Preferential and Rational Closure: A Comparative Study

Jürgen Dix[1] and Peter H. Schmitt[2]

[1] University of Koblenz, Department of Computer Science, Rheinau 1, 56075 Koblenz, FRG, dix@informatik.uni-koblenz.de

[2] University of Karlsruhe, Institute for Logic, Complexity and Deduction Systems, POBox 6980, 76128 Karlsruhe, FRG, pschmitt@ira.uka.de

Abstract. While there exist meanwhile a great number of nonmonotonic systems and while there are much more sophisticated applications than just the famous "Tweety" example for nonmonotonic reasoning, there still is no systematic analysis of these examples. Besides such an analysis we think that the following question is of great importance and should be discussed in the future: "How much *nonmonotonicity* is required in a particular application domain in order to get the "right" results?". Although we do not treat these two problems explicitly, this paper is a first step towards doing so. The idea is to abstract from a particular representation and to ask for the properties a nonmonotonic consequence-relation $\mid\sim$ should satisfy in order to derive "reasonable" conclusions in a given situation.

Using the syntactical framework of [Kraus et al., 1990], we first collect some interesting examples of nonmonotonic reasoning (well-known in the relevant literature) and then we test the *preferential* as well as the *rational* closure (see [Lehmann and Magidor, 1992] and [Kraus et al., 1990]). We have implemented the rational closure and we show that the computation of the preferential closure can be elegantly reduced to the computation of the rational closure.

We also consider some possibilities to incorporate not only *weak knowledge* (Default-knowledge) but also *hard knowledge* (facts that are true regardless of defaults). We show that all these possibilities are equivalent with respect to the preferential closure, but inequivalent for the rational closure.

1 Introduction

While there are meanwhile a number of working deduction systems for predicate logic and some of these have reached a very mature state, *nonmonotonic* deduction systems are very rare. Besides some prototypical systems for *inheritance hierarchies*, which can handle only problems formalized in a very resctricted language (usually disjunctions are not allowed), the only systems we know of are those by Poole and Ginsberg and another system of Stefan Brass ([Braß, 1992]).

These systems are not restricted to propositional logic, but can also handle problems formalized in predicate logic. Of course this is only possible to a limited extent, since the nonmonotonic logics used are undecidable for full predicate logic.

At first sight it may seem that many problems arising in the area of automated plan generation, would require for their easy formulation the use of function symbols. But a closer look reveals that in most cases additional, very restrictive assumptions are made, such as "domain-closure" or restriction to the Herbrand universe, which make it possible to express these problems in propositional logic, though admitedly using some "coding-overload".

One of the main motivations for the development of default logic was to find a logic that is closer to the way humans solve problems. Specifically humans are very good in dealing with the *frame problem* arising in planning tasks. Default logics were designed in particular to cope with this problem. Considering the difficulties that even relatively simple examples from the *blocks' world scenario* still cause, it is appearent that there is a long way to go.

For a comprehensive survey on nonmonotonic logics we refer the reader to [Dix, 1989, Brewka, 1991, Brewka et al., 1993, Makinson, 1993].

The examples presented in our paper are all taken from [Lifschitz, 1989] and [Poole, 1991].

In this paper we restrict attention to the two approaches by Lehmann, Magidor ([Lehmann and Magidor, 1992]) and by Kraus, Lehmann and Magidor [Kraus et al., 1990]. Both approaches adopt the following set-up:

Default-base $\mathcal{K}$: A given situation is modeled by a (finite) set of *conditional assertions* of the form "$\alpha \mid\sim \beta$", where α, β are expressions in propositional logic. This set forms the "default-knowledge-base" $\mathcal{K}$. An assertion "$\alpha \mid\sim \beta$" may be read as "*normally (typically, in most cases, if there is no evidence to the contrary)* α implies β".

Closure of $\mathcal{K}$: Starting from the default-base $\mathcal{K}$ one attempts to derive additional conditional assertions. The set $\mathcal{K}$ of elements of the form "$\alpha \mid\sim \beta$" thus leads to a, usually bigger, set again made up of elements of the form "$\alpha \mid\sim \beta$". The set of all conditional assertions derived from $\mathcal{K}$ is called the closure of $\mathcal{K}$. We will consider several different closure operators.

Query ?-$c \mid\sim d$: Finally we can ask queries about $\mathcal{K}$. These are again of the form "$\gamma \mid\sim \delta$" and will be answered by "yes" or "no", depending on wether "$\gamma \mid\sim \delta$" is in the closure of $\mathcal{K}$ or not.

Let us have a look at some illustrating examples.

Example 1 Specificity.

1. Penguins are birds.
2. Penguins normally don't fly.
3. Birds normally do fly.
4. Tweety is a bird, Fred is a penguin.

In classical logic the statements on Example 1 would be contradictory, since we could deduce that penguins fly and that penguins don't fly. Intuitively we do not want the statement “penguins normally fly” to be derivable. Penguins are a special kind of birds. The more specific information about penguins should override the information about birds in general. Following this principle we also expect that “Tweety flies” and “Fred does not fly” are derivable.

Example 2 Multiple Exceptions.

$$
\begin{aligned}
\mathcal{K}_2 = \{ \; & \text{animal} \;\mid\!\sim\; \neg\, \text{fly}, \\
& \text{bird} \;\mid\!\sim\; \text{animal}, \\
& \text{bird} \;\mid\!\sim\; \text{fly}, \\
& \text{bat} \;\mid\!\sim\; \text{animal}, \\
& \text{bat} \;\mid\!\sim\; \text{fly}, \\
& \text{ostrich} \;\mid\!\sim\; \text{bird}, \\
& \text{ostrich} \;\mid\!\sim\; \neg\, \text{fly} \\
\} &
\end{aligned}
$$

In this example birds are an exception within the class of all animals since they fly , while ostriches are an exceptions within the class of all birds since they don’t fly. Intuitively we expect animal $\mid\!\sim \neg$ bird to be derivable.

Example 3 Avoiding Exceptions.
The data base is the same as in Example 1.

Suppose we know a bird named “Sam” which wasn’t mentioned before. If we adopted the rule, not to assume that “Sam” belongs to an exceptional class unless we learn this explicitely, we would conclude that “Sam” is not a penguin. In the present context flying birds are normal and penguins are exceptional since they don’t fly.

Problems arising with this rule will be shown in the next example.

Example 4 Extraneous Information.

1. Birds are either penguins or colibries or canaries.
2. Birds normally fly.
3. Birds normally enjoy property E.
4. Pengiuns normally don’t fly.
5. Colibries normally don’t enjoy property E.

Let us assume we know that Tweety is a bird. Following the rule of avoiding exceptions, as in the previous example, we derive that Tweety is a canary, since this is the only kind of birds of which no exceptions are known. In particular we conclude “Tweety flies”.

But if we add the extraneous information that also canaries are exceptions with respect to property E, we can no longer derive “Tweety flies”.

Example 5 Disjunction.

1. Penguins are birds.

2. Penguins normally don't fly.
3. Birds normally fly.
4. Tweety is a penguin or Fred is a penguin.

Intuitively we want the assertion "Tweety and Fred fly" not to be derivable.

Many researchers have, based on examples like those given above, proposed formal systems for nonmonotonic reasoning. A general framework, which would allow to classify and compare different systems or even would lead to an agreement on the "correct" formalization, is still missing. Gabbay [Gabbay, 1984] was the first to suggest not to derive nonmonotonic logics solely from concrete examples, but to investigate general properties also. His idea was to concentrate on the consequence relations determined by systems of nonmonotonic logic (see also his follow-up paper [Gabbay, 1991]). If it is not possible to find one definite consequence relation that a majority would find acceptable, one could at least try to find minimal requirements for nonmonotonic logics that everybody could agree on. We will state Gabbay's postulates later in this paper. This line of research was further developed by Shoham (on the semantical level) and by Makinson (on the axiomatic level): see [Shoham, 1986] and [Makinson, 1989, Makinson, 1993]. For the moment we give two examples of the properties of consequence relations we have in mind.

Example 6 Contraposition.
Let the statement "If α then normally β" be given.

Should we be allowed to derive "If $\neg\beta$ then normally $\neg\alpha$" from this ?

Example 7 Transitivity.

1. β normally follows from α.
2. γ normally follows from β.

Should we be allowed to conclude that normally γ follows from α?

An agreement on the indispensible or desirable postulates for nonmonotonic logics is at present not in sight. We believe that the more general question, whether nonmonotonic logic can be formalized at all, to be the decisive question in this area. If the answer to this question turns out to be negative then the occupation with nonmonotonicity will remain forever speculative and restricted to the description of individual phenomena. But we believe there are indications that a successful formalization of nonmonotonic logic will emerge, at least in restricted domains. The search for general postulates for nonmonotonic logic can not only proceed one way, from concrete examples to formal properties, but it also has to use the way back. Now that in the papers by Kraus, Lehmann and Magidor, referred to above, first attractive formalizations have been presented, it is time to confront these again with concrete examples. This will be done in Section 5.
Another important issue addressed in this paper is the question

How do we distinguish between hard *and* default- *knowledge?*

In most applications there are not only assumptions that are true *by default*, but there are also hard facts. This leads to a partitioning of all assumptions into a theory Σ_s representing hard facts and uncertain facts $\mathcal{K}$.

While in [Kraus et al., 1990] a possibility to represent hard knowledge was provided by an appropriate choice of the set $\mathcal{U}$ of worlds ($\mathcal{U}$ corresponds to Σ_s via $\Sigma_s = Th(\mathcal{U})$), this is no longer the case for the successor paper. Here Lehmann and Magidor represent hard knowledge α as $\neg\alpha \mid\sim false$ in $\mathcal{K}$. We will investigate alternative approaches later in this paper.

2 The Formalism of Kraus, Lehmann and Magidor

2.1 Notation

We will use throughout the paper the following notation:

Fml	the set of all formulas
Δ, Σ, Φ	arbitrary sets of formulas
$\alpha, \beta, \gamma, \delta$	arbitrary formulas
$\alpha \mid\sim \beta$	conditional assertions
Σ_s	*hard* facts; i.e. formulas $\alpha \rightarrow \beta$ instead of $\alpha \mid\sim \beta$
$\mathcal{K}$	knowledge base: a finite set of conditional assertions
$\mathcal{K}^{cum}$	the cumulative closure: System $\mathcal{C}$
$\mathcal{K}^{pref}$	the preferential closure: System $\mathcal{P}$
$\mathcal{K}^{rat}$	the rational closure

2.2 The Syntactic Frame

We start with the definition of the language we want to work with. Let a signature for propositional logic be given. The set of *propositional* formulas in the given signature will be denoted by $\mathcal{L}$. We stipulate that $\mathcal{L}$ is in particular closed under the connectives $\neg, \vee, \wedge, \rightarrow$ and $\leftrightarrow$. As basic connectives we adopt as usual $\{\neg, \vee\}$: all other connectives can be expressed in terms of these.

