A Logic for Legal Hierarchies

Pierre-Yves Schobbens*

Fac. Univ. Notre-Dame de la Paix
Institut d'Informatique
Rue Grandgagnage 21
B-5000 Namur, Belgium
pys@info.fundp.ac.be

Abstract

The theory of non-monotonic reasoning has interesting applications for the formalization and automated use of legal concepts, specially:

- *drawing conclusions from a logically inconsistent, but hierarchic, regulations [1, 30];*
- *similarly, establishing facts from a set of inconsistent testimonies, partially ranked by confidence;*
- *using presumptions (such as the presumption of innocence) in the face of possibly contradictory evidence.*

In this paper, we use a logic [37, 38], that ranks contradictory formulae using two new paraconsistent variants of conjunction: "but" and "on the other hand". Its algebraic proof theory is presented.

1 Introduction

When representing legal knowledge (laws, norms, etc.) in expert systems, the knowledge analyst is often faced with logical contradictions between different law fragments. Expert systems based on classical logic collapse when faced with contradictions, since *any* consequence can be drawn from a single contradiction in classical logic, spoiling thus the whole knowledge base in face of a single fault. On the other hand, many legal texts use logical contradiction and hierarchy as a systematic structuring mean[29]. Fortunately, this problem is solvable in most modern expert systems, using meta-knowledge on the precedence of legal texts.

A similar problem occurs in the fact base, when contradictory testimonies must be used to try to establish facts; again, this problem is often solved using meta-knowledge. Lastly, the same mechanism can be used to model presumption, which can be understood as resolving a contradiction between a rule in the knowledge base and stronger evidence from the fact base.

Although the operational solutions usually found in expert systems often allow a correct treatment, they lack a logical basis. In this paper, we propose a logic to draw sound conclusions from a legal hierarchy, and examine its logical properties.

The need for such a logic has been recognized for long by theorists of law [27, 18], but the mathematical basis (non-monotonic logics) has been developed only recently, starting with [1]; see [25, 33] for an overview.

2 Logical basis

In this section, we outline the mathematical basis of our approach: the formal definitions are given in appendix. Our framework, like [1, 11], is parameterized by the underlying logic. The examples of this paper will use classical logic for simplicity, but an accurate modelling of legal reasoning requires a temporal and deontic logic [16]. This parameter logic must be presented in the form of a model theory, in the style of the *institutions* of [12]. In contrast, [1] require a consequence relation (a proof theory). As a supplementary parameter, our theory require the definition of a *closeness* between models, a "partially ordered distance" (A.1).

Example 2.1 The measure of closeness proposed by [6] for propositional logic is the number of predicates having a different truth value in each of the two models; this definition:

- does not extend to first-order logic;
- is sensitive to replication (using two different names for the same predicate.)

Example 2.2 (A.1) Here, we use the following two sets of predicates value as a measure of closeness: those that are false in the first model and true in the second, and conversely. Closeness is measured by set inclusion: when both sets are empty, the two models are perfectly close (and indeed identical.)

*This research was mostly carried out at Centre National de la Recherche Scientifique, Centre de Recherche en Informatique de Nancy (CRIN).

Example 2.3 (A.2) The extension of this measure to first-order logic involve some subtleties to deal adequately with equality [36].

2.1 Syntax and semantics

Given such a model theory and a closeness between models, our main object of study is the *poset*: a partial *reliability* order [1] among *witnesses*.

Example 2.4 Poirot tries to resolve a case where important documents have been stolen from an office. He is faced with 3 witnesses: Anna, Bob and Cliff. He has no reason to give more credit to Bob or Cliff, but knows that Anna, the room maid, is a more reliable witness. So the reliability order is here:
Anna $<_W$ Bob, Anna $<_W$ Cliff.

Each witness has a *testimony*: a formula of the original logic.

Example 2.5
- Anna says: "I have seen Mr. Bob entering the office."
- Bob says: "I did not enter the office. I have seen Cliff entering the office."
- Cliff says: "I did not enter the office. I have seen somebody (either Anna or Bob) entering the office."

Taking A, B, C as atomic propositions for "Anna entered the office", etc., we obtain:

- $A_t = \Phi(Anna) = B$
- $B_t = \Phi(Bob) = (\neg B \wedge C)$
- $C_t = \Phi(Cliff) = (\neg C \wedge (A \Leftrightarrow \neg B))$

opWhen dealing with paraconsistent sets of regulations, the terms *legal hierarchy, precedence, law, logical content* are used, respectively.

