Minimal belief and negation as failure: A feasible approach

Antje Beringer and **Torsten Schaub**
FG Intellektik, TH Darmstadt, Alexanderstraße 10,
D-6100 Darmstadt, Germany
{antje,torsten}@intellektik.informatik.th-darmstadt.de

Abstract

Lifschitz introduced a logic of minimal belief and negation as failure, called MBNF, in order to provide a theory of epistemic queries to nonmonotonic databases. We present a feasible subsystem of MBNF which can be translated into a logic built on first order logic and negation as failure, called FONF. We give a semantics for FONF along with an extended connection calculus. In particular, we demonstrate that the obtained system is still more expressive than other approaches.

Introduction

Lifschitz [1991; 1992][1] introduced a logic of minimal belief and negation as failure, MBNF, in order to provide a theory of epistemic queries to nonmonotonic databases. This approach deals with self-knowledge and ignorance as well as default information.

From one perspective, MBNF relies on concepts developed by Levesque [1984] and Reiter [1990] for database query evaluation. In these approaches, databases are treated as first order theories, whereas queries may also contain an epistemic modal operator. In addition to query-answering *from* a database, this modal operator allows for dealing with queries *about* the database. From another perspective, Lifschitz' approach relies on the system GK developed by Lin and Shoham [1990], which uses two epistemic operators accounting for the notion of "minimal belief" and "negation as failure". Thus, MBNF can be seen as an extension of GK, which identifies their epistemic operator for minimal belief with the ones used by Levesque and Reiter.

MBNF is very expressive. Apart from asking what a database knows, it permits expressing default knowledge and axiomatizing the closed world assumption [Reiter, 1977] and integrity constraints [Kowalski, 1978]. Furthermore, Lifschitz [1992] established close relationships to logic programming, default logic [Reiter, 1980] and circumscription [McCarthy, 1980].

However, Lifschitz' approach is purely semantical and mainly intended to provide a unifying framework for several nonmonotonic formalisms. Consequently, there is no proof theory yet. We address this gap by identifying a subsystem of MBNF and translating it into a feasible system by relying on the fact that in many cases negation as failure is expressive enough to account for the different modalities in MBNF. The resulting system is called FONF (first order logic with negation as failure). We demonstrate that it provides a versatile approach to epistemic query answering for nonmonotonic databases, which are first order theories enriched by beliefs and default statements. Furthermore, we give a clear semantics of FONF along with an extended connection calculus for FONF. Also, we show that FONF is still more expressive than PROLOG and a competing approach [Reiter, 1990].

[1] In what follows, we rely on the more recent approach.

Minimal belief and negation as failure

MBNF deals with an extended first order language including two independent modal operators B and not. B is of epistemic nature and represents the notion of "minimal belief", whereas not captures the notion of "negation as failure". A theory T, or *database*, is a set of sentences. α, β denote sentences; F, G denote formulas. A *positive* formula (or theory) does not contain not. An *objective* one contains neither B nor not.

For instance, given an ornithological database, we can formalize the default that "we believe that birds fly, unless there is evidence to the contrary", as

$$\forall x(\mathsf{B}\mathit{bird}(x) \wedge \mathsf{not}\neg \mathit{fly}(x) \rightarrow \mathsf{B}\mathit{fly}(x)).$$

Now, the idea is to interpret our beliefs by a set of "possible worlds", ie. $\mathsf{B}\mathit{bird}(\mathit{Tweety})$ is true iff Tweety is a bird in all possible worlds and $\mathsf{not}\neg\mathit{fly}(\mathit{Tweety})$ is true iff Tweety flies in some possible world.

Formally, the truth of a formula is defined wrt a triple $(w, W_{\mathsf{B}}, W_{\mathsf{not}})$, where w is a first order interpretation, or simply *world*, representing "the real world", W_{B} is a set of "possible worlds" defining the meaning of beliefs formalized with B, and W_{not} serves for the same purpose in case of not. w, W_{B} and W_{not} share the same universe, but W_{B} and W_{not} do not necessarily include w. Intuitively, this means that beliefs need not be consistent with reality. Thus, Tweety may be believed to fly without actually flying.