Models and satisfiability of sets of formulas are defined as usual. Finally we fix a set $\mathcal{U}$ of models, which is intended to represent *hard* background knowledge via $\Sigma_s = Th(\mathcal{U})$. Following [Kraus et al., 1990] we require:

> **Compactness relative to $\mathcal{U}$**: A set Δ of formulas is satisfiable by a model in $\mathcal{U}$, if all finite subsets of Δ are satisfiable by a model in $\mathcal{U}$.

Definition 1 $\alpha \mid\sim \beta$**.**
Let α and β be formulas. The pair $\alpha \mid\sim \beta$ is called a *condional assertion* and will be read as "α implies β nonmonotonically ". In other words: if α is true, we *normally* accept β to be also true .

Next we explain how to formalize a given situation, which we want to take as a basis for further nonmonotonic deductions, using conditional assertions. We distinguish three kinds of information.

1. The first kind of information are hard facts, e.g. “Dogs are animals”, or definitions, e.g. “to be a child” means “not to be grown-up”. This information is given by ordinary propositional formulas Σ_s. $\mathcal{U}$ then consists of all models satisfying Σ_s.
2. The second kind of information consists then of default rules,*conditional assertions*, e.g. “birds normally fly”. The set $\mathcal{K}$ of default rules is called the *knowledge base*.
3. The third kind of information describes the knowledge about a particular situation, e.g. “it is a bird”) and will be represented by the formula α in the query $\alpha \mid\sim \phi$.

In the following we will represent successively the three most important closure operators from the two papers by Kraus, Lehmann and Magidor: cumulative, preferetial and rational closure.

2.3 Cumulative Reasoning: $\mathcal{K}^{cum}$

The system to be presented, called *$\mathcal{C}$-System* , is based on a suggestion of D. Gabbay. It is the weakest among the systems treated by Kraus, Lehmann and Magidor and was first considered by Makinson in a slightly more general setting (see [Dix and Makinson, 1992] for the relationship between Makinsons' and Kraus, Lehmann and Magidors' approach). We will call system one weaker than system two, if all rules derivable in system one can also be derived in system two. $\mathcal{C}$ stands for *cumulative*.

Definition 2 System $\mathcal{C}$.
A binary relation $\mid\sim$ on formulas will be called a cumulative consequence relation, or cumulative for short, if it is closed under the inference rules *Reflexivity, Left Logical Equivalence, Right Weakening, Cut* and *Cautious Monotonicity*.

Let us have a closer look at the rules named in this definition.

$$\frac{}{\alpha \mid\sim \alpha} \qquad (\textit{Reflexivity})$$

$$\frac{\vdash \alpha \Leftrightarrow \beta,\ \alpha \mid\sim \gamma}{\beta \mid\sim \gamma} \qquad (\text{Left Logical Equivalence})$$

Left Logical Equivalence expresses the requirement that logically equivalent formulas should have the same nonmontonic consequences. Here $\vdash$ denotes the classical logical consequence relation restricted to $\mathcal{U}$. More precisely $\vdash \phi$ is interpreted as $\Sigma_s \vdash \phi$ in classical logic. Using the wellknown completeness theorem for classical logic, we can also write “$\models \phi$”. For ease of notation the reference to $\mathcal{U}$ or Σ_s will be supressed, unless it is absolutely crucial.

$$\frac{\vdash \alpha \rightarrow \beta,\ \gamma \mid\!\sim \alpha}{\gamma \mid\!\sim \beta} \qquad \textit{(Right Weakening)}$$

Right Weakening may be paraphrased to say that any logical consequence of a nonmonotonic consequence is again a nonmonotonic consequence. *Reflexivity* and *Right Weakening* suffice to derive $\alpha \mid\!\sim \beta$ from $\alpha \vdash \beta$, i.e. the nonmonotonic consequence relation is an extension of the classical consequence relation.

$$\frac{\alpha \wedge \beta \mid\!\sim \gamma,\ \alpha \mid\!\sim \beta}{\alpha \mid\!\sim \gamma} \qquad \textit{(Cut)}$$

Cut expresses the method of added hypothesis. When we try to prove γ from α , we adopt an additional hypothesis β, try to derive γ from $\alpha \wedge \beta$ and then try to get rid of the hypothesis again by proving that β follows from α.

$$\frac{\alpha \mid\!\sim \beta,\ \alpha \mid\!\sim \gamma}{\alpha \wedge \beta \mid\!\sim \gamma} \qquad \textit{(Cautious Monotonicity)}$$

For a monotone consequence relation $\mid\!\sim$ the assertion $\alpha \wedge \beta \mid\!\sim \gamma$ could always be derived from $\alpha \mid\!\sim \gamma$. *Cautious Monotonicity* is a relaxation of this requirement in that monotony only applies, when we add assumptions that have already been proved. Thus cautious monotonicity may be thought of as sanctioning the use of lemmas in proofs.

Lemma 3 Cumulativity.
The rules Cut and Cautious Monotonicity may be expressed in one rule as follows: If $\alpha \mid\!\sim \beta$ is true, then the consequences of α and $\alpha \wedge \beta$ coincide.

Definition 4 $\mathcal{K}^{cum}$.
$\alpha \mid\!\sim \beta$ is an element of $\mathcal{K}^{cum}$ if and only if there exists a derivation of $\alpha \mid\!\sim \beta$ from $\mathcal{K}$ using the rules in $\mathcal{C}$.

Some Derived Rules In System $\mathcal{C}$

The following rules can be derived in $\mathcal{C}$.

$$\frac{\alpha \mid\!\sim \beta,\ \alpha \mid\!\sim \gamma}{\alpha \mid\!\sim \beta \wedge \gamma} \qquad \textit{(And)}$$

$$\frac{\alpha \mid\!\sim \beta \rightarrow \gamma,\ \alpha \mid\!\sim \beta}{\alpha \mid\!\sim \gamma} \qquad \textit{(MPC)}$$

$$\frac{\alpha \mid\!\sim \beta,\ \beta \mid\!\sim \alpha,\ \alpha \mid\!\sim \gamma}{\beta \mid\!\sim \gamma} \qquad \textit{(Equivalence)}$$

Here is a derivation of the last rule: $\alpha \mathrel{|\!\sim} \beta$ and $\alpha \mathrel{|\!\sim} \gamma$ substituted in *Cautious Monotonicity* yields

$$\frac{\alpha \mathrel{|\!\sim} \beta,\ \alpha \mathrel{|\!\sim} \gamma}{\alpha \wedge \beta \mathrel{|\!\sim} \gamma}.$$

This result together with *Left Logical Equivalence* leads to:

$$\frac{\vdash \alpha \wedge \beta \leftrightarrow \beta \wedge \alpha,\ \alpha \wedge \beta \mathrel{|\!\sim} \gamma}{\beta \wedge \alpha \mathrel{|\!\sim} \gamma}.$$

This and $\beta \mathrel{|\!\sim} \alpha$ substituted in *Cut*:

$$\frac{\beta \wedge \alpha \mathrel{|\!\sim} \gamma,\ \beta \mathrel{|\!\sim} \alpha}{\beta \mathrel{|\!\sim} \gamma}.$$

2.4 Preferential Reasoning: $\mathcal{K}^{pref}$

In this section we present a system stronger than system $\mathcal{C}$, called system $\mathcal{P}$.

Definition 5 System $\mathcal{P}$.
System $\mathcal{P}$ consists of all rules of $\mathcal{C}$ together with the additional rule:

$$\frac{\alpha \mathrel{|\!\sim} \gamma,\ \beta \mathrel{|\!\sim} \gamma}{\alpha \vee \beta \mathrel{|\!\sim} \gamma} \qquad \text{(Or)}$$

A consequence relation closed under all rules of $\mathcal{P}$ will be called *preferential*.

Or may be read as requiring that any formula which is both a consequence of α and a consequence of β, should also be a consequence of the disjunction $\alpha \vee \beta$. The interplay of *or* and the rules of $\mathcal{C}$ accounts for the strength of system $\mathcal{C}$. As an example we consider:

The rule

$$\frac{\alpha \wedge \beta \mathrel{|\!\sim} \gamma}{\alpha \mathrel{|\!\sim} \beta \rightarrow \gamma} \qquad \textit{(S)}$$

is derivable from $\mathcal{P}$.

A proof is quite simple: From $\alpha \wedge \beta \mathrel{|\!\sim} \gamma$ we get, using *Right Weakening*,

$$\alpha \wedge \beta \mathrel{|\!\sim} \beta \rightarrow \gamma$$

In addition $\alpha \wedge \neg\beta \mathrel{|\!\sim} \beta \rightarrow \gamma$ is true. From both we obtain via *or* and *Left Logical Equivalence*

$$\alpha \mathrel{|\!\sim} \beta \rightarrow \gamma.$$

Definition 6 $\mathcal{K}^{pref}$.
$\alpha \mathrel{|\!\sim} \beta$ is an element of $\mathcal{K}^{pref}$ if and only if, there exists a derivation of $\alpha \mathrel{|\!\sim} \beta$ from $\mathcal{K}$ using the rules from $\mathcal{P}$.

How do people invent logical inference rules like those for the cumulative and preferential closure? We believe that above all they rely on their intuition – they will try by careful introspection to uncover logical principles employed in everyday-reasoning and in addition pay attention to analogies with classical logic. Despite the vague nature of this method it is nevertheless quite successful. But there is another way to arrive at logical inference rules or to justify hypothesized inference rules: to give a model theoretic semantics. As an example of this approach we will explain *preferential models*.

A preferential model $\mathcal{W}$ consists of a set W, whose elements are called worlds or states and a strict partial order $\prec$ on W. A relation $m_1 \prec m_2$ for worlds $m_1, m_2 \in W$ is meant to express the fact that world m_1 is more important than world m_2. Normally we place ourselves in world m_1 while m_2 is considered an exception. In the situation described in Example 2 we may distinguish worlds m_1, m_2 and m_3. Each world is determined by the set of formulas that are true in it.

m_1 :	"is an animal"	m_2 :	"is an animal"	m_3 :	"is an animal"
	"doesn't fly"		"flies"		"doesn't fly"
	"is not a bird"		"is a bird"		"is a bird"
	"is not a bat"		"is a bat"		"is not a bat"
	"is not an ostrich"		"is not an ostrich"		"is an ostrich"

It has become costumary in examples of this kind to presuppose the existence of an object t, whose properties are written down or are to be derived. In the description t itself does not appear explicitely.

We want to have $m_1 \prec m_2$, since in the considered situation flying animals are an exception. Furthermore we want $m_2 \prec m_3$ to be true, since non-flying birds are an exception. $\mathcal{W} = \langle \{m_1, m_2, m_3\}, \prec \rangle$ is an example of a preferential model. So far we have only looked at different worlds separately and it was clear which formulas are true for a particular world. But how do we define truth of conditional assertions $\alpha \mathrel{|\!\sim} \beta$ in the collection of all worlds $\mathcal{W}$? Having in mind the intended meaning of the relation $\prec$, the following definition suggests itself.

