We first show that under the hypothesis of the theorem the following claim holds
Claim: $\mathsf{Mods}(\varphi) = \rho(\mathsf{Mods}(\varphi))$.

The argument is as in the Claim in Theorem 2 but using Theorem 4.

Now, we have to prove that $\varphi \models \psi$ iff $\varphi \models \rho(\psi)$. $\varphi \models \psi$ iff $\mathsf{Mods}(\varphi) \models \psi$, iff, using Proposition 6, $\rho(\mathsf{Mods}(\varphi)) \models \rho(\psi)$. Since ρ is a symmetry of φ, by the Claim above, $\rho(\mathsf{Mods}(\varphi)) \models \rho(\psi)$ iff $\mathsf{Mods}(\varphi) \models \rho(\psi)$, iff $\varphi \models \rho(\psi)$. □

Notice that the notion of ρ-simulation in coinductive modal models is general enough to be applicable to a wide range of modal logics. However, our definition of ρ-simulation makes no assumption about the models being in the same class. Consider, for example, a model $\mathcal{M} \in \mathcal{C}_{\mathcal{H}(@)}$ and an arbitrary permutation $\rho = (i\ p)(\neg i\ \neg p)$ for $i \in \mathsf{NOM}$, $p \in \mathsf{PROP}$. By the defining condition $\mathcal{C}_{\mathcal{H}(@)}$, nominals in $\mathcal{M}$ are true at a unique element in the domain, but this does not necessary hold for $\rho(\mathcal{M})$, and hence $\rho(\mathcal{M})$ might not be in $\mathcal{C}_{\mathcal{H}(@)}$.

However, when working with the symmetries of a formula, this is not an issue, and we guarantee that every model $\mathcal{M}$ and its symmetric $\rho(\mathcal{M})$ are in the same model class. This is so since a symmetry of a formula is, implicitly, a *typed* permutation: it only maps symbols in such a way, that the resulting formula is a formula of the language. If this is not the case, either the formula at hand is not in the language of the logic, or the permutation is not a symmetry.

3.3 Layered Permutations

We now present the notion of *layered permutations* that, in cases where the modal language has a tree model property, enables us to develop a more flexible notion of symmetry.

First let us define the tree model property [35] for coinductive modal models.

Definition 30 (Paths in a model)**.** Given a model $\mathcal{M}$, a (finite) *path rooted at* $\mathcal{M}$ is a sequence $\pi = (\mathcal{M}_0, m_1, \mathcal{M}_1, \ldots, m_k, \mathcal{M}_k)$, for $m_i \in \mathsf{MOD}$ where $\mathcal{M}_0 = \mathcal{M}$, $k \geq 0$, and $\mathcal{M}_i \in R(m_i, w^{\mathcal{M}_{i-1}})$ for $i = 1, \ldots, k$. For a path $\pi = (\mathcal{M}_0, m_1, \mathcal{M}_1, \ldots, m_k, \mathcal{M}_k)$ we define $\mathit{first}(\pi) = \mathcal{M}_0$, $\mathit{last}(\pi) = \mathcal{M}_k$, and $\mathit{length}(\pi) = k$. For a path $\pi = (\mathcal{M}_0, m_1, \mathcal{M}_1, \ldots, m_k, \mathcal{M}_k)$, a model $\mathcal{N}$ and a modality $m \in \mathsf{MOD}$, such that $\mathcal{N} \in R(m, w^{\mathcal{M}_k})$, by $\pi m \mathcal{N} = (\mathcal{M}_0, m_1, \mathcal{M}_1, \ldots, m_k, \mathcal{M}_k, m, \mathcal{N})$ we denote the extension of π by $\mathcal{N}$ through m. We denote the set of all paths rooted at $\mathcal{M}$ as $\Pi[\mathcal{M}]$.

Definition 31 (Coinductive trees)**.** We say that a coinductive model is a tree if it has a unique path to every reachable model (every model in $\mathrm{Ext}(\mathcal{M})$). Formally we can define the class of all coinductive tree models, $\mathcal{C}_{\mathit{Tree}}$, with the following defining condition:

$$\mathcal{C}_{\mathit{Tree}} := P_{\mathit{Tree}}(\mathcal{M}) \Longleftrightarrow \mathit{last} : \Pi[\mathcal{M}] \mapsto \mathrm{Ext}(\mathcal{M}) \text{ is bijective.}$$

The *unraveling* construction (in its version for coinductive modal models) shown below always defines a model in $\mathcal{C}_{\mathit{Tree}}$.

Definition 32 (Model unraveling)**.** Given a model $\mathcal{M} = \langle w, W, R, V\rangle$, the *unraveling* of $\mathcal{M}$, (notation: $\mathcal{T}(\mathcal{M})$), is the rooted coinductive modal model $\mathcal{T}(\mathcal{M}) = \langle (\mathcal{M}), \Pi[\mathcal{M}], R', V'\rangle$ where $V'(\pi) = V(w^{\mathit{last}(\pi)})$, for all $\pi \in \Pi[\mathcal{M}]$, $R'(m, \pi) = \{\langle \pi', \Pi[\mathcal{M}], R', V'\rangle \mid \exists \mathcal{N} \in \mathsf{Mods}_W$ such that $\pi' = \pi m \mathcal{N}\}$ for $m \in \mathsf{MOD}, \pi \in \Pi[\mathcal{M}]$.

It is easy to verify that given a model $\mathcal{M}$, its unraveling $\mathcal{T}(\mathcal{M})$ is a tree ($\mathcal{T}(\mathcal{M}) \in \mathcal{C}_{Tree}$) and, as expected, $\mathcal{M}$ and $\mathcal{T}(\mathcal{M})$ are bisimilar.

In what follows, we use trees to define a more flexible type of symmetries that we call *layered symmetries*. The following gives a sufficient condition that enables us to work with layered symmetries.

Definition 33 (Tree witnesses)**.** We say that a class $\mathcal{C}$ of models has *tree witnesses* if for every model $\mathcal{M} \in \mathcal{C}$ exists a tree model $\mathcal{T} \in \mathcal{C}$ such that $\mathcal{M} \underline{\leftrightarrow} \mathcal{T}$.

From Definition 33 it follows that a class of models $\mathcal{C}$ closed under unravelings ($\mathcal{T}(\mathcal{M}) \in \mathcal{C}$ for all $\mathcal{M} \in \mathcal{C}$) has tree witnesses.

Logics defined over classes having tree witnesses have an interesting property: there is a direct correlation between the syntactical modal depth of the formula and the depth of a tree model satisfying it. In tree models, a notion of layer is induced by the depth (distance from the root) of the nodes in the model. Similarly, in modal formulas, a notion of layer is induced by the nesting of the modal operators. A consequence of this correspondence is that literals occurring at different layers of the formula are semantically independent of each other [40], i.e., at different layers the same literal can be assigned a different truth value.