Then, the truth of a formula F in MBNF is defined for the language of MBNF extended by names for all elements of the common universe.[2]

1. For atomic F, $(w, W_{\mathsf{B}}, W_{\mathsf{not}}) \models_{\text{MBNF}} F$ iff $w \models F$.
2. $(w, W_{\mathsf{B}}, W_{\mathsf{not}}) \models_{\text{MBNF}} \neg F$ iff $(w, W_{\mathsf{B}}, W_{\mathsf{not}}) \not\models_{\text{MBNF}} F$.
3. $(w, W_{\mathsf{B}}, W_{\mathsf{not}}) \models_{\text{MBNF}} F \wedge G$ iff $(w, W_{\mathsf{B}}, W_{\mathsf{not}}) \models_{\text{MBNF}} F$ and $(w, W_{\mathsf{B}}, W_{\mathsf{not}}) \models_{\text{MBNF}} G$.
4. $(w, W_{\mathsf{B}}, W_{\mathsf{not}}) \models_{\text{MBNF}} \exists X F(X)$ iff for some name ξ, $(w, W_{\mathsf{B}}, W_{\mathsf{not}}) \models_{\text{MBNF}} F(\xi)$.
5. $(w, W_{\mathsf{B}}, W_{\mathsf{not}}) \models_{\text{MBNF}} \mathsf{B}F$ iff $\forall w' \in W_{\mathsf{B}} : (w', W_{\mathsf{B}}, W_{\mathsf{not}}) \models_{\text{MBNF}} F$.
6. $(w, W_{\mathsf{B}}, W_{\mathsf{not}}) \models_{\text{MBNF}} \mathsf{not}F$ iff $\exists w' \in W_{\mathsf{not}} : (w', W_{\mathsf{B}}, W_{\mathsf{not}}) \not\models_{\text{MBNF}} F$.

The definition of a model in MBNF is restricted to the case where $W_{\mathsf{B}} = W_{\mathsf{not}}$. Therefore, Lifschitz introduces *structures* (w, W), where w is a world and W a set of worlds corresponding to W_{B} and W_{not}. In particular, he is only interested in $<$-maximal structures, where $(w, W) < (w', W')$ iff $W \subset W'$, since they express *"the idea of 'minimal belief': The larger the set of 'possible worlds' is, the fewer propositions are believed"* [Lifschitz, 1992]. Formally, a model in MBNF is defined by means of a fixed-point operator $\Gamma(T, W)$, which, given a theory T and a set of worlds W, denotes the set of all $<$-maximal structures (w, W') such that T is true in (w, W', W). Then, a structure (w, W) is an MBNF-*model* of T iff $(w, W) \in \Gamma(T, W)$.

In MBNF, theoremhood is only defined for positive formulas: A positive formula F is entailed by T, $T \models_{\text{MBNF}} F$, iff F is true in all models of T. Thus, query answering is also restricted to positive queries.[3] Notice that models of T need not be models of F. For instance, $\mathsf{B}p$ is true in all models of $\mathsf{B}(p \wedge q)$ and, hence, $\mathsf{B}(p \wedge q) \models_{\text{MBNF}} \mathsf{B}p$. However, no MBNF-model of $\mathsf{B}(p \wedge q)$ is an MBNF-model of $\mathsf{B}p$, since none of them is $<$-maximal in satisfying $\mathsf{B}p$. So, we have to distinguish carefully between formulas in a given theory and formulas being posed as queries to that theory.

FONF: A feasible approach to MBNF

We develop a feasible approach to MBNF by identifying a large subclass of MBNF, which allows for equivalent formalizations in first order logic plus negation as failure. This is the case whenever a theory T is complete for believed sentences, ie. whenever we have either $T \models_{\text{MBNF}} \mathsf{B}\alpha$ or $T \models_{\text{MBNF}} \neg\mathsf{B}\alpha$ for each α. In this case, first order logic with negation as failure is strong enough to capture also the notion of "minimal belief".

We thus identify a feasible subclass of MBNF for which we provide a translation into FONF, a first order logic with an additional negation as failure operator **not**. This translation preserves the above notion of completeness in the sense, that an MBNF-theory is complete for believed sentences iff the corresponding FONF-theory is.

