Expressivity and Control in Limited Reasoning

Marcelo Finger[1] and **Renata Wassermann**[2]

Abstract. Real agents (natural or artificial) are limited in their reasoning capabilities. In this paper, we present a general framework for modeling limited reasoning based on approximate reasoning and discuss its properties.

We start from Cadoli and Schaerf's approximate entailment. We first extend their system to deal with the full language of propositional logic. A tableau inference system is proposed for the extended system together with a sub-classical semantics; it is shown that this new approximate reasoning system is sound and complete with respect to this semantics. We show how this system can be incrementally used to move from one approximation to the next until the reasoning limitation is reached.

We note that although the extension is more expressive than the original system, it offers less control over the approximation process. We then suggest how we can recover control while keeping the increased expressivity.

Keywords: Resource-Bounded Reasoning, Automated Reasoning, Common-sense Reasoning, Deduction.

1 Introduction

Ideal agents know all the consequences of their beliefs. However, real agents are *limited* in their capabilities. Due to these limitations, a real rational agent must devise some strategy to make good use of the available resources.

Example 1 As a motivational example[3] consider Paul, who is finishing school and preparing himself for the final exams. He studied several different subjects, like Mathematics, Biology, Geography. His knowledge base contains (among others) the beliefs in Figure 1.

When Paul gets the exam, the first question is: *Do cows have molar teeth?*

Of course Paul cannot reason with all of his knowledge at once. First he recalls what he knows about cows and about molar teeth:

Cows eat grass.
Mammals have canine teeth or molar teeth.

From these two pieces of knowledge alone, he cannot answer the question. Since all he knows (explicitly) about cows is that they eat grass, he recalls what he knows about animals that eat grass:

Animals that eat grass do not have canine teeth.
Animals that eat grass are mammals.

Triangles are polygons.
Triangles with one right angle are Pythagorean.
Rectangles are polygons.
Rectangles have four right angles.
Cows eat grass.
Dogs are carnivore.
Animals that eat grass do not have canine teeth.
Carnivorous animals are mammals.
Mammals have canine teeth or molar teeth.
Animals that eat grass are mammals.
Mammals are vertebrate.
Vertebrates are animals.
Brazil is in South America.
Volcanic soil is fertile.

Figure 1. *Student's knowledge base*

From these, Paul can now derive that cows are mammals, that mammals have canine teeth or molar teeth, but that cows do not have canine teeth, hence cows have molar teeth.

The example shows that usually, a system does not have to check its whole knowledge base in order to answer a query. Moreover, it shows that the process of retrieving information is made gradually, and not in a single step. If Paul had to go too far in the process, he would not be able to find an answer, since the time available for the exam is limited. But this does not mean that if he was given more time later on, he would start reasoning from scratch: his partial (or approximate) reasoning would be useful and he would be able to continue from more or less where he stopped.

In this work we study a model of limited reasoning based on Cadoli and Schaerf's *approximate entailment* [9].

In particular, we study their family of logics S_3. Its initial formulation by Cadoli and Schaerf we call here $\text{CS}S_3$. $\text{CS}S_3$ only deals with formulas in negation normal form; it has been used to formalize approximate diagnosis [11] and belief revision [4]. However, the knowledge had to be encoded in clausal form. In this paper, we present an extension, which we call $\text{KE}S_3$, that covers full propositional logic.

Why do we need full propositional logic? It happens that each approximation step is characterized by a formal logic (see below). The final step of the approximations is classical logic, in which every formula is equivalent to one in clausal form. However, in *none* of the intermediate systems such equivalence holds.

S_3 is in fact a family of logics parameterized by a set S of *relevant propositions*. These logics approximate classical logic (CL) in the following sense. Let $\mathcal{P}$ be a set of propositions and

[1] Department of Computer Science, University of São Paulo, Brazil. mfinger@ime.usp.br
[2] Department of Computer Science, University of São Paulo, Brazil. renata@ime.usp.br
[3] The example is based on an example of [9].

