Logic-based Fusion of Legal Knowledge

Philippe Besnard, Éric Grégoire, Sébastien Ramon

▶ **To cite this version:**

Philippe Besnard, Éric Grégoire, Sébastien Ramon. Logic-based Fusion of Legal Knowledge. 15th International Conference on Information Fusion (FUSION'12), Jul 2012, Singapour, Singapore. pp.587-592. hal-00866521

HAL Id: hal-00866521
https://hal.science/hal-00866521

Submitted on 26 Sep 2013

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Logic-Based Fusion of Legal Knowledge

Philippe Besnard
IRIT CNRS UMR 5505
118 route de Narbonne
F-31065 Toulouse, France
besnard@irit.fr

Éric Grégoire Sébastien Ramon
CRIL CNRS UMR 8188 – Université d'Artois
Rue Jean Souvraz SP18
F-62307 Lens, France
{gregoire,ramon}@cril.univ-artois.fr

Abstract—**In this paper, we address several issues that are often overlooked in the fusion of various knowledge components when this knowledge is expressed in full standard logic. It is shown that not solving these issues can lead to inadequate inferences. As a case study, we consider the legal domain, which has been an application target for knowledge engineering for decades. Real-life examples based on the confrontation of two different legal systems about a same international agreement are investigated.**

High-level information fusion, knowledge fusion, logic-based fusion, knowledge engineering

I. INTRODUCTION

Formalizing laws and rules from the legal domain and implementing them in decision systems has been a specific application target of Artificial Intelligence research for decades [1]. For example, some early expert systems were related to the legal domain [2], which was also one early major motivation and application area for logic programming [3]. Ever since, the logic-based formalization and computer-implementation of laws and legal knowledge has been a domain in its own right, with its own research community [4-7] and agenda [8].

Whereas the logic-based formalization and implementation of legal knowledge has been extensively studied, little attention has been given so far to the problem of fusing or confronting several components of legal knowledge. In this paper, we aim at contributing to answering the question: how could several sources of legal knowledge encoded in several components be fused so that a standard-logic artificial agent could reason about it?

A major issue that is not addressed in this paper occurs when several knowledge components are logically conflicting, i.e. inconsistent. Indeed a standard-logic artificial (or human) reasoning agent would be entitled to draw any conclusion and its contrary from an inconsistent set of information. The handling of inconsistent premisses is a multi-facets domain of research in its own right [9], and much focus about the fusion of several information components has concentrated on the situation where these components are mutually logically conflicting, also in the legal domain [10].

On the contrary, less work has been devoted to situations where all the components are both individually and mutually consistent. In this case, it is most often assumed that unioning the components is what is needed [9], when no specific preference schema must prevail. On the contrary, we show that the mere union of consistent knowledge components can lead to unexpected conclusions: specific care must be provided to solve that.

In this context, two specific problems are identified and addressed in the paper: the *subsumption* issue and the *need to fuse rules themselves*.

As a case study, we consider real-life examples based on the confrontation of the Belgian and the French national legal systems about their own bilateral agreement (and its addenda) preventing any double taxation from occurring [11]. In the following, we express conditions for a resident in Belgium to be taxable or free from taxation in Belgium, according to the (sometimes) diverging French and Belgian points of view. Assume thus that we need to fuse two separate knowledge components. Both come from one of the involved legal systems, namely the Belgian one. In the first one, it is asserted that a Belgian resident pays taxes in Belgium whereas the second one merely asserts the same rule with a proviso that the individual must not be a French civil servant. Namely,

$Resident \Rightarrow Taxable$

$Resident \text{ and not } French_Civil_Servant \Rightarrow Taxable$

Both rules do not contradict one another from a logical point of view. According to the standard artificial intelligence approaches to belief fusion [13], the fused set of information should include both rules. However, the second rule is *more informative* than the first one, as it is intended to preclude the first rule from applying to French civil servants. If both rules are kept in the fused knowledge then nothing will prevent the first rule from applying, even for French civil servants. Clearly, this is not what is expected from the fusion process: the second rule is expected to prevail over the first one, as it is more informative. From a logical point of view, the second rule is a mere deductive consequence of the first one: i.e., it is logically *subsumed* by it. A smart fusion engine should detect such situations and propose the user to override the subsuming rules by the subsumed one. Unfortunately, subsumption does not necessarily appear in such an explicit and clear manner. Actually, in the worst case, subsumption links can only be discovered by considering all the possible logical interactions between all pieces of knowledge. This issue is one of the problems that are addressed in the paper.