As mentioned above, we have to distinguish between queries and sentences in a database. Accordingly, we define the following feasible subset of MBNF for queries and databases separately:

- A *feasible query* is an MBNF-formula q satisfying:
  1. q is positive.
  2. Each scope of an $\exists$ or $\neg$ in q is either purely subjective or purely objective.
  3. If the scope of a $\neg$ in q is subjective, then it must not contain free variables.
- A *feasible database* (*FDB*) is an MBNF-theory containing only *rules* of the form

  $$F_1 \wedge \ldots \wedge F_m \wedge \mathsf{not}\ F_{m+1} \wedge \ldots \wedge \mathsf{not}\ F_n \rightarrow F_{n+1}$$

  where for $n, m \geq 0$ each F_i $(i = 1, \ldots, n+1)$ is
  - either a disjunction-free MBNF-formula where the scope of $\neg$ is minimal and objective,[4]
  - or of the form $\mathsf{B}(G_1 \vee \ldots \vee G_k)$ where the G_i $(i = 1, \ldots, k)$ are objective formulas.

  F_{n+1} may also be an unrestricted objective formula.

These restrictions are not as strong as it seems at first sight: Even default rules, integrity constraints, and closed world axioms can be formalized within *FDB*s.

In [Lifschitz, 1992] an MBNF-formula F is translated into a first order formula F° to relate MBNF- and first order entailment: A second sort of so-called "world variables" is added to the first order language; appending one of them to each function and predicate symbol (as an additional argument), and introducing a unary predicate B whose argument is such a world variable. A world variables denotes the world in which a certain predicate or function symbol is interpreted and B accounts for the "accessibility" of a world from the actual world. However, the translation $^\circ$ is insufficient for creating a deduction method for MBNF. First, it deals only with positive formulas and, therefore, discards a substantial half of MBNF: The modal operator **not**. Second, only first order entailment carries over to MBNF but not vice versa. That is, roughly speaking, even for positive T and α, $T^\circ \models \alpha^\circ$ implies $T \models_{\text{MBNF}} \alpha$ but not vice versa. In this sense, the translation $^\circ$ is sound but incomplete.

Our approach addresses this shortcoming by translating feasible queries and databases into FONF. This has the following advantages: First, we deal with a much larger subset of MBNF. In particular, we can draw nonmonotonic conclusions by expressing B and **not** by a first order predicate *bel* and a negation as failure operator **not**. Second, our translation is truth-preserving. That is, for feasible queries and databases, FONF-entailment carries over to MBNF *and* vice versa.

[2] $\models$ without any subscript denotes first order entailment.

[3] As regards MBNF-queries, we rely on this restriction throughout the paper.

[4] Observe that one cannot distribute $\neg$ over B.

In this sense, the translation is sound and complete for feasible queries and databases. In the sequel, we give this translation and prove that it is truth-preserving.

Now, FONF-*formulas* are all formulas that can be built using the connectives and construction rules of first order logic and the unary operator not. The only constraint on FONF-formulas is that variables must not occur free in the scope of not.

The translation * of feasible MBNF-queries and -databases into FONF-formulas is developed in analogy to [Lifschitz, 1992]. We use the predicate *bel* to translate the MBNF-operator B. Then, a feasible MBNF-formula F, ie. either a feasible query or a formula belonging to a *FDB*, is translated into the FONF-formula $F^\star$ in the following way.

- If F is objective, then

1. $F^\star$ is obtained by appending the world variable V to each function and predicate symbol in F.

- else (ie. if F is non-objective)

2. $(\neg F)^\star = \mathtt{not}\ F^\star$.
3. $(F \circ G)^\star = F^\star \circ G^\star$ for $\circ = \wedge, \vee$ or $\rightarrow$.
4. $(\mathcal{Q}\ F)^\star = \mathcal{Q}\ F^\star$ for $\mathcal{Q} = \exists$ or $\forall$.
5. $(\mathsf{B}F)^\star = \forall V\ (bel(V) \rightarrow F^\star)$.
6. $(\mathsf{not}\ F)^\star = \mathtt{not}(\mathsf{B}F)^\star$.

Observe that feasible queries must not contain not, so that then Condition 6 does not apply. The translation * depends on the notion of feasible formulas which obey syntactical restrictions. Thus, we have to account for all connectives. As an example, translating $\neg\alpha \vee \neg\mathsf{B}\beta$ (α, β objective) into FONF yields $\neg\alpha^\star \vee \mathtt{not}(\forall V\ (bel(V) \rightarrow \beta^\star))$, which shows that the combination $\neg\mathsf{B}$ is translated using negation as failure, namely not, whereas pure negation $\neg$ is kept.

