Relational Dual Tableau Decision Procedure for Modal Logic K [*]

Golińska-Pilarek, Joanna[1], Muñoz-Velasco, Emilio[2] and Mora-Bonilla, Angel[2]
[1] Institute of Philosophy, University of Warsaw and National Institute of Telecommunications, Poland.
`j.golinska@uw.edu.pl`
[2] Dept. Applied Mathematics, University of Málaga, Málaga, Spain.
`{emilio, amora}@ctima.uma.es`

January 2, 2011

Abstract

We present a dual tableau system, $\mathcal{RLK}$, which is itself a deterministic decision procedure verifying validity of K-formulas. The system is constructed in the framework of the original methodology of relational proof systems, determined only by axioms and inference rules, without any external techniques. Furthermore, we describe an implementation of the system $\mathcal{RLK}$ in Prolog, and we show some of its advantages.
Keywords: modal logic, dual tableau methods, decision procedures, relational proof systems

1 Introduction

Relational dual tableau systems are based on Rasiowa-Sikorski diagrams for first-order logic [14]. The common language of most of relational dual tableaux is the logic of binary relations which is a logical counterpart to the class RRA of (representable) relation algebras introduced by Tarski [16]. The formulas of the classical logic of binary relations are intended to represent statements saying that two objects are related. Relations are specified in the form of relational terms. Terms are built from relational variables and/or relational constants with relational operations of union, intersection, complement, composition, and converse.

Relational dual tableaux are powerful tools for verification of validity as well as for proving entailment, model checking (i.e., verification of truth of a statement

[*]This work is partially supported by the Spanish research projects TIN2009-14562-C05-01 and P09-FQM-5233. The first author of the paper is partially supported by the Polish Ministry of Science and Higher Education grant IP2010 010170.

in a particular fixed finite model), and satisfaction (i.e., verification that a statement is satisfied by some fixed objects of a finite model). A comprehensive survey on applications of dual tableaux methodology to various theories and logics can be found in [13]. Dual tableaux are deduction systems alternative to other formalisms such as tableaux, Gentzen sequent calculi, resolution or Hilbert-style systems. A discussion of relationships between dual tableaux and those systems is presented in [13]. The main advantage of relational methodology is the possibility of representation within a uniform formalism the three basic components of formal systems: syntax, semantics, and deduction apparatus. Hence, the relational approach provides a general framework for representation, investigation, and implementation of theories with different languages and/or semantics.

The last years, modal logics have become very popular and extremely useful in many areas of computer science, thus many efforts have been made to design effective and simple-to-use decision procedures for decidable modal logics. Modal logics could be very helpful in modelling dynamic and reactive systems such as bio-inspired systems and can be applied to soft-computing scenarios [3,4,10,15]. For example, some Description Logics are syntactic variants of modal logics and have applications in bio-inspired systems [17], and Connectionist Logics [2] combine the strengths of modal logics and neural networks. In this paper we focus on logic K which is the minimal normal and decidable modal logic. We present a relational proof system in dual tableau style for the logic K, we show its soundness and completeness, and that it is itself a deterministic decision procedure for K. Finally, we described its implementation in Prolog. In the literature decision procedures for logic K have been intensively studied over the years [5,9]. The most popular decision procedures for logic K are based on tableau systems. However, the methodology of tableau systems does not provide any method for finding direct proofs of valid formulas. Furthermore, the most of tableaux for modal logics provide algorithms deciding unsatisfiability, but they are nondeterministic in nature. Thus, they use some heuristic techniques combined with intelligent backtracking, backjumping or simplifications, but all these techniques are *external* to the tableau itself. We present a dual tableau system, $\mathcal{RLK}$, which is *itself* a decision procedure verifying validity of K-formulas. The system is constructed in the framework of the original methodology of relational proof systems, determined only by axioms and inference rules, without any external techniques. In addition, the nice feature of the system $\mathcal{RLK}$ is its uniqueness: it generates in a deterministic way only one proof tree for a given formula. The system $\mathcal{RLK}$ presented in this paper determines an essential improvement of the system given in [7,12]. In the construction of the system $\mathcal{RLK}$ we use special dual clause representation of modal formulas and we reduce the number of the rules, hence the system is more effective than the system described in [7].

The paper is organized as follows. In Section 2, we present the relational formalization of modal logic K. In Section 3, we present $\mathcal{RLK}$-dual tableau system, its rules and axioms. Soundness and completeness of the system are studied in Section 4. The implementation of the system is presented in Section 5. Final remarks and prospects of future work are described in Section 6.