A second problem is linked to the following phenomenon. Actually, the two separate knowledge components also respectively contain:

Resident and not French_Civil_Servant ⇒ *Taxable*
Resident and not Work_in_France ⇒ *Taxable*

Clearly, none of these two rules subsumes the other one. Each of them belongs to a separate subpart of the legislation and is intended to encode an exception to the law that requires residents to pay taxes in Belgium.

Now, if we have to merge the two knowledge components, there is a clash of possible intuitions about the fusion process that should handle these two rules.

On the one hand, when the two rules express some sufficient conditions for a resident to be taxable, we must include both rules in the fused knowledge. In this way, nothing would prevent e.g. the first rule from concluding that a resident who is not a French civil servant but works in France to be taxable. However, this interpretation can be objected.

Indeed, it might be claimed that either being a French civil servant or working in France should be enough for blocking the law requiring a resident to pay taxes in Belgium. Under this interpretation, the two rules should be fused together and replaced by a unique rule: *Resident and not Work_in_France and not French_Civil_Servant* ⇒ *Taxable.* Indeed, this new rule would correctly encode that that either working in France or being a French civil servant is a sufficient condition for preventing the system to conclude that this resident is taxable in Belgium.

In the paper, we suggest a writing policy for rules that avoids this clash of intuitions, as much as possible. When and how rules should be fused together in this context is thus another concern of the paper.

II. LOGIC-BASED FRAMEWORK

To concentrate on the aforementioned conceptual problems, we consider the very simple framework of standard Boolean logic. Let $\mathcal{L}$ be a language of formulas over a finite alphabet $\mathcal{P}$ of Boolean variables, also called *atoms*. Atoms are noted a, b, c,… The ∧, ∨, ¬ and ⇒ symbols represent the standard conjunctive, disjunctive, negation and material implication connectives, respectively. A *literal* is an atom or a negated atom. Formulas are built in the usual way from atoms, connectives and parentheses; they are noted (resp. f, g, h,…). From a syntactical point of view, a *knowledge component KC* will be a set of formulas of $\mathcal{L}$. A formula is in conjunctive normal form (CNF) when expressed as a conjunction of clauses, where a clause is a disjunction of literals. Let a multi-set $\{KC_1,\ldots,KC_n\}$ of $n > 1$ knowledge components to be fused. The semantical concepts needed in the paper are as follows. Let Ω denote the set of all *interpretations* of $\mathcal{L}$, which are functions assigning either *true* or *false* to every atom. A *model* ω of *KC* is an interpretation of Ω that satisfies every formula of *KC*. *KC* is *consistent* when its set of models is not empty. $KC \vdash f$ expresses that the formula f can be deduced from *KC*, i.e. that it is *true* in all models of *KC*. We opt for a semantical (vs. a purely syntactical) regard of *KC*. Under this point of view, *KC* is identified with the set of all its deductive consequences.

Two central concepts in this paper are the *strict implicant* and the *subsumption* ones, defined as follows. Let f and g be two formulas. f is a strict implicant of g iff $f \vdash g$ but $g \not\vdash f$. *KC* subsumes g iff $KC \vdash f$ for some strict implicant f of g

A word of caution might also be needed for readers familiar with rule-based systems but not with logic. We exploit the full (sound and complete) inferential capability of Boolean logic, i.e. we do not only simply allow for mere forward and backward chaining on ⇒ as in traditional rule-based systems. For example, from the rule $f \Rightarrow g$ and $\neg g$, we infer $\neg f$ using contraposition. Also, the reader not familiar with logic should always keep in mind that a rule of the form $f \wedge g \wedge \neg h \Rightarrow i \vee \neg j$ is logically equivalent to $\neg f \vee \neg g \vee h \vee i \vee \neg j$, and will be treated as such.