Based on this ordering, we define the models of a poset as the models that are closest to the models of the testimonies, according to the usual lexicographic ordering:

$$\vec{h} \leq_G \vec{h}' \Leftrightarrow \forall\, w \in W, h_w \leq h'_w \vee \exists\, t <_W w,\;\; h_t < h'_t$$

where h_w represents the distance between the candidate model, and a model of w's testimony. So a model is better than another if for any witness, either the model is closer to some model of the testimony, or it has a good excuse: a more reliable testimony to satisfy strictly better.

Example 2.6 Here, as the testimonies are contradictory, no model can satisfy all of them, but some models are more likely than others. A model where nobody entered the office (A, B, C are all false) is clearly less likely than a model where only Cliff entered the office, since at least this one agrees with the testimony of Bob. However, a model where only Bob entered the office is even more likely, since this one agrees with the testimony of Anna, and Anna is more reliable.

To have a textual expression of posets, we introduce operators, that are posets with names instead of some testimonies. A name is just a placeholder that will later be replaced by a poset. The two operators mentioned in the abstract are the simplest operators of this kind, with two placeholders: "but" (noted /) gives precedence to its second argument, while "on the other hand" (noted $||$) treats its two arguments as equally reliable. We also have the single-node empty poset as operators. These operators are complete, in the sense that any finite poset is equivalent to some expression of our syntax. Our definitions thus give as models of ϕ_1/ϕ_2 the models of ϕ_2 that are the closest to ϕ_1. When ϕ_1 and ϕ_2 are consistent, this reduces to $\phi_1 \wedge \phi_2$.

Example 2.7 Poirot's mental poset can now be described as $(B_t||C_t)/A_t$, pronounced:
Bob says: "I did not enter the office. I have seen Cliff entering the office." On the other hand Cliff says: "I did not enter the office. I have seen somebody (either Anna or Bob) entering the office." But Anna says: "I have seen Mr. Bob entering the office."

This expression can now be solved by algebraic manipulations, see 2.2.

The operation $[G]$ takes a poset G, and returns its class of minimal models. Since, in our approach, sentences are considered as defining classes of models, $[G]$ can thus be used as a sentence of the original language, and in particular used as a testimony.

In summary, the syntax of our logic is:
$G ::= G_1/G_2 \mid G_1||G_2 \mid \varnothing \mid \phi$
$\phi ::= \phi_o \mid [G]$
where G, G_1, G_2 are expressions denoting posets, ϕ is a formula of the extended logic (denoting a class of models), ϕ_o is a formula of the original logic (e.g., propositional logic).

2.2 Proof theory

From the semantics, we can derive the validity of the following proof rules. We use equivalence of posets (A.2.2), and note that our operators form a kind of ring:

Im	$G\|\|G \equiv G$	
As	$G_1\|\|(G_2\|\|G_3) \equiv (G_1\|\|G_2)\|\|G_3$	
Id	$G\|\|\varnothing \equiv G$	
IdT	$G\|\|\top \equiv G$	if $G \neq \varnothing$
Ab	$G\|\|\bot \equiv \bot$	
Com	$G_1\|\|G_2 \equiv G_2\|\|G_1$	
ImB	$(G/G) \equiv G$	
AsB	$G_1/(G_2/G_3) \equiv (G_1/G_2)/G_3$	
*Id*1	$\varnothing/G \equiv G$	
*Id*2	$G/\varnothing \equiv G$	
*IT*1	$\top/G \equiv G$	if $G \neq \varnothing$
*IT*2	$G/\top \equiv G$	if $G \neq \varnothing$
*Ab*1	$\bot/G \equiv \bot$	
*Ab*2	$G/\bot \equiv \bot$	
Dis	$G_1/(G_2\|\|G_3) \equiv (G_1/G_2)\|\|(G_1/G_3)$	

This equivalence is a congruence for all our operations:

RG	$\phi_1 \Leftrightarrow \phi_2 \vdash \phi_1 \equiv \phi_2$
RO	$G_1 \equiv G_2 \vdash G_1 \vert\vert G \equiv G_2 \vert\vert G$
$RB1$	$G_1 \equiv G_2 \vdash G_1/G \equiv G_2/G$
$RB2$	$G_1 \equiv G_2 \vdash G/G_2 \equiv G/G_1$
RA	$G_1 \equiv G_2 \vdash [G_1] \Leftrightarrow [G_2]$

All rules of the original logic (e.g. propositional logic) apply, even to formulae of the extended language.