Definition 7 Truth in Preferential Models.

$$\alpha \mathrel{|\!\sim}_{\mathcal{W}} \beta$$

if and only if

$$\text{for every } m \in min(\alpha) : m \models \beta,$$

where $min(\alpha) = \{m \in W : m \models \alpha, m \text{ is minimal with respect to } \prec\}$ and $\alpha \mathrel{|\!\sim}_{\mathcal{W}} \beta$ means that "$\alpha \mathrel{|\!\sim} \beta$ is true in $\mathcal{W} = \langle W, \prec \rangle$".

It can easily be checked that according to this definition all conditional assertions from Example 2 are true in the preferential model given above. Even more, it will be observed that the relation $\alpha \mathrel{|\!\sim}_{\mathcal{W}} \beta$ satisfies all postulates of system $\mathcal{P}$. It can be seen as a strong argument in favour of system $\mathcal{P}$ that also the reverse implication can be established:

Theorem 8 Completeness Theorem for $\mathcal{P}$.
For every consequence relation $\mid\sim$ *satisfying all the rules of system* $\mathcal{P}$, *there exists a preferential model* $\mathcal{W}$ *such that* $\mid\sim_{\mathcal{W}} \; = \; \mid\sim$.

This is not the place to go into all details of preferential models, but there are two remarks we do not want to skip. In the general case, where the set W of all worlds may also be infinite, an additional requirement has to be imposed on the partial odering $\prec$, which ensures that $min(\alpha)$ always exists. Furthermore it is essential for the completeness theorem to allow for different worlds in W, satisfying exactly the same formulas.

Using appropriate modifications of the notion of a preferential model it is possible to obtain completeness theorems for the cumulative closure and the rational closure.

2.5 The Rational Closure: $\mathcal{K}^{rat}$

The differences between the systems $\mathcal{K}^{cum}$ and $\mathcal{K}^{pref}$ on one hand and $\mathcal{K}^{rat}$ on the other hand are considerable:

$\underline{\mathcal{K} \leadsto \mathcal{K}^{cum}}$**:** This closure operation, as well as $\mathcal{K} \leadsto \mathcal{K}^{pref}$ is *monotone*, i.e.

$$\mathcal{K} \subseteq \mathcal{K}' \implies \mathcal{K}^{cum} \subseteq (\mathcal{K}')^{cum}.$$

The reason for this is that both operations are defined by *inference rules* of the form:

"if $\alpha \mid\sim \beta$ and $\gamma \mid\sim \delta$ then $\zeta \mid\sim \eta$".

$\mathcal{K}^{cum}$ and $\mathcal{K}^{pref}$ are the least sets containing $\mathcal{K}$ and closed under a set of inference rules of this form.

$\underline{\mathcal{K} \leadsto \mathcal{K}^{rat}}$**:** This operation is *nonmonotonic* and not so easy to describe. But Lehmann and Magidor give in [Lehmann and Magidor, 1992] a very effective and intuitive algorithm to compute $\mathcal{K}^{rat}$. We will restrict ourselves to the explanation of this algorithm.

Two important concepts have to be introduced before we can give the definition of the algorithm that computes $\mathcal{K}^{rat}$:

1. the notion of an *exceptional formula*,
2. the notion of *rank* for formulas.

Definition 9 Exceptional Formulas.
A formula α is called *exceptional* for $\mathcal{K}$, if *true* $\mid\sim \neg\alpha \in \mathcal{K}^{pref}$. A conditional assertion $\alpha \mid\sim \beta$ is called *exceptional* for $\mathcal{K}$, if α is exceptional for $\mathcal{K}$ ist.

Thus α is exceptional for $\mathcal{K}$ means that normally $\neg\alpha$ holds true (given $\mathcal{K}$). The set of all exceptional conditional assertions in $\mathcal{K}$ will be denoted by *E($\mathcal{K}$).* Iterating the operator E leads to a sequence C_0, C_1, C_2, ...

$$C_0 := \mathcal{K},\ C_{i+1} := E(C_i).$$

This is a decreasing sequence, i.e. $C_{i+1} \subseteq C_i$ is true for all i. Since $\mathcal{K}$ is finite there is an index n, such that

$$C_1 \supset C_2 \supset ... \supset C_{n-1} \supset C_n = C_{n+1} = ...$$

To prove that a formula α is exceptional may seem difficult at first sight since it involves the use of the preferential closure. But the following theorem indicates an easy way to compute exceptionality.

Definition 10 $\widetilde{\mathcal{K}}$**.**
Let $\widetilde{\mathcal{K}}$ be the set obtained from $\mathcal{K}$ by replacing all occurences of $\mid\!\sim$ with the material implication $\rightarrow$. Thus $\widetilde{\mathcal{K}}$ is a set of ordinary propositional formulas.

Theorem 11 Lehmann/Magidor.
Let $\mathcal{K}$ be a knowledge base and α a formula. α is exceptional for $\mathcal{K}$ iff $\widetilde{\mathcal{K}} \models \neg\alpha$.

This theorem shows how to use a propositional theorem prover to compute the rational closure.

Before we continue to develop further the theory of exceptional formulas and their relation to the rational closure, we will try to clarify the meaning of this concept by giving an example. Let us look again at the Example 2 given above: Let C_0 be the set of all conditional assertions in this example. It can easily be seen that $\neg$ bird, $\neg$ bat and $\neg$ ostrich are logical consequences of $\widetilde{C_0}$. Thus "bird", "bat" and "ostrich" are exceptional for C_0:

$$\begin{aligned} C_1 = \{ & \text{bird} \mid\!\sim \text{animal}, \\ & \text{bird} \mid\!\sim \text{flies}, \\ & \text{bat} \mid\!\sim \text{animal}, \\ & \text{bat} \mid\!\sim \text{flies}, \\ & \text{ostrich} \mid\!\sim \text{bird}, \\ & \text{ostrich} \mid\!\sim \neg\, \text{flies} \\ \} & \end{aligned}$$

With respect to C_1 only "ostrich" is exceptional. Therefore:

$$C_2 = \{\text{ostrich} \mid\!\sim \text{bird},\ \text{ostrich} \mid\!\sim \neg\, \text{flies}\}$$

Finally $C_3 = \emptyset$. This demonstrates that the formal notion of an exceptional formula corresponds quite well with the phenomenon of multiple exceptions explained in the introduction.

Given a formula α we may ask for every C_i wether it is exceptional for C_i. If α is exceptional for $C_0, ..., C_{k-1}$ but not exceptional for C_k, C_{k+1}, ... , then k is called the *rank* of α. Notice that when α is not exceptional for C_k, then it is also not exceptional for all C_i with i $\geq$ k. If α is exceptional for all C_i (i $\geq$ 0) then α is said to have no rank.

Definition 12 $\mathcal{K}^{rat}$**.**
Let $\mathcal{K}$ be a knowledge base. $\mathcal{K}^{rat}$ consists of all conditional assertions $\alpha \mid\sim \beta$, such that either

1. the rank of α is strictly smaller than the rank of $\alpha \wedge \neg\beta$ (this includes the case that α has a rank and $\alpha \wedge \neg\beta$ does not) or
2. α has no rank.

2.6 Lehmann/Magidor's Algorithm for $\mathcal{K}^{rat}$

Definition 12 directly leads to the following algorithm for the computation of $\mathcal{K}^{rat}$. The original algorithm given in [Lehmann and Magidor, 1992] contained a bug. We present our corrected version:

$\mathcal{K}$: Knowldege base,
?- $\alpha \mid\sim \beta$: query to $\mathcal{K}$

C := $\mathcal{K}$;
while (α exceptional for C and E(C) $\neq$ C) do
 C := E(C) ;
if (α is exceptional for C)
 output "yes" {definition 1.1.2, α has a rank}
else
 if ($\alpha \wedge \neg\beta$ is exceptional for C)
  output "yes" {definition 1.1.2, case 1}
 else
  output "no";

This algorithm outputs *yes* iff $\alpha \mid\sim \beta \in \mathcal{K}^{rat}$ and can easily be implemented in Prolog. We use the predicate

```
rat_cl([L1],[a,b]).
```

Here L1 denotes the list of all conditional assertions in the knowledge base (represented as pairs [x,y]) and [a,b] is the query to the knowledge base.

2.7 The Relationship between $\mathcal{K}^{pref}$ and $\mathcal{K}^{rat}$

As we have seen, $\mathcal{K}^{rat}$ can be computed conveniently by the above algorithm. In addition we discovered the following interesting relationship between $\mathcal{K}^{rat}$ and $\mathcal{K}^{pref}$:

$$\alpha \mid\sim \beta \ \in \mathcal{K}^{pref} \ \ \underline{\text{iff}} \ \ \alpha \mid\sim \mathit{false} \ \in (\mathcal{K} \cup \{\alpha \mid\sim \neg\beta\})^{rat}$$

This follows from Lemma 41, Corollary 3 and the definition of $\mathcal{K}^{rat}$ in the article [Lehmann and Magidor, 1992]. This relationship makes it possible to compute also $\mathcal{K}^{pref}$ very efficently by reducing it to $\mathcal{K}^{rat}$.

```
p_cl(K,[A,B]) :- rat_cl([[A,-B]|K],[A,a & -a]).
```

3 Representing Hard Knowledge Σ_s

Since Lehmann and Magidor's approach starts from a knowledge base that already is made up of formulas of the form $\alpha \mid\sim \beta$, there remains the considerable problem to translate knowledge given in natural language or in another non-formalized fashion into formulas. Even hard, dependable knowledge has to be represented somehow using $\mid\sim$. Consider e.g. the English phrase *penguins are birds*, a hard fact without exceptions. So far we have , following common practice, stored this assertion as $penguin \mid\sim bird$ in the knowledge base. The same applies to $penguin \mid\sim \neg fly$. But only the assertion $bird \mid\sim fly$ is modeled appropriately by this representation. This weak point in the formalism by Lehmann and Magidor led us to experiment with other representations and modifications of the algorithm.

In [Kraus et al., 1990] a possibility to represent hard knowledge was provided in the form of the set $\mathcal{U}$ of worlds ($\mathcal{U}$ corresponds to Σ_s in our terminology via $\Sigma_s = Th(\mathcal{U})$). This was not continued in the successor paper, where Lehmann and Magidor instead represent hard knowledge α by including the assertion $\neg\alpha \mid\sim false$ in $\mathcal{K}$.

We will consider the following possibilites to combine hard knowledge and default knowledge:

1. $\mathcal{U}$**:** The method described in [Kraus et al., 1990] using $\mathcal{U}$.

2. $\mathcal{K} = \mathcal{H} \cup \mathcal{W}$**:** A modified algorithm for $\mathcal{K}^{rat}$. The possibility to reduce the computation of $\mathcal{K}^{pref}$ to a computation involving $\mathcal{K}^{rat}$ only works for $\mathcal{U} = Mod(\{true\})$. But a careful analysis of the algorithm shows, how to modify it to obtain the same reduction for arbitrary sets $\mathcal{U}$.
The modified algorithm takes as input a knowledge base $\mathcal{K}$, which is partitioned into *hard* (dependable) and *soft* (default) knowledge: $\mathcal{K} = \mathcal{H} \cup \mathcal{W}$.