III. POLICY FOR WRITING RULES IN A FUSION PERSPECTIVE

In the following, we resort to McCarthy's *Abnormality* notation [14] to emphasize and encode possible exceptions to rules (without switching here to a non-monotonic logic). Let $\mathcal{Ab}$ be a subset of $\mathcal{P}$. Its elements are noted $Ab_1,\ldots,Ab_m$ and called abnormality variables. They are intended to represent exceptions. For example, the rule "A resident who is not a French civil servant pays taxes" can be represented by the formulas:

Resident $\wedge \neg Ab_1 \Rightarrow$ *Taxable*
French_Civil_Servant $\Rightarrow Ab_1$

In the representation of legal knowledge, abnormality variables will appear in many rules to allow the representation of exceptions to the rules. We require that each KC_i uses a different subset of $\mathcal{Ab}$, and that these subsets share an empty intersection.

Actually, we expect the lawyer to encode rules with exceptions using this mechanism. As we have seen in the introduction, some circumstances will require the fusion of rules. In the next Sections, we will justify the fact that, in the general case, we only fuse formulas whose CNF version, on the one hand, contains only positive abnormality literals and, on the other hand, share identical subsets of other literals. Examples are *Resident* $\wedge \neg Ab_1 \Rightarrow$ *Taxable* and *Resident* $\wedge \neg Ab_2 \Rightarrow$ *Taxable*, which should fuse into *Resident* $\wedge \neg Ab_1 \wedge \neg Ab_2 \Rightarrow$ *Taxable*.

In this respect, two questions arise. First, what if the lawyer wants to encode the above two rules in *KC* as separate items? In the example, how could he (she) translate his (her) opinion that $\neg Ab_1$ and $\neg Ab_2$ are two sufficient conditions for a *Resident* to be taxable? Second, how could he (she) encode a rule like *Resident* $\wedge \neg Ab \Rightarrow$ *Taxable* inside a *KC* and prevent this rule from being fused with a similar one in another *KC*?

Concerning the first question: a solution is simply to allow such rules to coexist in a same *KC*. If the lawyer has not opted for their fused form in *KC*, then this would be interpreted as he (she) has decided that the abnormality variables translate various sufficient conditions. Several techniques can be suggested to solve the second question. The simplest one

would consist in introducing an additional (negative) $\neg Ab_i$ literal in the clausal form of the rule. In this way, according to our limited fusion-of-rules policy, which concerns clauses without negative $\neg Ab_i$ literals, these rules will never be fused together with other rules coming from other *KC*s.

IV. ADDRESSING THE SUBSUMPTION ISSUE

Actually, making sure that a more informative formula is not subsumed and thus "hidden" by other ones is a problem that must be handled both at the time of creating a knowledge component and when fusing several of them. Thus, we recommend applying the following technique at both steps. Also, keep in mind that formulas need not be explicitly present in *KC*, but can be mere logical consequences from it.

In order for a formula not to be subsumed in *KC* or in the fused *KC*s, all strict implicants *f* of a *g* must be dropped from these sets of formulas. If such an implicant is only a mere implicit consequence of *KC*, then all ways to conclude it should deleted. Interestingly, when the clausal form of the *KC*s is considered, it is enough to consider the longest (in terms of the number of involved literals) strict implicants of *g*. For example, assume *KC* contains the formula

$$Resident \wedge \neg Ab_1 \wedge \neg Ab_2 \wedge \neg Ab_3 \Rightarrow Taxable$$

The CNF version of the formula is the clause $\neg Resident \vee Ab_1 \vee Ab_2 \vee Ab_3 \vee Taxable$. Ensuring $KC \vdash \neg Resident \vee Ab_1 \vee Ab_2 \vee Ab_3$ ensures e.g. that $KC \vdash \neg Resident \vee Ab_1$ (otherwise, it would follow from $KC \vdash \neg Resident \vee Ab_1$ that $KC \vdash \neg Resident \vee Ab_1 \vee Ab_2$ and $KC \vdash \neg Resident \vee Ab_1 \vee Ab_2 \vee Ab_3$). Thus, if we need to make sure that a formula made of *n* literals is not subsumed, we only need to make sure that none of its *n* longest sub-formulas holds. At this point, two natural questions arise. How can we make sure that a strict implicant is not entailed? Which formulas should be checked to know whether they are subsumed or not?