$S^0 \subseteq S^1 \subseteq \ldots \subseteq \mathcal{P}$; let $Th(L)$ indicate the set of theorems of a logic. Then, by means of successive approximations:

$$Th(S_3(\varnothing)) \subseteq Th(S_3(S^0)) \subseteq Th(S_3(S^1)) \subseteq \ldots \subseteq Th(S_3(\mathcal{P}))$$

where $Th(S_3(\mathcal{P})) = Th(CL)$ is the set of classical theorems. From this property, we see that it suffices to prove a result in some S_3-approximation to have a classically valid theorem.

In [9], these approximate logics are defined by means of valuation semantics and algorithms for testing satisfiability. However, their formulation contained no strategy to incrementally increase S towards the closest approximation during theorem proving. We present a mechanism for *incremental approximation* for KES_3 that is also adapted to CSS_3.

Here, though not in the original formulation, we consider CSS_3 equipped with a resolution-style inference system and KES_3 equipped with a KE-tableaux system [6]. The set S plays the role of the *limitation in reasoning capabilities*. In such a setting, two questions naturally arise:

- Given S, what is $Th(S_3(S))$? This is what we call the *expressivity* of $S_3(S)$.
- How do we expand S to $S' \supset S$ in trying to prove a theorem? In other words, how do we *control* theorem proving before we exhaust our limited resources in reasoning?

The first property is a *static* property of the system, and the second is a *dynamic* one. The dynamic issue is to have an *anytime* method: one that can be stopped at any time, whenever we reach a limit for S-expansion.

We are going to show that, on the static side, the expressivity of CS$S_3(S)$ is smaller than that of KE$S_3(S)$. As a compensation, we are going to show that CS$S_3(S)$ offers more control than KE$S_3(S)$ even when operating over clausal form formulas. We then are going to extend KE$S_3(S)$ to recover the controlling capability, at the expense of reducing its static expressivity. The balance between expressivity and control is thus characterized, and the system KE$S_3(S)$ is shown to have a *fine-tuning* property, allowing its regulation in the balance of expressivity *vs.* control.

Due to the lack of space, all proofs are left for the full version of the paper.

2 Approximate Inference

In this section, we present Cadoli and Schaerf's system and extend it to deal with full propositional logic.

Notation: Let $\mathcal{P}$ be a countable set of propositional letters. We concentrate on the classical propositional language $\mathcal{L}_C$ formed by the usual boolean connectives $\rightarrow$ (implication), $\wedge$ (conjunction), $\vee$ (disjunction) and $\neg$ (negation).

Throughout the paper, we use lowercase Latin letters to denote propositional letters, lowercase Greek letters to denote formulas, and uppercase letters (Greek or Latin) to denote sets of formulas.

Let $S \subset \mathcal{P}$ be a finite set of propositional letters. We abuse notation and write that, for any formula $\alpha \in \mathcal{L}_C$, $\alpha \in S$ if all its propositional letters are in S. A *propositional valuation* v_p is a function $v_p : \mathcal{P} \rightarrow \{0, 1\}$.

2.1 Cadoli and Schaerf's Proposal

We briefly present here the notion of *approximate entailment* and summarize the main results obtained in [9].

Schaerf and Cadoli define two approximations of classical entailment: $\models^1_S$ which is complete but not sound, and $\models^3_S$ which is classically sound but incomplete. Here we deal only with the latter. In the trivial extreme of approximate entailment, i.e., when $S = \mathcal{P}$, classical entailment is obtained. At the other extreme, when $S = \varnothing$, $\models^3_S$ corresponds to Levesque's logic for explicit beliefs [8], which bears a connection to relevance logics such as those of Anderson and Belnap [1].

In an S_3 assignment, if $p \in S$, then p and $\neg p$ get opposite truth values, while if $p \notin S$, p and $\neg p$ do not both get 0, but may both get 1. The name S_3 comes from the three possible truth assignments for pairs p, $\neg p$ outside S. The set of formulas for which we are testing entailment is assumed to be in clausal form. Satisfiability, entailment, and validity are defined in the usual way.

The following example illustrates the use of approximate entailment. Since $\models^3_S$ is sound but incomplete, it can be used to approximate $\models$, i.e., if for some S we have that $B \models^3_S \alpha$, then $B \models \alpha$.