3. $\neg\alpha \mid\sim false$**:** Lehmann's method to represent hard facts α by $\neg\alpha \mid\sim false$.

4. $(\alpha \wedge \Sigma_s) \mid\sim \beta$**:** Our method: $\mathcal{K}$ contains only default knowledge, the hard knowledge Σ_s is incorporated into the query: instead of the query "$\alpha \mid\sim \beta$" we ask "$(\alpha \wedge \Sigma_s) \mid\sim \beta$".

5. $true \mid\sim \alpha$**:** Represent α by $true \mid\sim \alpha$. This possibility is only included for the sake of completeness. Intuitively a hard fact α cannot be adequately represented as a default $true \mid\sim \alpha$.

Using the semantical framework developed from Kraus, Lehmann and Magidor in [Kraus et al., 1990], it can be shown that all five approaches are equivalent with respect to $\mathcal{K}^{pref}$:

Theorem 13 Equivalence of 1. - 5. for $\mathcal{K}^{pref}$.
All five possibilities for representing hard knowledge are equivalent with respect to $\mathcal{K}^{pref}$.

Proof. Using Kraus, Lehmann and Magidor's Completeness Theorem for the preferential closure, we know that an assertion $\gamma \mid\sim \delta$ is in the preferential closure if and only if all minimal worlds (which are nothing else but classical models) satisfying γ also satisfy δ: see also Definition 7.

Possibility 1 states that (by definition of the worlds) every world (therefore also every minimal world) is a model of Σ_s.

Possibility 3 equivalently ensures that any minimal world is a model of Σ_s (note that a world is complete and consistent). The same is true for possibility 5: as every world is consistent, this condition implies that any minimal world is a model of Σ_s.

If we incorporate Σ_s in the query (possibility 4), then we restrict automatically to worlds satisfying Σ_s.

Possibility 2 is easily seen to be equivalent by inspecting the modified algorithm.

As to $\mathcal{K}^{rat}$, obviously 2. and 3. are equivalent but 3., 4. and 5. are pairwise inequivalent (condition 1 does not make any sense). We will illustrate this in Section 5 using the examples introduced there.

We consider the different behaviour of $\mathcal{K}^{pref}$ and $\mathcal{K}^{rat}$ concerning 1. - 5. interesting and argue that the reasons for this should be investigated in more depth.

3.1 Rational Closure With Hard Facts

In this section we present our modified algorithm and show that this algorithm (which is a particular method to present hard knowledge) is equivalent to Lehmann's method of representing hard facts α as $\neg\alpha \mid\!\sim false$.

Let us illustrate our algorithm with the help of Example 1. We distinguish in $\mathcal{K}$ between *hard* and *soft* statements: $\mathcal{K} = \mathcal{H} \cup \mathcal{W}$. In our example

$$\mathcal{H} = \{penguin \mid\!\sim bird\},\ \mathcal{W} = \{bird \mid\!\sim fly, penguin \mid\!\sim \neg fly\}.$$

The computation of $E(C)$ and the test for exceptionality of a formula α is still done with respect to $\mathcal{H} \cup \mathcal{W}$, but after the computation of $E(C)$ we always add the set $\mathcal{H}$:

$$E(C) := E(C) \cup \mathcal{H}.$$

This ensures that hard facts area always contained in $E(C)$: $\mathcal{H} \subset C_i$ which justifies their name: *hard* in contrast to *soft* knowledge.

The same effect occurs, if the formula $\neg(penguin \to bird) \mid\!\sim false$ is put into $\mathcal{W}$ instead of putting $penguin \mid\!\sim bird$ into $\mathcal{H}$: in $\widetilde{\mathcal{W}}$ the formula $\neg(penguin \to bird) \mid\!\sim false$ is equivalent to $\neg penguin \vee bird$ (the translation of $penguin \mid\!\sim bird$ in $\widetilde{\mathcal{W}}$).

The modified algorithm can therefore be stated as follows:

$\mathcal{K} = \mathcal{H} \cup \mathcal{W}$: Knowledge-Base,
?- $\alpha \mid\!\sim \beta$: query to $\mathcal{K}$

C := $\mathcal{H} \cup \mathcal{W}$;
while (α is exceptional for C and $E(C) \cup \mathcal{H} \neq$ C) do
 C := E(C) $\cup$ $\mathcal{H}$;
if (α is exceptional for C)
 output "yes" {Definition 12, α has no rank}
else
 if ($\alpha \wedge \neg\beta$ is an exceptional formula for C)
   output "yes" {Definition 12, Case 1}
 else
   output "no";

The associated programs `rat_hard_cl` and `p_hard_cl` have the following syntax:

```
rat_hard_cl(W,H,[A,B])

p_hard_cl(W,H,[A,B]) :- rat_hard_cl([[A,-B]|W],H,[A,a & -a]).
```

Here W is a list of all soft facts and H is the list of all hard facts.

Note that our modified algorithm can also be used for the computation of the preferential closure. In this case, he coincides with the algorithm described in [Kraus et al., 1990] for appropriately chosen $\mathcal{U}$.

3.2 Canonical Transformation Into Propositional Logic

Still another possibility is to transform a given problem in an uniform way to a problem in propositional logic. Instead of considering

$$bird(x) \mid\!\sim fly(x)$$

we replace all predicates by their instances and introduce new propositional constants. In this way, a given finite set of first-order formulae can be transformed into an equivalent finite set of propositional formulae, if the underlying Herbrand-Universe is finite: this condition is always fullfilled in our examples.

For the example considered in this section, we introduce new constants V_{fred} and V_{tweety}. V_{fred}, resp. V_{tweety}, stand for *the individual fred is a bird*, resp. *the individual tweety is a bird*. The statement $bird \mid\!\sim fly$ is formalized by $V_{fred} \mid\!\sim F_{fred}$ and $V_{tweety} \mid\!\sim F_{tweety}$.

A disadvantage of this formalization is that for the set C_i with $E(C_i) = C_i$ (computed by the algorithm) the corresponding set $\widetilde{C_i}$ is in most cases inconsistent. Therefore every query is (with respect to C_i) exceptional and the algorithm always answers "yes".

In our particular example we have

$$\mathcal{K} \supset \{ V_{tweety} \mid\sim F_{tweety},\ V_{fred} \mid\sim F_{fred}, P_{fred} \mid\sim \neg F_{fred},\ P_{tweety} \mid\sim \neg F_{tweety} \}$$

independently of how the hard statement "penguins are birds" is incorporated into $\mathcal{K}$: in any case, the set $\widetilde{\mathcal{K}}$ needed in the algorithm is inconsistent.

In any reasonable problem the following constellation occurs:

- *defeasible:* α normally implies β,
- *hard:* γ implies α,
- *hard:* γ implies $\neg\beta$.

This is typical for nonmonotonic problems: γ is the exception to $\alpha \mid\sim \beta$. Any formalization where γ is added (either as a hard or as a defeasible statement) results in an inconsistent set $\widetilde{\mathcal{K}}$. Any query is therefore exceptional and contained in the rational closure. We consider this to be a shortcoming of $\mathcal{K}^{rat}$. This problem also applies to $\mathcal{K}^{pref}$, since the preferential closure can be reduced to $\mathcal{K}^{rat}$ (see Section 2.7).

This sort of inconsistency can be avoided for $\mathcal{K}^{rat}$ if we consider the *modified* query, i.e. possibility 4. of Section 3. Unfortunately it will turn out that this method is much too weak for nearly all our examples in Section 5. Even for this modified query, $\mathcal{K}^{pref}$ still is inconsistent and answers any query with "yes": see Theorem 13.

4 First Results

4.1 $\mathcal{K}^{rat}$

We first consider $\mathcal{K}_1 := \{\ penguin \mid\sim bird,\ penguin \mid\sim \neg fly,\ bird \mid\sim fly\ \}$. The algorithm computes

Yes for	fly	$\mid\sim \neg penguin$	and *No* for	$bird \wedge \neg fly$	$\mid\sim$	$penguin$
	$\neg fly$	$\mid\sim \neg bird$		$bird \wedge \neg fly$	$\mid\sim$	$\neg penguin$
	$\neg fly$	$\mid\sim \neg penguin$		$penguin$	$\mid\sim$	fly
	$bird$	$\mid\sim \neg penguin$				
	$\neg bird$	$\mid\sim \neg penguin$				
	$bird \wedge penguin$	$\mid\sim \neg fly$				
	$black \wedge penguin$	$\mid\sim \neg fly$				
	$bird \wedge green$	$\mid\sim fly$				

We also consider an example from [Kraus et al., 1990]:

$$\begin{array}{lll}\mathcal{K}_2 := \{\ animal & \mathrel{|\!\sim} & \neg fly \\ \quad bird & \mathrel{|\!\sim} & animal \\ \quad bird & \mathrel{|\!\sim} & fly \\ \quad bat & \mathrel{|\!\sim} & animal \\ \quad bat & \mathrel{|\!\sim} & fly \\ \quad ostrich & \mathrel{|\!\sim} & bird \\ \quad ostrich & \mathrel{|\!\sim} & \neg fly\ \}\end{array}$$

We have *Yes* for

$$\begin{array}{lll}animal \wedge \neg bird \wedge \neg bat & \mathrel{|\!\sim} & \neg fly \\ bird \wedge \neg ostrich & \mathrel{|\!\sim} & fly \\ bat & \mathrel{|\!\sim} & fly \\ ostrich & \mathrel{|\!\sim} & \neg fly.\end{array}$$

In addition the famous *Nixon-Diamond* is correctly handled by the rational closure:

$$\mathcal{K}_3 := \{quaker \mathrel{|\!\sim} pacifist, republican \mathrel{|\!\sim} \neg pacifist\}$$

We get *Yes* for

$$quaker \wedge \neg republican \mathrel{|\!\sim} pacifist \text{ and } republican \wedge \neg quaker \mathrel{|\!\sim} \neg pacifist.$$

4.2 $\mathcal{K}^{pref}$

For $\mathcal{K}_1$ our algorithm answers

$$\begin{array}{llll}\text{“yes” for} & fly & \mathrel{|\!\sim} & \neg penguin \\ & bird & \mathrel{|\!\sim} & \neg penguin \\ & bird \wedge penguin & \mathrel{|\!\sim} & \neg fly\end{array}$$

and *no* for

$$\begin{array}{lll}\neg fly & \mathrel{|\!\sim} & \neg bird \\ \neg fly & \mathrel{|\!\sim} & \neg penguin \\ \neg bird & \mathrel{|\!\sim} & \neg penguin \\ penguin \wedge black & \mathrel{|\!\sim} & \neg fly \\ bird \wedge green & \mathrel{|\!\sim} & fly \\ bird \wedge \neg fly & \mathrel{|\!\sim} & penguin \\ bird \wedge \neg fly & \mathrel{|\!\sim} & \neg penguin \\ penguin & \mathrel{|\!\sim} & fly\end{array}$$

As to $\mathcal{K}_2$ there is no difference between $\mathcal{K}^{rat}$ and $\mathcal{K}^{pref}$, whereas no rational consequence of $\mathcal{K}_3$ is contained in $\mathcal{K}^{pref}$. This illustrates the greater *deductive power* of $\mathcal{K}^{rat}$.