2 Relational formalization of logic K

In this section, we define the relational logic, RL_K, appropriate for expressing K-formulas in their dual clause representation. Recall that the vocabulary of language of logic K consists of: the symbols from the set $\mathbb{V} = \{p_1, p_2, p_3, \ldots\}$, which is an ordered countable infinite set of propositional variables indexed with natural numbers; the classical propositional operations of negation $\neg$ and conjunction $\wedge$; and the modal propositional operation $\Diamond$ called the *possibility* operation. The set of K-formulas is the smallest set including the set of propositional variables and closed with respect to all the propositional operations. We call formulas of the form p or $\neg p$ *classical literals* and we use letters like a, b, c (possible with indices) as their meta-representations. By $\neg a$ we denote the formula $\neg p$ if $a = p$, and p if $a = \neg p$. A K-*model* is a structure $\mathcal{M} = (U, R, m)$ such that U is a non-empty set (of states), R is a binary relation on U, m is the meaning function such that $m(p) \subseteq U$, for every propositional variable $p \in \mathbb{V}$. The relation R is referred to as the *accessibility relation*. The *satisfaction relation* is defined as usual in modal logics. Thus, $\mathcal{M}, s \models \varphi$ denotes the fact that a formula φ is satisfied in a model $\mathcal{M}$ by a state s. A K-formula φ is said to be *true* in a K-model $\mathcal{M} = (U, R, m)$, $\mathcal{M} \models \varphi$, whenever for every $s \in U$, $\mathcal{M}, s \models \varphi$, and it is K-*valid* whenever it is true in all K-models.

Now, we introduce special dual clauses representing modal formulas. The relational representation of modal formulas is a key tool in the construction of the system presented in this paper. A *simple dual clause* is a K-formula either of the form $a_1 \wedge \ldots \wedge a_m$, $\Diamond a \wedge b$, $\neg\Diamond a \wedge b$, or $\neg\Diamond a$, where $a, b, a_1, \ldots, a_m$ are classical literals and $m \geq 1$. Simple dual clauses of the form $a_1 \wedge \ldots \wedge a_m$, $m \geq 2$, $\Diamond a \wedge b$, $\neg\Diamond a \wedge b$ are referred to as *conjunctive dual clauses*. A *dual clause* is a K-formula of the form $\Diamond^s \phi$, where ϕ a simple dual clause and $s \geq 0$. Although the vocabulary of the K-language does not contain disjunction, we will consider the meta-disjunction of dual clauses $\phi_1 \vee \ldots \vee \phi_r$. The meta-disjunction $\phi_1 \vee \ldots \vee \phi_r$ is said to be K-valid if and only if for every K-model $\mathcal{M} = (U, R, m)$ and for every $s \in U$ there exists $i \in \{1, \ldots, r\}$ such that $\mathcal{M}, s \models \phi_i$. The following result states that every K-formula can be transformed into an equivalent meta-disjunction of dual clauses. It is a dual form of the result from [11]. In fact, it can be proved that it is true for any normal modal logic.

Theorem 1 *For every K-formula φ, there exists a finite set $\{\phi_1, \ldots, \phi_r\}$ of dual clauses, such that φ is K-valid if and only if the meta-disjunction $\phi_1 \vee \ldots \vee \phi_r$ is K-valid. Moreover, the set $\{\phi_1, \ldots, \phi_r\}$ can be obtained in quadratic time.*

For a K-formula φ, its corresponding meta-disjunction of dual clauses, ensured by Theorem 1, is referred to as its *dual clause representation* and denoted by $form(\varphi)$.