Example 2 (Formalization of Example 1) Let B be (part of) the student's knowledge base and let α represent the exam question: *do cows have molar teeth?*. We want to check whether $B \models \alpha$, where $\alpha = \neg$`cow` $\vee$ `molar-teeth`:

$B = \{\neg$`cow` $\vee$ `grass-eater`,
$\neg$`dog`$\vee$ `carnivore`,
$\neg$`grass-eater` $\vee$ $\neg$`canine-teeth`,
$\neg$`carnivore` $\vee$ `mammal`,
$\neg$`mammal` $\vee$ `canine-teeth` $\vee$ `molar-teeth`,
$\neg$`grass-eater` $\vee$ `mammal`,
$\neg$`mammal` $\vee$ `vertebrate`,
$\neg$`vertebrate` $\vee$ `animal`$\}$.

In [9] it is shown that for $S = \{$`grass-eater`, `mammal`, `canine-teeth`$\}$, we have that $B \models^3_S \alpha$, hence $B \models \alpha$.

Theorem 1 ([9]) *There is an algorithm for deciding $B \models^3_S \alpha$ in $O(|B|.|\alpha|.2^{|S|})$ time.*[4]

This algorithm can be seen as a resolution method applied only to clauses where all literals are in S.

The good point of Schaerf and Cadoli's system is that they present an incremental algorithm to test for S_3 entailment as new elements are added to S. But there are two major limitations in their results:

1. The system is restricted to $\rightarrow$-free formulas and in negation normal form. In [4] it is noted that the standard translation of formulas into clausal form does not preserve truth-values under the non-standard semantic of S_3.
2. The set S must be guessed at each step of the approximation; no method is given for the atoms to be added to S. Some heuristics for a specific application are presented in [12], but nothing is said about the general case.

2.2 Extending Approximate Inference

In this section, we present an extension of S_3 to full propositional logic. We first extend Cadoli and Schaerf's semantics

[4] The result above depends on a polynomial time satisfiability algorithm for formulas in clausal form. This result has been extended in [2] for formulas in negation normal form, but is not extendable to formulas in arbitrary forms [3].

to all propositional formulas.

Definition 1 An S_3*-valuation* v_S^3 is a function, $v_S^3 : \mathcal{L}_C \to \{0,1\}$, that extends a propositional valuation v_p (i.e., $v_S^3(p) = v_p(p)$), satisfying the following restrictions:

$$
\begin{array}{lllll}
(\wedge) & v_S^3(\alpha \wedge \beta) = 1 & \Leftrightarrow & v_S^3(\alpha) = v_S^3(\beta) = 1 \\
(\vee) & v_S^3(\alpha \vee \beta) = 0 & \Leftrightarrow & v_S^3(\alpha) = v_S^3(\beta) = 0 \\
(\to) & v_S^3(\alpha \to \beta) = 0 & \Leftrightarrow & v_S^3(\alpha) = 1 \text{ and } v_S^3(\beta) = 0 \\
(\neg_1) & v_S^3(\neg\alpha) = 0 & \Rightarrow & v_S^3(\alpha) = 1 \\
(\neg_2) & v_S^3(\neg\alpha) = 1, \alpha \in S & \Rightarrow & v_S^3(\alpha) = 0
\end{array}
$$

Validity and satisfiability are defined as usual. The S_3-*entailment* relationship between a set of formulas B and a formula α is represented as $B \models_S^3 \alpha$ and holds if every valuation v_S^3 that simultaneously satisfies all formulas in B also satisfies α.

Lemma 1 *For any S, any S-valid formula in S_3 is classically valid.*

Theorem 2 *The following are properties that full S_3 inherits from classical logic:*

- *Modus Ponens is valid: $\alpha, \alpha \to \beta \models_S^3 \beta$.*
- *The deduction theorem holds: $B \models_S^3 \alpha$ iff $\models_S^3 \bigwedge B \to \alpha$.*
- *The excluded middle is valid: $\models_S^3 \alpha \vee \neg\alpha$.*

Theorem 3 *The following are non-classical properties of full S_3:*

- *The principle of contradiction* is not valid*: $\not\models_S^3 \neg(\alpha \wedge \neg\alpha)$.*
- *$\alpha \to \beta$ is not equivalent to $\neg\alpha \vee \beta$.*

A sound and complete axiomatization of the full S_3 was given in [7], where it was also compared with da Costa's Paraconsistent Logics [5]. We now turn to a more computational proof method based on KE-tableaux.

