Logic Engineering. The Case of Description and Hybrid Logics

Areces, C.E.

Publication date
2000

Chapter 5
Improving Reasoning Methods

Luchando por una verdad
pero que sea rentable.

from "El Ente," Los Visitantes

As a warming up to the purely theoretical work we will do in Chapters 6 and 7, we will now show how ideas from description and hybrid logics can be put to work with benefit even when the subject is purely modal. In particular, aided by the notions of nominals or labeling, we will show how to define well behaved direct resolution methods for modal languages. This "case study" is a clear example of how the additional flexibility provided by the ability to name states can be used to greatly simplify reasoning methods. In addition, we can build over the basic resolution system and obtain extensions for description and hybrid languages.

Reasoning methods for modal-like languages, can be broadly divided in two categories: direct and indirect. Indirect methods start by translating modal formulas into some first-order language preserving satisfiability, and then take advantage of reasoning methods for FO [de Rijke *et al.*, 2000]. Direct methods instead, work directly on modal formulas devising specialized algorithms for each modal language [Fitting, 1983].

The most developed reasoning methods for modal and modal-like languages today are direct methods, and they are mainly tableau based. Most indirect methods use first-order resolution. In contrast, *direct modal resolution* methods are poorly developed. By drawing on what we have learned in previous chapters about hybrid and description languages, we will provide a direct resolution-based proof procedure for modal languages which improves many aspects of previous proposals. After explaining in detail the resolution method for basic modal languages we will discuss extensions in the three fields of modal, description and hybrid logics. The main characteristics of the new resolution method can be summarized as follows:

- by using labeled formulas it avoids the complexity of earlier direct resolution-based methods for modal logic.
- it does not involve skolemization beyond the use of constants;
- it does not involve translation into large undecidable languages, working directly on modal, hybrid or description logic formulas instead;
- as far as we know, its extension to DLs is the first to account for knowledge base inference by means of a direct resolution approach;
- it is flexible and conservative in more than one sense: it allows the amalgamation of different ideas. In particular it incorporates the method of prefixes used in tableaux into resolution in such a way that different heuristics and optimizations devised in either field are applicable.

5.1 Resolution

Resolution, introduced originally for FO in [Robinson, 1965], is the most widely spread reasoning method for first-order logic today: most of the available automatic theorem provers for FO are resolution based. The elegance of the resolution method and its appeal for implementation rely on its bare simplicity.

Let us discuss the propositional case. To check whether a propositional formula φ is inconsistent, we first turn it into *clausal form*. To this aim, write φ in conjunctive normal form

$$\varphi = \bigwedge_{l \in L} \bigvee_{m \in M} \psi_{(l,m)},$$

and let the clause set associated with φ be

$$\mathit{ClSet}(\varphi) = \{\{\psi_{(l,m)} \mid m \in M\} \mid l \in L\}.$$

Now define $\mathit{ClSet}^*(\varphi)$ as the smallest set containing $\mathit{ClSet}(\varphi)$ and closed under a unique, very simple to grasp rule,

$$\frac{Cl_1 \cup \{N\} \in \mathit{ClSet}^*(\varphi) \quad Cl_2 \cup \{\neg N\} \in \mathit{ClSet}^*(\varphi)}{Cl_1 \cup Cl_2 \in \mathit{ClSet}^*(\varphi)} \text{ (RES)}.$$

If $\{\} \in \mathit{ClSet}^*(\varphi)$, then φ is inconsistent. The intuition behind the (RES) rule is as follows: given that either N or $\neg N$ is always the case in any model they can be "cut away" if the sets of clauses are conjoined. The aim of the whole method is to "cut away everything" and arrive to the empty set.

The resolution method seems to be specially devised for a dumb machine able to crunch symbols quickly. The only computational cost is a search for complementary atoms in the set of clauses. Of course, actual system implementations for first-order logic are not "dumb" at all. On the contrary, the field has developed into an extensive community, with an impressive collection of methods, optimizations, etc. [Bibel and Schmitt, 1998; Robinson and Voronkov, 2000].

In contrast, modern modal theorem provers, as well as the fastest description logic provers we mentioned in Chapter 2, are generally based on tableau methods. Strangely enough, nowadays resolution and modal languages seem to be related only when indirect methods are used. In translation-based resolution calculi for modal logics, one translates modal languages into a large background language (typically first-order logic), and devises strategies that guarantee termination for the fragment corresponding to the original modal language [Fermüller *et al.*, 1993; Hustadt, 1999; de Nivelle *et al.*, 2000; Areces *et al.*, 2000d]. First-order resolution provers like BLIKSEM or SPASS handle modal formulas in this way. This approach has both advantages and disadvantages with respect to the tableau approach. On the one hand we can translate many systems into the same background language and hence explore different, and also combined, systems without the need to modify the prover. But empirical tests show that the price to pay is high [Horrocks *et al.*, 2000a; Areces *et al.*, 2000d]. The undecidability of the full background language usually shows up in degraded performance on the modal fragments, and first-order provers can hardly emulate their tableau based competitors.

It is natural to wonder why *direct* resolution methods for modal languages don't figure in the picture. Designing resolution methods that can directly (that is, without having to perform translations) be applied to modal logics, received some attention in the late 1980s and early 1990s [Enjalbert and Fariñas del Cerro, 1989; Mints, 1989; de Nivelle, 1994]. Also the first (non-clausal) resolution methods for temporal languages go back to that period with the work of Abadi and Manna [1985]. Recently, new results on clausal temporal resolution have been presented (see [Dixon *et al.*, 2000]). But even though we might sometimes think of modal languages as a "simple extension of propositional logic," direct resolution for modal languages has proved a difficult task: in basic modal languages the resolution rule has to operate *inside* boxes and diamonds to achieve completeness. This leads to more complex systems, less elegant results, and poorer performance, ruining the one-dumb-rule spirit of resolution.

5.1.1 Direct Resolution for Modal Languages

To understand exactly how we can use hybrid and description logic ideas to improve direct modal resolution, we introduce the system presented by Enjalbert and Fariñas del Cerro in [1989]. We first provide some definitions and notation from [Enjalbert and Fariñas del Cerro, 1989], as they are not completely standard.

A modal formula is in *disjunctive normal form* if it is a (possibly empty) disjunction of the form

$$\varphi = \bigvee L_i \vee \bigvee \Box D_j \vee \bigvee \Diamond A_k,$$

where each L_i is a literal, each D_j is in disjunctive normal form, and each A_k is in conjunctive normal form. A modal formula is in *conjunctive normal form* if it is a conjunction $\varphi = \bigwedge C_i$, where each C_i is in disjunctive normal form. A formula in disjunctive normal form is called a *clause*. The empty clause is denoted as $\perp$. We identify a conjunction $C_1 \wedge \ldots \wedge C_n$ with the set $(C_1, \ldots, C_n)$. Clearly any modal formula is equivalent to a clause, and from now on we need only consider clauses.