Considering a formula as a graph is a transparent operation:

$Nop \quad [\phi] \Leftrightarrow \phi$

Finally, we have two powerful rules:

OR	$[G[n := (\phi_1 \vee \phi_2)]] \Rightarrow ([G[n := \phi_1]] \vee [G[n := \phi_2]])$
$K7$	$([G_1/\phi_1] \wedge [G_2/\phi_2]) \Rightarrow [op(G_1, G_2)/(\phi_1 \wedge \phi_2)]$

In the first, G is an operator containing the name n. This rule reflects the fact that our logic is model-based.
In the second, op is a pure binary operator; in particular, it can be any combination of $/, ||, \varnothing$.

The rules above do not depend on a specific closeness: they are applicable in any default institution. The following rules can be derived (a proof indication is on the right):

WS	$[G/\phi] \Rightarrow \phi$	$(K7)$
And	$([G/\phi_1] \wedge [G/\phi_2]) \Rightarrow [G/(\phi_1 \wedge \phi_2)]$	$(K7)$
$K3$	$([G] \wedge E) \Rightarrow [G/E]$	$(K7)$
ORB	$[G/(\phi_1 \vee \phi_2)] \Rightarrow ([G/\phi_2] \vee [G/\phi_2])$	$(K7;\ OR)$
ABA	$[G/\phi_1] \wedge \phi_2 \Rightarrow [G/(\phi_1 \wedge \phi_2)]$	$(K7)$
HL	$[\phi_1/\phi_2] \Leftrightarrow (\phi_1 \vee [\phi_1/\phi_2]) \wedge \phi_2$	$(K7)$
IMP	$[G_1] \Rightarrow [G_2] \vdash [G_1] \Rightarrow [op(G_1, G_2)]$	$(K7)$
OI	$\neg\phi_1 \vdash \neg[\phi_1 \vert\vert \phi_2]$	(Ab)
$BI1$	$\neg\phi_1 \vdash \neg[\phi_1/\phi_2]$	$(Ab2)$
$BI2$	$\neg\phi_2 \vdash \neg[\phi_1/\phi_2]$	$(Ab1)$
$CO1$	$\phi_1 \vee \phi_2 \vdash [\phi_1/\phi] \vee [\phi_2/\phi] \Leftrightarrow \phi$	$(K7, OR)$
$CO2$	$\phi_1 \vee \phi_2 \vdash [G] \Rightarrow [G/\phi_1] \vee [G/\phi_2]$	$(ORB, BT1)$
CM	$\dfrac{[G/\phi_1] \Rightarrow \phi_2 \quad [G/(\phi_1 \wedge F2)] \Rightarrow \phi_3}{[G/\phi_1] \Rightarrow \phi_3}$	$(K7)$

When the institution is connected, i.e. there is a morphism between any pair of models (as is the case with our propositional institution) we may use:

$L \quad [G_1/G_2] \Rightarrow [G_1/[G_2]]$

Note that these rules are invariant under uniform substitution: by replacing a propositional symbol by a formula in a theorem, we obtain another theorem.

All rules above are sound, but clearly not complete for any specific default institution. To treat our propositional institution, the following rule alone is already complete:

$OG \quad [G] \Leftrightarrow \bigvee\{\psi \mid d(\psi, \Psi) \in Min(d(\psi, \Psi))\}$

where:

- ψ is a consistent conjunction of literals (notationally identified with a set of literals).
- Ψ is a labelling of W (the witnesses of G) by consistent conjunctions of literals, such that $\Psi(w) \Rightarrow \Phi(w)$.
- The *distance* $d(\psi, \Psi)$ is the labelling of G given by $d(\psi, \Psi)(w) = \{l \mid l \in \Psi(w) \wedge (\neg l) \in \psi\}$, with the constraint that any literal l appearing in a Ψ must be decided upon in ψ: $l \in \psi$ or $(\neg l) \in \psi$.
- The order between distances is given by the lexicographic combination of set inclusion.

From this rule an algorithm to reduce extended formulae to propositional formulae is easily derived [37], showing the decidability of the logic.

For our first-order institution, the second-order rule $2E$ (described in appendix B) is complete.

3 Legal applications

This section shows through simple examples how our logic is can be used in several areas of legal reasoning.

3.1 Ranking testimonies

Example 3.1 We are now in position to solve our example 2.4. Poirot's poset is: $((\neg B \wedge C) || (\neg C \wedge (A \Leftrightarrow \neg B)))/B$. The reader can check, using the model-theoretic definitions, that the models of this poset are the same as those of $\neg A \wedge B$.

The implicit reasoning performed by our logic is the following: Anna is reliable, so that B is true (we apply rule WS). Cliff says that either Anna or Bob entered the office, and is not contradicted. Knowing that Bob entered the office, Anna did not. About Cliff, we have two contradictory testimonies, and no one is more reliable than the other, so we do not conclude (but certainly we do not want to conclude every formula, as with classical conjunction).