The independence between literals at different layers enables us to give a more flexible notion of a permutation that we call *layered permutation*.

Definition 34 (Layered permutation)**.** A *layered permutation* $\bar{\rho}$ is a, possibly empty, finite sequence of permutations $\langle \rho_1, \ldots, \rho_k \rangle$. Let $|\langle \rho_1, \ldots, \rho_k \rangle| = k$ be the length of $\bar{\rho}$ ($\langle\rangle$ has length 0). For $1 \leq i \leq n$, $\bar{\rho}_i$ is the sub-sequence that starts from the i^{th} element of $\bar{\rho}$ (in particular, $\bar{\rho}_i = \langle\rangle$ for $i > |\bar{\rho}|$ and $\bar{\rho}_1 = \bar{\rho}$). Let $\bar{\rho} = \langle \rho_1, \ldots, \rho_k \rangle$ then $\bar{\rho}(i)$ is ρ_i if $1 \leq i \leq |\bar{\rho}|$ and $\bar{\rho}(i) = \rho_{Id}$ otherwise, for ρ_{Id} the identity permutation.

A layered permutation is *consistent* if all its permutations are consistent.

Let φ be a modal formula. We define the application of a layered permutation $\bar{\rho}$ to φ as follows. Define $\langle\rangle(\varphi) = \varphi$, and for $|\bar{\rho}| \geq 1$:

$$\begin{array}{lcl} \bar{\rho}(a) & = & \bar{\rho}(1)(a) \\ \bar{\rho}(\neg\varphi) & = & \neg\bar{\rho}(\varphi) \\ \bar{\rho}(\varphi \vee \psi) & = & \bar{\rho}(\varphi) \vee \bar{\rho}(\psi) \\ \bar{\rho}([m]\varphi) & = & [\bar{\rho}(1)(m)]\bar{\rho}_2(\varphi). \end{array}$$

Where $\bar{\rho}(1)(m) = \bar{\rho}(1)(m(a)) = m(\bar{\rho}(1)(a))$ if m is indexed by a, and $\bar{\rho}(1)(m) = m$ otherwise.

A layered permutation $\bar{\rho}$ is a symmetry of a modal CNF formula φ if $\bar{\rho}(\varphi) = \varphi$, when φ is represented using set notation.

Notice that applying a layered permutation to a formula is well defined even if the modal depth of the formula is greater than the size of the layered permutation.

Layered permutations let us use a different permutation at each modal depth, which can capture symmetries that non layered permutations cannot define.

Example 8. Consider the formula $\varphi = (p \vee \Box(p \vee \neg r)) \wedge (\neg q \vee \Box(\neg p \vee r))$. If we only consider non-layered permutations then φ has no symmetry. However, the layered permutation $\langle \rho_1, \rho_2 \rangle$ generated by $\rho_1 = (p\ \neg q)(\neg p\ q)$ and $\rho_2 = (p\ \neg p)(r\ \neg r)$ is a symmetry of φ.

From now on we can mostly repeat the work we did in the previous section to arrive to results similar to Theorems 4 and 5 but involving layered permutation, with one caveat: the obvious extension of the notion of permuted model $\rho(\mathcal{M})$ to layered permutations is ill defined if $\mathcal{M}$ is not a tree. Hence, we need the additional requirement that the class C of models has tree witnesses for the result to go through.

Definition 35 (Layered permutation of a model)**.** Let $\bar{\rho}$ be a layered permutation and $\mathcal{M} = \langle w, W, R, V\rangle$ a tree model. Then $\bar{\rho}(\mathcal{M}) = \langle w, W, R', V'\rangle$, where $V'(v) = \bar{\rho}(1)(L_{V(v)}) \cap \mathsf{ATOM}$ for all $v \in W$, and $R'(m, v) = \{\bar{\rho}_2(\mathcal{N}) \mid \mathcal{N} \in R(\bar{\rho}(1)(m), v)\}$ for all $m \in \mathsf{MOD}$ and $v \in W$. For M a class (or a set) of tree models, $\bar{\rho}(M) = \{\bar{\rho}(\mathcal{M}) \mid \mathcal{M} \in M\}$.

We can now extend the notion of ρ-simulation to layered permutations.

Definition 36 ($\bar{\rho}$-simulation)**.** Let $\bar{\rho}$ be a layered permutation, a *$\bar{\rho}$-simulation* between tree models $\mathcal{M} = \langle w, W, R, V\rangle$ and $\mathcal{M}' = \langle w', W', R', V'\rangle$ is a family of relations $Z_{\bar{\rho}_i} \subseteq \mathrm{Ext}(\mathcal{M}) \times \mathrm{Ext}(\mathcal{M}')$, $1 \leq i$, that satisfies the following conditions:

Root: $\mathcal{M} Z_{\bar{\rho}_1} \mathcal{M}'$.

$\bar{\rho}$-Harmony: $\mathcal{S} Z_{\bar{\rho}_i} \mathcal{S}'$ implies $l \in L_{V^{\mathcal{S}}(w^{\mathcal{S}})}$ iff $\bar{\rho}_i(1)(l) \in L_{V^{\mathcal{S}'}(w^{\mathcal{S}'})}$.

Zig: If $\mathcal{S} Z_{\bar{\rho}_i} \mathcal{S}'$, then $\mathcal{T} \in R^{\mathcal{S}}(m, w^{\mathcal{S}})$ implies $\mathcal{T}' Z_{\bar{\rho}_{i+1}} \mathcal{T}'$ for some $\mathcal{T}' \in R^{\mathcal{S}'}(\bar{\rho}_i(1)(m), w^{\mathcal{S}'})$.

Zag: If $\mathcal{S} Z_{\bar{\rho}_i} \mathcal{S}'$, then $\mathcal{T}' \in R^{\mathcal{S}'}(m, w^{\mathcal{S}'})$ implies $\mathcal{T} Z_{\bar{\rho}_{i+1}} \mathcal{T}'$ for some $\mathcal{T} \in R^{\mathcal{S}}(\bar{\rho}_i(1)^{-1}(m), w^{\mathcal{S}})$.

We say that two models $\mathcal{M}$ and $\mathcal{M}'$ are $\bar{\rho}$-similar (notation: $\mathcal{M} \underline{\rightarrow}_{\bar{\rho}} \mathcal{M}'$), if there is a $\bar{\rho}$-simulation between them.

Definition 36 does not make any assumption about the size of the layered permutation. It is well defined even if the layered permutation at hand is the empty sequence. In that case, it just behaves as the identity permutation at each layer and the relation defines a bisimulation between the models.