The following examples of applications of the resolution rule "in modal contexts" are discussed in [Enjalbert and Fariñas del Cerro, 1989]

$$\frac{\Box(p \vee q) \qquad \Diamond\neg p}{\Diamond(\neg p, q)} \qquad\qquad \frac{\Box(p \vee q) \qquad \Box\neg p}{\Box q}.$$

Both inferences are sound, and are clearly instances of the (RES) rule. But if we attempt to apply a similar rule to the clauses $\Diamond(p \vee q)$ and $\Diamond\neg p$ to derive $\Diamond(\neg p, q)$ we don't preserve soundness. Also, inferences with only one premise seem to be needed, as for example

$$\frac{\Diamond(\neg p, p \vee q)}{\Diamond(\neg p, p \vee q, q)}.$$

In line with these intuitions, the following resolution system is introduced and proved complete for **K**. Define inductively two relations on clauses $\Sigma(A, B) \rightarrow C$ (C is a direct resolvent of A and B) and $\Gamma(A) \rightarrow C$ (C is a direct resolvent of A), as indicated in Figure 5.1, where α, β, κ, δ_1, δ_2 are clauses, Ψ, Φ are sets (conjunctions) of clauses, and (α, Ψ) denotes the result of appending the clause α to the set Ψ.

Define the simplification relation $A \approx B$ as the least congruence containing

Axioms	
(A1) $\Sigma(p, \neg p) \to \bot$ (A2) $\Sigma(\bot, \alpha) \to \bot$	
Σ-Rules	**Γ-Rules**
$(\vee)\ \dfrac{\Sigma(\alpha, \beta) \to \kappa}{\Sigma(\alpha \vee \delta_1, \beta \vee \delta_2) \to \kappa \vee \delta_1 \vee \delta_2}$	$(\Diamond 1)\ \dfrac{\Sigma(\alpha, \beta) \to \kappa}{\Gamma(\Diamond(\alpha, \beta, \Phi)) \to \Diamond(\alpha, \beta, \kappa, \Phi)}$
$(\Box\Diamond)\ \dfrac{\Sigma(\alpha, \beta) \to \kappa}{\Sigma(\Box\alpha, \Diamond(\beta, \Psi)) \to \Diamond(\beta, \kappa, \Psi)}$	$(\Diamond 2)\ \dfrac{\Gamma(\alpha) \to \beta}{\Gamma(\Diamond(\alpha, \Phi)) \to \Diamond(\beta, \alpha, \Phi)}$
$(\Box\Box)\ \dfrac{\Sigma(\alpha, \beta) \to \kappa}{\Sigma(\Box\alpha, \Box\beta) \to \Box\kappa}$	$(\vee)\ \dfrac{\Gamma(\alpha) \to \beta}{\Gamma(\alpha \vee \kappa) \to \beta \vee \kappa}$
	$(\Box)\ \dfrac{\Gamma(\alpha) \to \beta}{\Gamma(\Box\alpha) \to \Box\beta}$

Figure 5.1: Resolution rules

$$\begin{aligned} \Diamond\bot &\approx \bot \\ \bot \vee D &\approx D \\ (\bot, E) &\approx \bot \\ A \vee A \vee D &\approx A \vee D. \end{aligned}$$

For any formula F there is a unique F' such that $F \approx F'$ and F' cannot be simplified further. We call F' the normal form of F. C is a *resolvent* of A and B (respectively A) iff there is some C' such that $\Sigma(A, B) \to C'$ (respectively, $\Gamma(A) \to C'$) and C is the normal form of C'. We write $\Sigma(A, B) \Rightarrow C$ (respectively, $\Gamma(A) \Rightarrow C$) if C is a resolvent of A and B (respectively, of A).

Given a set of clauses S, let *ClSet*$^*(S)$ be the smallest set containing S and closed under resolvents of elements in *ClSet*$^*(S)$. We say that D is a *resolution consequence* of a set of clauses S (notation $S \vdash D$) iff $D \in$ *ClSet*$^*(S)$.

THEOREM 5.1. *For S a set of clauses and D a clause, $S \vdash D$ iff $\models_{\mathbf{K}} S \to D$.*

So much for the one-rule-spirit of resolution. Let us go through an example to better understand how this resolution method works.

EXAMPLE 5.2. Consider the formula $\Diamond(p \wedge (\neg p \vee \Box r \vee q)) \wedge \Box\neg q \wedge \Box\Diamond\neg r$. In the resolution proof below we underline the literals on which resolution takes place, and simplify many steps for succinctness.

1. $(\Diamond(\underline{p}, \underline{\neg p} \vee \Box r \vee q), \Box\neg q, \Box\Diamond\neg r)$
   by (A1), ($\vee$) and ($\Diamond$1)
2. $(\Diamond(p, \neg p \vee \Box r \vee q, \Box r \vee \underline{q}), \Box\underline{\neg q}, \Box\Diamond\neg r)$
   by (A1), ($\vee$) and ($\Box\Diamond$)
3. $(\Diamond(p, \neg p \vee \Box r \vee q, \Box r \vee q, \Box\underline{r}), \Box\neg q, \Box\Diamond\underline{\neg r})$
   by (A1) and two applications of ($\Box\Diamond$)
4. $(\Diamond(p, \neg p \vee \Box r \vee q, \Box r \vee q, \Box r, \Diamond(\neg r, \bot)), \Box\neg q, \Box\Diamond\neg r)$
   by the simplification $\Diamond\bot = \bot$, (A2) and ($\Diamond$1)
5. $\bot$

As we see, the direct resolution method for modal logics presented in [Enjalbert and Fariñas del Cerro, 1989] (and similarly those in [Fariñas del Cerro, 1982; Mints, 1989; de Nivelle, 1994] perform resolution "inside" modalities (in a similar way as how new tableaux have to be started in non-prefixed tableaux systems).

In the next sections we develop a direct resolution method for modal, description and hybrid logics that retains as much as possible of the lean one-rule character of traditional resolution methods. The key idea introduced here, from a basic modal logic perspective, is to use labels to decorate formulas with additional information. Labels allow us to make information explicit and resolution can then always be performed at the "top level." From a description or hybrid logic perspective we have just taking advantage of the new expressive power that individuals/nominals provide.