Now, we define the relational logic $\mathsf{RL_K}$. The vocabulary of the language of $\mathsf{RL_K}$ consists of the symbols from the following pairwise disjoint sets: $\mathbb{OV} = \{z_0, z_1, \ldots\}$, an ordered countable infinite set of object variables indexed with natural numbers; $\mathbb{RV} = \{P_1, P_2, \ldots\}$, an ordered countable infinite set of relational variables indexed with natural numbers; $\{R\}$, the set consisting of the relational constant R representing the accessibility relation from K-models; and $\{-, \cap, ;\}$, the set of relational operations, where $-$ is the complement operation, $\cap$ is the intersection operation, and ; is the composition operation. The set of relational terms is defined as follows. To begin with, the set of terms of the form P and $-P$, for a relational variable P, is referred to as the set of *relational literals*. We will use letters like A, B, C (possible with indices) as their meta-representations. By $-A$ we denote the term $-P$ if $A = P$, and P if $A = -P$. The set of *simple relational terms*, $\mathbb{SRT}$, is the smallest set which includes relational literals and satisfies the following two conditions: if $A_1, \ldots, A_m$, $m \geq 1$, are relational literals, then $A_1 \cap \ldots \cap A_m \in \mathbb{SRT}$; and if A and B are relational literals, $(R\,;A) \cap B$, $-(R\,;A) \cap B$, $-(R\,;A) \in \mathbb{SRT}$. Terms of the form $A_1, \ldots, A_m$, $m \geq 2$, $(R\,;A) \cap B$, and $-(R\,;A) \cap B$ are referred to as *conjunctive terms*. The set of *relational terms*, $\mathbb{RT}$, is the smallest set which includes simple relational terms such that, if $T \in \mathbb{SRT}$, then $R^s\,;T \in \mathbb{RT}$, where $R^s\,;T$ is defined as usual. $\mathsf{RL_K}$-formulas are of the form z_iTz_0, where z_i, z_0 are object variables, $i \geq 1$, and T is a relational term. A formula of the form z_iAz_0, for a relational literal A, is called a *literal formula*. A formula z_iTz_0, where T is a simple relational term (resp. a conjunctive term) is referred to as a *simple formula* (resp. *conjunctive formula*). An $\mathsf{RL_K}$-*model* is a structure $\mathcal{M} = (U, R, m)$, where U is a non-empty set, R is a binary relation on U, and m is the meaning function such that: $m(P) = X \times U$, where $X \subseteq U$, for every relational variable P; $m(R) = R$, i.e., R is the interpretation of the relational constant R; m extends to all the compound relational terms as usual. Let $\mathcal{M} = (U, R, m)$ be an $\mathsf{RL_K}$-model. A *valuation* in $\mathcal{M}$ is any function $v\colon \mathbb{OV} \to U$. An $\mathsf{RL_K}$-formula z_iTz_0 is *satisfied* in an $\mathsf{RL_K}$-model $\mathcal{M}$ by a valuation v, $\mathcal{M}, v \models z_iTz_0$ whenever $(v(z_i), v(z_0)) \in m(T)$. A formula is *true* in $\mathcal{M}$ whenever it is satisfied by all the valuations in $\mathcal{M}$, and it is $\mathsf{RL_K}$-*valid* whenever it is true in all $\mathsf{RL_K}$-models. A finite set of $\mathsf{RL_K}$-formulas $\{\varphi_1, \ldots, \varphi_n\}$ is $\mathsf{RL_K}$-valid whenever for every $\mathsf{RL_K}$-model $\mathcal{M}$ and for every valuation v in $\mathcal{M}$ there exists $i \in \{1, \ldots, n\}$ such that $\mathcal{M}, v \models \varphi_i$.

Now, we define the translation of K-dual clauses into relational terms. The translation starts with a one-to-one assignment of relational variables to the propositional variables. More precisely, for every $i \geq 1$, we define $\tau'(p_i) = P_i$. Then the translation τ of K-dual clauses is defined inductively as follows: $\tau(p) = \tau'(p)$, for any propositional variable $p \in \mathbb{V}$; $\tau(\neg\phi) = -\tau(\phi)$; $\tau(\phi \wedge \phi') = \tau(\phi) \cap \tau(\phi')$; $\tau(\Diamond\phi) = (R\,;\tau(\phi))$. Clearly, the translation τ is well defined, that is for every K-dual clause ϕ, there exists a relational term T such that $\tau(\phi) = T$. Furthermore, the translation τ assigns to each K-dual clause a right ideal relation, i.e., a relation which is of the form $X \times U$, for some $X \subseteq U$. Therefore, if $\mathcal{M}, v \models z_iTz_0$, for some $i \geq 1$ and a relational term T, then for every $x \in U$, $(v(z_i), x) \in m(T)$.

Hence, the variable z_0 represents elements of the universe U. The following result states the relationships between modal formulas and their relational representation.