Theorem 6. *Let φ be a modal CNF formula and $\bar{\rho}$ a symmetry of φ. Then $\mathcal{M} \models \varphi$ iff $\bar{\rho}(\mathcal{M}) \models \varphi$.*

Proof. The proof is similar to the one for Theorem 4 but using layered permutations. □

Theorem 7. *Let φ and ψ be modal formulas and let $\bar{\rho}$ be a consistent layered permutation, and let C be a class of models with tree witnesses. If $\bar{\rho}$ is a symmetry of φ then for any ψ we have that $\varphi \models_C \psi$ iff $\varphi \models_C \bar{\rho}(\psi)$.*

Proof. We first show that under the hypothesis of the theorem the following two properties hold.

Claim 1**:** $\mathsf{Mods}_{C \cap C_{Tree}}(\varphi) = \bar{\rho}(\mathsf{Mods}_{C \cap C_{Tree}}(\varphi))$.

The argument is the same as for the Claim in Theorem 5 but using layered permutations.

Claim 2: $\mathsf{Mods}_C(\varphi) \models_C \varphi$ iff $\mathsf{Mods}_{C\cap C_{Tree}}(\varphi) \models_C \varphi$.

The left-to-right direction follows trivially from the fact that $\mathsf{Mods}_{C\cap C_{Tree}}(\varphi) \subseteq \mathsf{Mods}_C(\varphi)$. For the other direction, assume $\mathsf{Mods}_{C\cap C_{Tree}}(\varphi) \models_C \varphi$ and $\mathsf{Mods}_C(\varphi) \not\models_C \varphi$. Then there exists $\mathcal{M} \in \mathsf{Mods}_C(\varphi)$ such that $\mathcal{M} \not\models_C \varphi$. But $\mathcal{M} \underline{\leftrightarrow} \mathcal{T}$, and $\mathcal{T} \in \mathsf{Mods}_{C\cap C_{Tree}}(\varphi)$. Hence $\mathcal{T} \not\models_C \varphi$ which contradicts our assumption.

It remains to prove that $\varphi \models_C \psi$ if and only if $\varphi \models_C \bar{\rho}(\psi)$. By definition, $\varphi \models_C \psi$ iff $\mathsf{Mods}_C(\varphi) \models_C \psi$, iff, by Claim 2, $\mathsf{Mods}_{C\cap C_{Tree}}(\varphi) \models_C \psi$, iff, using a layered version of Prop. 6, $\bar{\rho}(\mathsf{Mods}_{C\cap C_{Tree}}(\varphi)) \models_C \bar{\rho}(\psi)$. Since $\bar{\rho}$ is a symmetry of φ, by Claim 1, $\bar{\rho}(\mathsf{Mods}_{C\cap C_{Tree}}(\varphi)) \models_C \bar{\rho}(\psi)$ iff $\mathsf{Mods}_{C\cap C_{Tree}}(\varphi) \models_C \bar{\rho}(\psi)$, iff, by Claim 2, $\mathsf{Mods}_C(\varphi) \models_C \bar{\rho}(\psi)$ then $\varphi \models_C \bar{\rho}(\psi)$. □

4 Symmetry Detection for Modal Logics

We now focus on detecting symmetries for modal formulas. Unless stated otherwise, we work with modal CNF formulas considering them as sets of sets (see Section 2), although we might write them as usual for the sake of clarity.

Let us start by presenting the needed definitions.

Definition 37 (atoms and indexes functions)**.** Let φ be a modal CNF formula, C a clause and $a \in \mathsf{ATOM}$. The function atoms that returns the set of atoms occurring in φ is defined as follows:

$$\begin{array}{lcl}
\mathsf{atoms}(a) & = & \{a\} \\
\mathsf{atoms}(\neg a) & = & \{a\} \\
\mathsf{atoms}([m]C) & = & \mathsf{atoms}(C) \\
\mathsf{atoms}([m(a)]C) & = & \{a\} \cup \mathsf{atoms}(C) \\
\mathsf{atoms}(\neg[m]C) & = & \mathsf{atoms}(C) \\
\mathsf{atoms}(\neg[m(a)]C) & = & \{a\} \cup \mathsf{atoms}(C) \\
\mathsf{atoms}(C) & = & \bigcup_{l\in C} \mathsf{atoms}(l) \\
\mathsf{atoms}(\varphi) & = & \bigcup_{C\in\varphi} \mathsf{atoms}(C).
\end{array}$$

By $\mathsf{atoms}(\varphi, i)$ we denote the set of atoms occurring in φ at modal depth i. We define $\mathsf{lits}(\varphi) = \{\neg a \mid a \in \mathsf{atoms}(\varphi)\} \cup \mathsf{atoms}(\varphi)$.

The function indexes that returns the set of atoms indexing a modality in φ is defined as follows:

$$\begin{array}{lcl}
\mathsf{indexes}(a) & = & \emptyset \\
\mathsf{indexes}(\neg a) & = & \emptyset \\
\mathsf{indexes}([m]C) & = & \mathsf{indexes}(C) \\
\mathsf{indexes}(\neg[m]C) & = & \mathsf{indexes}(C) \\
\mathsf{indexes}([m(a)]C) & = & \{a\} \cup \mathsf{indexes}(C) \\
\mathsf{indexes}(\neg[m(a)]C) & = & \{a\} \cup \mathsf{indexes}(C) \\
\mathsf{indexes}(C) & = & \bigcup_{l\in C} \mathsf{indexes}(l) \\
\mathsf{indexes}(\varphi) & = & \bigcup_{C\in\varphi} \mathsf{indexes}(C).
\end{array}$$

To detect the symmetries of a formula we will construct an undirected labeled finite graph such that the automorphism group of the graph is isomorphic to the symmetry group of the formula.

Definition 38. An undirected, labeled finite multi-graph is a structure $G = \langle N, \{E_i\}_{i\in I}, l\rangle$ where N, the set of *nodes*, is a finite non-empty set; E_i, the *edge-relations*, are symmetric relations over $N \times N$; and $l : N \mapsto L$ is the *labeling function* mapping nodes into a finite set L of labels.

To construct the graph we need to identify the clauses in a formula and label the nodes correctly to avoid detecting automorphisms in the graph that do not correspond to symmetries in the formula.

Definition 39 (Identification function)**.** Let φ be a modal CNF formula. An *identification function* $(\_)^{id} : \mathsf{clauses}(\varphi) \mapsto I$ is an injective function that assigns to each clause in φ a unique identifier over an arbitrary set I. We define $\mathsf{ids}(\varphi) = \{C^{id} \mid C \in \mathsf{clauses}(\varphi)\}$.

