A Model–based Reasoning Approach to Circumscription

Peter Fröhlich [1] **and Wolfgang Nejdl** [1]

Abstract. In this paper we show how model–based reasoning techniques can be used to solve circumscription entailment problems without computing the circumscription axiom. We characterize the circumscription by a small subset of the minimal models computed by a revision function. Using these models a filtering function decides, whether a given formula is entailed by the circumscription. We describe, how these functions can be efficiently implemented. Finally we demonstrate the power of our approach by applying it to current formalisms from the reasoning about action and change domain.

1 Introduction

Circumscription is one of the most popular formalisms in non–monotonic reasoning. To decide, if a formula follows from the theory $\mathcal{T}$ circumscribed in P, most approaches compute the circumscription axiom and (if possible) reduce it to a first order sentence [8]. Then monotonic logic is used to do the prove. Another approach is to characterize the semantics of circumscription by minimal models. This characterization has been used by Ginsberg to create an ATMS–based circumscriptive theorem prover [6]. In this paper we present a more direct approach for exploiting this characterization, which uses model–based reasoning techniques to compute minimal models of a given theory. By dividing the minimal models into equivalence classes we only need to compute a representative subset of the minimal models. These few models are then used by a filtering function to decide, if a given formula φ follows from circumscription. The proof is done by showing that there is no minimal model, in which $\neg\varphi$ holds. This refutation idea was also used by Przymusinski [11], whose query answering method is based on resolution. We provide efficient algorithms for our approach which make use of the model–based reasoning techniques introduced by Chou and Winslett [2]. We show that these algorithms are sound and complete for fixed–domain theories [9]. Our approach avoids problems with the symbolic manipulation and reduction of the circumscription axiom and provides an efficient way to handle current applications. We underline this by implementing current formalisms for reasoning about action and change in our system.

2 The Model–based Approach

2.1 Definition of Minimal Models

We assume familiarity with circumscription in this paper. For a model M and a predicate symbol K we write $M|K|$ to denote the extension of K in M. In [10], McCarthy defines Formula Circumscription, which allows the extensions of certain predicates to vary during minimization. We will first consider the special case of circumscribing a theory $\mathcal{T}$ in a predicate P, while varying the extensions of all other predicates and generalize this to other variants of circumscription is section 4. The semantics for this case of circumscription can be characterized by the $<_P$ minimal models of $\mathcal{T}$.

Definition 2.1 *Consider models M_1 and M_2. $M_1 \leq_P M_2$, iff*

1. *M_1 and M_2 have the same domain and agree in the interpretation of the constant symbols.*
2. *$M_1|P| \subseteq M_2|P|$*

*We write $M <_P N$, iff $M \leq_P N$ and not $N \leq_P M$. A model M is called a $<_P$–*minimal model *of the theory $\mathcal{T}$, iff there is no model N of $\mathcal{T}$, such that $N <_P M$. We will denote the set of $<_P$–minimal models of $\mathcal{T}$ by MinModels$_P(\mathcal{T})$.*

The following theorem found by Lifschitz [8] shows the connection between circumscription and the set of $<_P$–minimal models.

Theorem 1 *The formula φ follows from the circumscription of $\mathcal{T}$ in P, while varying all other predicates, iff φ holds in all $<_P$–minimal models of $\mathcal{T}$.*

Thus, if we knew all $<_P$–minimal models of a given theory, we could answer queries concerning the circumscribed theory just by looking at the models. However, computing and storing all $<_P$–minimal models is usually very inefficient. We will describe a method for answering queries which needs only a few models, by introducing an equivalence relation on models.

2.2 Computing Minimal Models

Two models are defined to be equivalent wrt. P, if the extension of P is the same in both models.

Definition 2.2 *Two models M and M' are P–*equivalent *(denoted by $M \sim_P M'$), iff $M|P| = M'|P|$. By $[M]$, we denote the set of all models P–equivalent to M, i.e. $[M] = \{M' | M \sim_P M'\}$. For a set $\mathcal{M}$ of models we define $\mathcal{M}/{\sim_P} := \{[M] | M \in \mathcal{M}\}$.*

Each equivalence class is represented by (stored as) some model $M \in [M]$. Thus we work on finite sets of models representing $MinModels_P(\mathcal{T})/{\sim_P}$.