In order to show that this translation is truth-preserving, we look at the semantics of FONF and define satisfiability wrt a set of worlds W:

$$W \models_{\text{FONF}} \alpha \text{ iff } \forall w \in W: (w, W) \models_{\text{FONF}} \alpha,$$

where the truth value of a FONF-formula wrt a structure (w, W) is defined in the following way:

- If F is objective, then

1. $(w, W) \models_{\text{FONF}} F$ iff $w \models F$.

- else (ie. if F is non-objective)

2. $(w, W) \models_{\text{FONF}} \neg F$ iff $(w, W) \not\models_{\text{FONF}} F$.
3. $(w, W) \models_{\text{FONF}} F \wedge G$ iff $(w, W) \models_{\text{FONF}} F$ and $(w, W) \models_{\text{FONF}} G$.
4. $(w, W) \models_{\text{FONF}} \mathtt{not} F$ iff $\exists w' \in W$: $(w', W) \not\models_{\text{FONF}} F$.

FONF can be seen as an extension of extended logic programs [Gelfond and Lifschitz, 1990]. Accordingly, FONF-models extend the semantics of extended logic programs to the first order case: For a FONF-theory T, and a set of worlds W, we develop a set of objective formulas T^W from T by

1. deleting all rules, where not α occurs in the body while $W \models_{\text{FONF}} \alpha$ holds.
2. deleting all remaining subformulas of the form not α.

Then, W is a FONF-*model* for T if it consists of all first order models of T^W. A sentence α is entailed by a FONF-theory T, $T \models_{\text{FONF}} \alpha$, iff α is true in all FONF-models of T. Then, we obtain the equivalence between query-answering in MBNF and FONF for *FDB*s.

Theorem 1 *For feasible* MBNF-*databases* T *and feasible* MBNF-*queries* α : $T \models_{\text{MBNF}} \alpha$ *iff* $T^\star \models_{\text{FONF}} \alpha^\star$.

As a corollary, we get that the translation * preserves completeness for believed sentences in the above sense. The translation proposed in [Lifschitz, 1992] satisfies only one half of the above result since it only provides completeness for "monotonically answerable queries". Moreover, Lifschitz deals with a more restricted fragment of MBNF, which excludes, for example, the use of negation as failure for database sentences.

A connection calculus for FONF

We develop a calculus for FONF based on the connection method [Bibel, 1987], an affirmative method for proving the validity of a formula in disjunctive normal form (DNF). These formulas are displayed two-dimensionally in the form of *matrices.* A matrix is a set of sets of literals. Each element of a matrix represents a *clause* of a formula's DNF. In order to show that a sentence α is entailed by a sentence T, we have to check whether $\neg T \vee \alpha$ is valid. In the connection method this is accomplished by path checking: A *path* through a matrix is a set of literals, one from each clause. A *connection* is an unordered pair of literals with the same predicate symbol, but different signs. A connection is *complementary* if both literals are identical except for their sign. Now, a formula, like $\neg T \vee \alpha$, is valid iff each path through its matrix contains a complementary connection under a global substitution.

First, we extend the definition of matrices and literals. An *NM-Literal* is an expression not M, where M is an *NM*-matrix. An *NM-matrix* is a matrix containing normal and *NM*-literals. Although these structures seem to be rather complex, we only deal with normal form matrices, since *NM*-literals are treated in a special way during the deduction process.

The definition of classical normal form matrices relies on the DNF of formulas. Here, we deal with formulas in *disjunctive* FONF-*normal form* (FONF-DNF) by treating subformulas like not α as atoms while transforming formulas into DNF. Then these α's are transformed recursively into FONF-DNF and so forth. An *NM*-matrix M_F represents a quantifier-free FONF-formula F in FONF-DNF as follows.

1. If F is a literal then $M_F = \{\{F\}\}$.
2. If $F = \mathtt{not}\ G$ then $M_F = \{\{\mathtt{not}\ M_G\}\}$.
3. If $F = F_1 \wedge \ldots \wedge F_n$ then $M_F = \{\bigcup_{i=1}^{n} M_{F_i}\}$.