5.1.2 Labeled Resolution

In this section we introduce a direct resolution proof procedure for the basic multi-modal logic $\mathbf{K}_m$. In what follows, we assume fixed a modal similarity type $\mathcal{S} = \langle \mathsf{REL}, \mathsf{PROP} \rangle$, together with a hybrid/description logic similarity type (without state variables) $\mathcal{S}' = \langle \mathsf{REL}, \mathsf{PROP}, \mathsf{LAB} \rangle$ where LAB is a countably infinite set of nominals/individuals.

DEFINITION 5.3. [Weak negation normal form] Define the following rewriting procedure *wnnf* on modal formulas

i. $\neg\neg\varphi \overset{wnnf}{\rightsquigarrow} \varphi$,
ii. $\langle R \rangle \varphi \overset{wnnf}{\rightsquigarrow} \neg([R]\neg\varphi)$,
iii. $(\varphi_1 \vee \varphi_2) \overset{wnnf}{\rightsquigarrow} \neg(\neg\varphi_1 \wedge \neg\varphi_2)$.

For any formula φ, *wnnf* converges to a unique normal form *wnnf*(φ) which is logically equivalent to φ. If we take $\vee$ and $\langle R \rangle \varphi$ as defined operators, then *wnnf* is slightly more than an expansion of definitions.

DEFINITION 5.4. [Clauses] A *clause* is a set Cl such that each element of Cl is a *labeled formula* of the form $t : \varphi$ or $(t_1, t_2) : R$ for $t, t_1, t_2 \in \mathsf{LAB}$, $R \in \mathsf{REL}$ and φ a basic multi-modal formula. Let φ be a basic multi-modal formula. The set S_φ of clauses corresponding to φ is simply $\{\{a : wnnf(\varphi)\}\}$, for a an arbitrary label in LAB.

Notice that formulas in a clause can be seen as assertions in a description language. Let Cl be a clause, and $\mathcal{I} = \langle \Delta, \cdot^{\mathcal{I}} \rangle$ be a description logic interpretation on $\mathcal{S}'$, we write $\mathcal{I} \models Cl$ if $\mathcal{I} \models \bigvee Cl$. A set of clauses S is *satisfiable* if there is interpretation $\mathcal{I}$ such that for all $Cl \in S$, $\mathcal{I} \models Cl$.

The following proposition is straightforward,

PROPOSITION 5.5. *Let φ be a basic multi-modal formula and S_φ its corresponding set of clauses. Then φ is satisfiable iff S_φ is satisfiable.*

PROOF. For the left to right implication, given a model $\mathcal{M}$ and $m \in M$ such that $\mathcal{M}, m \Vdash \varphi$, just define $a^{\mathcal{I}} = m$ and give any interpretation to others elements in LAB. For the other direction, just drop the interpretation of elements in LAB. QED

$$(\wedge)\ \frac{Cl \cup \{t\!:\!\varphi_1 \wedge \varphi_2\}}{\begin{array}{c}Cl \cup \{t\!:\!\varphi_1\}\\ Cl \cup \{t\!:\!\varphi_2\}\end{array}} \qquad (\neg\wedge)\ \frac{Cl \cup \{t\!:\!\neg(\varphi_1 \wedge \varphi_2)\}}{Cl \cup \{t\!:\!\mathit{wnnf}(\neg\varphi_1), t\!:\!\mathit{wnnf}(\neg\varphi_2)\}}$$

$$(\mathrm{RES})\ \frac{Cl_1 \cup \{t\!:\!\varphi\} \quad Cl_2 \cup \{t\!:\!\neg\varphi\}}{Cl_1 \cup Cl_2}$$

$$([R])\ \frac{Cl_1 \cup \{t_1\!:\![R]\varphi\} \quad Cl_2 \cup \{(t_1,t_2)\!:\!R\}}{Cl_1 \cup Cl_2 \cup \{t_2\!:\!\varphi\}}$$

$$(\neg[R])\ \frac{Cl \cup \{t\!:\!\neg[R]\varphi\}}{\begin{array}{c}Cl \cup \{(t,n)\!:\!R\}\\ Cl \cup \{n\!:\!\mathit{wnnf}(\neg\varphi)\}\end{array}}, \text{ where } n \text{ is new.}$$

Figure 5.2: Labeled resolution rules

Figure 5.2 provides a set of rules transforming sets of clauses into sets of clauses.

If you read the rules with the standard translation ST of Definition 1.19 in the back of your mind, the meaning of $([R])$ and $(\neg[R])$ will be immediately clear. $([R])$ is needed to account for the "hidden" negation in the guard of the quantifier in the translation of the box, and in that sense it is indeed a standard resolution rule which cuts away complementary binary literals. On the other hand, $(\neg[R])$ can be seen as a mild kind of skolemization which only involves the introduction of constants. From this point of view we can consider the $(\wedge)$, $(\neg\wedge)$ and $(\neg[R])$ rules as preprocessing the input formula and feeding it into the resolution rules (RES) and $(\neg[R])$. Equivalently, we can view the system as intermingling the reduction towards a standard clausal form with the resolution steps as in [Fitting, 1990]. One immediate advantage of this method is that resolution can be performed not only on literals, but on complex formulas.

DEFINITION 5.6. [Deduction] A *deduction* of a clause Cl from a set of clauses S is a finite sequence $S_1, \ldots, S_n$ of sets of clauses such that $S = S_1$, $Cl \in S_n$ and each S_i (for $i > 1$) is obtained from S_{i-1} by adding the consequent clauses of the application of one of the resolution rules in Figure 5.2 to clauses in S_{i-1}. Cl is a *consequence* of S if there is a deduction of Cl from S. A deduction of $\{\}$ from S is a *refutation* of S.

The set $\mathit{ClSet}^*(S)$, defined as the smallest set containing S and all its consequences, need not be finite because the rule $(\neg[R])$ can introduce infinitely many clauses which only differ on the label. By restricting $(\neg[R])$ to be "fired only once" in a way similar as how is done for constraint systems in Table 2.2, we can ensure finiteness of $\mathit{ClSet}^*(S)$, and hence termination of the search for consequences.

Before moving on, let's redo Example 5.2 in the new resolution system. Again we underline the part of the formula where a rule applies. Notice that we are now explicitly showing *all* steps.

EXAMPLE 5.7.