As an example, we list the output of p_cl and rat_cl for $\mathcal{K}_2$. Note that the rank of a query is computed with respect to the corresponding knowledge base.

```
?- p_cl([[a,-f],[b,a],[b,f],[bat, a],[bat, f],
         [o,b],[o,-f]],
        [a & -b & -bat,-f]).
   ''Yes'', query has no rank, n = 0

?- rat_cl([[a,-f],[b,a],[b,f],[bat, a],[bat, f],
           [o,b],[o,-f]],
          [a & -b & -bat,-f]).
   ''Yes'', rank of the query w.r.t. K = 0;


?- p_cl([[a,-f],[b,a],[b,f],[bat, a],[bat, f],
         [o,b],[o,-f]],
        [b & -o,f]).
   ''Yes'', query has no rank, n = 2

?- rat_cl([[a,-f],[b,a],[b,f],[bat, a],[bat, f],
           [o,b],[o,-f]],
          [b & -o,f]).
   ''Yes'', rank of the query w.r.t. K = 1;


?- p_cl([[a,-f],[b,a],[b,f],[bat, a],[bat, f],
         [o,b],[o,-f]],
        [bat,f]).
   ''Yes'', query has no rank, n = 3

?- rat_cl([[a,-f],[b,a],[b,f],[bat, a],[bat, f],
           [o,b],[o,-f]],
          [bat,f]).
   ''Yes'', rank of the query w.r.t. K = 1;


?- p_cl([[a,-f],[b,a],[b,f],[bat, a],[bat, f],
         [o,b],[o,-f]],
        [o,-f]).
   ''Yes'', query has no rank, n = 2

?- rat_cl([[a,-f],[b,a],[b,f],[bat, a],[bat, f],
           [o,b],[o,-f]],
          [o,-f]).
   ''Yes'', rank of the query w.r.t. K = 2;
```

5 $\mathcal{K}^{pref}$ versus $\mathcal{K}^{rat}$: A Comparison

This last section summarizes and illustrates (using additional examples) the results of the preceding sections concerning *similarities*, *differences*, *handling* und *understanding* of the two sytems $\mathcal{K}^{pref}$ and $\mathcal{K}^{rat}$.

We use the following four program-calls:

`p_cl([K],[a,b])`: `a` $\mid\!\sim$ `b` $\in \mathcal{K}^{pref}$? Here all formulae are defeasible: there is no possibility to state hard knowledge Σ_s.

`p_hard_cl([W],[H],[a,b])`: `a` $\mid\!\sim$ `b` $\in \mathcal{K}^{pref}$? Here we distinguish between hard and defeasible formulae: $\mathcal{K} = \mathcal{W} \cup \mathcal{H}$.

`rat_cl([K],[a,b])`: `a` $\mid\!\sim$ `b` $\in \mathcal{K}^{rat}$? Here all formulae are defeasible: there is no possibility to state hard knowledge Σ_s.

`rat_hard_cl([W],[H],[a,b])`: `a` $\mid\!\sim$ `b` $\in \mathcal{K}^{rat}$? Here we distinguish between hard and defeasible formulae: $\mathcal{K} = \mathcal{W} \cup \mathcal{H}$.

Let us now explain in detail the different methods of formalization. We distinguish between two entirely different formalizations:

I. Pred.-Logic Formalization: Transform a problem given in predicate logic in an uniform way to a problem in propositional logic. Instead of considering

$$bird(x) \mid\!\sim fly(x)$$

we replace all predicates by their instances and introduce new propositional constants. In this way, a given finite set of first-order formulae can be transformed into an equivalent finite set of propositional formulae, if the underlying Herbrand-Universe is finite: this condition is always fullfilled in our examples.

As already noticed in Section 3.2 both the rational and the preferential closure answer "yes" for nearly all queries due to the inconsistency. Only the rational closure together with the modified query avoids this inconsistency: we therefore only consider this case.

II. Usual formalization: In this formalization we do not make any conceptual difference between predicates and their instances. It corresponds to the formalizations given in [Lehmann and Magidor, 1992] and [Kraus et al., 1990]. A disadvantage, however, is that also hard knowledge $\alpha \rightarrow \beta$ has to be coded as a defeasible assertion $\alpha \mid\!\sim \beta$. So we also investigate the possibility to distinguish explicitly between hard and soft formulae:

hard vs. soft: $\mathcal{K}$ splits into *hard* and *soft* formulae: $\mathcal{K} = \mathcal{H} \cup \mathcal{W}$. We still get two subcases:

mod. algorithm: we apply the modified algorithm for $\mathcal{K}^{pref}$ and for $\mathcal{K}^{rat}$ (i.e. `rat_hard_cl` and `p_hard_cl`), or

mod. query: we modify the query. Instead of "$\alpha \mid\!\sim \beta$" we consider "$(\alpha \wedge \Sigma_s) \mid\!\sim \beta$". As usual Σ_s stands for the (*hard*) formulae. The knowledge base itself only contains *defeasible* assertions. With respect to the preferential closure we get the same results as for the just mentioned distinction of hard and soft formulae (this follows immediately from Theorem 8 and Definition 7). It suffices therefore to test just `rat_cl`.

5.1 Lifschitz' Benchmark-Examples

Example 8 Lifschitz, A1.

1. A and B are heavy blocks,
2. Heavy blocks are normally located on the table.
3. A is not on the table.

Is B on the table? We expect the answer "Yes".

I. Pred.-Logic formalization: `ha`, `hb` mean that A, resp. B, are heavy. `ta`,`tb` mean that A, resp. B, are on the table. Due to the inconsistency we only consider the rational closure with the modified query:

$\mathcal{K}'$: $\mathcal{K}' = \mathtt{ha} \mid\!\sim \mathtt{ta}, \mathtt{hb} \mid\!\sim \mathtt{tb}$

?-: $(\mathtt{ha\ \&\ hb\ \&\ -ta}) \mid\!\sim \mathtt{tb}$

```
?- rat_cl([[ha,ta],[hb,tb]],
          [ha & hb & -ta,tb]).

  ``No'', rank of the query w.r.t. K = 1;
```

II. Usual formalization: `a`, `b` mean A, B are blocks. `heavy` means "to be heavy" and `table` means "to be located on the table".

$\mathcal{K}$: $\mathtt{a} \mid\!\sim \mathtt{heavy}, \mathtt{b} \mid\!\sim \mathtt{heavy}, \mathtt{heavy} \mid\!\sim \mathtt{table}, \mathtt{a} \mid\!\sim \mathtt{-table},$

?-: $\mathtt{b} \mid\!\sim \mathtt{table}, \mathtt{a} \mid\!\sim \mathtt{-b}$

```
?- p_cl([[a,heavy],[b,heavy],[heavy,table],[a,-table]],
        [b,table]).
  ``No'', rank of the query w.r.t. K  = 1;


?- p_cl([[a,heavy],[b,heavy],[heavy,table],[a,-table]],
        [a,-b]).
  ``No'', rank of the query w.r.t. K  = 1;


?- rat_cl([[a,heavy],[b,heavy],[heavy,table],[a,-table]],
          [b,table]).
  ``Yes'', query has no rank, n = 0;

?- rat_cl([[a,heavy],[b,heavy],[heavy,table],[a,-table]],
          [a,-b]).
  ``No'', rank of the query w.r.t. K  = 1;
```

hard vs. soft: $\mathcal{K}$ splits into *hard* and *soft* formulae: $\mathcal{K} = \mathcal{H} \cup \mathcal{W}$.

$\mathcal{W}$: $\mathtt{heavy} \mid\!\sim \mathtt{table},$

$\mathcal{H}$: $\mathtt{a} \mid\!\sim \mathtt{heavy}, \mathtt{b} \mid\!\sim \mathtt{heavy}, \mathtt{a} \mid\!\sim \mathtt{-table}$

– *modified algorithm:*
?-: `b` $\mid\!\sim$ `table`

```
?- p_hard_cl([[heavy,table]],
             [[a,heavy],[b,heavy],[a,-table]],
             [b,table]).
 ''No'', rank of the query w.r.t. K  = 0;

?- rat_hard_cl([...],[...],
               [b,table]).
 ''Yes'', rank of the query w.r.t. K  = 0;

?- rat_hard_cl([[heavy,table]],
               [[a,heavy],[b,heavy],[a,-table]],
               [a,-b]).
 ''No'', rank of the query w.r.t. K  = 0;
```

– *modified query:*
$\mathcal{K}'$: $\mathcal{K}' = \mathcal{W}$
?-: `(b & a =>  heavy  & b =>  heavy  &  a =>  -table)` $\mid\!\sim$ `table`

```
?- rat_cl([[heavy,table]],
                [b & (a =>  heavy) & (b =>  heavy)
                   & (a =>  -table),table]).

 ''Yes'', rank of the query w.r.t. K  = 0;
```

Example 9 Lifschitz, A2.

1. A and B are heavy blocks,
2. Heavy blocks are normally located on the table.
3. A is not on the table.
4. B is red.

Is B on the table? We expect the answer "Yes".

I. Pred.-Logic formalization: `ha`, `hb` mean that A, resp. B, are heavy. `ta,tb` mean that A, resp. B, are on the table. `rb` means that B is red.
Due to the inconsistency we only consider the rational closure with the modified query:
$\mathcal{K}'$: $\mathcal{K}' =$ `ha` $\mid\!\sim$ `ta, hb` $\mid\!\sim$ `tb`
?-: `(ha & hb & -ta & rb)` $\mid\!\sim$ `tb`

```
?- rat_cl([[ha,ta],[hb,tb]],
          [ha & hb & -ta & rb,tb]).

 ''No'', rank of the query w.r.t. K = 1;
```

II. Usual formalization: `a`, `b` mean A, B are blocks. `heavy` means "to be heavy", `table` means "to be on the table" and `red` means "to be red".

$\mathcal{K}$: `a` $\mid\!\sim$ `heavy`, `b` $\mid\!\sim$ `heavy`, `heavy` $\mid\!\sim$ `table`, `a` $\mid\!\sim$ `-table`, `b` $\mid\!\sim$ `red`,
?-: `b` $\mid\!\sim$ `table`

```
?- p_cl([[a,heavy],[b,heavy],[heavy,table],[a,-table],[b,red]],
       [b,table]).

  ``No'', rank of the query w.r.t. K  = 1;

?- rat_cl([[a,heavy],[b,heavy],[heavy,table],[a,-table],[b,red]],
         [b,table]).

  ``Yes'', query has no rank, n = 0;
```

hard vs. soft: $\mathcal{K}$ splits into *hard* and *soft* formulae: $\mathcal{K} = \mathcal{H} \cup \mathcal{W}$.