Definition 2.3 *A set $\mathcal{M}$ of models is a* Transversal *of MinModels$_P(\mathcal{T})/{\sim_P}$, iff it contains exactly one model out of every equivalence class in MinModels$_P(\mathcal{T})/{\sim_P}$.*

Now consider a (possibly empty) set $\mathcal{T}$ of formulas for which we have a transversal $\mathcal{M}$ of the minimal models. We want to insert new knowledge φ into the theory $\mathcal{T}$ and thus obtain a transversal of the minimal models of $\mathcal{T} \cup \{\varphi\}$. We call a function, which computes this new transversal, a *revision function*.

---

[1] University of Hannover, Lange Laube 3, 30159 Hannover, Germany
{froehlich, nejdl}@kbs.uni–hannover.de

ECAI 96. *12th European Conference on Artificial Intelligence*
Edited by W. Wahlster
Published in 1996 by John Wiley & Sons, Ltd.

Definition 2.4 *Consider a first order language $\mathcal{L}$, where $\mathcal{I}_\mathcal{L}$ is the set of all finite interpretations. Let $\mathcal{C}$ be a class of theories in $\mathcal{L}$. Let f_P be a function, which takes a theory, a set of models and a formula as parameters and produces a set of models: $f_P : \mathcal{C} \times 2^{\mathcal{I}_\mathcal{L}} \times \mathcal{L} \to 2^{\mathcal{I}_\mathcal{L}}$.*

f_P is called a Revision Function *for $\mathcal{C}$, iff for all $\mathcal{T} \in \mathcal{C}$, $\varphi \in \mathcal{L}$ (such that $\mathcal{T} \cup \{\varphi\} \in \mathcal{C}$) and all transversals $\mathcal{M} \subseteq \mathcal{I}_\mathcal{L}$ of MinModels$_P(\mathcal{T})/{\sim_P}$:*

$f_P(\mathcal{T}, \mathcal{M}, \varphi)$ is a transversal of MinModels$_P(\mathcal{T} \cup \{\varphi\})/\sim_P$.

In section 3 we will introduce an efficient revision function Rev_P and show its completeness for a large class of theories. The rest of this section is dedicated to the question of how to use a revision function to compute minimal models and decide entailment under circumscription. Our first observation is that we can use the revision function directly to compute a transversal of the minimal models by executing the revision $Rev_P(\emptyset, \{\emptyset\}, \bigwedge_{F \in \mathcal{T}} F)$. This method will be used in the PMON example (see section 5.1). In other applications we want to compute minimal models in multiple steps. That is, we already have a transversal $\mathcal{M}$ of the minimal models for a part $\mathcal{T}_0$ of the theory and we use $Rev_P(\mathcal{T}_0, \mathcal{M}, \bigwedge_{F \in \mathcal{T}_1} F)$ to compute the minimal models of $\mathcal{T}_0 \cup \mathcal{T}_1$. One reason for doing so can be the appearance of new knowledge. In this case we want to reuse the minimal models of the old theory to compute the new minimal models.

Another reason for computing minimal models in several steps is efficiency. In reasoning about action and change we can for example use standard model generation techniques to create a model of the static world, without regarding the action axioms. Then we revise this model with the action axioms. This technique is shown in section 5.2 using Baker's approach to non–monotonic reasoning. We also use our circumscription method for diagnosis and value prediction in model–based diagnosis. In this area we can use simulation tools to construct an initial model.

2.3 Deciding Entailment under Circumscription

Until now, we defined a method for computing the equivalence classes of the minimal models of a theory $\mathcal{T}$. These equivalence classes can be used to decide entailment under circumscription. The decision procedure is an application of theorem 1 by Lifschitz: Before we describe the general decision procedure, note that answering queries concerning P is now trivial, since a transversal of the $<_P$–minimal models contains a model for each minimal extension of P.

Proposition 2.5 *Let P be a predicate symbol of arity n and $\mathcal{M}$ a transversal of MinModels$_P(\mathcal{T})/{\sim_P}$. $P(x_1, \ldots, x_n)$ follows from the circumscription of $\mathcal{T}$ in P varying all other predicates, iff $\forall M \in \mathcal{M} : M \models P(x_1, \ldots, x_n)$*

To prove that an arbitrary formula φ is entailed by the circumscription of $\mathcal{T}$ in P, we show that $\neg\varphi$ does not hold in any $<_P$–minimal model of $\mathcal{T}$. Our method for deciding entailment makes use of a *Filtering Function*. We can filter a set of equivalence classes with a formula φ by eliminating all equivalence classes, which do not contain a model of φ.