Definition 40 (Admissible labeling)**.** Let φ be a modal CNF formula. An *admissible labeling* is a function $l : \mathsf{lits}(\varphi) \cup \mathsf{ids}(\varphi) \mapsto L$ mapping literals and clause identifiers of φ to a finite set of labels L, such that the following holds:

1. All atom literals have the same label.
2. Given C and D, clauses at modal depth 0, then $l(C^{id}) = l(D^{id})$.
3. Given $[m]C$ and $[m]D$ then $l(C^{id}) = l(D^{id})$.
4. Given $\neg[m]C$ and $\neg[m]D$ then $l(C^{id}) = l(D^{id})$.
5. Given $[m]C$ and $\neg[m]D$ then $l(C^{id}) \neq l(D^{id})$.
6. Given $[m(a)]C$ and $[m(b)]D$ then $l(C^{id}) = l(D^{id})$.
7. Given $\neg[m(a)]C$ and $\neg[m(b)]D$ then $l(C^{id}) = l(D^{id})$.
8. Given $[m(a)]C$ and $\neg[m(a)]D$ then $l(C^{id}) \neq l(D^{id})$.
9. The sets of labels associated to atom literals, clauses at modal depth 0 and modal literals, are pairwise disjoint, i.e., an atom literal, a clause at modal depth 0 and a modal literal do not share a label.

In what follows we call *global permutation (symmetry)* a permutation (symmetry) as defined by Definition 23, and *layered permutation (symmetry)* a layered permutation (symmetry) as defined by Definition 34.

We now present two graph constructions for detecting global and layered symmetries, respectively, in modal CNF formulas. Both constructions are based on the MIN3C construction for propositional CNF formulas [11]. Unlike MIN3C, in our graphs we use two types of edges, coloring is more complex since we have to deal with different modalities, and we fix the representation of binary clauses and Boolean consistency by using a vertex and two edges to model binary clauses and an edge between the positive literal and negative literal vertices to model Boolean consistency.

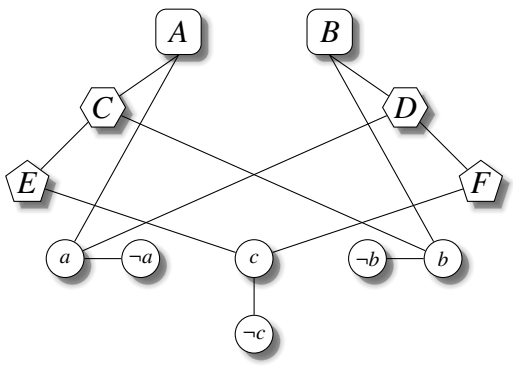
Definition 41 (Global graph construction)**.** Let φ be a modal CNF formula. Let l be an admissible labeling and let id be an identification function.

The graph $GG^{id,l}(\varphi) = (N, E_1, E_2, l)$ corresponding to φ, l and id is the smallest labeled graph satisfying the following conditions:

1. For each $a \in \mathsf{atoms}(\varphi)$:
   (a) There are nodes a and $\neg a$ in N with labels $l(a)$ and $l(\neg a)$.
   (b) There is an edge between a and $\neg a$ in E_1.
2. For each clause C at modal depth 0 there is a node C^{id} in N with label $l(C^{id})$.
3. For each atom literal k occurring as a disjunct in a clause C, there is an edge between C and k in E_1.
4. For each modal literal $[m]D$ ($\neg[m]D$) occurring as a disjunct in a clause C:
   (a) There is a node D^{id} in N with label $l(D^{id})$.
   (b) There is an edge between C^{id} and D^{id} in E_1.
   (c) If m is indexed by an atom variable a, then there is an edge between D^{id} and a in E_2.

$GG^{id,l}(\varphi)$ has $2 \times |\mathsf{atoms}(\varphi)| + |\mathsf{clauses}(\varphi)|$ nodes.

Example 9. Consider the formula $\varphi = (a \vee [m](b \vee \neg[m]c)) \wedge (b \vee [m](a \vee \neg[m]c))$. Figure 3 shows its associated labeled graph $GG^{id,l}(\varphi)$. Notice that in this case there is no modality indexed by an atom variable and therefore, $E_2 = \emptyset$. Labels are represented by shapes in the figure and assigned according to the conditions in Definition 40. Thus, clauses, A and B, which are clauses at modal depth 0, have the same label (square) by condition 2; clauses C, D have the same label (hexagon) by condition 3; clauses E, F have the same label (pentagon) by condition 4; and all atom literals have the same label (circle) by condition 1.



$A = (a \vee [m](b \vee \neg[m]c))$
$B = (b \vee [m](a \vee \neg[m]c))$
$C = [m](b \vee \neg[m]c)$
$D = [m](a \vee \neg[m]c)$
$E = \neg[m]c$
$F = \neg[m]c$

Figure 3: Global graph construction example.

The graph has one non-trivial automorphism $\pi = (A\ B)(C\ D)(E\ F)(a\ b)\ (\neg a\ \neg b)$ which corresponds to the symmetry $\rho = (a\ b)(\neg a\ \neg b)$ of φ.

We now show that each symmetry of the formula φ corresponds to an automorphism of the graph $GG^{id,l}(\varphi)$.

Proposition 7. *Let φ be a modal CNF formula and $GG^{id,l}(\varphi) = (N, E_1, E_2, l)$ the graph corresponding to φ. For each symmetry ρ of φ there exists an automorphism π_ρ of $GG^{id,l}(\varphi)$.*

Proof. We have to prove that given a symmetry ρ of φ, we can build an automorphism π_ρ of $GG^{id,l}(\varphi)$. Let us define π_ρ as follows:

$$\pi_\rho(x) = \begin{cases} \rho(x) & \text{if } x \in \mathsf{lits}(\varphi) \\ \rho(C)^{id} & \text{if } x = C^{id} \in \mathsf{ids}(\varphi). \end{cases}$$

Since ρ is a symmetry, π_ρ is a well defined bijection. Moreover, π_ρ is a permutation over $N = \mathsf{lits}(\varphi) \cup \mathsf{ids}(\varphi)$. We have to prove the following properties for a an atom literal and C, D clauses in φ:

1. $(a, \sim a) \in E_1$ iff $(\pi_\rho(a), \sim\pi_\rho(a)) \in E_1$.

   $(\rightarrow)$: If $(a, \sim a) \in E_1$ then a or $\sim a$ occur in φ, and since ρ is a symmetry of φ, $\rho(a)$ or $\sim\rho(a)$ occur in φ, and therefore, by construction $(\rho(a), \sim\rho(a)) = (\pi_\rho(a), \sim\pi_\rho(a)) \in E_1$.

   $(\leftarrow)$: Same argument as before.