In order to check whether a clause is subsumed or not, a first straightforward preprocessing step could check whether any of its *n* strict longest sub-clauses is actually present in the components. This can be performed efficiently in O(*n*,*m*), where *m* is the total number of clauses in the components. Obviously enough, this would not allow the detection of more complex subsumption links. From a computational point of view, checking whether a formula subsumes another one is co-NP-complete, and thus intractable in the worst case. However, recent dramatic progress in Boolean and search makes it often possible to get answers within seconds, especially thanks to powerful SAT solvers [16], which check whether a set of clauses is consistent or not. In [15], some of us have experimented a method that proves powerful to address the subsumption issue. The goal is not only to answer whether the formula is subsumed or not, but also to deliver clauses that can be required to be dropped in order to delete the subsumption links. The technique is based on the SAT-solvers technology and on methods [17] for delivering so-called MUSEs (Minimal Unsatisfiable Subsets). Assume we need to check whether *KC* (that contains *g*) subsumes *g* through a strict implicant *f* of *g*. In the positive case, the set $KC \cup \{\neg f\}$ is inconsistent. Then, the solver looks for the MUSes of this set, namely the (cardinality-)minimal subsets of clauses, that are inconsistent. Making sure that at least one clause in each of the MUSes is dropped ensures that the subsumption link is blocked. Dropping such clauses can be automatic, or the lawyer can be asked whether he (she) really wants to drop them, or even be given the choice of selecting the clauses to be dropped in the MUSes. When he (she) prefers keeping these MUSes intact, he (she) is then conducted to revise his (her) former requirement about the subsumption-freeness status of *g*. This solver provides efficient results even for huge *KCs*, provided that the number of MUSes remains small [15].

Now, it should be clear that such a process remains nevertheless computationally heavy. Accordingly, we should restrict its use for checking the subsumed status of a limited number of clauses. In this respect, we would suggest the lawyer to restrict the application of this process to formulas that, at the same time, are explicitly present in the components and that contain Ab_i variables, translating explicit rules with possible exceptions. Furthermore, only rules where he (she) would not accept less Ab_i to be included in the rules should be checked about their subsumption status.

Finally, a word of caution, the search for subsuming formulas of a clause *g* should consider all the information in the components without the facts that correspond to literals mentioned in *g*, since those facts naturally subsume *g*. Indeed, assume $\neg resident$ is established in the components. This fact should not be dropped because it subsumes a rule of the form $resident \wedge \ldots \wedge \ldots \Rightarrow \ldots$

V. FUSING RULES THEMSELEVES

In the general case, a rule of the form $a \wedge \neg Ab \Rightarrow b$ needs not be explicitly present in a knowledge component *KC*: it might be an implicit logical consequence of *KC*. Moreover, as soon as e.g. $\neg a$ is *true*, e.g. $a \wedge \neg Ab \Rightarrow b$, $a \Rightarrow b$ and $a \Rightarrow \neg b$ can also be deduced from *KC*. Accordingly, for each *possible* rule, checking whether it should be fused or not, is an intractable task.

We thus check rules that are explicitly present in the components, only. Our very limited target concerns the rules that only differ through their negative literals $\neg Ab_i$ (positive Ab_i in the clausal form of the rule) and, at the same time, belong to different knowledge components. For example $a \wedge \neg Ab_1 \Rightarrow b$ and $a \wedge \neg Ab_2 \Rightarrow b$ are candidates to form $a \wedge \neg Ab_1 \wedge \neg Ab_2 \Rightarrow b$. Having identified such rules, we either ask the lawyer if he (she) accepts such a fusion of the rules to occur, or more bluntly, if the lawyer agrees so we might adopt a policy requiring them to be fused automatically.

For example, assume:

KC_1: $Work_in_France \wedge \neg Ab_1 \Rightarrow \neg Taxable$
$Frontier_Status \Rightarrow Ab_1$

KC_2: $Work_in_France \wedge \neg Ab_2 \Rightarrow \neg Taxable$
$Belgian \wedge French_Civil_Servant \Rightarrow Ab_2$ [1]

[1] This is the French version of the agreement (French *Conseil d'État* 9/11/1987 N°51075). The Belgians opposed this interpretation for a while: according to the Belgians, Belgians who are both French Civil Servants and Belgian residents should not pay taxes in Belgium.