Definition 2.6 *Consider a language $\mathcal{L}$ and a class of theories $\mathcal{C}$ in $\mathcal{L}$. Let $\mathcal{T}$ be a theory, φ a formula and $\mathcal{M}$ a set of models.*

$$\Pi_P(\mathcal{T}, \mathcal{M}, \varphi) := \{[M] | M \in \mathcal{M} \wedge \exists N \in [M] : N \models \varphi\}$$

A function $f_P : \mathcal{C} \times 2^{\mathcal{I}_\mathcal{L}} \times \mathcal{L} \to 2^{\mathcal{I}_\mathcal{L}}$ is called a Filtering Function *for $\mathcal{C}$, iff for all $\mathcal{T} \in \mathcal{C}$, $\varphi \in \mathcal{L}$ (such that $\mathcal{T} \cup \{\varphi\} \in \mathcal{C}$) and for every transversal $\mathcal{M}$ of MinModels$_P(\mathcal{T})/{\sim_P}$:*

$f_P(\mathcal{T}, \mathcal{M}, \varphi)$ is a transversal of $\Pi_P(\mathcal{T}, M, \varphi)$.

A filtering function $Filter_P$ will be defined in section 3.5 as a simplified version of the revision function Rev_P. Using the filtering function, we can decide whether an arbitrary formula φ follows from the circumscription by first computing a transversal of the minimal models and then filtering with $\neg\varphi$. If no equivalence class remains after the filtering, there is by definition no minimal model, in which $\neg\varphi$ holds. Thus, φ follows from the circumscription by theorem 1.

Theorem 2 *Let f_P be a filtering function for a class $\mathcal{C}$ of theories. Let $\mathcal{T} \in \mathcal{C}$ be a theory, $\mathcal{M}$ a transversal of MinModels$_P(\mathcal{T})/{\sim_P}$ and φ a formula such that $\mathcal{T} \cup \{\varphi\} \in \mathcal{C}$.*

φ follows from the circumscription of $\mathcal{T}$ in P while varying all other predicates, iff $f_P(\mathcal{T}, \mathcal{M}, \neg\varphi) = \emptyset$.

Detailed proofs for all results presented in this paper can be found in the longer technical report [5]. Recently, new methods for reasoning about action and change have been introduced [12, 7] which first minimize a certain predicate in a part of the theory (usually the domain axioms) and then filter the resulting models using another part of the theory (usually the observations). This can be formalized in our approach by applying the filtering operation twice. The first filtering uses the observations to prune the inappropriate models from the set of minimal models. To prove a query φ, a second filtering step with $\neg\varphi$ is used. See sections 5.1 and 5.2 for examples.

3 Algorithms for Revision and Filtering

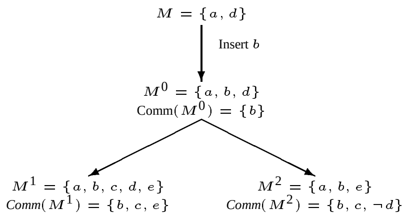
In the previous section we did not specify how to compute $Rev_P(\mathcal{T}, \mathcal{M}, \varphi)$ and $Filter_P(\mathcal{T}, \mathcal{M}, \varphi)$. In this section we define algorithms for computing these functions and show their completeness with respect to fixed–domain theories. The material in this section was inspired by Chou and Winsletts work on implementing model–based belief revision [2].

3.1 The Language

Two limitations are inherent in the model–based paradigm. (1) We can not handle infinite models. (2) We can not handle infinitely many models. Both problems are avoided, if we limit ourselves to languages without function symbols and theories which include the unique name assumption and the domain closure assumption. Such theories are called *Fixed–Domain Theories*. In fixed–domain theories all models are isomorphic to Herbrand models [9], thus it is sufficient to consider only the minimal Herbrand models in theorem 1, when using fixed–domain theories. Our algorithms handle two extensions: functions symbols with Herbrand interpretation and infinite sorts. When using infinite sorts, we have to make sure that existential quantifiers only range over finite intervals, otherwise limitation (2) would be violated. The extensions are useful for reasoning with situation calculus and formalizing infinite integer time.