2. $(a, C^{id}) \in E_1$ iff $(\pi_\rho(a), \pi_\rho(C^{id})) \in E_1$.

   $(\rightarrow)$: Assume $(a, C^{id}) \in E_1$, then a occurs as a disjunct in the clause C. Since ρ is a symmetry of φ, $\rho(a)$ occurs as a disjunct in $\rho(C)$, and both occur in φ. Then $(\rho(a), \rho(C)^{id}) = (\pi_\rho(a), \pi_\rho(C^{id})) \in E_1$.

   $(\leftarrow)$: Assume $(\pi_\rho(a), \pi_\rho(C^{id})) \in E_1$, then $\pi_\rho(a) = \rho(a)$ occurs a disjunct in $\rho(C)$. Since ρ is a symmetry of φ, a occurs as disjunct in C and both occur in φ, then $(a, C^{id}) \in E_1$.

3. $(C^{id}, D^{id}) \in E_1$ iff $(\pi_\rho(C^{id}), \pi_\rho(D^{id})) \in E_1$.

   $(\rightarrow)$: Assume $(C^{id}, D^{id}) \in E_1$, then either C or D occurs as a modal literal within the other. Assume that D occurs as a modal literal in C, then, $C = \{\ldots, [m]D, \ldots\}$. Since ρ is a symmetry of φ, $\rho(C) = \{\ldots, \rho([m]D), \ldots\} = \{\ldots, [\rho(m)]\rho(D)), \ldots\}$ occurs in φ. Then $(\rho(C)^{id}, \rho(D)^{id}) = (\pi_\rho(C^{id}), \pi_\rho(D^{id})) \in E_1$.

   $(\leftarrow)$: It follows by the same argument.

4. $(a, C^{id}) \in E_2$ iff $(\pi_\rho(a), \pi_\rho(C^{id})) \in E_2$.

   $(\rightarrow)$: Assume $(a, C^{id}) \in E_2$, then C occurs as a modal literal indexed by a, i.e., $[m(a)]C$ occurs in φ. Since ρ is a symmetry of φ, $\rho([m(a)]C) = [m(\rho(a))]\rho(C)$ occurs in φ. Hence $(\rho(a), \rho(C)^{id}) = (\pi_\rho(a), \pi_\rho(C^{id})) \in E_2$.

   $(\leftarrow)$: It follows by the same argument.

5. If $\pi_\rho(x) = y$ then $l(x) = l(y)$.

   If follows from the fact that l is an admissible labeling and ρ a symmetry.

□

We show that any automorphism of $GG^{id,l}(\varphi)$ induces a symmetry of φ.

Proposition 8. *Let φ be a modal CNF formula and $GG^{id,l}(\varphi) = (N, E_1, E_2, l)$ the graph corresponding to φ. For each automorphism π of $GG^{id,l}(\varphi)$ there exists a symmetry ρ_π of φ.*

Proof. Let us define $\rho_\pi = \pi \restriction \mathsf{lits}(\varphi)$. To prove that ρ_π is a symmetry of φ, we have to prove the following:

1. ρ_π is a consistent permutation, i.e., $\rho_\pi(\sim a) = \sim\rho_\pi(a)$ for $a \in \mathsf{ALIT}$.

   By construction $(a, \sim a) \in E_1$, since π is an automorphism of $GG^{id,l}(\varphi)$ it follows that $(\pi(a), \pi(\sim a)) \in E_1$ iff $\pi(\sim a) = \sim\pi(a)$, which implies that $\rho_\pi(\sim a) = \sim\rho_\pi(a)$.

2. $\rho_\pi(\varphi) = \varphi$.

   It follows directly from the construction of $GG^{id,l}(\varphi)$ and the fact that l is an admissible labeling which only allows the automorphism to permute nodes having the same label. Therefore, atom literals are permuted with atom literals, clauses at modal depth 0 with clauses at modal depth 0 and modal literals to modal literals of the same modality and polarity.

□

Theorem 8 (Correctness)**.** *Let φ be a modal CNF formula and $GG^{id,l}(\varphi) = (N, E_1, E_2, l)$ the graph corresponding to φ. Then every symmetry ρ of φ corresponds one-to-one to an automorphism π of $GG^{id,l}(\varphi)$.*

Proof. Immediate from Proposition 7 and 8. □

Theorem 8 ensures that the proposed graph construction is correct and, therefore, no spurious symmetry, i.e., automorphisms of the graph that do not correspond to symmetries of the formula, is detected.

Since we developed this construction in the coinductive framework and it makes no assumption about the class of models on which we interpret the formulas, it can be used as a *template* from which to derive graph constructions for concrete modal logics. For some modal logics, e.g., basic modal logic, we can use the algorithm as it is defined. However, for other modal logics, this has to be done carefully. Further constraints might be required to ensure that properties enforced by the particular class of models intended, are respected.

We now extend the graph construction from Definition 41 to detect layered symmetries. Recall that in modal logics defined over classes having tree witnesses, literals occurring at different modal depths can be considered independently. Therefore, we can define a layered permutation with a different permutation at each modal depth (see Definition 34).

Definition 42 (Layered graph construction)**.** Let φ be a modal CNF formula. Let l be an admissible labeling and let id be an identification function.

The graph $LG^{id,l}(\varphi) = (N, E_1, E_2, l')$ corresponding to φ, l and id is the smallest labeled graph satisfying the following conditions:

1. The labeling function l', that extends the labeling function l to work with pairs in $\mathsf{lits}(\varphi)\times\mathbb{N}$, is defined as:
$$\text{ł}'(x) = \begin{cases} \text{ł}(y) & \text{if } x = (y, i) \\ \text{ł}(x) & \text{otherwise .} \end{cases}$$

2. For each atom variable $a \in \mathsf{indexes}(\varphi)$:
   (a) There are nodes a and $\neg a$ in N with labels $l'(a)$ and $l'(\neg a)$.
   (b) There is an edge between a and $\neg a$ in E_1.

3. For each atom variable $a \in (\mathsf{atoms}(\varphi, i)\backslash\mathsf{indexes}(\varphi))$ with $0 \leq i \leq \mathsf{md}(\varphi)$:
   (a) There are nodes (a, i) and $(\neg a, i)$ in N with labels $l'((a, i))$ and $l'((\neg a, i))$.
   (b) There is an edge between (a, i) and $(\neg a, i)$ in E_1.

4. For each clause C at modal depth 0 there is a node C^{id} in N with label $l'(C^{id})$.

5. For each atom literal k occurring as a disjunct in a clause C at modal depth i, there is an edge between C and k in E_1, if $k \in \mathsf{indexes}(\varphi)$, or between C and (k, i) in E_1, otherwise.