Fusing the first two rules yields

KC_{Fused}: $Work_in_France \wedge \neg Ab_1 \wedge \neg Ab_2 \Rightarrow \neg Taxable$
$Frontier_Status \Rightarrow Ab_1$
$Belgian \wedge French_Civil_Servant \Rightarrow Ab_2$

Not fusing them would allow us to conclude e.g. that a resident in Belgium who is French, works in France and has the special frontier worker status is free from taxation in Belgium, which would be a wrong interpretation according to both countries.

Now, it might happen that both formulas $a \wedge \neg Ab_1 \Rightarrow b$ and $a \wedge \neg Ab_2 \Rightarrow b$ do coexist in a same KC_i. In this specific case, we assume that the lawyer has thus expressed two different sufficient conditions for b to be derived from a. Accordingly, we do not merge them. If his (her) goal was to express that these two conditions must be met at the same time, he (she) would have had to express $a \wedge \neg Ab_1 \wedge \neg Ab_2 \Rightarrow b$ directly.

It is straightforward to program a preprocessor that checks for these kinds of rules to be fused and that are coming from several components, since it resorts to a mere syntactical search among formulas. Obviously, replacing formulas by fused ones can be done in constant time. For example, when formulas are in CNF, constructing a fused component from several ones can be done in O(n log n), where n is the total number of clauses in the components. When the clauses are sorted according to a lexicographic order, this complexity even deflates to O(n).

Due to the specific role for Ab_i variables, we shall not recommend to fuse other types of formulas in the general case. Let us illustrate this through the main other possible forms of interactions between formulas involving Ab_i.

First, in the general case, it is not acceptable to fuse together formulas whose antecedents differ on other literals than abnormality ones. We will thus not fuse together $a \wedge \neg Ab_1 \Rightarrow b$ and $a \wedge c \wedge \neg Ab_2 \Rightarrow b$. To illustrate this, assume that the first formula asserts that a resident pays taxes and that the second one asserts that a resident who has a special frontier-worker status pays taxes. Then, merging both formulas into $a \wedge c \wedge \neg Ab_1 \wedge \neg Ab_2 \Rightarrow b$ would only allow us to infer that a resident pays taxes when he is a frontier-worker; residents who are not working in France would thus be free from taxation in Belgium: this would clearly be a wrong conclusion.

Also, we will not fuse together formulas that are identical except w.r.t. their positive Ab_i literals: in the general case, we should not fuse $a \wedge Ab_1 \Rightarrow b$ and $a \wedge Ab_2 \Rightarrow b$ into $a \wedge Ab_1 \wedge Ab_2 \Rightarrow b$. Indeed, as Ab_i represent exceptions, this would require the simultaneous occurrence of the exceptions encoded by Ab_1 and Ab_2, in order to infer b from a. Clearly, this would be an unacceptable weakening of the information described in the initial formulas. For example, according to the French point of view, we could have:

KC_1: $Work_in_France \wedge Ab_1 \Rightarrow Taxable$
$Frontier_Status \Rightarrow Ab_1$

KC_2: $Work_in_France \wedge Ab_2 \Rightarrow Taxable$
$Belgian \wedge French_Civil_Servant \Rightarrow Ab_2$

Fusing the first rules of the KCs would lead to

KC_{Fused}: $Work_in_France \wedge Ab_1 \wedge Ab_2 \Rightarrow Taxable$
$Frontier_Status \Rightarrow Ab_1$
$Belgian \wedge French_Civil_Servant \Rightarrow Ab_2$

This would require the two sufficient conditions for a person working in France to be taxable to be satisfied at the same time in order for being taxable in Belgium (namely having the frontier worker status and being both a Belgian and a French civil servant should be satisfied), which was clearly not the intended meaning of the expressed knowledge. A same analysis can be held w.r.t. pairs of formulas of the form ($Ab_1 \Rightarrow b$, $Ab_2 \Rightarrow b$).