4. If $F = F_1 \vee \ldots \vee F_n$ then $M_F = \bigcup_{i=1}^{n} M_{F_i}$.

For instance, $p(X) \vee (q(a) \wedge \texttt{not}\ (r(Y) \wedge q(Y) \vee p(a)))$ has the following matrix representation:

$$\left[\begin{array}{ccc} p(X) & & q(a) \\ & \texttt{not} & \left[\begin{array}{cc} r(Y) & p(a) \\ q(Y) & \end{array}\right] \end{array}\right]$$

In order to define a nonmonotonic notion of complementarity, we introduce so-called *adjunct matrices*. They are used for resolving the complex structures in *NM*-matrices during the deduction process. Given an *NM*-matrix $M = \{C_1, \ldots, C_n\}$ where $C_n = \{L_1, \ldots, L_m, \texttt{not}\ N\}$ the adjunct matrix M_N is defined as $M_N = (M \setminus C_n) \cup N$, or two-dimensionally:

$$\begin{array}{cc} \mathrm{M} & \mathrm{M}_N \\ \left[\begin{array}{cccc} \vdots & & \vdots & L_1 \\ C_1 & \cdots & C_{n-1} & \vdots \\ \vdots & & \vdots & \texttt{not}\ [\mathrm{N}] \end{array}\right] & \left[\begin{array}{cccc} \vdots & & \vdots & \\ C_1 & \cdots & C_{n-1} & \mathrm{N} \\ \vdots & & \vdots & \end{array}\right] \end{array}$$

An *NM*-matrix is *NM-complementary* if each path through the matrix is *NM*-complementary. A path p is *NM*-complementary if

- p contains a connection $\{K, L\}$ which is complementary under unification or
- p contains an *NM*-literal $\texttt{not}\ N$ with N being an *NM*-matrix. If the adjunct matrix M_N is not *NM*-complementary, then p is *NM*-complementary.

The deduction algorithm relies on the standard connection method, except that if a path contains an *NM*-literal then the same deduction algorithm is started recursively with the corresponding adjunct matrix. Let M be an *NM*-matrix and let $\mathcal{P}^M$ be the set of all paths through M. Then the *NM*-complementarity of M is checked by checking all paths in $\mathcal{P}^M$ for *NM*-complementarity. This is accomplished by means of the procedure nmc in the following informal way for a set of paths $\mathcal{P}'$ and a matrix M'.

$\texttt{nmc}(\mathcal{P}', M')$

- If $\mathcal{P}' = \emptyset$, then $\texttt{nmc}(\mathcal{P}', M')$= "yes"
- else choose $p \in \mathcal{P}'$.
  - If p is classically complementary with connection $\{K\sigma, L\sigma\}$ and unifier σ, then $\texttt{nmc}(\mathcal{P}', M') = \texttt{nmc}((\mathcal{P}' - \{p \mid \{K, L\} \in p\})\sigma, M')$
  - else
    - if there exists an *NM*-literal $\texttt{not}\ N \in p$ such that $\texttt{nmc}(\mathcal{P}^{M_N}, M_N)$= "no", then $\texttt{nmc}(\mathcal{P}', M')$=$\texttt{nmc}(\mathcal{P}' - \{p \mid \texttt{not}\ N \in p\}, M')$
    - else $\texttt{nmc}(\mathcal{P}', M')$= "no"

Initially, nmc is called with $\mathcal{P}^M$ and M, namely $\texttt{nmc}(\mathcal{P}^M, M)$. Then, we obtain the following result.

Proposition 1 *If the algorithm terminates with "yes", then the NM-matrix is NM-complementary.*

So far, we have only considered quantifier-free FONF-formulas in FONF-DNF. But how can an arbitrary FONF-formula F be treated within this method? First of all, F must be transformed into FONF-*Skolem normal form*, analogously to [Bibel, 1987]. We denote the result of FONF-skolemization of a formula F by $\mathcal{S}(F)$.

Now, FONF-formulas can be represented as matrices. If we have a FONF-database, we require that rules containing free variables in the scope of **not** (which is actually not allowed in FONF) are replaced by the set of their ground instances before skolemization.

However, the above algorithm has its limitations due to its simplicity. First, in MBNF and FONF, it is necessary to distinguish between sentences occurring in the database and those serving as queries. Representing a database together with a query as an *NM*-matrix removes this distinction. Second, the above algorithm cannot deal with FONF-theories possessing multiple FONF-models. This requires a separate algorithmic treatment of alternative FONF-models.