6. For each modal literal $[m]D$ ($\neg[m]D$) occurring as a disjunct in a clause C:
   (a) There is a node D^{id} in N with label $l'(D^{id})$.
   (b) There is an edge between C^{id} and D^{id} in E_1.
   (c) If m is indexed by an atom variable a, then there is an edge between D^{id} and a in E_2.

$LG^{id,l}(\varphi)$ has $2 \times \mathsf{indexes}(\varphi) + 2 \times |\mathsf{atoms}(\varphi)| \times (\mathsf{md}(\varphi) + 1) + |\mathsf{clauses}(\varphi)|$ nodes.

This construction differs from Definition 41 in the way literals are handled since it treats differently the atoms indexing modalities and atoms not indexing modalities. Atoms indexing modalities are considered "global", i.e., just a pair of nodes are added for this type of atoms (see item 2). On the other hand, for atoms not indexing modalities, if the same literal occurs at different modal depths, the construction treats these occurrences independently adding distinct literal nodes (see item 3) including the modal depth information, which we will use later to build the layered permutation corresponding to the formula.

Example 10. Consider the following modal CNF formula $\varphi = (\neg a \vee [m(i)]b \vee [m(i)]\neg b) \wedge (\neg b \vee [m(j)]a \vee [m(j)]\neg a)$, where $m(i)$ and $m(j)$ represent modalities indexed by the atoms i and j respectively. Figure 4 shows its graph $LG^{id,l}(\varphi)$ (labels are represented by shapes, nodes of the form (l, i) by l_i, edges in E_1 and E_2 by single and double lines respectively).
A set of generators for the automorphism group of $LG^{id,l}(\varphi)$ is:

$$\begin{aligned} \pi_1 &= (C\ D)(b_1\ \neg b_1) \\ \pi_2 &= (E\ F)(a_1\ \neg a_1) \\ \pi_3 &= (A\ B)(C\ E)(D\ F)(a_0\ b_0)(\neg a_0\ \neg b_0)(a_1\ b_1)(\neg a_1\ \neg b_1)(i\ j)(\neg i\ \neg j). \end{aligned}$$

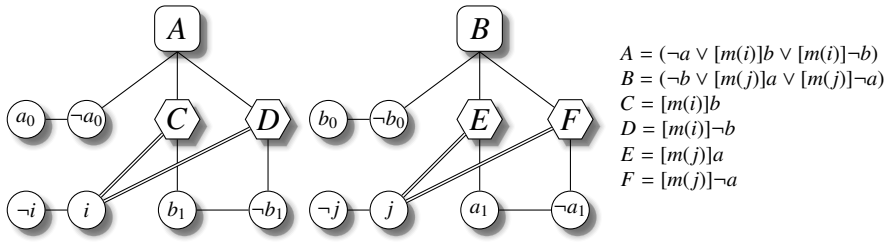


Figure 4: Layered construction example.

To obtain the corresponding layered permutation for φ we build the permutations at each modal depth using the depth information present in each node a_i and adjoining the cycles involving atoms indexing modalities, since they are considered "global permutations". We obtain the following:

$$\bar{\rho}_1 = \langle \rho_{Id}, (b\ \neg b)\rangle$$
$$\bar{\rho}_2 = \langle \rho_{Id}, (a\ \neg a)\rangle$$
$$\bar{\rho}_3 = \langle (a\ b)(\neg a\ \neg b)(i\ j)(\neg i\ \neg j), (a\ b)(\neg a\ \neg b)(i\ j)(\neg i\ \neg j)\rangle,$$

where ρ_{Id} is the identity permutation.

Theorem 9 (Correctness)**.** *Let φ be a modal CNF formula and $LG^{id,l}(\varphi) = (N, E_1, E_2, l)$ the graph corresponding to φ. Then every symmetry $\bar{\rho}$ of φ corresponds one-to-one to an automorphism π of $LG^{id,l}(\varphi)$.*

Proof. It follows from a straightforward generalization of Propositions 7 and 8. □

4.1 Experimental Evaluation

We now empirically evaluate the graph constructions to see how often symmetries appear in modal benchmarks and how hard it is to actually find them.

An evaluation of the effects of using symmetry information in a tableaux algorithm for the basic modal logic is available in [41].

4.1.1 Implementation

For testing purposes, we restrict ourselves to the basic modal logic. Our test set contains 378 instances, distributed in 9 problem classes, from the Logics Workbench Benchmark for *K* (LWB) [42].

The LWB is a structured test set that has been extensively used to test modal provers and therefore it constitutes a good starting point to assess our detection algorithms. It should be mentioned, however, that most of the formulas in this test set became too easy for state of the art provers, even though it still contains instances which remain difficult. An extensive evaluation of the detection algorithms on more challenging test sets is done in [41].

We implemented the global and layered graph constructions into the tool `sy4ncl`. `sy4ncl` is a command line tool, written in Haskell, that takes a basic modal logic formula, builds the

selected graph (global or layered) and outputs it in a suitable format, for further processing by an automorphisms detection tool, along with a mapping from vertices to literals and statistics about the constructed graph.

To search for symmetries we use the graph automorphism detection tool `Bliss` [43]. `Bliss` takes a graph specification and returns a set of generators for the automorphism group of the graph. If the graph has non-trivial automorphisms, we then reconstruct the symmetries of the formula using the mapping generated by `sy4ncl`.

All tests were ran on an Intel Core i7 2.93GHz with 16GB of RAM and a timeout of 10 seconds for both graph construction and symmetry detection. The tools and benchmarks are available at:

`http://cs.famaf.unc.edu.ar/~ezequiel/sml-am`.

4.1.2 Results

Table 2 summarizes the results for the LWB test set using both constructions (global and layered). Columns $\#In$, $\#To$ and $\#Sy$ are the number of instances in the test set, the number of instances that timeouted and the number of instances with at least one non-trivial symmetry, respectively. Columns T_G and T_S are the total time, in seconds, needed to create the graph and search for automorphisms, for **all** the instances, respectively.

	$\#In$	$\#To$	$\#Sy$	T_G	T_S
Global	378	0	135	9.83	1.18
Layered	378	0	208	9.80	1.80

Table 2: Symmetries in the LWB test set.

Table 2 shows that many symmetric instances exists in the LWB test set and that the time required to compute the symmetries (graph time plus search time) is negligible. It also confirms our claim that by using the layered construction we could detect more symmetries. Indeed, with the layered construction, we find 73 more symmetric instances than with the global construction.

Table 3 shows detailed results for the LWB test set. Column M is the median number of detected generators in each problem class. This value gives an approximated idea of how symmetric are the instances in a problem class.