Theorem 2 *For all* K*-dual clauses* $\phi_1, \ldots, \phi_r$*,* $r \geq 1$*,* $\phi_1 \vee \ldots \vee \phi_r$ *is* K*-valid iff* $\{z_1\tau(\phi_1)z_0, \ldots, z_1\tau(\phi_r)z_0\}$ *is* $\mathsf{RL_K}$*-valid.*

Given a K-formula φ such that $form(\varphi) = \phi_1 \vee \ldots \vee \phi_r$, $r \geq 1$, we define its translation as $\tau(\varphi) \overset{\mathrm{df}}{=} \{z_1\tau(\phi_1)z_0, \ldots, z_1\tau(\phi_r)z_0\}$. Thus, the translation τ transfers every K-formula into a finite set of $\mathsf{RL_K}$-formulas. Due to Theorem 1 and Theorem 2, we get:

Theorem 3 *For every* K*-formula* φ*,* φ *is* K*-valid iff* $\tau(\varphi)$ *is* $\mathsf{RL_K}$*-valid.*

Now, we define a kind of lexicographical *order* on the set of all conjunctive terms. Although there are many ways of defining an order on all terms, to get a deterministic dual tableau, we need only an order on conjunctive terms. Let terms A and B be relational literals. Then $A < B$ whenever either of the following holds: $A = P_i$ and $B = P_j$ and $i < j$, or $A = P_i$ and $B = -P_j$, or $A = -P_i$ and $B = -P_j$ and $i < j$, where P_i, P_j are relational variables and $i, j \in \mathbb{N}$. Let $m, k > 1$ and let $A_1, \ldots, A_m, B_1, \ldots, B_k$ be relational literals. Then $A_1 \cap \ldots \cap A_m < B_1 \cap \ldots \cap B_k$ whenever either $m < k$ (i.e., the length of $A_1 \cap \ldots \cap A_m$ is less than the length of $B_1 \cap \ldots \cap B_k$) or $m = k$ and there exists $i \in \{1, \ldots, m\}$ such that $A_i < B_i$ and $A_j = B_j$ for all $j < i$. For all relational literals A, B, C, and D, we define $(R\,;A) \cap B < (R\,;C) \cap D$ (resp. $-(R\,;A) \cap B < -(R\,;C) \cap D$) if either $A < C$ or $A = C$ and $B < D$. Finally, let $m \geq 1$. For all relational literals $A_1, \ldots, A_m$, A, B, C, and D, we require $A_1 \cap \ldots \cap A_m < (R\,;A) \cap B < -(R\,;C) \cap D$. Now, we extend the order $<$ to all $\mathsf{RL_K}$-conjunctive formulas. We define: $z_i\phi z_0 < z_j\psi z_0$ if and only if either $i < j$, or $i = j$ and $\phi < \psi$. The following proposition is a direct consequence of the above definition.

Proposition 4 *The set of all* $\mathsf{RL_K}$*-conjunctive formulas is well ordered by* $<$*.*

If X is a finite set of $\mathsf{RL_K}$-formulas, then an $\mathsf{RL_K}$-conjunctive formula $z_i\phi z_0$ is said to be *minimal with respect to* X whenever $z_i\phi z_0 < z_j\psi z_0$, for all $\mathsf{RL_K}$-conjunctive formulas $z_j\psi z_0 \in X$.

3 Relational dual tableau for modal logic K

In this section, we present a relational dual tableau, $\mathcal{RLK}$, which can be used for verification of validity of K-formulas in their dual clause representation. The system $\mathcal{RLK}$ is determined by axiomatic sets of $\mathsf{RL_K}$-formulas and rules which apply to finite sets of $\mathsf{RL_K}$-formulas. The axiomatic sets take the place of axioms. A finite set of $\mathsf{RL_K}$-formulas is said to be $\mathsf{RL_K}$-*axiomatic* whenever it is a superset of $\{z_iTz_0, z_i{-}Tz_0\}$, for some $i \geq 1$ and $\mathsf{RL_K}$-term T. Sets which are not axiomatic are referred to as *non-axiomatic*. Clearly, every axiomatic set is $\mathsf{RL_K}$-valid.

The rules of $\mathcal{RLK}$-system have the following general form: $(*)\ \dfrac{\Gamma}{\Gamma_1 \mid \ldots \mid \Gamma_n}$, where $\Gamma, \Gamma_1, \ldots, \Gamma_n$, $n \geq 1$, are finite non-empty sets of RL_{K}-formulas. A rule of the form $(*)$ is *applicable* to a finite set Y if and only if $Y = \Gamma$ and there exists $i \in \{1, \ldots, n\}$ such that $\Gamma_i \neq Y$, that is, an application of a rule must introduce a new formula. If $n > 1$, then a rule of the form $(*)$ is an n-fold branching rule. In a rule, the set above the line is referred to as its *premise* and the set(s) below the line is (are) its *conclusion(s)*. A variable z_i, $i \geq 1$, which appears in a conclusion of a rule and does not appear in the premise, is called a *new variable* introduced by an application of a rule. $\mathcal{RLK}$-dual tableau consists of decomposition rules $(\cap)$ and (K) which are described below.