2.3 Tableaux for Approximate Inference

KE-tableaux were introduced by D'Agostino [6] as a principled computational improvement over Smullyan's Semantic Tableaux [10].

KE tableaux deal with T- and F-signed formulas: $T\ \alpha$ and $F\ \alpha$. For each connective, there are at least one T- and one F-linear expansion rules. Linear expansion rules always have a *main premise*, and may also have an auxiliary premise. They may have one or two consequences. The only branching rule is the *Principle of Bivalence*, stating that something cannot be true and false at the same time. Figure 2 shows KE-tableau expansion rules for classical logic.

The final line in Figure 2 presents the *Principle of Bivalence* (PB), stating that any formula α is either true of false. The use of PB is highly non-deterministic, so it is normally used according to a *branching heuristic*: PB is used to generate the auxiliary premise for a two-premised rule. To show that $\alpha_1, \ldots, \alpha_n \vdash \beta$ we start with the initial tableau with a column containing $T\ \alpha_1$, …, $T\ \alpha_n$, $F\ \beta$ and develop the tableau by applying the expansion rules in Figure 2. A branch is closed if it contains both $F\ \alpha$ and $T\ \alpha$, for some formula α. The sequent is *deducible* if we can *close* all branches in the tableau.

$$
\frac{T\ \alpha \to \beta \quad T\ \alpha}{T\ \beta}\ (T \to_1) \qquad
\frac{T\ \alpha \to \beta \quad F\ \beta}{F\ \alpha}\ (T \to_2) \qquad
\frac{F\ \alpha \to \beta}{T\ \alpha \quad F\ \beta}\ (F \to)
$$

$$
\frac{F\ \alpha \wedge \beta \quad T\ \alpha}{F\ \beta}\ (F\wedge_1) \qquad
\frac{F\ \alpha \wedge \beta \quad T\ \beta}{F\ \alpha}\ (F\wedge_2) \qquad
\frac{T\ \alpha \wedge \beta}{T\ \alpha \quad T\ \beta}\ (T\wedge)
$$

$$
\frac{T\ \alpha \vee \beta \quad F\ \alpha}{T\ \beta}\ (T\vee_1) \qquad
\frac{T\ \alpha \vee \beta \quad F\ \beta}{T\ \alpha}\ (T\vee_2) \qquad
\frac{F\ \alpha \vee \beta}{F\ \alpha \quad F\ \beta}\ (F\vee)
$$

$$
\frac{T\ \neg\alpha}{F\ \alpha}\ (T\neg) \qquad
\frac{F\ \neg\alpha}{T\ \alpha}\ (F\neg)
$$

$$
\frac{}{T\ \alpha \quad F\ \alpha}\ (\mathrm{PB})
$$

Figure 2. KE-rules for classical logic

KES_3 Tableaux. To construct a KE-tableau system for S_3, we keep all classical rules except rule $(T\ \neg)$, which is changed in KES_3 to:

$$
\frac{T\ \neg\alpha}{F\ \alpha} \quad \text{provided } \alpha \in S
$$

The $(T\ \neg)$-expansion of a branch is only allowed if it contains its antecedent *and the proviso is satisfied*, that is, the formula in question belongs to S. This makes our system immediately subclassical, for any tableau that closes for KES_3 also closes for classical logic. So KES_3 is correct and incomplete with respect to classical logic.

Theorem 4 *KES_3 is sound and complete with respect to the semantics presented in Section 2.2.*

Let us examine an example.

Example 3 Figure 3 shows that $\to$ is not definable in terms of $\vee$ and $\neg$ in KES_3 for $S = \varnothing$.

Note that the left tableau for $\alpha \to \beta \vdash \neg\alpha \vee \beta$ is exactly the same as for classical logic.