This result can also be obtained by rule OG. We have two implicants for $\Psi(\mathit{Cliff})$, namely (1) $A \wedge \neg B \wedge \neg C$ and (2) $\neg A \wedge B \wedge \neg C$. For the others witnesses we can merely take $\Psi_i(w) = \Phi(w)$.

- Taking $\psi_1 = \neg A \wedge B \wedge C$:
  - $d(\psi_1, \Psi_2)(Anna) = \varnothing$
  - $d(\psi_1, \Psi_2)(Bob) = \neg B$
  - $d(\psi_1, \Psi_2)(\mathit{Cliff}) = \neg C$
- Taking $\psi_2 = \neg A \wedge B \wedge \neg C$:
  - $d(\psi_2, \Psi_2)(Anna) = \varnothing$
  - $d(\psi_2, \Psi_2)(Bob) = \neg B \wedge C$
  - $d(\psi_2, \Psi_2)(\mathit{Cliff}) = \varnothing$

These distances are incomparable for the lexicographic ordering. We easily check that other values for ψ and Ψ are not minimal.

Putting brackets around a poset is not innocuous: it can be thought as reporting conclusions drawn from other sources, without citing them. In [11, 35], an operation $[\phi_1/\phi_2]$ was introduced, without recognizing the nature of its constituents. The combination of several testimonies often gave paradoxical results, as first noted in [15], and shown in the following examples.

Example 3.2 The example of [15] is the following: Poirot arrives at night in a city. He meets a witness telling him that they are two restaurants in town, where they are, and that their opening hours ensure that at least one of them is always open. He goes to the closest one, and from far away he sees the lights on. But when trying to push the door, he sees that the door is locked and that there is nobody inside.

Let A, B represent the fact that the closest (resp. farthest) restaurant is open. The detective first learns $A \vee B$, then (more reliably) A, and finally $\neg A$. If we use the operation $[../..]$ to integrate new, more reliable information, we obtain first $[(A \vee B)/A] = A$, and then $[A/\neg A] = \neg A$, so that the detective should paradoxically believe nothing about restaurant B. But if we use the operator $../..$, we obtain $(A \vee B)/A/\neg A$, which has the same models as $\neg A \wedge B$, as intuitively expected.

Example 3.3 Assume that our detective sent a young assistant to question Bob and Cliff. From their testimonies he concludes $[(\neg B \wedge C)||(\neg C \wedge (A \Leftrightarrow \neg B))] = (A \wedge \neg B)$. Our logic reconstructs the assistant reasoning: Whether Cliff entered the office is disputed. But nobody contests that either Anna or Bob did, nor that Bob did not. So Anna did.

He then reports his opinion to the detective, who concludes $(A \wedge \neg B)/B = (A \wedge B)$: since he does not know why his assistant believes Anna entered the office, he does not cancel that conclusion.

Example 3.4 The Attorney General has a supplementary but highly unreliable testimony: a tramp reporting that Cliff entered the office. If he takes the opinion of his detective (which has no opinion about this, as seen above) and of the tramp, he will conclude that Cliff entered the office. Formally $C/[((\neg B \wedge C)||(\neg C \wedge (A \Leftrightarrow \neg B)))/B]$ is $C/(\neg A \wedge B)$, which is simply $C \wedge \neg A \wedge B$: Since no contradictions are present, "but" reduces to "and".

If the Attorney had questioned the witnesses directly, he would instead have the mental poset: $C/((\neg B \wedge C)||(\neg C \wedge (A \Leftrightarrow \neg B)))/B$, from which he can conclude that C is disputed between Bob and Cliff, and that the testimony of the tramp adds no value to the one of Bob. So he would conclude $\neg A \wedge B$, like the detective.

3.2 Hierarchies of laws

An interesting application of our logic allows drawing conclusions from a contradictory set of laws, where a precedence is known.

Example 3.5 The legal tradition admits that constitutional law (C) has precedence over common law (M), and over administrative regulations (A); no precedence is admitted between the last two. This precedence can be described by $(M||A)/C$, where C, M, A are expressions of our logic, probably containing further precedence operators.

Assume that:

- the constitution states that expressing a personal opinion is not reprehensible;
- the common law states that expressing publicly a slanderous personal opinion is reprehensible;
- an administrative regulations says that this is reprehensible, or at least prejudicial.