$\mathcal{W}$: `heavy` $\mid\!\sim$ `table`,
$\mathcal{H}$: `a` $\mid\!\sim$ `heavy`, `b` $\mid\!\sim$ `heavy`, `a` $\mid\!\sim$ `-table`, `b` $\mid\!\sim$ `red`

– *modified algorithm:*

?-: `b` $\mid\!\sim$ `table`

```
?- p_hard_cl([[heavy,table]],
             [[a,heavy],[b,heavy],[a,-table],[b,red]],
             [b,table]).
 ``No'', rank of the query w.r.t. K  = 0;

?- rat_hard_cl([...],[...],
               [b,table]).
 ``Yes'', rank of the query w.r.t. K = 0;
```

– *modified query:*

$\mathcal{K}'$: $\mathcal{K}' = \mathcal{W}$
?-: `(b & a=>heavy  & b=>heavy &  a=>-table) & b=>red)` $\mid\!\sim$ `table`

```
?- rat_cl([[heavy,table]],
          [b & (a => heavy) & (b => heavy) & (a => -table)
             & (b => red), table]).
 ``Yes'', rank of the query w.r.t. K  = 0;
```

Example 10 Lifschitz, A3.

1. A and B are heavy blocks,
2. Heavy blocks are normally located on the table.
3. A is not on the table.
4. Heavy blocks are normally red.
5. B is not red.

Is B on the table and is A red.

I. Pred.-Logic formalization: Notions as above. Due to the inconsistency we only consider the rational closure with the modified query:

$\mathcal{K}'$: $\mathcal{K}' =$ ha $\mid\!\sim$ ta, hb $\mid\!\sim$ tb, ha $\mid\!\sim$ ra, hb $\mid\!\sim$ rb
?-: (ha & hb & -ta & -rb) $\mid\!\sim$ tb, (ha & hb & -ta & -rb) $\mid\!\sim$ ra

```
?- rat_cl([[ha,ta],[hb,tb],[ha,ra],[hb,rb]],
          [ha & hb & -ta & -rb,tb]).
  ``No'', rank of the query w.r.t. K = 1;

?- rat_cl([[ha,ta],[hb,tb]],
          [ha & hb & -ta & -rb,ra]).
  ``No'', rank of the query w.r.t. K = 1;
```

II. Usual formalization: Notions as above.
$\mathcal{K}$: a $\mid\!\sim$ heavy, b $\mid\!\sim$ heavy, heavy $\mid\!\sim$ table, a $\mid\!\sim$ -table, heavy $\mid\!\sim$ red, b $\mid\!\sim$ -red
?-: b $\mid\!\sim$ table, a $\mid\!\sim$ red

```
?- p_cl([[a,heavy],[b,heavy],[heavy,table],[a,-table],
         [heavy,red],[b,-red]],[b,table]).
  ``No'', rank of the query w.r.t. K  = 1;

?- p_cl([...],[a,red]).
  ``No'', rank of the query w.r.t. K  = 1;

?- rat_cl([...],[b,table]).
  ``No'', rank of the query w.r.t. K  = 1;


?- rat_cl([...],[a,red]).
  ``No'', rank of the query w.r.t. K  = 1;
```

hard vs. soft: $\mathcal{K}$ splits into *hard* and *soft* formulae: $\mathcal{K} = \mathcal{H} \cup \mathcal{W}$.
$\mathcal{W}$: heavy $\mid\!\sim$ table, heavy $\mid\!\sim$ red
$\mathcal{H}$: a $\mid\!\sim$ heavy, b $\mid\!\sim$ heavy, a $\mid\!\sim$ -table, b $\mid\!\sim$ -red

– *modified algorithm:*
  ?-: b $\mid\!\sim$ table, a $\mid\!\sim$ red

```
?- p_hard_cl([[heavy,table],[heavy,red]],
             [[a,heavy],[b,heavy],[a,-table],[b,-red]],
             [b,table]).
  ``No'', rank of the query w.r.t. K  = 0;

?- p_hard_cl([...],[...],[a,red]).
  ``No'', rank of the query w.r.t. K  = 0;

?- rat_hard_cl([...],[...],[b,table]).
  ``No'', rank of the query w.r.t. K  = 0;
```

```
?- rat_hard_cl([...],[...],[a,red]).
 ‘‘No’’, rank of the query w.r.t. K  = 0;
```

– *modified query:*

$\mathcal{K}'$: $\mathcal{K}' = \mathcal{W}$

?-: (b & a=>heavy & b=>heavy & a=>-table) $\mid\!\sim$ table, (a & a =>heavy & b =>heavy & a =>-table) $\mid\!\sim$ red

```
?- rat_cl([[heavy,table],[heavy,red]],
          [b & (a =>  heavy) & (b =>  heavy)
             & (a =>  -table) & (b => -red), table]).
 ‘‘No’’, rank of the query w.r.t. K = 1;

?- rat_cl([[heavy,table],[heavy,red]],
           [a & (a =>  heavy) & (b =>  heavy)
              & (a =>  -table) & (b => -red), red]).
 ‘‘No’’, rank of the query w.r.t. K = 1;
```

Example 11 Lifschitz, A6.

1. A, B and C are heavy blocks,
2. Heavy blocks are normally located on the table.
3. At least one of A or B is not located on the table.

Exactly one of the blocks A, B is not on the table.

I. Pred.-Logic formalization: Notions as above. Due to the inconsistency we only consider the rational closure with the modified query:

$\mathcal{K}'$: $\mathcal{K}' =$ ha $\mid\!\sim$ ta, hb $\mid\!\sim$ tb, hc $\mid\!\sim$ tc

?-: (-ta v -tb) $\mid\!\sim$ ((ta & -tb) v (tb & -ta)

```
?- rat_cl([[ha,ta],[hb,tb],[hc,tc]],
          [-ta v -tb,(ta & -tb) v (tb & -ta)]).
 ‘‘No’’, rank of the query w.r.t. K = 0;
```

II. Usual formalization: Notions as above.

$\mathcal{K}$: a $\mid\!\sim$ heavy, b $\mid\!\sim$ heavy, c $\mid\!\sim$ heavy, heavy $\mid\!\sim$ table, (a v b) $\mid\!\sim$ -table,

?-: (a & -b) v (-a & b) $\mid\!\sim$ -table

```
?- p_cl([[a,heavy],[b,heavy],[c,heavy],[heavy,table],
         [a v b,-table]],
        [(a & -b) v (-a & b),-table]).
 ‘‘No’’, rank of the query w.r.t. K  = 2;

?- rat_cl([[a,heavy],[b,heavy],[c,heavy],[heavy,table],
           [a v b,-table]],
          [(a & -b) v (-a & b),-table]).
 ‘‘Yes’’, rank of the query w.r.t. K = 1;
```

hard vs. soft: $\mathcal{K}$ splits into *hard* and *soft* formulae: $\mathcal{K} = \mathcal{H} \cup \mathcal{W}$.

$\mathcal{W}$: `heavy` $\mid\!\sim$ `table`,

$\mathcal{H}$: `a` $\mid\!\sim$ `heavy`, `b` $\mid\!\sim$ `heavy`, `c` $\mid\!\sim$ `heavy`, `a v b` $\mid\!\sim$ `-table`

– *modified algorithm:*

?-: `(a & -b) v (-a & b)` $\mid\!\sim$ `-table`

```
?- p_hard_cl([[heavy,table]],
             [[a,heavy],[b,heavy],[c,heavy],
              [a v b,-table]],
             [(a & -b) v (-a & b),-table]).
  ``Yes'', query has no rank, n = 2

?- rat_hard_cl([...],[...],[(a & -b) v (-a & b),-table]).
  ``Yes'', query has no rank, n = 2
```

– *modified query:*

$\mathcal{K}'$: $\mathcal{K}' = \mathcal{W}$

?-: `((a&-b)v(-a & b) & a=>heavy & b=>heavy & a=>-table)` $\mid\!\sim$ `-table`

```
?- rat_cl([[heavy,table]],
          [((a & -b) v (-a & b)) & (a => heavy) & (b => heavy)
               & (c => heavy) & ((a v b) => -table), -table]).
  ``Yes'', rank of the query w.r.t. K  = 1;
```

5.2 More Examples

Example 12 Specificity.

1. Penguins are birds.
2. Penguins normally don't fly.
3. Birds normally fly.
4. Tweety is a bird, Fred is a penguin.

Intuitively we would like to conclude "Tweety flies" and "Fred does not fly".

I. Pred.-Logic formalization: We introduce the following new constants: P_{fred}, V_{fred}, P_{tweety}, V_{tweety}, F_{fred} and F_{tweety}. Defaults are represented by instantiating with all terms of the underlying language.
Due to the inconsistency we only consider the rational closure with the modified query:

$\mathcal{K}'$: $\mathcal{K}' = \{$`P_{fred}` $\mid\!\sim$ $-$`F_{fred}`, `P_{tweety}` $\mid\!\sim$ $-$`F_{tweety}`,
`V_{fred}` $\mid\!\sim$ `F_{fred}`, `V_{tweety}` $\mid\!\sim$ `F_{tweety}`$\}$

?-: `((P_{tweety}=>V_{tweety}) & (P_{fred}=>V_{fred})) & V_{tweety} & P_{fred})` $\mid\!\sim$ `(F_{tweety})`,

`((P_{tweety}=>V_{tweety}) & (P_{fred}=>P_{fred})) & V_{tweety} & P_{fred})` $\mid\!\sim$ `(-F_{fred})`

```
?- rat_cl([[penguin_fred,-fly_fred],[penguin_tweety,-fly_tweety],
           [bird_fred,fly_fred],[bird_tweety,fly_tweety]],
          [(penguin_tweety => bird_tweety) &
           (penguin_fred => bird_fred) &
            bird_tweety &  penguin_fred, fly_tweety]).
 ''Yes'', rank of the query w.r.t. K = 0;

?- rat_cl([[penguin_fred,-fly_fred],[penguin_tweety,-fly_tweety],
           [bird_fred,fly_fred],[bird_tweety,fly_tweety]],
          [(penguin_tweety => bird_tweety) &
           (penguin_fred => penguin_fred) &
            bird_tweety &  penguin_fred, -fly_fred]).
 ''Yes'', rank of the query w.r.t. K = 0;
```

II. Usual formalization:

$\mathcal{K}$: `penguin` $\mid\!\sim$ `bird`, `penguin` $\mid\!\sim$ `-fly`, `bird` $\mid\!\sim$ `fly`,
`tweety` $\mid\!\sim$ `bird`, `fred` $\mid\!\sim$ `penguin`,

?-: `penguin` $\mid\!\sim$ `fly`, `tweety` $\mid\!\sim$ `fly`, `fred` $\mid\!\sim$ `-fly`

```
?- p_cl([[penguin,bird],[penguin,-fly],[bird,fly],
         [tweety,bird],[fred,penguin]],[penguin,fly]).
 ''No'', rank of the query w.r.t. K = 1;

?- p_cl([..],[tweety,fly]).
 ''No'', rank of the query w.r.t. K = 1;

?- p_cl([..],[fred,-fly]).
 ''No'', rank of the query w.r.t. K = 2;


?- rat_cl([...],[penguin,fly]).
 ''No'', rank of the query w.r.t. K = 1;

?- rat_cl([..],[tweety,fly]).
 ''Yes'', rank of the query w.r.t. K = 0;

?- rat_cl([..],[fred,-fly]).
 ''Yes'',  rank of the query w.r.t. K = 1;
```

hard vs. soft: $\mathcal{K}$ splits into *hard* and *soft* formulae: $\mathcal{K} = \mathcal{H} \cup \mathcal{W}$.