Table 3 shows that the LWB test set presents a behavior that matches our expectations: the existence of symmetries is driven by the codification used in each problem class. Many problem classes (`k_branch`, `k_path`, `k_grz`, `k_ph` and `k_poly`) exhibit many symmetric instances, while others exhibit none (`k_d4`, `k_dum`, `k_t4p`) or few symmetric instances (`k_lin`). Also notice the effect of using the layered construction. For some problem classes (`k_branch`, `k_ph` and `k_poly`) both graph constructions yield practically the same results, with the layered version detecting a few more symmetries per instances than the global version. However, in the `k_path` and `k_grz` classes, differences are more evident.

		Global		Layered	
Class	$\#In$	$\#Sy$	M	$\#Sy$	M
k_branch	42	42	11.5	42	11.5
k_d4	42	0	-	0	-
k_dum	42	0	-	0	-
k_grz	42	2	1	42	4.5
k_lin	42	1	1	1	1
k_path	42	9	1.75	42	34.5
k_ph	42	39	1	39	1
k_poly	42	42	16	42	18
k_t4p	42	0	-	0	-

Table 3: Symmetries in the LWB test set detailed by problem class.

To gain an insight on the size of the resulting graphs, Table 4 provides detailed information for the top 10 largest graphs generated using both constructions. Columns *md*, $\#V$ and $\#Cl$ are the modal depth, the number of propositional variables and the number of clauses (clauses at modal depth 0 plus clauses occurring in modal literals) in the input formula. Columns $\#N$ and $\#E$ are the number of nodes and the number of edges in the resulting graph, respectively. Column $T_{\text{G+S}}$ is the total time, in seconds, required to build the graph and to search for automorphisms in it.

As can be seen in Table 4, the graph building algorithms and the search for automorphisms are efficient, requiring a negligible amount of time even for graphs containing thousands of nodes and edges (e.g., k_branch_p_21). The table also shows that layered graphs are slightly bigger than global graphs due to the duplication of literal nodes occurring at different modal depths.

5 Discussion

We presented the theoretical foundations for exploiting and detecting symmetries in modal logics.

First we focused on the basic modal logic and established two results: that symmetries of a formula partition the model space into equivalence classes containing only models satisfying the formula or only models not satisfying it, and that symmetries can be used as an inference mechanism. The key notion for proving these results is that of ρ-simulation, which can be seen as a relaxed form of bisimulation that takes into account the effects of a permutation ρ.

We extended the results to a broad range of modal logics using the coinductive framework and introduced a more flexible notion of symmetry, i.e., layered symmetries, for modal logics defined over model classes having tree witnesses.

				Global			Layered		
Formula	*md*	*#V*	*#Cl*	*#N*	*#E*	$T_{\text{G+S}}$	*#N*	*#E*	$T_{\text{G+S}}$
k_branch_p_21	22	991	40119	42101	48678	0.37	44081	49668	0.38
k_branch_n_21	22	991	40097	42079	48656	0.38	44059	49646	0.36
k_ph_n_21	2	484	14368	15336	20186	0.24	15336	20186	0.22
k_ph_p_21	2	484	14368	15336	20186	0.24	15336	20186	0.21
k_poly_n_21	65	100	10726	10926	11086	0.05	11054	11150	0.07
k_poly_p_21	64	99	10401	10599	10756	0.06	10723	10818	0.06
k_path_n_21	22	6	4416	4428	4754	0.02	4680	4880	0.06
k_path_p_21	21	6	4104	4116	4430	0.02	4356	4550	0.06
k_t4p_n_21	46	5	2786	2796	3045	0.01	2892	3093	0.01
k_lin_n_21	3	239	1662	2140	3245	0.03	2464	3407	0.03

Table 4: Top 10 largest graphs.

We then introduced graph constructions to detect global and layered symmetries in modal CNF formulas. We proved that the graph constructions are correct, i.e., every detected automorphism of the resulting graph corresponds to a symmetry of the original formula. Since the constructions are developed in the coinductive framework, they can be seen as templates from which to derive concrete implementations for concrete modal logics.

We tested the constructions on basic modal logic benchmarks. Experimental results showed that symmetries do exists in modal benchmarks, and, as expected, the presence of symmetries highly depends on the problem codification. Results also showed that detecting symmetries is relatively inexpensive even for large graphs.

The next step would be to see how to profit from the presence of symmetries in modal formulas in an automated prover. A preliminary evaluation of the effect of using symmetric information in a tableaux algorithm for the basic modal logic is available in [41]. Also, more testing in richer modal logics, e.g., hybrid logic, is necessary.

Acknowledgements This work was partially supported by grants ANPCyT-PICT-2008-306, ANPCyT-PICT-2010-688, ANPCyT-PICT-2013-2011, the FP7-PEOPLE-2011-IRSES Project "Mobility between Europe and Argentina applying Logics to Systems" (MEALS) and the Laboratoire International Associé "INFINIS".

Bibliography

[1] C. Areces, D. Gorín, Coinductive models and normal forms for modal logics (or how we learned to stop worrying and love coinduction), Journal of Applied Logic 8 (4) (2010)

305–318.

[2] G. Darvas, Symmetry. Cultural-historical and ontological aspects of science-arts relations. The natural and man-made world in an interdisciplinary approach, Basel: Birkhäuser, 2007.

[3] J. Harrison, Without loss of generality, in: S. Berghofer, T. Nipkow, C. Urban, M. Wenzel (Eds.), Proceedings of the 22nd International Conference on Theorem Proving in Higher Order Logics, Vol. 5674 of LNCS, Springer-Verlag, Munich, Germany, 2009, pp. 43–59.

[4] K. Sakallah, Symmetry and Satisfiability, Vol. 185 of Frontiers in Artificial Intelligence and Applications, IOS Press, 2009, Ch. 10, pp. 289–338.

[5] B. Benhamou, L. Sais, Theoretical study of symmetries in propositional calculus and applications, in: Automated Deduction (CADE-11), 1992, pp. 281–294.

[6] B. Krishnamurthy, Short proofs for tricky formulas, Acta Informatica 22 (3) (1985) 253–275.

[7] C. Brown, L. Finkelstein, P. Purdom Jr, Backtrack searching in the presence of symmetry, in: Applied algebra, algebraic algorithms and error-correcting codes, Springer, 1989, pp. 99–110.

[8] J. Crawford, A theoretical analysis of reasoning by symmetry in first-order logic, in: Proceedings of AAAI Workshop on Tractable Reasoning, San Jose, CA, 1992, pp. 17–22.

[9] B. Benhamou, L. Sais, Tractability through symmetries in propositional calculus, Journal of Automated Reasoning 12 (1) (1994) 89–102.

[10] J. Crawford, M. Ginsberg, E. Luks, A. Roy, Symmetry-breaking predicates for search problems, in: Proceedings of KR 1996, 1996, pp. 148–159.