1.	$T\ \alpha \to \beta$	1.	$T\ \neg\alpha \vee \beta$
2.	$F\ \neg\alpha \vee \beta$	2.	$F\ \alpha \to \beta$
3.	$F\ \neg\alpha$, from 2	3.	$T\ \alpha$, from 2
4.	$\boxed{F\ \beta}$, from 2	4.	$F\ \beta$, from 2
5.	$T\ \alpha$, from 3	5.	$T\ \neg\alpha$, from 1,4
6.	$\boxed{T\ \beta}$, from 1,5		?
	$\times$		

Figure 3. Undefinability of $\to$ in terms of $\vee$, $\neg$ in KES_3

However, the tableau on the right for $\neg\alpha \vee \beta \vdash \alpha \to \beta$ cannot be closed for the rule on $T\ \neg\alpha$ cannot be applied for $\alpha \notin S$. We get stuck, as there are no further rules to be applied, meaning that the input sequent is not provable.

One important feature of the open tableau in Figure 3 is that if, at the point that it gets stuck, we insert the propositional letters of α in the set S, the tableau expansion can proceed as in classical logic. In fact, the tableau then closes after a single step. This shows that the sequent $\neg\alpha \vee \beta \vdash \alpha \rightarrow \beta$ is deducible if $\alpha \in S$ (and nothing needs to be said about β).

What we have actually done is to change the logic we are operating with during the KE-tableau expansion by adding a formula to S. That formula was chosen so that a stuck tableau could proceed classically. This actually makes us move one step closer to classical logic. S works as the *limited resource* that *controls* the proof. Classical logic is reached when all atoms are in S.

This simple procedure is the KES_3 incremental way of doing approximate theorem proving.

3 Expressivity Gains and Control Losses

In this section we compare KES_3 with the Cadoli-Schaerf (CSS_3) method with regards to *expressivity* (i.e. the theorems proved for the same set S) and to the *control* that the set S exerts over the proof development.

3.1 Dynamic Properties

A property that tells us in which direction to expand our limited resource to achieve a goal is a *dynamic* property of the method. KES_3 provides a method for expanding S in trying to prove a theorem, namely: *"If a branch is closed due to a blocked use of* $(T\neg)$*, add the blocking formula* α *to* S*, so as to unblock that branch."*

Cadoli and Schaerf did not provide a dynamic extension for their system. However, to be honest with their excellent work, such a dynamic behaviour can be easily provided in analogy to ours. In CSS_3, we can resolve $\alpha \vee l$ with $\neg l \vee \beta$ only if $l \in S$, which gives us the dynamic rule: *"If resolution is blocked due to the absence of resolvents in* S*, add a potential resolvent* l *to* S*, so as to unblock resolution."*

With that formulation, we compare the dynamics of KES_3 and CSS_3 for conjunctive normal form formulas. We note that both methods are highly non-deterministic in their behaviour of choosing branch expansion rules and resolvents.

Suppose the size of a CSS_3 proof is measured by the number of resolution steps, and the size of a KES_3 is measured by the number of expansion rules applied.

Theorem 5 *Let* B, α *be a set of clauses and a clause. In a proof of* $B \vdash \alpha$*, KE*S_3 *can linearly simulate the dynamics of CS*S_3*, generating the same* S*.*

The theorem is proved by rewriting the resolution steps as application of inference rules in a KES_3 tableau. Also, in [9] every atom in α was implicitly considered part of S, so in order to compare both systems, we have to start with those atoms in S. Every possible approximation $S^0 \subset \ldots \subset S^k$ in CSS_3 is also possible in KES_3. However, because KES_3 deals with a larger language, several transformational tricks can be used in KES_3 to improve its *static expressivity* which cannot be simulated by CSS_3.

3.2 Static Expressivity

The *static expressivity* of a method is the set of theorems it can prove with a fixed limited resource. In our case, we are going to compare the *set of theorems that can be proved* with a given S.

The idea is to use the larger language of KES_3 to rewrite $\neg l \vee \alpha$ as $l \rightarrow \alpha$. Since α may be a large disjunction, we may not know a priori which negative literal to transform, so this transformation is assumed to be applied "on the fly" during theorem proving.

We can now show that for a fixed S, and formulas in $\rightarrow$-clausal form, KES_3 can prove more theorems.