1. $\{i\!:\!\underline{\neg\Box}\neg(p \wedge \neg(p \wedge \neg\Box r \wedge \neg q))\}, \{i\!:\!\Box\neg q\}, \{i\!:\!\Box\neg\Box r\},$ by $(\neg\Box)$
2. $\{R(i,j)\}, \{j\!:\!(p\underline{\wedge}\neg(p \wedge \neg\Box r \wedge \neg q))\}, \{i\!:\!\Box\neg q\}, \{i\!:\!\Box\neg\Box r\},$ by $(\wedge)$

3. $\{R(i,j)\}, \{j{:}p\}, \{j{:}\underline{\neg}(p\underline{\wedge}\neg\Box r\underline{\wedge}\neg q)\}, \{i{:}\Box\neg q\},\ \{i{:}\Box\neg\Box r\},$ by $(\neg\wedge)$
4. $\{R(i,j)\}, \{j{:}\underline{p}\}, \{j{:}\underline{\neg p}, j{:}\Box r, j{:}q\}, \{i{:}\Box\neg q\},\ \{i{:}\Box\neg\Box r\},$ by (RES)
5. $\{R(i,j)\}, \{j{:}\overline{\Box r, j{:}q}\}, \{i{:}\underline{\Box}\neg q\},\ \{i{:}\Box\neg\Box r\},$ by $(\Box)$
6. $\{j{:}\Box r, j{:}\underline{q}\}, \{j{:}\underline{\neg q}\},\ \{j{:}\neg\Box r\},$ by (RES)
7. $\{j{:}\underline{\Box r}\},\ \{j{:}\underline{\neg\Box r}\},$ by (RES)
8. $\{\}$.

It is straightforward to prove that the resolution rules in Figure 5.2 preserve satisfiability. That is, given a rule, if the premises are satisfiable, then so are the conclusions. In Section 5.2.2, we will extend the system to deal with knowledge bases in the description language $\mathcal{ALCR}$, and prove there in detail, soundness, completeness and termination.

5.2 Extensions and Variations

The system we have just introduced can be extended in different directions. In this section we discuss first how to account for modal systems different from $\mathbf{K}_m$. The next step — one which should by now come very naturally to us — is to internalize into the object language the labels we used to assist resolution. In particular, we will extend the calculus above to deal with (simple, acyclic) knowledge bases in $\mathcal{ALCR}$. Finally, we briefly discuss extensions for hybrid languages.

5.2.1 Modal Logics

From a traditional modal point of view we often want to consider systems above $\mathbf{K}_m$. We choose systems **T**, **D**, and **4** as examples. Each system is axiomatically defined as an extension of the basic system **K** by the addition of an axiom scheme which characterizes certain property of the accessibility relation.

Name	Axiom Scheme	Accessibility Relation
T	$p \to \Diamond p$	reflexivity: $\forall x.R(x,x)$
D	$\Box p \to \Diamond p$	seriality: $\forall x \exists y.R(x,y)$
4	$\Diamond\Diamond p \to \Diamond p$	transitivity: $\forall xyz.(R(x,y) \wedge R(y,z) \to R(x,z))$

Corresponding to each of the axioms we add a new resolution rule.

$$(\mathbf{T}) \quad \frac{Cl \cup \{t{:}\Box\varphi\}}{Cl \cup \{t{:}\varphi\}}$$

$$(\mathbf{D}) \quad \frac{Cl \cup \{t{:}\Box\varphi\}}{Cl \cup \{t{:}\neg\Box \mathit{wnnf}(\neg\varphi)\}}$$

$$(\mathbf{4}) \quad \frac{Cl_1 \cup \{t_1{:}\Box\varphi\} \quad Cl_2 \cup \{(t_1,t_2){:}R\}}{Cl_1 \cup Cl_2 \cup \{t_2 : \Box\varphi\}}.$$

Soundness for these systems is immediate:

THEOREM 5.8. *The resolution methods obtained by adding the rules* (**T**), (**D**) *and* (**4**), *are sound with respect to the class of models where the relation R is reflexive, serial and transitive, respectively.*

For completeness and termination we should modify the constructions in Section 5.2.2 (in particular (**4**) needs a mechanism of cycle detection); this can be done using methods from [Enjalbert and Fariñas del Cerro, 1989].

THEOREM 5.9. *The resolution methods obtained by adding the rules* (**T**), (**D**) *and* (**4**), *are complete and terminate with respect to the class of models where the relation is reflexive, serial and transitive, respectively.*

5.2.2 Description Logics

In this section we will spell out the details of a labeled resolution system to decide consistency of simple, acyclic knowledge bases in the description logic $\mathcal{ALCR}$ (see Section 2.2.2). We assume fixed a description logic signature $\langle \mathsf{CON}, \mathsf{ROL}, \mathsf{IND}\rangle$ together with an additional countable set of labels LAB.

The new definition of weak negation normal form is simply a notational variation, obtained by exchanging $\vee$ by $\sqcup$, $\wedge$ by $\sqcap$, etc. Again, for any concept C, *wnnf* always converges to a unique normal form which we denote as *wnnf*(C). The definition of clauses and set of clauses associated to a knowledge base are only slightly more involved.

DEFINITION 5.10. A *clause* is a set Cl such that each element of Cl is either a concept assertion of the form $t{:}C$ where $t \in \mathsf{IND} \cup \mathsf{LAB}$, or a role assertion of the form $(t_1, t_2){:}R$, where t_1, t_2 are in $\mathsf{IND} \cup \mathsf{LAB}$.

We will use the notation $t{:}C$ for concept assertions and $(t_1, t_2){:}R$ for role assertions. We use the notation $\bar{t}{:}N$ to refer both to concept and role assertions.

A formula in a clause is a *literal* if it is either a role assertion, a concept or negated concept assertion on an atomic concept, or a universal or negated universal concept assertion. The notions of model for a clause and for a set of clauses are as in Definition 5.4.

Let $\Sigma = \langle T, A\rangle$ be a knowledge base with simple, acyclic definitions. As we discussed in Section 2.2.2, any such knowledge base can be transformed into an "unfolded" equivalent knowledge base of the form $\langle \{\}, A\rangle$. Hence, from now on we will only consider knowledge bases with empty T-boxes.

DEFINITION 5.11. [Set of clauses of a knowledge base] The set $S_\Sigma = \langle \{\}, A\rangle$ of clauses corresponding to Σ is the smallest set such that

- if $a{:}C_1 \sqcap \cdots \sqcap C_n = wnnf(a{:}C)$ for $a{:}C \in A$ then $\{a{:}C_i\} \in S_\Sigma$,
- if $(a, b){:}R_1 \sqcap \cdots \sqcap R_n \in A$ then $\{(a, b){:}R_i\} \in S_\Sigma$.