$\mathcal{W}$: `bird` $\mid\!\sim$ `fly`, `penguin` $\mid\!\sim$ `-fly`

$\mathcal{H}$: `penguin` $\mid\!\sim$ `bird`, `tweety` $\mid\!\sim$ `bird`, `fred` $\mid\!\sim$ `penguin`

– *modified algorithm:*

?-: `tweety` $\mid\!\sim$ `fly`, `fred` $\mid\!\sim$ `-fly`,

```
?- p_hard_cl([[bird,fly],[penguin,-fly]],
             [[penguin,bird],[tweety,bird],
              [fred,penguin]],
             [tweety,fly]).
 ``No'', rank of the query w.r.t. K = 0;

?- p_hard_cl([...],[...],[fred,-fly]).
 ``No'', rank of the query w.r.t. K = 0;


?- rat_hard_cl([...],[...],[tweety,fly]).
 ``Yes'', rank of the query w.r.t. K = 0;

?- rat_hard_cl([...],[...],[fred,-fly]).
 ``Yes'', rank of the query w.r.t. K = 0;
```

– *modified query:*
  $\mathcal{K}'$: $\mathcal{K}' = \mathcal{W}$
  ?-: `tweety & (penguin => bird) & (tweety => bird) & (fred => penguin)` $\mid\!\sim$ `fly`,
  ?-: `fred & (penguin => bird) & (tweety => bird) & (fred => penguin)` $\mid\!\sim$ `-fly`,

```
?- rat_cl([[bird,fly],[penguin,-fly]],
         [tweety & (penguin => bird) & (tweety => bird)
                 & (fred => penguin), fly]).
  ``Yes'', rank of the query w.r.t. K = 0;

?- rat_cl([[bird,fly],[penguin,-fly]],
          [fred & (penguin => bird) & (tweety => bird)
                & (fred => penguin), -fly]).
  ``No'', rank of the query w.r.t. K = 1;
```

Example 13 Contraposition.

Let the statement "If α then normally β" be given.

Should we be allowed to derive "If $\neg\beta$ then normally $\neg\alpha$" from this ?

II. Usual formalization:

$\mathcal{K}$: `a` $\mid\!\sim$ `b`
?-: `-b` $\mid\!\sim$ `-a`

```
?- p_cl([[a,b]],[-b,-a]).
 ``No'', rank of the query w.r.t. K = 1;

?- rat_cl([[a,b]],[-b,-a]).
 ``Yes'', rank of the query w.r.t. K = 0;
```

Example 14 Avoiding Exceptions.

1. Penguins are birds.
2. Penguins normally don't fly.
3. Birds normally fly.

Suppose we know a bird named "Tweety". Should we deduce that "Tweety" is no penguin? "Yes".

II. Usual formalization:

$\mathcal{K}$: `penguin` $\mid\!\sim$ `bird`, `penguin` $\mid\!\sim$ `-fly`, `bird` $\mid\!\sim$ `fly`

?-: `tweety` $\mid\!\sim$ `-penguin`, `tweety` $\mid\!\sim$ `fly`, `tweety` $\mid\!\sim$ `-fly`,

```
?- p_cl([[penguin,bird],[penguin,-fly],[bird,fly]],
        [tweety,-penguin]).
 ''No'', rank of the query w.r.t. K = 1;

?- p_cl([..],[tweety,fly]).
 ''No'', rank of the query w.r.t. K = 0;

?- p_cl([..],[tweety,-fly]).
 ''No'', rank of the query w.r.t. K = 0;


?- rat_cl([...],[tweety,-penguin]).
 ''Yes'', rank of the query w.r.t. K = 0;

?- rat_cl([..],[tweety,fly]).
 ''No'', rank of the query w.r.t. K = 0;

?- rat_cl([..],[tweety,-fly]).
 ''No'', rank of the query w.r.t. K = 0;
```

hard vs. soft: $\mathcal{K}$ splits into *hard* and *soft* formulae: $\mathcal{K} = \mathcal{H} \cup \mathcal{W}$.

$\mathcal{W}$: `penguin` $\mid\!\sim$ `-fly`, `bird` $\mid\!\sim$ `fly`

$\mathcal{H}$: `penguin` $\mid\!\sim$ `bird`,

– *modified algorithm:*

?-: `tweety` $\mid\!\sim$ `-penguin`, `tweety` $\mid\!\sim$ `fly`, `tweety` $\mid\!\sim$ `-fly`,

```
?- p_hard_cl([[penguin,-fly], [bird, fly]],
             [[penguin,bird]],
             [tweety,-penguin]).
  ''No'', rank of the query w.r.t. K = 0;

?- p_hard_cl([...],[...],   [tweety,fly]).
  ''No'', rank of the query w.r.t. K = 0;
```

```
?- p_hard_cl([...],[...],   [tweety,-fly]).
  ‘‘No’’, rank of the query w.r.t. K = 0;

?- rat_hard_cl([...],[...], [tweety,-penguin]).
  ‘‘Yes’’, rank of the query w.r.t. K = 0;

?- rat_hard_cl([...],[...], [tweety,fly]).
  ‘‘No’’, rank of the query w.r.t. K = 0;

?- rat_hard_cl([...],[...], [tweety,-fly]).
  ‘‘No’’, rank of the query w.r.t. K = 0;
```

– *modified query:*
$\mathcal{K}'$: $\mathcal{K}' = \mathcal{W}$
?-: `tweety & penguin => bird` $\mid\!\sim$ `-penguin`,
`tweety & penguin => bird` $\mid\!\sim$ `fly`,
`tweety & penguin => bird` $\mid\!\sim$ `-fly`,

```
?- rat_cl([[penguin,-fly], [bird, fly]],
          [tweety & (penguin => bird), -penguin]).
 ‘‘Yes’’, rank of the query w.r.t. K = 0;

?- rat_cl([[penguin,-fly], [bird, fly]],
          [tweety & (penguin => bird), fly]).
 ‘‘No’’, rank of the query w.r.t. K = 0;

?- rat_cl([[penguin,-fly], [bird, fly]],
          [tweety & (penguin => bird), -fly]).
 ‘‘No’’, rank of the query w.r.t. K = 0;
```

Example 15 Extraneous Information.

1. Birds are either penguins or colibries or canaries.
2. Birds normally fly.
3. Birds normally enjoy property E.
4. Pengiuns normally don’t fly.
5. Colibries normally don’t enjoy property E.

Let us assume we know that Tweety is a bird. Following the rule of avoiding exceptions, as in the previous example, we derive that Tweety is a canary, since this is the only kind of birds for which no exceptions are known. In particular we conclude “Tweety flies”.

But if we add the extraneous information that also canaries are an exceptions with respect to a new property E', we can no longer derive “Tweety flies”.

II. Usual formalization:

$\mathcal{K}$: `bird` $\mid\!\sim$ `(penguin v colibri v canary)`,
`(penguin v colibri v  canary)` $\mid\!\sim$ `bird`,

`bird` $\mid\!\sim$ `fly`, `bird` $\mid\!\sim$ `e`,
`penguin` $\mid\!\sim$ `-fly`, `colibri` $\mid\!\sim$ `-e`,

?-: `tweety` $\mid\!\sim$ `canary`, `tweety` $\mid\!\sim$ `fly`

```
?- p_cl([[bird, penguin v colibri v canary],[bird,fly],
        [penguin v colibri v canary,bird],[bird,e],
        [penguin,-fly],[colibri,-e]],[tweety,fly]).
 ``No'', rank of the query w.r.t. K = 0;

?- p_cl([...],[tweety,canary]).
 ``No'', rank of the query w.r.t. K = 0;

?- rat_cl([...],[tweety,fly]).
 ``No'', rank of the query w.r.t. K = 0;

?- rat_cl([...],[tweety,canary]).
 ``No'', rank of the query w.r.t. K = 0;
```

$\mathcal{K}'$: As we already got the false results for $\mathcal{K}$ it makes no sense to add `bird` $\mid\!\sim$ `e'` and `canary` $\mid\!\sim$ `-e'`.

hard vs. soft: $\mathcal{K}$ splits into *hard* and *soft* formulae: $\mathcal{K} = \mathcal{H} \cup \mathcal{W}$.

$\mathcal{W}$: `bird` $\mid\!\sim$ `fly`, `bird` $\mid\!\sim$ `e`,
`penguin` $\mid\!\sim$ `-fly`, `colibri` $\mid\!\sim$ `-e`

$\mathcal{H}$: `bird` $\mid\!\sim$ `(penguin v colibri v  canary)`,
`(penguin v colibri v  canary)` $\mid\!\sim$ `bird`

– *modified algorithm:*

?-: `tweety` $\mid\!\sim$ `canary`, `tweety` $\mid\!\sim$ `fly`

```
?- p_hard_cl([[bird,fly],[bird,e],[penguin,-fly],
              [colibri,-e]],
             [[bird, penguin v colibri v canary],
              [penguin v colibri v canary,bird]],
             [tweety,canary]).
   ``No'', rank of the query w.r.t. K = 0;

?- p_hard_cl([...],[...],[tweety,fly])
   ``No'', rank of the query w.r.t. K = 0;


?- rat_hard_cl([...],[...],[tweety,canary])
   ``No'', rank of the query w.r.t. K = 0;

?- rat_hard_cl([...],[...],[tweety,fly])
   ``No'', rank of the query w.r.t. K = 0;
```

$\mathcal{K}'$: As we already got the false results for $\mathcal{K}$ it makes no sense to add `bird` $\mid\!\sim$ `e'` and `canary` $\mid\!\sim$ `-e'`.