For any finite set X of RL_{K}-formulas, for every $i \geq 1$, and for all formulas $\varphi_1, \ldots, \varphi_m$, $m \geq 2$,

$(\cap)\quad \dfrac{X \cup \{z_i(\varphi_1 \cap \ldots \cap \varphi_m)z_0\}}{X \cup \{z_i\varphi_1 z_0\} \mid \ldots \mid X \cup \{z_i\varphi_m z_0\}}$ $\quad z_i(\varphi_1 \cap \ldots \cap \varphi_m)z_0$ is minimal with respect to X

$(K)\quad \dfrac{X \cup K_1 \cup K_2}{X \cup G_1 \cup G_2}$ $\quad$ where X does not contain any conjunctive formula; K_i and G_i, for $i \in \{1, 2\}$, are defined as follows.

$K_1 \overset{\text{df}}{=} \{z_n-(R\,;A_j)z_0\}_{j \in J_1}$, where $J_1 = \{0, \ldots, m-1\}$ for $m \geq 1$, A_j are relational literals for every $j \in J_1$, and $n \geq 1$ is such that for every $i \geq 1$ and for all relational literals B, if $z_i-(R\,;B)z_0 \in X$, then $i > n$;

$K_2 \overset{\text{df}}{=} \{z_n(R^{s_j}\,;T_j)z_0\}_{j \in J_2}$, where J_2 is a finite (possibly empty) set of indices, $s_j \geq 1$, and T_j are simple relational terms;

$G_1 = \{z_{n_1}-A_1 z_0, \ldots, z_{n_1+m-1}-A_{m-1}z_0\}$, where $n_1 \geq 1$ is the smallest index such that object variable z_{n_1} does not occur in the premise of the rule (K);

$G_2 = \{z_{n_1}(R^{s_j-1}\,;T_j)z_0, \ldots, z_{n_1+m-1}(R^{s_j-1}\,;T_j)z_0\}_{j \in J_2}$.

Given a finite set of RL_{K}-formulas $\{z_iT_1z_0, \ldots, z_iT_nz_0\}$, $i, n \geq 1$, successive applications of the rules result in a tree whose nodes consist of finite sets of RL_{K}-formulas. More precisely, an *$\mathcal{RLK}$-proof tree* of $\{z_iT_1z_0, \ldots, z_iT_nz_0\}$ is a tree such that: $\{z_iT_1z_0, \ldots, z_iT_nz_0\}$ is the root of the tree, each node except the root is obtained by an application of a rule to its predecessor node, and a node does not have successors whenever its set of formulas is RL_{K}-axiomatic or none of the rules applies to it. Observe that the rule (K) can be applied to a finite set of the form $X \cup K_1 \cup K_2$ only if the rule $(\cap)$ cannot be applied to it. Thus, the algorithm for applications of the rules is as follows: first, check whether the rule $(\cap)$ applies to a set, if not try to apply the rule (K). Furthermore, due to the order on formulas, on each step of the construction of the tree, a formula (resp. formulas) to which the rule $(\cap)$ (resp. (K)) can be applied is uniquely determined. A branch of an $\mathcal{RLK}$-proof tree is *closed* if it contains a node with an RL_{K}-axiomatic set of formulas. An $\mathcal{RLK}$-proof tree is closed if and only if all of its branches are closed. A finite set of RL_{K}-formulas is *$\mathcal{RLK}$-provable* whenever there is a closed $\mathcal{RLK}$-proof tree of it, which is then referred to as its *$\mathcal{RLK}$-proof.* Observe that every $\mathcal{RLK}$-proof tree is finite. Furthermore, the rules of $\mathcal{RLK}$-dual tableau guarantee that:

Theorem 5 *Let $X = \{z_iT_1z_0, \ldots, z_iT_nz_0\}$, for $i, n \geq 1$, be a finite set of* RL_{K}-*formulas. Then, the set X has exactly one finite $\mathcal{RLK}$-proof tree.*

4 Soundness and completeness

In this section we study soundness and completeness of $\mathcal{RLK}$-system. A rule of the form $\dfrac{\Gamma}{\Gamma_1 \mid \ldots \mid \Gamma_n}$ is said to be $\mathsf{RL_K}$-*correct* whenever Γ is $\mathsf{RL_K}$-valid if and only if for every $i \in \{1, \ldots, n\}$ the set Γ_i is $\mathsf{RL_K}$-valid. In the proof of soundness, we use the fact that the rules of $\mathcal{RLK}$-dual tableau are $\mathsf{RL_K}$-correct, that is they preserve and reflect validity of sets of its premise and conclusions:

Proposition 6 *The* $\mathsf{RL_K}$*-axiomatic sets are* $\mathsf{RL_K}$*-valid and the* $\mathsf{RL_K}$*-rules are* $\mathsf{RL_K}$*-correct.*

The proof of the above proposition is omitted, because of the lack of space.