We can identify in S_Σ a (possibly empty) subset of clauses RA of the form $\{(a, b) : R\}$ which we call *role assertions*, and for each label a a (possibly empty) subset CA_a of clauses of the form $\{a{:}C\}$ which we call *concept assertions for a*. Because of the format of a knowledge base it is impossible to find in S_Σ *mixed* clauses containing both (in disjunction) concept and role assertions. Furthermore there are no disjunctive concept assertions on different labels, i.e., there is no clause Cl in S_Σ such that $Cl = Cl' \cup \{a : C_1\} \cup \{b{:}C_2\}$ for $a \neq b$. We will take advantage of these properties in the first steps of the completeness proof.

$$(\sqcap)\quad \frac{Cl \cup \{\bar{t}\!:\!N_1 \sqcap N_2\}}{\begin{array}{c}Cl \cup \{\bar{t}\!:\!N_1\}\\ Cl \cup \{\bar{t}\!:\!N_2\}\end{array}} \qquad (\neg\sqcap)\quad \frac{Cl \cup \{t\!:\!\neg(C_1 \sqcap C_2)\}}{Cl \cup \{t\!:\!\mathit{wnnf}(\neg C_1), t\!:\!\mathit{wnnf}(\neg C_2)\}}$$

$$(\text{RES})\quad \frac{Cl_1 \cup \{\bar{t}\!:\!N\} \quad Cl_2 \cup \{\bar{t}\!:\!\neg N\}}{Cl_1 \cup Cl_2}$$

$$(\forall)\quad \frac{Cl_1 \cup \{t_1\!:\!\forall R.C\} \quad Cl_2 \cup \{(t_1,t_2)\!:\!R\}}{Cl_1 \cup Cl_2 \cup \{t_2\!:\!C\}}$$

$$(\neg\forall)\quad \frac{Cl \cup \{t\!:\!\neg\forall R.C\}}{\begin{array}{c}Cl \cup \{(t,n)\!:\!R\}\\ Cl \cup \{n\!:\!\mathit{wnnf}(\neg C)\}\end{array}},\ \text{where } n \text{ is new.}$$

Figure 5.3: Labeled resolution rules for $\mathcal{ALCR}$

Proving that Σ is consistent if and only if S_Σ has a model is straightforward. Figure 5.3 shows the labeled resolution rules, but this time recast for the language $\mathcal{ALCR}$. Before proving soundness, completeness and termination we present a simple example of resolution in our system.

EXAMPLE 5.12. Consider the following description. Ignoring some fundamental genetic laws, suppose that children of tall people are blond (1). Furthermore, all Tom's daughters are tall (2), but he has a non-blond grandchild (3). Can we infer that Tom has a son (4)?

$$\begin{array}{cc}
(0) & \mathtt{FEMALE} \doteq \neg\mathtt{MALE}\\
(1) & \mathtt{TALL} \sqsubseteq \forall\mathtt{Child.BLOND}\\
(2) & \mathtt{tom}\!:\!\forall\mathtt{Child.}(\neg\mathtt{FEMALE} \sqcup \mathtt{TALL})\\
(3) & \mathtt{tom}\!:\!\exists\mathtt{Child.}\exists\mathtt{Child.}\neg\mathtt{BLOND}\\
\hline
(4) & \mathtt{tom}\!:\!\exists\mathtt{Child.MALE.}
\end{array}$$

As is standard, we use a new proposition letter REST-TALL to complete the partial definition in (1) and we resolve with the negation of the formula we want to infer. After unfolding and applying *wnnf* we obtain the following three clauses

1. $\{\mathtt{tom}\!:\!\forall\mathtt{Child.}\neg(\neg\mathtt{MALE} \sqcap \neg((\forall\mathtt{Child.BLOND}) \sqcap \mathtt{REST\text{-}TALL}))\}$
2. $\{\mathtt{tom}\!:\!\neg\forall\mathtt{Child.}\forall\mathtt{Child.BLOND}\}$
3. $\{\mathtt{tom}\!:\!\forall\mathtt{Child.}\neg\mathtt{MALE}\}$.

Now we start resolving,

4.	$\{\mathtt{s}\!:\!\neg\forall\mathtt{Child.BLOND}\}$	by $(\neg\forall)$ in 2
5.	$\{(\mathtt{tom},\mathtt{s})\!:\!\mathtt{Child}\}$	by $(\neg\forall)$ in 2
6.	$\{\mathtt{s}\!:\!\neg\mathtt{MALE}\}$	by $(\forall)$ in 3
7.	$\{\mathtt{s}\!:\!\neg(\neg\mathtt{MALE} \sqcap \neg((\forall\mathtt{Child.BLOND}) \sqcap \mathtt{REST\text{-}TALL})\}$	by $(\forall)$ in 1
8.	$\{\mathtt{s}\!:\!\mathtt{MALE}, \mathtt{s}\!:\!((\forall\mathtt{Child.BLOND}) \sqcap \mathtt{REST\text{-}TALL})\}$	by $(\neg\sqcap)$ in 7
9.	$\{\mathtt{s}\!:\!((\forall\mathtt{Child.BLOND}) \sqcap \mathtt{REST\text{-}TALL})\}$	by (RES) in 6 and 8
10.	$\{\mathtt{s}\!:\!\forall\mathtt{Child.BLOND}\}$	by $(\sqcap)$ in 9
11.	$\{\mathtt{s}\!:\!\mathtt{REST\text{-}TALL}\}$	by $(\sqcap)$ in 9
12.	$\{\}$	by (RES) in 4 and 10.

THEOREM 5.13. [Soundness] *The rules described in Figure 5.3 are sound. That is, if Σ is a knowledge base, then S_Σ has a refutation only if Σ is unsatisfiable.*

PROOF. We prove that labeled resolution rules preserve satisfiability. We only discuss $(\neg\forall)$. Let $\mathcal{I}$ be a model of the antecedent. If $\mathcal{I}$ is a model of Cl we are done. If $\mathcal{I}$ is a model of $t:\neg\forall R.C$, then there exists d in the domain, such that $(t^{\mathcal{I}}, d) \in R^{\mathcal{I}}$ and $d \in \neg C^{\mathcal{I}}$. Let $\mathcal{I}'$ be identical to $\mathcal{I}$ except perhaps in the interpretation of n where $n^{\mathcal{I}'} = d$. As n is a new label, also $\mathcal{I}' \models t:\neg\forall R.C$. But now $\mathcal{I}' \models Cl \cup \{(t,n):R\}$ and $\mathcal{I}' \models Cl \cup \{n:wnnf(\neg C)\}$. QED

We now prove completeness. We follow the approach used in [Enjalbert and Fariñas del Cerro, 1989]: given a set of clauses S we aim to define a structure T_S such that

(†) if S is satisfiable, a model can be effectively constructed from T_S; and
(††) if S is unsatisfiable, a refutation can be effectively constructed from T_S.