In the general case, it does neither seem acceptable to fuse two formulas that are identical except that the first one refers to e.g. Ab_1 and the other one to $\neg Ab_2$, without knowing the actual priority or specificity relation linking Ab_1 and Ab_2. For the sake of the generality, we will neither merge them. For example, according to the French point of view, we could write

KC_1: $French_Civil_Servant \wedge Ab_1 \Rightarrow \neg Taxable$
$French \Rightarrow Ab_1$

KC_2: $French_Civil_Servant \wedge \neg Ab_2 \Rightarrow \neg Taxable$
$Belgian \Rightarrow Ab_2$

Merging the first two rules would lead to

KC_{Fused}: $French_Civil_Servant \wedge Ab_1 \wedge \neg Ab_2 \Rightarrow \neg Taxable$
$French \Rightarrow Ab_1$
$Belgian \Rightarrow Ab_2$

and would not allow us to infer anymore that e.g. a French civil servant who is Dutch should be free from taxes in Belgium. In this specific case, not being Belgian is more important than being French in order for being free from taxes in Belgium according to the French point of view. However, it is easy to find situations where *not condition*$_1$ should be given a lower priority than *condition*$_2$. Accordingly, for the sake of generality, we shall not merge such kinds of rules.

Since it has been assumed that each KC_i uses its own subset of Ab_i propositions and that these subsets share an empty intersection, pairs of formulas of ($a \Rightarrow Ab_1$, $b \Rightarrow Ab_1$) and ($a \Rightarrow \neg Ab_1$, $b \Rightarrow \neg Ab_1$) can only occur in a same KC_i. The first pair of formulas generally expresses sufficient conditions for an exception to occur. In the general case, we will not fuse them so that the exceptions should occur simultaneously in order for the inference to be permitted. For example, we could have, according to the French point of view:

KC_1: $Work_in_France \wedge \neg Ab_1 \Rightarrow \neg Taxable$
$Frontier_Status \Rightarrow Ab_1$
$Belgian \wedge French_Civil_Servant \Rightarrow Ab_1$

Fusing the last two rules to yield

KC_{Fused}: $Work_in_France \wedge \neg Ab_1 \Rightarrow \neg Taxable$
$Frontier_Status \wedge Belgian \wedge French_Civil_Servant \Rightarrow Ab_1$

would allow us to infer that a French having the special frontier worker status working in France is free from taxation, which is a wrong conclusion.

Formulas of the type $a \Rightarrow \neg Ab_1$ describe conditions for not having an exception. Since in the general case, we cannot list

such conditions exhaustively and since we assume that they occur in a same KC_i to be fused, we however assume that the lawyer has decided to give them a status of sufficient condition (otherwise, he (she) would have had to merge them into $(a \wedge b) \Rightarrow \neg Ab_1$). Finally, we will neither fuse together pairs of formulas taken from different categories discussed in this Section.

VI. PERSPECTIVES AND CONCLUSONS

Although this research has taken the legal domain as a case study, its findings and results can be applied to many other domains. Especially, they might be of interest to various areas traditionally covered by the information fusion research community, including control, supervision and decision systems in the defense domains.

Let us give some examples in that respect. First, consider the subsumption issue. Assume that we need to fuse two different knowledge systems. The first-one includes the rule "*It the target is located within the security zone then open fire*". The second one is more precise and recommends the same decision, except when the target is friendly. A sound and complete standard logic-based decision system that would include both rules would recommend, based on the first rule, to open fire on the target, even when it is a friendly one. Clearly, the second rule is a logical consequence of the first one. However, we do not want it to be preempted by the first, less precise, one. It is important to note that both rules should not necessarily be explicit: they might be mere deductive conclusions that can be derived from some other information. Also, the subsumption issue and the way we solve it do not rely on the syntactical form of the involved formulas. Let us illustrate that this also occurs for e.g. disjunctive formulas. Assume that a recognition system concludes *enemy-tank or enemy-SUV*. Assume that a second system concludes *enemy-tank or enemy-SUV or friendly-truck*. The second sentence is a logical consequence of the first one (indeed, whenever the first formula is true, the second one is also true). If we want the second disjunction to prevail over the first one, then we need to retract any ability to infer the first one.