3.2 Removing Inconsistencies

Consider a given model M of a theory $\mathcal{T}$. We want to augment $\mathcal{T}$ by new information. At this stage we explain how to add a literal L (positive or negative) to the theory. Later we will generalize this to the case of adding an arbitrary formula φ. As an example, we consider the theory $\mathcal{T} := \{a, b \to c \vee \neg d, a \wedge b \to e\}$ with a given model $M := \{a, d\}$. We want to revise this theory by adding the new literal b. Our revision is exhaustive in the sense that M is repaired in all possible ways. In our example we would do the following:



In the figure above we first inserted the new literal b into M giving M^0. We denote the set of truth values which we changed in model N compared to the initial model by $Comm(N)$ (**Comm**itted literals), e.g. $Comm(M^0) = \{b\}$ in the above example. Obviously model M^0 contradicts $\mathcal{T}$ because the formulas $b \to c \vee \neg d$ and $a \wedge b \to e$ are not satisfied. We create models M^1 and M^2 by changing M^0 in a minimal way, so that these inconsistencies are removed. Let us formalize this repair step.

Definition 3.1 *An* Inconsistency *for a theory $\mathcal{T}$ is a minimal set I of ground literals such that $\mathcal{T} \cup I$ is inconsistent.*

Inconsistencies in Herbrand models can be found efficiently [5]. M^0 contains the inconsistencies $I_1 = \{b, \neg c, d\}$ and $I_2 = \{a, b, \neg e\}$. By definition, no model of $\mathcal{T} \cup \{b\}$ contains either I_1 or I_2. Consequently, any model of $\mathcal{T} \cup \{b\}$ differs from M^0 in at least one literal from I_1 and at least one literal from I_2. We generate several models, in which we change the truth values of some literals to remove the current inconsistencies. We never change the truth value of a given literal twice. Furthermore it never makes sense to change a truth value, which is already determined by the facts in the theory.

Definition 3.2 *The* Reduced Inconsistency I' *for an inconsistency I is obtained by removing all committed literals and all literals, which are part of the theory $\mathcal{T}$, from I.*

In our example the reduced inconsistencies are $I_1' = \{\neg c, d\}$ and $I_2' = \{\neg e\}$ To change at least one literal per reduced inconsistency we first construct *hitting sets* of the reduced inconsistencies, i.e. sets which contain one literal from each reduced inconsistency, then we negate the truth values of all literals in the hitting set.

Definition 3.3 *A* hitting set *for a set of (reduced) inconsistencies $\{I_1, \ldots, I_n\}$ is a set HS, such that $\forall k \in \{1, \ldots, n\} : HS \cap I_k \neq \emptyset$. A hitting set is called* minimal*, iff there is no hitting set HS', such that $HS' \subset HS$.*

In our example the hitting sets are $\{\neg c, \neg e\}$ and $\{d, \neg e\}$. We define the repair step as a non–deterministic function $Step(\mathcal{T}, M)$, which produces a set of new models by negating the truth values of the literals in the minimal hitting set of the reduced inconsistencies.

Definition 3.4 *Let $\mathcal{T}$ be a theory and M an interpretation.*

1. *If $M \models \mathcal{T}$: Step$(\mathcal{T}, M) := \emptyset$.*
2. *If $M \not\models \mathcal{T}$:*
   (a) *If $\emptyset$ is a reduced minimal inconsistency, then Step$(\mathcal{T}, M) := \{\emptyset\}$.*
   (b) *If all reduced inconsistencies are non–empty, let $\{I_1, \ldots I_n\}$ be the set of reduced minimal inconsistencies of M wrt. $\mathcal{T}$.*

Step$(\mathcal{T}, M) := \{M' | M'$ is obtained from M by inverting the truth values of a minimal hitting set of $\{I_1, \ldots, I_n\}\}$

Case 2(a) corresponds to an inconsistency, which can no longer be repaired, because all literals in the inconsistency are already committed. It is sufficient for exhaustive repair to consider only the minimal hitting sets of the inconsistencies. The reason is that any model of $\mathcal{T}$ must differ from a given inconsistent model in (a superset of) one of the minimal hitting sets.

3.3 Revision Algorithm

Our method consists of applying a sequence of repair steps to every model $M \in \mathcal{M}$ and discarding the non-minimal models. By executing the repair steps in a best–first order we limit the generation of non–minimal models.

To repair a model M our algorithm iteratively applies the $Step$–function. It maintains a set WM of inconsistent models and a set S of consistent models (Solutions). Initially $WM = \{M\}$ and $S = \emptyset$. The algorithm now selects a $<_P$–minimal inconsistent model $\hat{M}$ from WM, i.e. $\hat{M}$ has the property that $\nexists M' \in WM \cup S : M' <_P \hat{M}$. $\hat{M}$ is deleted from WM and $Step(\mathcal{T}, \hat{M})$ is computed.