Theorem 7 (Soundness) *Let* $\Phi = \{z_i T_1 z_0, \ldots, z_i T_n z_0\}$*, for* $i, n \geq 1$*, be a finite set of* $\mathsf{RL_K}$*-formulas. Then, if* Φ *is* $\mathcal{RLK}$*-provable, then it is* $\mathsf{RL_K}$*-valid.*

Proof Let $\Phi = \{z_i T_1 z_0, \ldots, z_i T_n z_0\}$, $i, n \geq 1$, be a finite $\mathcal{RLK}$-provable set of formulas. Then, there exists an $\mathcal{RLK}$-proof tree of Φ such that each of its branches is closed, that is it ends with an $\mathsf{RL_K}$-valid set of formulas. Thus, by Proposition 6, going from the bottom to the top of the tree, we conclude that Φ is $\mathsf{RL_K}$-valid.

Theorem 8 (Completeness) *Let* $\Phi = \{z_i T_1 z_0, \ldots, z_i T_n z_0\}$*, for* $i, n \geq 1$*, be a finite set of* $\mathsf{RL_K}$*-formulas. Then, if* Φ *is* $\mathsf{RL_K}$*-valid, then it is* $\mathcal{RLK}$*-provable.*

The proof of the above theorem follows from the proof of the analogous theorem presented in [7]. Now, by Theorems 3, 7, and 8, we get:

Theorem 9 (Soundness and Completeness) *For every* K*-formula* φ*,* φ *is* K*-valid iff* $\tau(\varphi)$ *is* $\mathcal{RLK}$*-provable.*

The above theorem and Theorem 5 imply:

Theorem 10 $\mathcal{RLK}$ *is a sound and complete decision procedure for the logic* K.

5 Implementation of the prover

In this section, we outline the new prover developed in Prolog based on the theoretical foundations presented in this paper. A key point for improving the efficiency of our prover is the management of the modal clauses. Then, we establish the algorithm to translate modal formula to modal clauses, following the ideas presented in [8]. After this, we translate the modal clauses to relational terms, as defined above. We use $p, q, \ldots$ or $\neg p, \neg q, \ldots$ to be classical literals; $a, b, c, etc.$ denote classical literals; and ψ, φ to represent the non classical literals.

First, we translate the modal formulas to negative normal form, *nnf*, and then the translation of a modal formula in *nnf* to modal clauses is obtained by the following algorithm, where $\Box^s$ represents $\overbrace{\Box \ldots \Box}^{s}$.

Function `m2c` (ψ) Modal 2 Clauses

```
Output:  set of modal clauses - translation of $\psi$
  begin
      if $\psi$ is modal clause then
          return $\psi$
      else
          switch the value of $\psi$ do
              case [1] $\psi$ is $\Box^s[\varphi_1 \wedge \ldots \wedge \varphi_k]$
                  return $m2c(\Box^s[\varphi_1]), \ldots, m2c(\Box^s[\varphi_k])$
              case [2] $\psi$ is $\Box^s[a, \Box(\varphi)]$ or $\Box^s[a, \diamond(\varphi)]$
                  return $\Box^s[a, \Box(p)], m2c(\Box^{s+1}[\neg p, m2c(\varphi)])$
                or respectively
                  return $\Box^s[a, \diamond(p)], m2c(\Box^{s+1}[\neg p, m2c(\varphi)])$
              case [3] $\psi$ is $\Box^s[a_1, \ldots, a_h, \varphi_1, \ldots, \varphi_k]$, $k > 2$ or $(k = 1, h > 1)$
                  Introduce $k$ new primitive propositions: $p_1, \ldots p_k$
                  return $\Box^s[a_1, \ldots, a_h, p1, \ldots, p_k], m2c(\Box^s[\neg p_1, \varphi_1]), \ldots, m2c(\Box^s[\neg p_k, \varphi_k])$
              case [4] $\psi$ is $\Box^s[a, \varphi \wedge \zeta]$
                  return $m2c(\Box^s[a, \varphi]), m2c(\Box^s[a, \zeta])$
          end
      end
  end
```