We address this shortcoming by slightly restricting the definition of feasible queries and databases instead of providing a much more complicated algorithm. Thus, we introduce *determinate* queries and databases.[5] Determinate queries are feasible MBNF-queries in DNF which are either objective or consist only of one non-objective disjunct. Determinate databases are *FDB*s which do not contain circular sets of rules, like $\{\mathsf{not}\ p \rightarrow \mathsf{B}q,\ \mathsf{not}\ q \rightarrow \mathsf{B}p\}$, because only such circular sets cause multiple models in FONF. This restriction is not as serious as it might seem at first sight: First, non-objective disjunctive queries can be posed by querying the single disjuncts separately. Second, we doubt that we loose much expressiveness by forbidding circular rules. So, if we consider this kind of databases and queries, we can use the nonmonotonic connection method for query-answering:

Proposition 2 *For determinate* FONF*-theories T and* FONF*-sentences α: $T \models_{\text{FONF}} \alpha$ iff the NM-matrix of $\mathcal{S}(\neg T \vee \alpha)$ is NM-complementary.*

Together with Theorem 1, we obtain the following.

Theorem 2 *For determinate* MBNF*-databases T and determinate* MBNF*-queries α: $T \models_{\text{MBNF}} \alpha$ iff the NM-matrix of $\mathcal{S}(\neg T^\star \vee \alpha^\star)$ is NM-complementary.*

Hence, we obtain a deduction method for a quite large subset of MBNF: Given a determinate MBNF-database T and -query α, we check whether $T \models_{\text{MBNF}} \alpha$ holds by

1. translating T into $T^\star$ and α into $\alpha^\star$,
2. replacing free variables with all constants occurring in the database,
3. skolemizing $\neg T^\star \vee \alpha^\star$ yielding $\mathcal{S}(\neg T^\star \vee \alpha^\star)$,
4. testing the resulting matrix for *NM*-complementarity.

[5] This expression will also be used for the corresponding FONF-queries and -databases.

Finally, let us consider an example illustrating our approach. Consider the following MBNF-database T:

$$\begin{array}{l} \mathsf{B}(\mathit{teaches}(\mathit{anne},\mathit{bio}) \vee \mathit{teaches}(\mathit{sue},\mathit{bio})) \\ \mathsf{not}\ \mathit{teaches}(X,\mathit{bio}) \rightarrow \neg\ \mathit{teaches}(X,\mathit{bio}) \end{array}$$

which we write in short notation as

$$\mathsf{B}(t(a,b) \vee t(s,b)) \wedge (\mathsf{not}\ t(X,b) \rightarrow \neg t(X,b)).$$

Recall that the second conjunct is considered as an abbreviation for the set of its ground instances. Consider the following query:

Is it true that Anne doesn't teach biology?

This query, say α, is formalized in MBNF as $\neg t(a,b)$. Notice that T and α constitute determinate MBNF-expressions. Now, we have to verify whether $T \models_{\text{MBNF}} \alpha$ holds. According to the closed world axiom given by not $t(X,b) \rightarrow \neg t(X,b)$, saying that a person does not teach biology unless proven otherwise, we expect a positive answer.

Following the four steps above, we first translate T and α into FONF and obtain the FONF-theory $T^\star$

$$\begin{array}{l} \forall V\ \mathit{bel}(V) \rightarrow t(a,b,V) \vee t(s,b,V) \\ \texttt{not}\ [\forall V\ \mathit{bel}(V) \rightarrow t(X,b,V)] \rightarrow \neg t(X,b,V) \end{array}$$

along with the query $\alpha^\star = \neg t(a,b,V)$. Then, the theory is negated yielding $\neg T^\star$. After replacing X by the constants a, s and b[6] in $\neg T^\star$, we obtain $\mathcal{S}(\neg T^\star \vee \alpha^\star)$ by FONF-skolemization which is (with Skolem constants w_i $(i = 1,2,3)$)

$$\begin{array}{l} \mathit{bel}(V) \wedge \neg t(a,b,V) \wedge \neg t(s,b,V) \\ \texttt{not}\ [\neg \mathit{bel}(w_2) \vee t(a,b,w_2)] \wedge t(a,b,w_1) \\ \texttt{not}\ [\neg \mathit{bel}(w_3) \vee t(s,b,w_3)] \wedge t(s,b,w_1) \\ \neg t(a,b,w_1). \end{array}$$

This FONF-formula has the following matrix representation (if we ignore the drawn line)

$$\begin{bmatrix} \mathit{bel}(V) & \texttt{not}\ N_1 & \texttt{not}\ N_2 & \neg t(a,b,w_1) \\ \neg t(a,b,V) & t(a,b,w_1) & t(s,b,w_1) & \\ \neg t(s,b,V) & & & \end{bmatrix}$$

with the submatrices $N_1 = [\neg \mathit{bel}(w_2)\quad t(a,b,w_2)]$ and $N_2 = [\neg \mathit{bel}(w_3)\quad t(s,b,w_3)]$.