If $Step(\mathcal{T}, \hat{M}) = \emptyset$ then $\hat{M}$ is consistent, i.e. $\hat{M} \models \mathcal{T}$. $\hat{M}$ is inserted into the solutions set and can be used to delete non–minimal models from WM: All models, in which the extension of P is equivalent to or larger than in $\hat{M}$ can be deleted, because we want to obtain only one solution for each minimal $\sim_P$-class. If $\mathcal{N} := Step(\mathcal{T}, \hat{M}) \neq \emptyset$, then the models from $\mathcal{N}$ are inserted into WM. If $Step(\mathcal{T}, \hat{M}) = \{\emptyset\}$, no further repair step can be applied to the inconsistent model $\hat{M}$ and it is discarded. The revision of M is complete, if $WM = \emptyset$. This algorithm ($RepairModel$) has to be applied to every model in the set of initial models $\mathcal{M}$. Finally, another $<_P$–minimality check is performed on all result models.

Algorithm 3.5 *Rev$_P(\mathcal{T}, \mathcal{M}, L)$*

FUNCTION *Rev$_P(\mathcal{T}, \mathcal{M}, L)$*
Solutions $:= \emptyset$;
FOR EACH $M \in \mathcal{M}$ DO
    Solutions $:=$ *Solutions* $\cup$ *RepairModel*$(\mathcal{T}, M, L)$;
FOR EACH $S \in$ *Solutions* DO
    IF $\exists S' \in$ *Solutions*$\backslash S : S' <_P S$ THEN
        Solutions $:=$ *Solutions*$\backslash S$;
RETURN *Solutions*;

FUNCTION *RepairModel*$(\mathcal{T}, M, L)$
Insert L into M giving M^0;
WM $:= \{M^0\}$; $S := \emptyset$;
WHILE *WM* $\neq \emptyset$ DO
    Select a $<_P$–minimal model $\hat{M}$ from *WM*;
    WM $:=$ *WM*$\backslash\{\hat{M}\}$;
    $\mathcal{N} :=$ *Step*$(\mathcal{T} \cup \{L\}, \hat{M})$;
    IF $\mathcal{N} = \emptyset$ THEN
        IF $\nexists M \in S : M <_P \hat{M}$ THEN
            $S := S \cup \hat{M}$;
            WM $:=$ *WM*$\backslash\{M \in$ *WM* $: \hat{M} <_P M \vee \hat{M} \sim_P M\}$;
    ELSE IF $\mathcal{N} \neq \{\emptyset\}$ THEN *WM* $:=$ *WM* $\cup\, \mathcal{N}$;
    ELSE Discard $\hat{M}$
RETURN S;

Rev_P can be easily generalized to revision with arbitrary formulas. Let $\mathcal{T}$ be a theory and $\mathcal{M}$ be a transversal of $MinModels_P(\mathcal{T})/{\sim_P}$. We want to revise $\mathcal{T}$ with a new formula φ. Let L be a literal, which occurs neither in $\mathcal{T}$ nor in $\mathcal{M}$, and $\mathcal{T}' := \mathcal{T} \cup \{L \to \varphi\}$. Since L does not occur in $\mathcal{M}$, $\mathcal{M}$ is also a transversal of $MinModels_P(\mathcal{T}')/{\sim_P}$.

Instead of revising $\mathcal{T}$ with φ, we can obtain a transversal of $MinModels_P(\mathcal{T} \cup \{\varphi\})$ by revising $\mathcal{T}'$ with L.

3.4 Properties of the Algorithm

In this section we give correctness and completeness results for fixed domain theories developed in [5]. We start by stating that all models generated by algorithm 3.5 are minimal models.

Proposition 3.6 *Let $\mathcal{T}$ a fixed–domain theory, φ a formula without function symbols and $\mathcal{M}$ a transversal of MinModels$_P(\mathcal{T})/\sim_P$. Then Rev$_P(\mathcal{T}, \mathcal{M}, \varphi) \subseteq$ MinModels$_P(\mathcal{T} \cup \{\varphi\})$.*

Additionally Rev_P is complete for fixed–domain theories:

Proposition 3.7 *Let $\mathcal{T}$ be a fixed–domain theory, φ a formula without functions symbols and $\mathcal{M}$ a transversal of MinModels$_P(\mathcal{T})/\sim_P$. Then for every $M \in$ MinModels$_P(\mathcal{T})$ there exists $N \in$ Rev$_P(\mathcal{T}, \mathcal{M}, \varphi)$, such that $M \sim_P N$.*

Propositions 3.6 and 3.7 together with the observation that at most one representative is computed for every equivalence class show that Rev_P computes a transversal of the minimal models and consequently is a revision function for fixed domain theories.