Figure 1

Example The modal formula $\Box(p \to q) \to (\Box p \to \Box q)$ is translated to relational clauses as follows.

```
?- modal2clauses.
formulaKLogic(implication(square(implication(p, q)),
            implication(square(p), square(q)))).
%First, to modal clauses
rclause(square(1, or([not(p), q]))).
rclause(square(1, p)).
rclause(diamond(1, not(q))).
%Second, modal clauses to relational clauses.
rel([1], opp(comp(r, opp(q))), x, y).
rel([1], comp(r, opp(p)), x, y).
rel([1], comp(r, inter([p, opp(q)])), x, y).
```

When the translation to relational clauses from modal clauses is obtained, the proof process runs as follows. Prolog builds a tree representing the proof process. First, it sorts the relations in the root node and applies a rule following the order defined above. After the application of the rule, it checks if there is an axiomatic set in any node of the tree. The process is repeated until either all nodes are closed, or none of the rules is applicable. The formula is valid whenever all the nodes are closed. A formula is represented as the Prolog fact: $rel([1], T, z_1, z_0)$. In node 1, it stores the formula $z_1 T z_0$. The rules of $\mathcal{RLK}$ are translated to clauses in Prolog. We emphasize that the efficiency of the prover presented in this paper is because we have an efficient set of input formulas that allow us to have only two rules in the prover: (K) and *intersection* rules. The inference engine is executed recursively until all the leaves are closed. The formulas are ordered and the rules are applied. If a rule includes new formulas in a leaf, the engine of the prover adds these relations in the adequate position in order to preserve the ordering defined previously. The engine of the prover uses the mechanism of pattern machine of Prolog to detect if exists an axiomatic set in any leaf of the tree. In this case, it deletes the corresponding leaf and informs the user. Then, the prover detect if the formula is valid or not valid.

Example The modal formula $\Box(p \to q) \to (\Box p \to \Box q)$ from the previous example, is proved by $\mathcal{RLK}$ implementation by applying only 2 rules. Our previous version $RePML_K$ applied 7 rules, and Lotrec and TWB applied 9 and 7 rules, respectively [1,6,12].

```
[rel([1], opp(comp(r, opp(q))), x, y),
rel([1], comp(r, opp(p)), x, y),
rel([1], comp(r, inter([p, opp(q)])), x, y)]
___________________________________________    K Rule
[rel([1], q, w1, y), rel([1], opp(p), w1, y),
rel([1], inter([p, opp(q)]), w1, y)]

[rel([1], inter([p, opp(q)]), w1, y)]
_________________________________________    Intersection Rule
rel([1, 1], p, w1, y) | rel([1, 2], opp(q), w1, y)

 ::::: Axiomatic set (close leaf): [1, 2]
[rel([1, 2], opp(q), w1, y), rel([1, 2], q, w1, y)]
 ::::: Axiomatic set (close leaf): [1, 1]
[rel([1, 1], opp(p), w1, y), rel([1, 1], p, w1, y)
 :::::  Variables used: [w1, x, y]
 :::::  VALID.
 :::::  Total Rules applications: 2
used_rules([1], k, [rel(opp(comp(r, opp(q))), x, y), ...
used_rules([1], inter, [rel([1], inter([p, opp(q)]), w1, y)])
```

Finally, consider Figure 2 where we show the result of a small comparative of the number of rules applied by our prover $\mathcal{RLK}$, Lotrec, TWB, and $RePML_K$. Observe that we improve the number the applied rules in all the cases.

Modal formula	Lotrec	TWB	$RePML_K$	$\mathcal{RLK}$	Output
$\Diamond p \to \Diamond p$	1	3	1	1	valid
$\Diamond p \to \Box(\Box\neg q \vee \Diamond q)$	6	5	6	2	valid
$\neg(\Diamond p \wedge \Box\neg p)$	4	3	3	1	valid
$\neg(\Diamond p \wedge \Diamond(\Diamond q \wedge \neg\Diamond q))$	4	5	3	2	valid
$\Box(a \to b) \to (\Box a \to \Box b)$	7	7	6	2	valid
$(\neg\Box\neg p0 \leftrightarrow \Diamond p0)$	7	8	8	1	valid
$\Box p1 \to p1$	1	1	2	0	not valid
$\Box p1 \to \Box\Box p1$	6	3	6	2	not valid
$\neg\Box p1 \to \Box\neg\Box p1$	4	3	6	2	not valid

Figure 2: Comparative of number of rules applied in provers for modal logic K.