It remains to be checked whether this matrix is *NM*-complementary in order to prove $T \models_{\text{MBNF}} \alpha$. The first connection starting from the query $\neg t(a,b,w_1)$ is shown by the drawn line. It remains to be tested, if all paths through the *NM*-literal $\texttt{not}\ N_1$ are *NM*-complementary. So, the adjunct matrix has to be built yielding the following matrix (with N_2 as above), which must not be *NM*-complementary for a successful proof.

$$\begin{bmatrix} \mathit{bel}(V) & \neg \mathit{bel}(w_2) & t(a,b,w_2) & \texttt{not}\ N_2 & \neg t(a,b,w_1) \\ \neg t(a,b,V) & & & t(s,b,w_1) & \\ \neg t(s,b,V) & & & & \end{bmatrix}$$

[6]For simplicity, we omit the last case in this example, as it obviously does not contribute to the proof.

During the proof for the above adjunct matrix a copy of the first clause has to be generated. We get the substitution $\sigma = \{V_1 \backslash w_1, V_2 \backslash w_2\}$, where V_1 occurs in the first copy and V_2 in the second one. The resulting matrix contains the (non-complementary) path

$$\begin{array}{lll} \{\neg t(a,b,w_1), & \neg t(s,b,w_2), & \neg \mathit{bel}(w_2), \\ \ t(a,b,w_2), & t(s,b,w_1), & \neg t(a,b,w_1)\}. \end{array}$$

The first two literals stem from the two copies of the first clause of the adjunct matrix. The four others belong to the remaining clauses of the adjunct matrix.

Since this path through the adjunct matrix is not complementary, the *NM*-literal not N_1 in the original matrix is *NM*-complementary. Therefore, all paths through the original matrix are *NM*-complementary. Accordingly, we have proven that $T \models_{\text{MBNF}} \alpha$ holds and thus that Anne doesn't teach biology.

In order to illustrate the difference between queries to and about the database in presence of the closed world assumption (CWA), consider the query, say β,

Is it known that Anne doesn't teach biology?

Now, we expect a negative answer, since the used formalization of the CWA only affects objective formulas. It avoids merging propositions about the world (like objective formulas in the database) and propositions about the database, which causes inconsistencies when using the "conventional" CWA [Reiter, 1977] in the presence of incomplete knowledge.

β is formalized in MBNF as $\mathsf{B}\neg t(a,b)$ yielding the skolemized FONF-formula $\neg \mathit{bel}(w_4) \vee \neg t(a,b,w_4)$. Treating β and the above formula T according to the four aforementioned steps results in the following matrix with submatrices N_1 and N_2 as defined above:

$$\begin{bmatrix} \mathit{bel}(V) & \texttt{not}\ N_1 & \texttt{not}\ N_2 & \neg \mathit{bel}(w_4) & \neg t(a,b,w_4) \\ \neg t(a,b,V) & t(a,b,w_1) & t(s,b,w_1) & & \\ \neg t(s,b,V) & & & & \end{bmatrix}$$

Looking at the three paths without *NM*-literals, it can be easily seen that at least one of them will never be complementary (regardless of how V is instantiated). Consequently, the matrix is not *NM*-complementary, which tells us that β is not an MBNF-consequence of T. That is, even though we were able to derive that Anne does not teach biology, we cannot derive that it is *known* that Anne does not teach biology. This is because the used closed world axioms merely affect what is derivable and not what is known. However, observe that the opposite query, namely $\neg\mathsf{B}\neg t(a,b)$ is answered positively. Certainly, we could obtain different answers by using different closed world axioms.

The above algorithm has been implemented using a PROLOG implementation of the connection method. The program takes *NM*-matrices and checks whether they are *NM*-complementary. It consists only of five PROLOG-clauses. Interestingly, the first four clauses constitute a full first order theorem prover, and merely the fifth clause deals with negation as failure. This extremely easy way of implementation is a benefit of the restriction to determinate queries and -databases.