Theorem 3 *Rev$_P$ is a revision function for fixed–domain theories.*

Algorithm 3.5 also computes minimal Herbrand models in the presence of functions symbols and infinite sorts. When we convert it from a best first to a breadth first algorithm by changing the management of the node list WM we can show that it always finds a set of finite minimal Herbrand models, if it exists.

3.5 Filtering Algorithm

For the filtering we use the same $Step$–repair function as for the revision. Instead of minimizing the extension of P, we now hold the previously minimized extension of P fixed, by regarding the extension of P as part of the theory. It is sufficient to find one repaired consistent model for each minimal model given.

Algorithm 3.8 *Filter$_P(\mathcal{T}, \mathcal{M}, L)$*

FUNCTION *Filter$_P(\mathcal{T}, \mathcal{M}, L)$*
Solutions $:= \emptyset$;
FOR EACH $M \in \mathcal{M}$ DO
  $\mathcal{T}' := \mathcal{T} \cup \bigcup_{\vec{x}:M \models P(\vec{x})} P(\vec{x}) \cup \bigcup_{\vec{x}:M \not\models P(\vec{x})} \neg P(\vec{x})$;
  Solutions := *Solutions* $\cup$ *RepairModel*$'(\mathcal{T}', M, L)$;
RETURN *Solutions*;

FUNCTION *RepairModel*$'(\mathcal{T}', M, L)$
Insert L into M giving M^0;
WM $:= \{M^0\}$; $S := \emptyset$;
WHILE *WM* $\neq \emptyset$ DO
  Select a model $\hat{M}$ from *WM*;
  WM := *WM*$\backslash\{\hat{M}\}$;
  $\mathcal{N}$:= *Step*$(\mathcal{T}' \cup \{L\}, \hat{M})$;
  IF $\mathcal{N} = \emptyset$ THEN $S := \{\hat{M}\}$; *WM* $:= \emptyset$;
  ELSE IF $\mathcal{N} \neq \{\emptyset\}$ THEN *WM* := *WM* $\cup\, \mathcal{N}$;
  ELSE Discard $\hat{M}$;
RETURN S;

For the filtering algorithm we can show analogous results as for the revision algorithm:

Theorem 4 *Filter$_P$ is a filtering function for fixed–domain theories.*

4 Variants of Circumscription

Parallel Circumscription of predicates $P_1; \ldots; P_n$ can be realized simply by replacing the order $<_P$ by $<_{P_1;\ldots;P_n}$ in the implementation of our revision function. This order is defined by

Definition 4.1 *Let M_1 and M_2 be models. $M_1 \leq_{P_1;\ldots;P_n} M_2$, iff*

1. *M_1 and M_2 have the same domain and agree in the interpretation of the constant symbols.*
2. *For all $i \in \{1, \ldots, n\}$: $M_1|P_i| \subseteq M_2|P_i|$.*

We can show that our revision function can minimize using any order that is expressible as set–inclusion of a set of atoms. Since $M_1 <_{P_1;\ldots;P_n} M_2$, iff $\bigcup_{i\in\{1,\ldots,n\}} M_1|P_i| \subset \bigcup_{i\in\{1,\ldots,n\}} M_2|P_i|$ our approach works fine with parallel circumscription.

Fixed predicates can be replaced by varying predicates using de Kleer's method [3]: Instead of holding predicate Q fixed during the minimization of P, we define $Q'(\vec{x}) \equiv \neg Q(\vec{x})$, and then minimize P, Q and Q' in parallel. *Prioritized Circumscription* can be directly handled in our approach by multiple revisions.

5 Examples

We have implemented both Sandewall's and Baker's approaches to reasoning about action and change. Furthermore we implemented Kartha's [7] extension to Baker's approach as well as diagnosis and prediction problems. We also solved examples from Ginsberg [6]. All examples can be found in [5]. The first two are also described in the following sections, because they underline different aspects of our approach. Both examples discussed in this paper are solved in less than 0.5 seconds, in general our Prolog implementation computes approximately 100 repair steps per second on a SparcStation 4. A more detailed evaluation is contained in [5].