But in our case we have to deal also with A-Box information, that is, with named objects (concept assertions) and fixed constraints on relations (role assertions). We will proceed in stages. To begin, we will obtain a first structure to account for named states and their fixed relation constraints. After that we can use a simple generalization of results in [Enjalbert and Fariñas del Cerro, 1989]. We base our construction on trees which will help in guiding the construction of the corresponding refutation proof.

Let Σ be a knowledge base and S_Σ its corresponding set of clauses. Let a be a label and CA_a the subset of CA of concept assertions concerning the label a. Construct inductively, for each CA_a, a binary tree T_a. Let the original tree u consist of the single node CA_a and repeat in alternation the following operations.

Operation A1. Repeat the following steps as long as possible:
- choose a leaf w. Replace any clause of the form $\{a:\neg(C_1 \sqcap C_2)\}$ by $\{a:wnnf(\neg C_1), a:wnnf(\neg C_2)\}$; and any clause of the form $\{a:C_1 \sqcap C_2\}$ by $\{a:C_1\}$ and $\{a:C_2\}$.

Operation A2. Repeat the following steps as long as possible:
- choose a leaf w of u and a clause Cl in w of the form $Cl = \{a:C_1, a:C_2\} \cup Cl'$;
- add two children w_1 and w_2 to w, where $w_1 = w\backslash\{Cl\} \cup \{\{a:C_1\}\}$ and $w_2 = w\backslash\{Cl\} \cup \{\{a:C_2\} \cup Cl'\}$.

The leaves of T_a give us the possibilities for "named states" in our model. We can view each leaf as a set S_a^j, representing a possible configuration for state a.

PROPOSITION 5.14. *Operation A (the combination of A1 and A2) terminates, and upon termination*
i. all the leaves $S_a^1, \ldots, S_a^n$ of the tree are sets of unit literal clauses,
ii. if all $S_a^1, \ldots, S_a^n$ are refutable, then CA_a is refutable,
iii. if one S_a^j is satisfiable, then CA_a is satisfiable.

PROOF. Termination is trivial. *i*) holds by virtue of the construction, and *ii*) is proved by induction on the depth of the tree. We need only realize that by simple propositional resolution, if the two children of a node w are refutable, then so is w. *iii*) is also easy. Informally, Operation A "splits" disjunctions and "carries along" conjunctions. Hence if some S_a^j has a model we have a model satisfying all conjuncts in CA_a and at least one of each disjunct. QED

We should now consider the set RA of role assertions. Let NAMES be the set of labels which appear in Σ. If a is in NAMES but CA_a is empty in S_Σ, define $S_a^1 = \{\{a : C, a : \neg C\}\}$ for some concept C. We will construct a set of sets of nodes $\mathcal{N} = \{N_i \mid N_i$ contains exactly one leaf of each $T_a\}$. Each N_i is a possible set of constraints for the named worlds in a model of S_Σ.

PROPOSITION 5.15. *If for all i, $\bigcup N_i \cup RA$ is refutable, then so is S_Σ.*

PROOF. If for all i, $\bigcup N_i \cup RA$ is refutable, then for some label a we have that for all S_a^j obtained from CA_a, $S_a^j \cup RA$ is refutable. Hence by Proposition 5.14, $CA_a \cup RA$ is refutable, and so is S_Σ. QED

For all i, we will now extend each set in N_i with further constraints. For each $S_a \in N_i$, start with a node w_a labeled by S_a.

Operation B1. Equal to *Operation A1.*

Operation B2. Repeat the following steps as long as possible:
- choose nodes w_a, w_b such that $\{(a,b):R\}$ in RA, $\{a:\forall R_i.C_i\} \in w_a$, $\{b:C_i\} \notin w_b$, where w_b is without children;
- add a child to w_b, $w_b' = w_b \cup \{\{b:C_i\}\}$.

Call N_i^* the set of all leaves obtained from the forest constructed in B.

PROPOSITION 5.16. *Operation B terminates, and upon termination*
  i. all nodes created are derivable from $\bigcup N_i \cup RA$, and hence if a leaf is refutable so is $\bigcup N_i \cup RA$,
  ii. if some $\bigcup N_i^*$ is satisfiable, then S_Σ is satisfiable.

PROOF. To prove termination, notice that in each cycle the quantifier depth of the formulas considered decreases. Furthermore, it is not possible to apply twice the operation to a node named by a and b and a formula $a:\forall R_i.C_i$.

As to i), each node is created by an application of the $(\forall)$ rule to members of $N_i \cup RA$ or clauses previously derived by such applications. To prove ii), let $\mathcal{I}$ be a model of N_i^*. Define $\mathcal{I}' = \langle \Delta', \cdot^{\mathcal{I}'} \rangle$ as $\Delta' = \Delta$, $a^{\mathcal{I}'} = a^{\mathcal{I}}$ for all labels a, $C^{\mathcal{I}'} = C^{\mathcal{I}}$ for all atomic concepts C, and $R^{\mathcal{I}'} = R^{\mathcal{I}} \cup \{(a^{\mathcal{I}}, b^{\mathcal{I}}) \mid \{(a,b):R\} \in RA\}$.

Observe that $\mathcal{I}'$ differs from $\mathcal{I}$ only in an extended interpretation of role symbols. By definition, $\mathcal{I}' \models RA$. It remains to prove that $\mathcal{I}' \models CA$. By Proposition 5.14, we are done if we prove that $\mathcal{I}' \models \bigcup N_i^*$. Since we only expanded the interpretation of relations, $\mathcal{I}$ and $\mathcal{I}'$ can only disagree on universal concepts of the form $a : \forall R.C$. By induction on the quantifier depth we prove this to be false.

Assume that $\mathcal{I}$ and $\mathcal{I}'$ agree on all formulas of quantifier depth less than n, and let $a:\forall R.C$ be of quantifier depth n, for $\{a:\forall R.C\} \in S_a^*$. Suppose $\mathcal{I}' \not\models \forall R.C$. This holds iff there exists b such that $(a^{\mathcal{I}'}, b^{\mathcal{I}'}) \in R^{\mathcal{I}'}$ and $\mathcal{I}' \not\models b:C$. By the inductive hypothesis, $\mathcal{I} \not\models b:C$. Now, if $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in R^{\mathcal{I}}$ we are done. Otherwise, by definition $\{(a,b):R\} \in RA$. But then $\{b : C\} \in S_b^*$ by construction and as $\mathcal{I} \models S_b^*$, we also have $\mathcal{I} \models b : C$ — a contradiction. QED

As we said above, each N_i^* represents the "named core" of a model of S. The final step is to define the non-named part of the model. The following operations are performed to each set in each of the N_i^* obtaining in such a way a forest F_i.