Conclusion

MBNF [Lifschitz, 1992] is very expressive and thus very intractable. Therefore, we have presented a feasible approach to minimal belief and negation as failure by relying on the fact that in many cases negation as failure is expressive enough to capture additionally the (nonmonotonic) notion of minimal belief. We have identified a substantial subsystem of MBNF: Feasible databases along with feasible queries. This subsystem allows for a truth-preserving translation into FONF, a first order logic with negation as failure. However, feasibility has its costs. For instance, FONF does not allow for "quantifying-in" **not**. Also, we have given a semantics of FONF by extending the semantics of extended logic programs [Gelfond and Lifschitz, 1990].

We have developed an extended connection calculus for FONF, which has been implemented in PROLOG. To our knowledge, this constitutes the first connection calculus integrating negation as failure. We wanted to keep our calculus along with its algorithm as simple as possible, so that it can be easily adopted by existing implementations of the connection method, like SETHEO [Letz *et al.*, 1992]. The preservation of simplicity has resulted in the restriction to determinate theories, which possess only single FONF-models. This restriction is comparable with the one found in extended logic programming, where one restricts oneself to well-behaved programs with only one model.

As a result, we can compute determinate queries to determinate databases in MBNF. This subset of MBNF is still expressive enough for many purposes: Apart from asking what a database knows, determinate queries and databases allow for expressing default rules, axiomatizing the closed world assumption and integrity constraints. Also, it seems that the restriction to determinate queries and databases can be dropped in the presence of a more sophisticated algorithm treating multiple FONF-models separately.

Moreover, our approach is still more expressive than others: First, FONF is more expressive than PROLOG: Since it is build on top of a first order logic, it allows for integrating disjunctions and existential quantification. Second, Reiter [1990] has proposed another approach, in which databases are treated as first order theories, whereas queries may include an epistemic modal operator. As shown in [Lifschitz, 1992], this is equivalent to testing whether $\mathsf{B}T \models_{\text{MBNF}} \mathsf{B}F$ holds for objective theories T and positive formulas F of MBNF. Obviously, Reiter's approach is also subsumed by determinate queries and databases, so that we can use our approach to implement his system as well.

Although we cannot account for MBNF in its entirety, our approach still deals with a very expressive and, hence, substantial subset of MBNF. Moreover, from the perspective of conventional theorem proving, our translation has shown how the epistemic facet of negation as failure can be integrated into automated theorem provers. To this end, it is obviously possible to implement FONF by means of other deduction methods, like resolution. In particular, it remains future work to compare resolution-based approaches to negation as failure to the approach presented here.

Acknowledgements

We thank S. Brüning, M. Lindner, A. Rothschild, S. Schaub, and M. Thielscher for useful comments on earlier drafts of this paper. This work was supported by *DFG, MPS (HO 1294/3-1)* and by *BMfT, TASSO (ITW 8900 C2)*.

References

Bibel, W. 1987. *Automated Theorem Proving*, Vieweg, 2nd edition.

Gelfond, M. and Lifschitz, V. 1990. Logic programs with classical negation. In Proc. *International Conference on Logic Programming*. 579–597.

Kowalski, R. 1978. Logic for data description. In Gallaire, H. and Minker, J., eds., In Proc. *Logic and Databases*, Plenum. 77–103.

Letz, R.; Bayerl, S.; Schumann, J.; and Bibel, W. 1992. SETHEO: A high-performance theorem prover. *Journal on Automated Reasoning*.

Levesque, H. 1984. Foundations of a functional approach to knowledge representation. *Artificial Intelligence* 23:155–212.

Lifschitz, V. 1991. Nonmonotonic databases and epistemic queries. In Myopoulos, J. and Reiter, R., eds., In Proc. *International Joint Conference on Artificial Intelligence*, Morgan Kaufmann 381–386.

Lifschitz, V. 1992. Minimal belief and negation as failure. Submitted.

Lin, F. and Shoham, Y. 1990. Epistemic semantics for fixed–points nonmonotonic logics. In Parikh, Rohit, ed., In Proc. *Theoretical Aspects of Reasoning about Knowledge*. 111–120.

McCarthy, J. 1980. Circumscription — a form of nonmonotonic reasoning. *Artificial Intelligence* 13(1–2):27–39.

Reiter, R. 1977. On closed world data bases. In Gallaire, H. and Nicolas, J.-M., eds., In Proc. *Logic and Databases*, Plenum. 119–140.

Reiter, R. 1980. A logic for default reasoning. *Artificial Intelligence* 13(1–2):81–132.

Reiter, R. 1990. On asking what a database knows. In Lloyd, J. W., ed., *Computational Logic*, Springer. 96–113.