5.1 PMON–Circumscription

In this section we show how to implement Sandewall's approach to reasoning about action and change [12]. We use PMON–circumscription (recently discussed by Doherty [4]). This example underlines the importance of reasoning with equivalence classes and the efficient use of fixed predicates. We will use the Russian Turkey Shoot Scenario: There are two fluents, a (alive) and l (loaded) and three actions, $Load$, $Spin$ (spinning the guns chamber) and $Fire$. The gun was loaded between timepoint 1 and 2, the spinning action performed between 3 and 4 and the gun fired between 5 and 6.

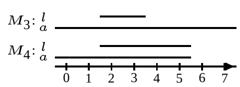
In Sandewalls framework the effect of actions is described by *Reassignment Formulas*, e.g. the loading action is described by $[1;2]l := T$ which means that the fluent l is assigned the value true during [1;2]. Reassignment formulas can include a condition. The firing action is described by the formula $[5]l \to [5,6](\neg a := T \wedge \neg l := T)$ denoting that a and l are false after the shooting if the gun was loaded at time 5. Before reasoning, the reassignment formulas are transformed into first order logic. Two predicates are used: $Holds(t, f)$, (fluent f holds at time t) and $Occlude(t, f)$ (fluent f may change its value at time point t as effect of an action). PMON–Circumscription first minimizes $Occlude$ while holding $Holds$ fixed. Then the resulting models are filtered with the observations and the nochange axiom, which states that a fluent f can only change its value at time t, if $Occlude(t, f)$ holds. This reasoning pattern can be directly simulated in our approach. However, fixing the whole extension of $Holds$ leads to a large amount of incomparable models. We extend the transformation to first order logic by generating a literal $Precond(I)$ for each conditional reassignment formula. The truth value of $Precond(I)$ is coupled with the truth value of the condition in the reassignment formula. In our example we generate such a literal for the $Fire$–action: $Precond(1) \equiv Holds(5, l)$. We obtain the correct results by holding $Precond$ fixed instead of $Holds$. The translation yields theory SCD:

$$\begin{aligned}
&\exists t(1 \le t < 2) \wedge \forall t'(t < t' \le 2 \rightarrow Holds(t', l)) \\
&\quad \wedge \forall t'(1 < t' \le 2 \rightarrow Occlude(t', l)) \\
&\exists t(3 \le t < 4) \wedge \forall t'(t < t' \le 4 \rightarrow (Holds(t', l) \vee \neg Holds(t', l))) \\
&\quad \wedge \forall t'(3 < t' \le 4 \rightarrow Occlude(t', l)) \\
&Holds(5, l) \rightarrow ((\exists t.5 \le t < 6 \wedge \forall t'(t \le t' < 6 \rightarrow \neg Holds(t', a))) \\
&\qquad \wedge \forall t'(5 < t' \le 6 \rightarrow Occlude(t', a)) \wedge \\
&\qquad (\exists t.5 \le t < 6 \wedge \forall t'(t \le t' < 6 \rightarrow \neg Holds(t', l))) \\
&\qquad \wedge \forall t'(5 < t' \le 6 \rightarrow Occlude(t', l))) \\
&Precond(1) \equiv Holds(5, l) \\
&Precond(1) \equiv \neg Precond'(1)
\end{aligned}$$

We first compute $Rev_{Occlude;Precond,Precond'}(\emptyset, \{\emptyset\}, SCD)$ and obtain two result models:

$$\begin{aligned}
M_1: &\quad \{Holds(2, l), Occlude(2, l), Occlude(4, l), Precond'(1)\} \\
M_2: &\quad \{Holds(2, l), Holds(5, l), Occlude(2, l), Occlude(4, l), \\
&\quad Occlude(6, a), Occlude(6, l), Precond(1)\}
\end{aligned}$$

By treating $Precond$ as fixed, we have obtained one model, in which the gun is loaded at time 5 and one, in which it is not loaded. These two models are representatives of two large equivalence classes of models, whose members differ in the extensions of $Holds$. Our equivalence class approach avoids storing all these models. The next reasoning step is filtering with the observations and the nochange axiom. We compute $Filter_{Occlude;Precond,Precond'}(SCD, \{M_1, M_2\}, OBS \cup NCP)$. Both equivalence classes survive the filtering. The remaining representatives are shown below:



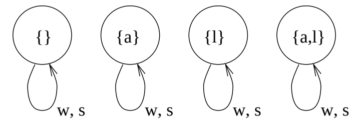
The two equivalence classes now only contain the two intended chronicle completions. If we ask whether the turkey is dead at time 6 by executing $Filter_{Occlude;Precond,Precond'}(SCD \cup OBS \cup NCP, \{M_3, M_4\}, \neg Holds(6, a))$, model M_4 remains. As intended, the query cannot be proved and the system remains unspecific about the question, whether the turkey stays alive. On the other hand, if we ask whether the gun is unloaded at time 6, the system successfully proves this query (both models M_3 and M_4 are eliminated through filtering with $Holds(6, l)$).