Fix N_i^*, and a. We construct a tree "hanging" from the corresponding $S_a^* \in N_i^*$. The condition that each node of the tree is named by either an individual or a new label (that is, all the formulas in a node have the same prefix) will be preserved as an invariant during the construction. Set the original tree u to S_a^* and repeat the following operations C1, C2 and C3 in succession until the end-condition holds.

Operation C1. Equal to *Operation A1.*

Operation C2. Equal to *Operation A2.*

Operation C3. For each leaf w of u,
- if for some concept we have $\{C\}, \{\neg C\} \in w$, do nothing;
- otherwise, since w is a set of unit clauses, we can write $w = \{\{t{:}C_1\}, \dots, \{t{:}C_m\}, \{t:\forall R_{k_1}.A_1\}, \dots, \{t:\forall R_{k_n}.A_n\}, \{t:\neg\forall R_{l_1}.P_1\}, \dots, \{t:\neg\forall R_{l_q}.P_q\}\}$. Form the sets $w_i = \{\{wnnf(t':\neg P_i)\}\} \cup S_i$, where t' is a new label, and $S_i = \{\{t':A_h\} \mid \{t:\forall R_i.A_h\} \in w\}$, and append each of them to w as children marking the edges as R_i links. The nodes w_i are called the *projections* of w.

End-condition. Operation C3 is inapplicable.

PROPOSITION 5.17. *Operation C cannot be applied indefinitely.*

DEFINITION 5.18. We call nodes to which Operation C1 or C2 has been applied of type 1, and those to which Operation C3 has been applied of type 2. The set of *closed nodes* is recursively defined as follows,

- if for some concept $\{t{:}C\}, \{t{:}\neg C\}$ are in w then w is closed,
- if w is of type 1 and all its children are closed, w is closed,
- if w is of type 2 and some of its children is closed, w is closed.

Let F_i be a forest that is obtained by applying Operations C1, C2, and C3 to N_i^* as often as possible. Then F_i is *closed* if any of its roots is closed.

LEMMA 5.19. *If one of the forests F_i is not closed then S_Σ has a model.*

PROOF. Let F_i be a non-closed forest. By a simple generalization of the results in [Enjalbert and Fariñas del Cerro, 1989, Lemma 2.7] we can obtain a model $\mathcal{I} = \langle\Delta, \cdot^{\mathcal{I}}\rangle$ of all roots S_a^* in F_i, from the trees "hanging" from them, ie., a model of $\bigcup N_i^*$. By Proposition 5.16, S_Σ has a model. QED

Lemma 5.19 establishes the property (†) we wanted in our structure T_S. To establish (††) we need a further auxiliary result.

PROPOSITION 5.20. *Let w be a node of type 2. If one of its projections w_i is refutable, then so is w.*

PROOF. Let w be a set of unit clauses $w = \{\{t:C_1\}, \ldots, \{t:C_m\}, \{t:\forall R_{k_1}.A_1\}, \ldots, \{t:\forall R_{k_n}.A_n\}, \{t:\neg\forall R_{l_1}.P_1\}, \ldots, \{t:\neg\forall R_{l_q}.P_q\}\}$. And let w_i be its refutable projection: $w_i = \{\{wnnf(t':\neg P_i)\}\} \cup S_i$, where t' is a new label, and $S_i = \{\{t':A_h\} \mid \{t:\forall R_i.A_h\} \in w\}$. We use resolution on w to arrive at the clauses in w_i from which the refutation can be carried out: Apply $(\neg\forall)$ to $\{t:\neg\forall R_i.P_i\}$ in w to obtain $\{t':wnnf(t':\neg P_i)\}$ and $\{(t,t'):R_i\}$. Now apply $(\forall)$ to all the clauses $\{t:\forall R_i.A_h\}$ in w to obtain $\{t':A_h\}$. QED

LEMMA 5.21. *In a forest F_i, every closed node is refutable.*

PROOF. For w a node in F_i, let $d(w)$ be the longest distance from w to a leaf.

If $d(w) = 0$, then w is a leaf, thus for some concept C, $\{t:C\}$ and $\{t:\neg C\}$ are in w. Using (RES) we immediately derive $\{\}$.

For the induction step, suppose the proposition holds for all w' such that $d(w') < n$ and that $d(w) = n$. If w is of type 1, let $w_1 = w \setminus \{Cl\} \cup \{Cl_1\}$ and $w_2 = w \setminus \{Cl\} \cup \{Cl_2\}$ be its children. By the inductive hypothesis there is a refutation for w_1 and w_2. By propositional resolution there is a refutation of w: repeat the refutation proof for w_2 but starting with w, instead of the empty clause we should obtain a derivation of Cl_2; now use the refutation of w_2. Suppose w is of type 2. Because w is closed, one of its projections is closed. Hence, by the inductive hypothesis it has a refutation. By Proposition 5.20, w itself has a refutation. QED

THEOREM 5.22. [Completeness] *The resolution method described above is complete: if Σ is a knowledge base, then S_Σ is refutable whenever Σ is unsatisfiable.*

PROOF. We only need to put together the previous pieces. If Σ does not have a model then neither does S_Σ. By Lemma 5.19 all the forests F_i obtained from S_Σ are closed, and by Lemma 5.21, for each N_i^*, one of the sets $S_{a_j}^*$ is refutable. By Proposition 5.16, for all i, $\bigcup N_i \cup RA$ is refutable. By Proposition 5.15, S_Σ is refutable. QED

Because we have shown how to *effectively* obtain a refutation from an inconsistent set of clauses we have also established termination. Notice that during the completeness proof we have used a *specific strategy* in the application of the resolution rules (crucially, the $(\neg\forall)$ rule is never applied twice to the same formula). By means of this strategy, we can guarantee termination of labeled modal resolution when verifying the consistency of any knowledge base in $\mathcal{ALCR}$.

THEOREM 5.23. [Termination] *Labeled resolution can effectively decide the consistency of simple, acyclic knowledge bases in $\mathcal{ALCR}$.*

We have spelled out in detail the method for the basic description logic $\mathcal{ALCR}$, the next natural step is to consider extensions. For instance, in [Calvanese *et al.*, 1997] some attention has been given to n-ary roles (in modal logic terms, n-ary modal operators). Our approach generalizes to this case without further problems.