– *modified query:*
$\mathcal{K}'$: $\mathcal{K}' = \mathcal{W}$
?-: tweety & bird <=> (penguin v colibri v canary) $\mid\!\sim$ canary,
tweety & bird <=> (penguin v colibri v canary) $\mid\!\sim$ fly

```
?- rat_cl([[bird, penguin v colibri v canary],[bird,e],
           [penguin v colibri v canary,bird],[bird,fly],
           [penguin,-fly],[colibri,-e],[bird,e'],[canary,-e'],
           [bird,e'],[canary,-e']],
          [( bird => (penguin v colibri v canary)) & tweety &
           ((penguin v colibri v canary) => bird),canary]).
 ‘‘No’’, rank of the query w.r.t. K = 0;

?- rat_cl([...]
          [( bird => (penguin v colibri v canary)) & tweety &
           ((penguin v colibri v canary) => bird),fly]).
 ‘‘No’’, rank of the query w.r.t. K = 0;
```

Example 16 Disjunction.

1. Penguins are birds.
2. Penguins normally don‘t fly.
3. Birds normally fly.
4. Tweety is a penguin or Fred is a penguin.

Intuitively we want the assertion “Tweety and Fred fly” not to be derivable.

II. Usual formalization:
$\mathcal{K}$: penguin $\mid\!\sim$ bird, penguin $\mid\!\sim$ -fly, bird $\mid\!\sim$ fly, tweety v fred $\mid\!\sim$ penguin
?-: tweety & fred $\mid\!\sim$ fly, tweety $\mid\!\sim$ fly, tweety $\mid\!\sim$ -fly

```
?- p_cl([[penguin,bird],[penguin,-fly],[bird,fly],
        [tweety v fred,penguin]],[tweety & fred,fly]).
 ‘‘No’’, rank of the query w.r.t. K = 1;

?- p_cl([..],[tweety,fly]).
 ‘‘No’’, rank of the query w.r.t. K = 1;


?- p_cl([..],[tweety,-fly]).
 ‘‘No’’, rank of the query w.r.t. K = 2;

?- rat_cl([...],[tweety & fred,fly]).
 ‘‘No’’, rank of the query w.r.t. K = 1;

?- rat_cl([..],[tweety,fly]).
 ‘‘No’’, rank of the query w.r.t. K = 1;
```

```
?- rat_cl([..],[tweety,-fly]).
 ``Yes'', rank of the query w.r.t. K = 1;


?- rat_cl([[penguin,bird],[penguin,-fly],[bird,fly],
           [tweety v fred,penguin]],[tweety & fred,-fly]).
 ``Yes'', rank of the query w.r.t. K = 1;
```

hard vs. soft: $\mathcal{K}$ splits into *hard* and *soft* formulae: $\mathcal{K} = \mathcal{H} \cup \mathcal{W}$

$\mathcal{W}$: `penguin` $\mid\!\sim$ `-fly`, `bird` $\mid\!\sim$ `fly`,

$\mathcal{H}$: `penguin` $\mid\!\sim$ `bird`, `tweety v fred` $\mid\!\sim$ `penguin`

– *modified algorithm:*

?-: `tweety & fred` $\mid\!\sim$ `fly`, `tweety` $\mid\!\sim$ `fly`

```
?- p_hard_cl([[penguin,-fly],[bird,fly]],
             [[penguin,bird],[tweety v fred,penguin]],
             [tweety & fred,fly]).
 ``No'', rank of the query w.r.t. K = 0;

?- rat_hard_cl([...],[...],[tweety & fred,fly]).
 ``No'', rank of the query w.r.t. K = 0;

?- rat_hard_cl([...],[...],[tweety ,fly]).
 ``No'', rank of the query w.r.t. K = 1;
```

– *modified query:*

$\mathcal{K}'$: $\mathcal{K}' = \mathcal{W}$

?-: `tweety&fred& penguin=>bird & (tweety v fred)=>penguin` $\mid\!\sim$ `fly`

```
?- rat_cl([[penguin,-fly],[bird,fly]],
          [(tweety & fred) & (penguin => bird)
                  & ((tweety v fred) => penguin),fly]).

 ``No'', rank of the query w.r.t. K = 1;
```

6 Conclusion

For some of the examples from Section 5 there does not seem to be a general agreement for the *right* answers. In the following table we have only collected those examples that possess a unique undisputable semantics. The entries "Yes", resp. "No", express that the answers of the respective closures are, resp. are not, in accordance with the underlying intuitions. The table only considers the *usual formalization*, i.e. the method II.

Let us summarize the most important observations:

Example	$\mathcal{K}^{pref}$	$\mathcal{K}^{rat}$	$(\mathcal{H} \cup \mathcal{W})^{pref}$	$(\mathcal{H} \cup \mathcal{W})^{rat}$	mod. query ($\mathcal{K}^{rat}$)
Lif., A1	no	yes	no	yes	yes
Lif., A2	no	yes	no	yes	yes
Lif., A6	no	yes	yes	yes	yes
Specificity	no	yes	no	yes	no
Avoid. Except.	no	yes	no	yes	yes
Disjunction	no	no	no	no	no

Table 1. The Behaviour of $\mathcal{K}^{pref}$ and $\mathcal{K}^{rat}$

I. Pred.-Logic formalization: In all examples $\widetilde{\mathcal{K}}$ was inconsistent both for $\mathcal{K}^{rat}$ and for $\mathcal{K}^{pref}$. This is very unsatisfying because every query will be answered with "Yes".

The modified query solves this inconsistency-problem for the rational closure, but gets only in Example 1 the intuitively right result: in all other examples it turns out to be too weak.

II. Usual formalization: The preferential closure is in all examples too weak and not able to infer the right results. Only in Example 11 the variant `p_hard_cl` computes the intended result.

The rational closure computes in all but Example 5 the intended answers. In addition, for $\mathcal{K}^{rat}$ the answers with respect to the modified query agree with those with respect to the usual query: only Example 1 is an exception.

`rat_hard_cl` was in no case better than `rat_cl`.

The formalization of hard formulae α as $true \mid\!\sim \alpha$ (for example $true \mid\!\sim (\alpha \to \beta)$ instead of $\alpha \mid\!\sim \beta$)) turned out to be too weak for all examples. We therefore omitted the corresponding program-calls.

As we said in the introduction, this paper is a first step towards discussing the questions "How much *nonmonotonicity* is required in a particular application domain in order to get the "right" results?" and "How does a particular formalization (*hard* versus *soft* knowledge) affect the results?".

We concentrated on two systems: $\mathcal{K}^{pref}$ and $\mathcal{K}^{rat}$. A detailed and clarifying interpretation of the results in Table 1 is still missing. Why, for example, is $\mathcal{K}^{rat}$ already as good as $(\mathcal{H} \cup \mathcal{W})^{rat}$? It seems that only artificially constructed examples lead to different results. Why is $(\mathcal{H} \cup \mathcal{W})^{pref}$ only in one single example better than $\mathcal{K}^{pref}$?

We hope that the ideas suggested in this paper will turn out useful and stimulating for further development and application of nonmonotonic reasoning to real-world problems.

7 Acknowledgements

The authors thank Karl Schlechta for carefully reading a draft and many helpful comments. An anonymous referee also pointed out some misprints.

References

[Braß, 1992] Braß, S. (1992). *Defaults in Deduktiven Datenbanken*. PhD thesis, University of Hannover.

[Brewka, 1991] Brewka, G. (1991). Cumulative Default Logic: In Defense of Nonmonotonic Inference Rules. *Artificial Intelligence*, 50:183–205.

[Brewka et al., 1993] Brewka, G., Dix, J., and Konolige, K. (1993). *Nonmonotonic Reasoning: An Overview*, volume forthcoming of *CSLI Lecture Notes*. Dikran.

[Dix, 1989] Dix, J. (1989). Some Tendencies in Non Monotonic Reasoning: I. The Main Approaches. In Dahn, B. I. and Wolter, H., editors, *Proceedings of the 7th Easter Conference on Model-Theory*, pages 81–105, 1086 Berlin, PSF 1297, DDR. Sektion Mathematik der Humboldt Universität.

[Dix, 1992a] Dix, J. (1992a). Default Theories of Poole-Type and a Method for Constructing Cumulative Versions of Default Logic. In Neumann, B., editor, *Proceedings ECAI*, pages 289–293. John Wiley & Sons.

[Dix, 1992b] Dix, J. (1992b). *Nichtmonotones Schließen und dessen Anwendung auf Semantiken logischer Programme*. PhD thesis, University of Karlsruhe.

[Dix and Makinson, 1992] Dix, J. and Makinson, D. (1992). The Relationship between KLM and MAK models for Nonmonotonic Inference Operations. *Journal of Logic, Language and Information*, Volume 1, No. 2:131–140.

[Dix et al., 1991] Dix, J., Merkle, S., Troppens, U., and Waelde, S. (1991). KLM's Abschlüsse K^{cum} und K^{pref} versus LM's Abschluss K^{rat}: Theorie und Implementierung. Technical report, Universität Karlsruhe, Fakultät für Informatik, Institut für Logik, Komplexität und Deduktionssysteme.

[Gabbay, 1984] Gabbay, D. (1984). Theoretical Foundations for Non-Monotonic Reasoning in Expert- Systems. In Apt, K., editor, *Logics and Models of Concurrent Systems*, pages 439–458. Springer.

[Gabbay, 1991] Gabbay, D. (1991). Theoretical Foundations for Non-Monotonic Reasoning, Part 2, Structured Non-Monotonic Theories. In *Proceedings of the Third Scandinavian Conference on AI, SCAI '91*, pages 19–40. IOS-Press, Amsterdam.

[Kraus et al., 1990] Kraus, S., Lehmann, D., and Magidor, M. (1990). Nonmonotonic Reasoning, Preferential Models and Cumulative Logics. *Artificial Intelligence*, 44(1):167–207.

[Lehmann and Magidor, 1992] Lehmann, D. and Magidor, M. (1992). What does a conditional knowledge base entail? *Artificial Intelligence*, 55:1–60.

[Lifschitz, 1989] Lifschitz, V. (1989). Benchmark Problems for Formal NMR, Version 2.00. In Reinfrank, de Kleer, Ginsberg, and Sandewall, editors, *Non-Monotonic Reasoning*, pages 202–219. Lecture Notes in Artificial Intelligence 346, Springer-Verlag.

[Makinson, 1989] Makinson, D. (1989). General Theory of Cumulative Inference. In Reinfrank, de Kleer, Ginsberg, and Sandewall, editors, *Non-Monotonic Reasoning*. Lecture Notes in Artificial Intelligence 346, Springer-Verlag.

[Makinson, 1993] Makinson, D. (1993). General Patterns in Nonmonotonic Reasoning. In Gabbay, D., Hogger, C., and Robinson, J., editors, *Handbook of Logic in Artificial*

Intelligence and Logic Programming Vol. 2, Nonmonotonic and Uncertain Reasoning, chapter 2.2. Oxford University Press.

[Poole, 1991] Poole, D. L. (1991). The effect of knowledge on belief: conditioning, specificity and the lottery paradox in default reasoning. *Artificial Intelligence*, 49:281–307.

[Shoham, 1986] Shoham, Y. (1986). *Reasoning about Change: Time and Causation from the Standpoint of Artificial Intelligence*. PhD thesis, Yale University.

Table of Contents