5.2 Baker's Formalism

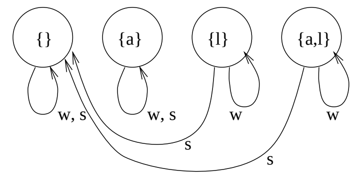
Baker proposed an approach for non–monotonic reasoning in the situation calculus [1]. He extends previous approaches by introducing an existence of situations axiom, which guarantees the existence of a situation for every consistent combination of truth values of the fluents. Additionally Baker varies the $Result$–function instead of $Holds$. We implement Baker's approach using a language with three sorts: $\mathcal{A}$ for actions, $\mathcal{F}$ for fluents and $\mathcal{S} := 2^{\mathcal{F}}$ for situations, which are characterized by the fluents that hold.[1] The usual $Result$–function of the situation calculus is replaced by a predicate $Result$, for which we postulate that it is functional by

$$\forall a \forall s \exists s'(Result(a, s, s') \wedge \forall s''(Result(a, s, s'') \rightarrow s' = s''))$$

Consider the Yale Shooting Problem with fluents a (Alive) and l (Loaded) and the actions W (Wait) and S (Shoot). We start reasoning in a trivial model M_0 of the domain axioms without the description of the actions, i.e. in a completely inert world, where the actions lead to no change in the truth values of the fluents:



No abnormality has to be assumed because all fluents are allowed to persist. Now we revise this model with the axiom ACT of the shoot action ($Rev_{Ab}(\mathcal{T} \backslash ACT, M_0, ACT)$). This axiom states that the turkey is dead and the gun is unloaded after shooting with a loaded gun. The following model M_1 is detected to be the minimal one.



Now we prove that the turkey is dead, when we start with a loaded gun and a living turkey then wait and then shoot.

$$\varphi : Result(W, \{a, l\}, s_1) \wedge Result(S, s_1, s_2) \rightarrow \neg Holds(a, s_2)$$

We prove this by $Filter_{Ab}(\mathcal{T}, \{M_1\}, \neg\varphi)$, which returns $\emptyset$.

REFERENCES

[1] A. B. Baker. Nonmonotonic reasoning in the framework of the situation calculus. *Artificial Intelligence*, 49:5–23, 1991.

[2] T. S.-C. Chou and M. Winslett. A model–based belief revision system. *Journal of Automated Reasoning*, 12:157–208, 1994.

[3] J. de Kleer. Eliminating the fixed predicates from a circumscription. *Artificial Intelligence*, 39(3):391–398, 1990.

[4] P. Doherty. Reasoning about action and change using occlusion. In *Proceedings of the 11th European Conference on Artificial Intelligence*, pages 401–405, 1994.

[5] P. Fröhlich and W. Nejdl. A model-based reasoning approach to circumscription. Technical report, Universität Hannover, Jan. 1996. http://www.kbs.uni-hannover.de/bib/bibliography.html.

[6] M. L. Ginsberg. A circumscriptive theorem prover. *Artificial Intelligence*, 39:209–230, 1989.

[7] G. N. Kartha. Two counterexamples related to bakers approach to the frame problem. *Artificial Intelligence*, 69:379–391, 1994.

[8] V. Lifschitz. Computing circumscription. In *Proc. of the Intl. Conf. on Artificial Intelligence*, pages 121–127, Los Angeles, Aug. 1985.

[9] W. Lukaszewicz. *Non–monotonic reasoning: formalization of commonsense reasoning*. Ellis Horwood, 1990.

[10] J. McCarthy. Applications of circumscription to formalizing commonsense knowledge. *Artificial Intelligence*, 28:89–118, 1986.

[11] T. C. Przymusinki. An algorithm to compute circumscription. *Artificial Intelligence*, 38:49–73, 1989.

[12] E. Sandewall. *Features and Fluents. The representation of Knowledge about Dynamical Systems. Volume I.* Oxford University Press, 1994.

[1] Certain fluent–combinations can be inconsistent with the domain axioms. These can be eliminated by an additional circumscription step [5]