6 Conclusions and future works

We have presented a sound and complete dual tableau system, $\mathcal{RLK}$, which is itself a deterministic decision procedure for modal logic K. The system $\mathcal{RLK}$ does not use any external technique such as backtracking, backjumping, etc. An advantage of general methodology of relational approach to deduction is modularity. We expect similar modularity in the construction of relational decision procedures. Thus, the system presented in the paper can be seen as the common core of all relational decision procedures for modal logics. We are already working on extensions of $\mathcal{RLK}$-system in order to get decision procedures for other normal modal logics, in particular for those in which the accessibility relation is assumed to be reflexive or symmetric or transitive. It seems that reflexivity and transitivity case can be solved within our approach. We implemented this system in Prolog and it works very well with the formulas we have introduced. We are also working on an user friendly front-end in order to improve its educational capabilities. Last, but not least, we have planned an exhaustive comparison with the best know provers for modal logic.

References

[1] P. Abate and R. Goré, *The Tableau Workbench*, Electronic Notes in Theoretical Computer Science Vol. 231, 55–67 (2009).

[2] A. d'Avila Garcez, L. C. Lamb and D. M. Gabbay. *Connectionist modal logic: Rep- resenting modalities in neuronal networks*. Theoretical Computer Science, 371, 34–53, 2007.

[3] E. Corchado and A. Herrero, *Neural visualization of network traffic data for intrusion detection*, Applied Soft Computing. doi:10.1016/j.asoc.2010.07.002.

[4] E. Corchado, A. Arroyo, and V. Tricio, *Soft computing models to identify typical meteorological days*, Logic Journal of the IGPL, Press July 21, 2010. doi:10.1093/jigpal/jzq035.

[5] M. Fitting, *Modal Proof Theory*, in: P. Blackburn, J. van Benthem, and F. Wolter (eds), Handbook of Modal Logic, Studies in Logic and Practical Reasoning, Volume 3, Elsevier, Amsterdam, 85–138 (2007).

[6] O. Gasquet, A. Herzig, D. Longin, and M. Sahade, *Lotrec: Logical tableaux research engineering companion*, Lecture Notes in Artificial Intelligence Vol. 3702, 318–322 (2005).

[7] J. Golińska-Pilarek, E. Muñoz-Velasco, and A. Mora, *A new deduction system for deciding validity in modal logic* K, Logic Jnl of IGPL, doi:10.1093/jigpal/jzq033, (2010).

[8] R. Goré and L. A. Nguyen, *Clausal Tableaux for Multimodal Logics of Belief*, Fundamenta Informaticae, Vol. 94(1), 21–40 (2009).

[9] I. Horrocks, U. Hustadt, U. Sattler, and R. Schmidt, *Computational Modal Logic*, in: P. Blackburn, J. van Benthem, and F. Wolter (eds), Handbook of Modal Logic, Studies in Logic and Practical Reasoning, Volume 3, Elsevier, Amsterdam, 181–245 (2007).

[10] L. Lima, P. Novais, R. Costa, J. Bulas Cruz, and J. Neves, *Group decision making and Quality-of-Information in e-Health systems*, Logic Jnl IGPL, doi:10.1093/jigpal/jzq029 (2010).

[11] G. Mints, *Gentzen-type Systems and Resolution Rules*, Lecture Notes in Computer Science Vol. 417, 198–231 (1988).

[12] A. Mora, E. Muñoz-Velasco, and J. Golińska-Pilarek, *Implementing a Relational Theorem Prover for Modal Logic* K, to appear in Int. Jnl. of Computer Mathematics (2010).

[13] E. Orłowska and J. Golińska-Pilarek, *Dual Tableaux: Foundations, Methodology, Case Studies*, Trends in Logic 36, Springer Science, doi: 10.1007/978-94-007-0005-5-21, a book in print (2011).

[14] H. Rasiowa and R. Sikorski, *On Gentzen theorem*, Fundamenta Mathematicae 48, 57-69 (1960).

[15] J. Sedano, L. Curiel, E. Corchado, E. de la Cal and J. R. Villar, *A soft computing method for detecting lifetime building thermal insulation failures*, Integrated Computer-Aided Engineering 17 (2), 103-115, IOS Press (2010).

[16] A. Tarski, *On the calculus of relations*, Journal of Symbolic Logic 6, 73-89 (1941).

[17] S. Yim, J. O. Wilson, D. W. Rosen, *Development of an Ontology for Bio-Inspired Design using Description Logics.* International Conference on Product Lifecycle Management 2008 (PLM08), Vol. 4, No. 1, 2008.