Theorem 6 *Let* B, α *be a set of formulas and a formula in CNF. Suppose CS*S_3 *proves* $B \vdash \alpha$ *with* S*. Suppose KE applies the transformation above to clauses with one or more negative literals. Then KE*S_3 *proves* $B \vdash \alpha$ *in time linear w.r.t. the time needed by CS*S_3*, with an* $S' \subseteq S$*; it is possible that* $S' \subset S$*.*

Proof Sketch: It suffices to note that, in the simulation of CSS_3-resolution, one needs not always add l to S' (see Figure 4). □

$T\ \neg l \vee \alpha$	$T\ l \rightarrow \alpha$
$T\ l \vee \beta$	$T\ l \vee \beta$
$F\ \alpha \vee \beta$	$F\ \alpha \vee \beta$
$F\ \alpha$	$F\ \alpha$
$F\ \beta$	$F\ \beta$
$T\ \neg l$	$F\ l$
$F\ l\ (l \in S)$	$T\ l$
$T\ l$	$\times$
$\times$	

Figure 4. Simulation of CSS_3 by KES_3

This does not only mean that the static expressivity of KES_3 is higher, but also that CSS_3 may not simulate any expansion $S^0 \subset \ldots \subset S^k$ in KES_3: KES_3 may proceed without the addition of new elements to S at points where CSS_3 is surely blocked and needs S to be expanded.

Example 4 Example 2 is redone below according to KES_3. The $\vee$-clauses have been a priori transformed to $\rightarrow$, but the same could have been done on-the-fly. Above the horizontal line is the knowledge base B and the denial of α. The branching rule PB is applied over `canine-teeth` after line 14.

$S = \varnothing$

1.	T `cow` $\rightarrow$ `grass-eater`	
2.	T `dog` $\rightarrow$ `carnivore`	
3.	T `canine-teeth` $\rightarrow \neg$`grass-eater`	
4.	T `carnivore` $\rightarrow$ `mammal`	
5.	T `mammal` $\rightarrow$ (`canine-teeth` $\vee$ `molar-teeth`)	
6.	T `grass-eater` $\rightarrow$ `mammal`	
7.	T `mammal` $\rightarrow$ `vertebrate`	
8.	T `vertebrate` $\rightarrow$ `animal`}	
9.	F `cow` $\rightarrow$ `molar-teeth`	
10.	T `cow`	$F_{\rightarrow} : 9$
11.	F `molar-teeth`	$F_{\rightarrow} : 9$
12.	T `grass-eater`	$T_{\rightarrow} : 1, 10$
13.	T `mammal`	$T_{\rightarrow} : 6, 12$
14.	T `canine-teeth` $\vee$ `molar-teeth`	$T_{\rightarrow} : 1, 10$

15'. T `canine-teeth` PB:T	15". F `canine-teeth` PB:F
16'. T $\neg$`grass-eater` $T_{\rightarrow} : 3, 15'$	16". T `molar-teeth` $T_{\vee} : 14, 15''$
S ={grass-eater}	$\times$
17'. F `grass-eater` $T_{\neg} : 16'$	
$\times$	

Note that the tableau starts with $S = \varnothing$. After line $16'$, it becomes blocked and S has to be expanded to allow for its closing. In that way, the tableau ends up with $S =$ {`grass-eater`}, a subset from the S computed in Example 2. It is interesting to note that this agrees with our motivational example, where Paul, besides his knowledge about cows and molar teeth, only had to take into account his knowledge about grass eaters.

What was the price payed for such an increase of static expressivity? The answer is: *loss of control* in the deduction process.

The *sensitivity* of a proof method depends on the set of new theorems ΔT we get when we move from S to $S \cup \Delta S$. Proof method 1 has *more control* than method 2 if it has more sensitivity, that is, if for the same ΔS, $\Delta T_1 \subseteq \Delta T_2$. Note that sensitivity and control are also dynamic properties.

In CSS_3, the set S has an effect over (i.e., controls) the set of atoms over which resolution can be applied. In KES_3, the set S controls the formulas over which $(T\neg)$ can be applied; by applying the transformation above, we eliminate $\neg$-formulas and thus reduce the control of S on KES_3 proofs. If we add to S an atom that only occurs non-negated in $B \vdash \alpha$, no new theorems are obtained in KES_3. We have thus shown the following:

Theorem 7

- *KES_3 is more expressive than CSS_3.*
- *CSS_3 has more control than KES_3.*

3.3 Recovering Control

In the previous section, we have seen that although the extension proposed to S_3 allows for more expressivity, we end up losing control over the resources used. Cadoli and Schaerf use resolution as the only inference rule and the set S determines the set of atoms over which resolution may be applied. In our system, modus ponens is valid even if S is empty, i.e., $\alpha \to \beta, \beta \vdash_{\mathrm{KE}S_3} \beta$.