Considering additional structure on roles is another possibility. We have limited ourselves to conjunction, but disjunction, negation, composition, etc. can be considered. And, of course, the addition of counting operators should be hight on our to-do list. A very attractive idea which matches nicely with the resolution approach is to incorporate a limited kind of unification on "universal labels" of the form $x\!:\!C$, to account for on the fly unfolding of definitions and more general T-Boxes. The use of such universal labels would make it unnecessary to perform a complete unfolding of the knowledge base as a pre-processing step. The leitmotiv would be "to do expansion by definitions only when needed in deduction." On the fly unfolding has already been implemented in tableaux based systems like KRIS [Baader *et al.*, 1994]. See also our discussion in Section 4.5.1.

5.2.3 Hybrid Logics

It's the turn of hybrid languages now. But of course, we have already been dealing with hybrid languages throughout the previous section: just remember the tight connections between description and hybrid logics that we built in Chapter 4.

But what about binders? Extending the system to account for hybrid sentences using $\downarrow$ is fairly straightforward. Consider the rules

$$(\downarrow)\ \frac{Cl_1 \cup \{t\!:\!\downarrow x.\varphi\}}{Cl_1 \cup \{t\!:\!\varphi[x/t]\}} \qquad\qquad (\neg\downarrow)\ \frac{Cl_1 \cup \{t\!:\!\neg\!\downarrow x.\varphi\}}{Cl_1 \cup \{t\!:\!\downarrow x.wnnf(\neg\varphi)\}}.$$

Notice that the rules transform hybrid sentences into hybrid sentences. If, in addition, we add the following rules to handle nominals

$$(\text{NOM})\ \frac{Cl_1 \cup \{t\!:\!i\} \qquad Cl_2 \cup \{i\!:\!\varphi\}}{Cl_1 \cup Cl_2 \cup \{t\!:\!\varphi\}} \qquad\qquad (\text{SYM})\ \frac{Cl \cup \{t : i\}}{Cl \cup \{i : t\}}$$

we obtain a complete calculus for sentences in $\mathcal{H}_{\mathsf{S}}(\downarrow)$. Of course, in this case we cannot expect a heuristic ensuring termination as the satisfiability problem for full $\mathcal{H}_{\mathsf{S}}(\downarrow)$ is undecidable. As we discuss in Section 4.5.4, we need strong restrictions in the language to achieve decidability like considering only sentences with non-nested occurrences of $\downarrow$ (see Theorem 7.10).

Let's work out a short example. We prove that $\downarrow x.\Diamond(x \wedge p) \rightarrow p$ is a tautology. Consider the negation of the formula in clausal form

$$\begin{array}{lll}
1. & \{i\!:\!\underline{\downarrow x}.\neg\Box\neg(x \wedge p)\}, \{i\!:\!\neg p\}, & \text{by } (\downarrow) \\
2. & \{i\!:\!\underline{\neg\Box}\neg(i \wedge p)\}, \{i\!:\!\neg p\}, & \text{by } (\neg\Box) \\
3. & \{R(i,j)\}, \{j\!:\!(i\underline{\wedge}p)\}, \{i\!:\!\neg p\}, & \text{by } (\wedge) \\
4. & \{j\!:\!\underline{i}\}, \{j\!:\!p\}, \{i\!:\!\neg p\}, & \text{by (SYM)} \\
5. & \{i\!:\!\underline{j}\}, \{j\!:\!\underline{p}\}, \{i\!:\!\neg p\}, & \text{by (NOM)} \\
6. & \{i\!:\!\underline{p}\}, \{i\!:\!\underline{\neg p}\}, & \text{by (RES)} \\
7. & \{\}. &
\end{array}$$

5.3 Reflections

In Section 2.4 we showed how constraint systems for instance checking could decide the different reasoning task we introduced in the previous sections. In Section 3.2 we argued how hybrid languages were able to internalize labeled deduction. The same ideas play a fundamental role in the labeled resolution systems we introduced in this chapter. Once again, individuals/nominals/labels together with the satisfiability operator : or @

are the key to achieve smooth and well behaved reasoning methods. And the systems we introduced in this chapter should have made clear that labeled resolution has many advantages with respect to previous direct resolution proposals, supporting our claim that description/hybrid logic ideas can indeed be used to improve reasoning methods. We complete the chapter with a discussion on a number of independent directions for future research.

Once labels are introduced the resolution method is very close to the tableaux approach, but we are still doing resolution. As we said, the rules ($\sqcap$), ($\neg\sqcap$) and ($\neg\forall$), prepare formulas to be fed into the resolution rules (RES) and ($\forall$). And the aim is still to derive the empty clause instead of finding a model by exhausting a branch. But, is this method any better than tableaux? We don't think this is the correct question to ask. We believe that we learn different things from studying different methods. For example, Horrocks and Patel-Schneider [1999] study a number of interesting optimizations of the tableaux implementation which were tested on the tableaux based theorem prover DLP. Some of their ideas can immediately be (or have already been) incorporated in our resolution method (lexical normalization and early detection of clashes, for instance), and others might perhaps be used in implementations of our method. On the other hand, optimizations for direct resolution such as those discussed in [Auffray *et al.*, 1990] can also be exploited in conjunction with the others. For example, in implementations of the resolution algorithm, strategies for selecting the resolving pairs are critical. This kind of heuristics has been investigated by Auffray *et al.* and some of their results easily extend to our framework. In certain cases, establishing completeness of these heuristics is even simpler because of our explicit use of resolution via labels.

The issue of heuristics is very much connected with complexity. The basic heuristic we used in the proof of Theorem 5.22 keeps the complete clause set "in memory" all the time and hence requires non-polynomial space. A similar situation occurs in clausal propositional resolution where the translation into clausal form can introduce an exponential blow up. We conjecture that a PSPACE heuristic for labeled resolution can be obtained by exploiting further the presence of labels (and given that we don't force a translation into full clausal form). Notice that labels and role assertions let us keep track of the accessibility relation and we can define the notion of "being a member of a branch." Now we can attempt to use the tree property of modal languages to guide resolution. We used similar ideas in [Areces *et al.*, 2000d] to improve the performance of translation based resolution provers.

The ideas behind labeled resolution are simple enough so that adapting available provers should not prove to be a very difficult task. It would be interesting to perform empirical testing on the performance of this resolution prover following the lines drawn in, for example [Horrocks *et al.*, 2000a], both in comparison with translation based resolution provers and those based on tableaux.

Finally, our completeness proof is constructive: when a refutation cannot be found we can actually define a model for the formula or knowledge base. Hence, our method can also be used for model extraction. How does this method perform in comparison with traditional model extraction from tableaux systems?