Examples requiring fusing rules also abound in the defense domain. Assume that we have two rules: *If the switch is on and the switch is not broken then the light is on* and *If the switch is on and the lamp-bulb is not broken then the light is on*. Clearly, we need to fuse both rules to yield *If the switch is on and the lamp-bulb is not broken and the switch is not broken then the light is on*. If we do not fuse the rules then we would be able to derive that the light is on when the switch in on, even when the lamp-bulb is broken. A similar problem occurs, as when additional conditions like *lamp-bulb is not broken* are actually assumptions by default. All the techniques proposed in this paper thus apply here.

The present work might need some technical extensions to better cope with these other domains. First of all, quantitative measures of uncertainty could be grafted to rules and taken into account in our fusion calculus, which was not necessary in the legal domain. Also, the expressivity of the logic could be extended in several directions. First, all the results in this paper are directly extendable to finite versions of first-order logic. Second, the inference mechanism could be extended to a non-monotonic one in order e.g. to represent exceptions by default. In this respect, [18] provides a first promising study in that direction. Note that the Abnormality apparatus is very suitable for a natural augmentation of the reasoning paradigms of logic with a reasoning-by-default one. Finally, it might also be of interest to graft this study to other expressively powerful representation mechanisms, such as description logic. We intend to pursue these promising lines of research in the future.

ACKNOWLEDGMENT

This work has been supported in part by the *Région Nord/Pas-de-Calais* and the EC through a FEDER grant.

REFERENCES

[1] L. Thorne McCarty, "Reflections on Taxman: An Experiment in Artificial Intelligence and Legal Reasoning,", Harvard Law Review, vol. 90(5), pp. 837-893, 1976.

[2] R.E. Susskind, Expert Systems in Law: A Jurisprudential Inquiry, Oxford University Press, 310 pp., 1987.

[3] M.J. Sergot, F. Sadri, R.A. Kowalski, F. Kriwaczek, P. Hammond and H.T. Cory, "The British Nationality Act as a Logic Program,", Communications of the ACM, vol. 29(5), pp. 370-386, 1986.

[4] Proceedings of The Thirteenth International Conference on Artificial Intelligence and Law (ICAIL 2011), Pittsburgh, PA, ACM Press, 2011

[5] Proceedings of the 24th Annual Conf. Jurix (JURIX 2011), K.M. Atkinson (ed.), Frontiers in Artificial Intelligence and Applications, vol. 235, 2011.

[6] Artificial Intelligence and Law, vol. 19, Springer, 2011.

[7] Jurix Foundation http://www.jurix.nl/

[8] http://infomedia.uib.no/deon2012/

[9] D. Zhang, É. Grégoire, "The landscape of inconsistency: a perspective", International Journal of Semantic Computing (IJSC), vol. 5(3), World Scientific Press, 2011.

[10] J. Carrère, L. Cholvy, F. Cuppens, Cl. Saurel., "Merging regulations: analysis of a practical example", International Journal of Intelligent Systems, Special Issue on Fusion, vol. 16(11), 2001.

[11] Bilateral Belgian-French Taxation Agreement, available from http://www.impot.gouv.fr/portal/deploiement/p1/fichedescriptive_1425/fichedescriptive_1425.pdf (in French)

[12] S. Konieczny, É. Grégoire, "Logic-based information fusion in artificial intelligence", Information Fusion, vol. 7(1), pp. 4-18, 2006.

[13] S. Konieczny, R. Pino Perez, "On the logic of merging", Proc. of the 6th International Conference on Principles of Knowledge Representation and Reasoning (KR'98), pp. 488-498, 1998.

[14] J. McCarthy, "Applications of circumscription to formalizing common-sense reasoning", Artificial Intelligence, vol. 29, pp. 89-116, 1986.

[15] Ph. Besnard, É. Grégoire, S. Ramon, "Enforcing Logically Weaker Knowledge in Classical Logic", Proceedings of the 5th International Conference on Knowledge Science Engineering and Management (KSEM'11), LNAI 7091, Springer, pp. 44-55, Irvine (CA), 2011.

[16] http://www.satlive.org/

[17] É. Grégoire, B. Mazure and C. Piette, "Using local search to find MSSes and MUSes", European Journal of Operational Research, vol. 199(3), pp. 640-646, 2009.

[18] Ph. Besnard, É. Grégoire, S. Ramon, "Overriding Subsuming Rules", Proc. of the 11th European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty (ECSQARU'11), LNAI 6717, Springer, Belfast, pp. 532-544, 2011.