[11] F. Aloul, A. Ramani, I. Markov, K. Sakallah, Solving difficult instances of Boolean satisfiability in the presence of symmetry, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 22 (9) (2003) 1117–1137.

[12] F. Aloul, I. Markov, K. Sakallah, Shatter: efficient symmetry-breaking for Boolean satisfiability, in: Design Automation Conference, IEEE, 2003, pp. 836–839.

[13] F. Aloul, K. Sakallah, I. Markov, Efficient symmetry breaking for Boolean satisfiability, IEEE Transactions on Computers 55 (5) (2006) 549–558.

[14] B. Benhamou, T. Nabhani, R. Ostrowski, M. Saidi, Enhancing clause learning by symmetry in SAT solvers, in: Proceedings of the 22nd IEEE International Conference on Tools with Artificial Intelligence (ICTAI), 2010, pp. 329–335.

[15] I. Gent, K. Petrie, J. Puget, Symmetry in constraint programming, in: F. Rossi, P. van Beek, T. Walsh (Eds.), Handbook of Constraint Programming, Elsevier, 2006, Ch. 10.

[16] D. Cohen, P. Jeavons, C. Jefferson, K. Petrie, B. Smith, Symmetry definitions for constraint satisfaction problems, in: Principles and Practice of Constraint Programming, Springer, 2005, pp. 17–31.

[17] F. Margot, Pruning by isomorphism in branch-and-cut, Mathematical Programming 94 (1) (2002) 71–90.

[18] F. Margot, Exploiting orbits in symmetric ILP, Mathematical Programming 98 (1-3) (2003) 3–21.

[19] M. Fox, D. Long, The detection and exploitation of symmetry in planning problems, in: IJCAI, Vol. 99, 1999, pp. 956–961.

[20] M. Fox, D. Long, Extending the exploitation of symmetries in planning, in: AIPS, 2002, pp. 83–91.

[21] E. Clarke, R. Enders, T. Filkorn, S. Jha, Exploiting symmetry in temporal logic model checking, Formal Methods in System Design 9 (1-2) (1996) 77–104.

[22] C. Ip, D. Dill, Better verification through symmetry, Formal methods in system design 9 (1-2) (1996) 41–75.

[23] A. Sistla, V. Gyuris, E. Emerson, SMC: a symmetry-based model checker for verification of safety and liveness properties, ACM Transactions on Software Engineering and Methodology (TOSEM) 9 (2) (2000) 133–166.

[24] D. Bošnački, D. Dams, L. Holenderski, Symmetric Spin, International Journal on Software Tools for Technology Transfer 4 (1) (2002) 92–106.

[25] G. Audemard, B. Mazure, L. Sais, Dealing with symmetries in Quantified Boolean Formulas, in: Proceedings of the 7th International Conference on Theory and Applications of Satisfiability Testing (SAT'04), 2004, pp. 257–262.

[26] G. Audemard, S. Jabbour, L. Sais, Efficient symmetry breaking predicates for Quantified Boolean Formulae, in: Proceedings of SymCon-Symmetry in Constraitns-CP workshop, 2007.

[27] G. Audemard, S. Jabbour, L. Sais, Symmetry breaking in Quantified Boolean Formulae, in: IJCAI, 2007, pp. 2262–2267.

[28] G. Audemard, A. Cimatti, A. Kornilowicz, R. Sebastiani, Bounded model checking for timed systems, in: Formal Techniques for Networked and Distributed Sytems (FORTE 2002), Springer, 2002, pp. 243–259.

[29] K. Roe, The heuristic theorem prover: Yet another SMT modulo theorem prover, in: Computer Aided Verification, Springer, 2006, pp. 467–470.

[30] D. Déharbe, P. Fontaine, S. Merz, B. Woltzenlogel Paleo, Exploiting symmetry in SMT problems, in: Automated Deduction (CADE-23), Vol. 6803 of LNCS, Springer Berlin Heidelberg, 2011, pp. 222–236.

[31] A. Donaldson, A. Miller, Automatic symmetry detection for model checking using computational group theory, in: Formal Methods, Springer, 2005, pp. 481–496.

[32] A. Miller, A. Donaldson, M. Calder, Symmetry in temporal logic model checking, ACM Computing Surveys 38 (3).

[33] A. Donaldson, Automatic techniques for detecting and exploiting symmetry in model checking, Ph.D. thesis, University of Glasgow (2007).

[34] M. Cohen, M. Dam, A. Lomuscio, H. Qu, A symmetry reduction technique for model checking temporal-epistemic logic., in: IJCAI, Vol. 9, 2009, pp. 721–726.

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

[36] P. Blackburn, J. van Benthem, F. Wolter, Handbook of Modal Logic, Vol. 3 of Studies in Logic and Practical Reasoning, Elsevier Science Inc., New York, NY, USA, 2006.

[37] J. Fraleigh, V. Katz, A first course in abstract algebra, Addison-Wesley world student series, Addison-Wesley, 2003.

[38] P. Patel-Schneider, R. Sebastiani, A new general method to generate random modal formulae for testing decision procedures, Journal of Artificial Intelligence Research 18 (2003) 351–389.

[39] C. Areces, B. ten Cate, Hybrid logics, in: P. Blackburn, F. Wolter, J. van Benthem (Eds.), Handbook of Modal Logics, Elsevier, 2006, pp. 821–868.

[40] C. Areces, R. Gennari, J. Heguiabehere, M. de Rijke, Tree-based heuristics in modal theorem proving, in: Proceedings of ECAI'2000, Berlin, Germany, 2000, pp. 199–203.

[41] C. Areces, E. Orbe, Symmetric blocking, Theoretical Computer Science. In Press. doi: http://dx.doi.org/10.1016/j.tcs.2015.06.020.

[42] P. Balsiger, A. Heuerding, S. Schwendimann, A benchmark method for the propositional modal logics K, KT, S4, Journal of Automated Reasoning 24 (3) (2000) 297–317.

[43] T. Junttila, P. Kaski, Engineering an efficient canonical labeling tool for large and sparse graphs, in: Proceedings of the Workshop on Algorithm Engineering and Experiments (ALENEX), 2007.

MEALS Partner Abbreviations

SAU: Saarland University, D

RWT: RWTH Aachen University, D

TUD: Technische Universität Dresden, D

INR: Institut National de Recherche en Informatique et en Automatique, FR

IMP: Imperial College of Science, Technology and Medicine, UK

ULEIC: University of Leicester, UK

TUE: Technische Universiteit Eindhoven, NL

UNC: Universidad Nacional de Córdoba, AR

UBA: Universidad de Buenos Aires, AR

UNR: Universidad Nacional de Río Cuarto, AR

ITBA: Instituto Técnológico Buenos Aires, AR