If we want to regain control, we can add a restriction to the application of modus ponens, so that we always need part of the formulas to be in S. We end up with rules like these:

$$\frac{\begin{array}{c} T\ \alpha \to \beta \\ T\ \alpha \end{array}}{T\ \beta \text{ if } \alpha \in S^T_{\to}} \qquad \frac{T\ \neg\alpha}{F\ \alpha \text{ if } \alpha \in S^T_{\neg}}$$

where $S = S^T_{\neg} \cup S^T_{\to}$. This blocks the use of the transformation rule in Theorem 6, and the two systems can clearly simulate each other.

Therefore, the systems now have the same static expressivity, and the same control over the approximations. Furthermore, we see that the KES_3 system can be *fine-tuned* for more expressivity or more control, depending on the application.

Example 5 If we apply the new rule $(T \to)$ to Example 4, every line in which $(T \to)$ was applied would cause an expansion in $S^T_{\to}$ at each such line, namely:

- `cow` is added at line 12
- `grass-eater` is added at line 13
- `mammal` is added at line 14
- `canine-teeth` is added at line $15'$

And since in [9] every atom in α was implicitly considered part of S, we end up with the same S-set as in Example 2.

4 Conclusions and future work

We have presented an extension of Cadoli and Schaerf's S_3 system that deals with full propositional logic. We have given a proof method based on KE-tableaux, where the proofs can be built incrementally when we add new atoms to the context set S. We have shown that while our system is reacher in terms of expressivity, it allows for less control in the approximation. We have then shown that control can be regained without loosing the expressivity.

Ongoing work includes the implementation of a theorem prover based on KES_3.

ACKNOWLEDGEMENTS

Marcelo Finger is partly supported by the Brazilian Research Council (CNPq), grant PQ 300597/95-5. Renata Wassermann is partly supported by CNPq grant 68.0099/01-8. This work was developed under the CNPq project APQ 468765/00-0.

REFERENCES

[1] A.R Anderson and N.D Belnap, *Entailment: The Logic of Relevance and Necessity, Vol. 1*, Princeton University Press, 1975.
[2] Marco Cadoli and Marco Schaerf, 'Approximate inference in default logic and circumscription', *Fundamenta Informaticae*, **23**, 123–143, (1995).
[3] Marco Cadoli and Marco Schaerf, 'The complexity of entailment in propositional multivalued logics', *Annals of Mathematics and Artificial Intelligence*, **18**(1), 29–50, (1996).
[4] Samir Chopra, Rohit Parikh, and Renata Wassermann, 'Approximate belief revision', *Logic Journal of the IGPL*, **9**(6), 755–768, (2001).
[5] Newton C.A. da Costa, 'Calculs propositionnels pour les systémes formels inconsistants', *Comptes Rendus d'Academie des Sciences de Paris*, **257**, (1963).
[6] Marcello D'Agostino, 'Are tableaux an improvement on truth-tables? — cut-free proofs and bivalence', *Journal of Logic, Language and Information*, **1**, 235–252, (1992).
[7] Marcelo Finger and Renata Wassermann, 'Approximate reasoning and paraconsistency', in *8th Workshop on Logic, Language, Information and Computation (WoLLIC'2001)*, pp. 77–86, (July 31–August 3 2001).
[8] Hector Levesque, 'A logic of implicit and explicit belief', in *Proceedings of AAAI-84*, (1984).
[9] Marco Schaerf and Marco Cadoli, 'Tractable reasoning via approximation', *Artificial Intelligence*, **74**(2), 249–310, (1995).
[10] Raymond M. Smullyan, *First-Order Logic*, Springer-Verlag, 1968.
[11] Annette ten Teije and Frank van Harmelen, 'Computing approximate diagnoses by using approximate entailment', in *Proceedings of KR'96*, (1996).
[12] Annette ten Teije and Frank van Harmelen, 'Exploiting domain knowledge for approximate diagnosis', in *Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence (IJCAI'97)*, ed., M. Pollack, pp. 454–459, Nagoya, Japan, (August 1997).