Using the atomic proposition R to represent that expressing publicly a slanderous personal opinion is reprehensible, and P, that it is prejudicial, we obtain $(R||(R \vee P))/\neg R$, which has the same models as $P \wedge \neg R$, as usually admitted.

Most laws contain implicit or explicit suppositions that have lower precedence, so that even within a single legal text a hierarchy may be present.

Example 3.6 [27] The French Civil Code only lists the cases where a settlement is void; the implicit assumption is, obviously, that any settlement not listed is valid. We have thus common sense (CS) telling us that acts (in particular, settlements) are normally valid: $(\forall x, act(x) \Rightarrow \neg void(x))$while the Civil Code ($CC$) gives some more specific opposite information, for instance:
"Any settlement between wife and husband is void"
$\forall x, (settlement(x) \wedge conjugal(x) \Rightarrow void(x)$
The Civil Code even contains places for explicit exceptions: "The marriage contract (MC) can, however, establish a different rule". So the expression treating this example is : $CS/CC/MC$.

3.3 Presumptions

Presumptions are clearly a non monotonic form of legal reasoning: their meaning can only be understood as a norm that has to be defeated by facts and other legal texts taking precedence.

Example 3.7 The Belgian law (BL) states that innocence is presumed:
$\forall x : person, f : crime. \neg guilty(x, f)$
However, the fiscal law (FL) inverse the onus of proof: in some cases, the defendant has to prove his innocence. This exception to a presumption can be represented by:
$\forall x, f. fiscal(f) \wedge suspected(x, f) \Rightarrow guilty(x, f)$
Similarly, the introduction of European directives (ED) into the Belgian civil law inverse the onus of proof for producers:
$\forall x, f, g. produces(x, g) \wedge causes(g, f) \Rightarrow guilty(x, f)$
In any case, these presumptions can be cancelled by testimonies T, that can be treated as shown above. The resulting expression is thus $BL/(FL||ED)/T$. This shows the interest of treating testimonies, exceptions and presumptions in a uniform logical setting.

4 Related work

The relations with non-monotonic logics, counterfactual conditionals (in particular [23]), databases updates [41] are described in [37]. They are strong links with model-theoretic logics like preferential logics [40, 19, 20] (see [31]), initial models and their extensions used in algebraic specifications

[12, 3, 17, 26] (see [36]), circumscription [21, 14] (see [36]). Actually our first-order institution (A.2) yields a new variant of circumscription dealing adequately with equality [36].

Here, we will only describe the practical differences with other logics for the specification of law. [25, 32] reviews most of them with application to legal reasoning in mind.

Many of these approaches (default logic [28], autoepistemic logic [22]) are *proof-oriented*: we talk about the provedness of facts. This is indeed adequate to deal with onus of proof, but in presence of disjunctive information, it may yield unexpected results.

Example 4.1 The presumption of innocence can be formalised in e.g. default logic [28] as
$inn = \; : M\ \neg liable(x, d) \vdash \neg liable(x, d)$
if it is not proved that person x is liable for damage d, then we assume (s)he is not
Furthermore, *companies are liable for damages caused by their employees during the normal course of their work, and for which they are themselves liable*:
$comp = liable(x, d) \wedge duringworkfor(x, f, c) \wedge caused(f, d) \Rightarrow liable(c, d)$.

Let's consider the case where two movers, *Murdoch* and *Matthew*, have broken a precious vase while working in a customer's house. None of them accepts the responsibility of the fact. So we know that:
$fact = liable(Mu, Broken) \vee liable(Ma, Broken)$
(and the obvious facts about causation, etc.). Default logic concludes that none is liable (as expected, due to the presumption of innocence) but also that the company is not liable, since none of its employees is liable. This conclusion is indeed supported by the letter of the law, but is intuitively (and jurisprudentially) unexpected.

In our approach, we use classical sentences instead of defaults:
$innocence = \forall x, d. \neg liable(x, d)$
We formalise the problem as:
$innocence / \forall x, f, c, d. comp / fact$.

We obtain two incomparable models, each satisfying $comp$ and $fact$. They differ by which mover is liable. So we do not conclude the liability of any of them. But the liability of the company can be concluded, because in each model it is liable.

Syntactically based logic (including [28, 1, 5, 13, 24]) are often unable to derive universal norms.

Example 4.2 Let's try to represent in default logic [28] the (now abrogated) Belgian law saying that:
Parents are liable for the deed of their minor children; ... but parents aren't penally liable for the deed of the children above 18.
First note that using a default
$liable(x, d) \wedge minor(x) \wedge child(x, y) : M\ liable(y, d) \vdash liable(y, d)$
won't work. We have first to remind that liable is merely an abbreviation for "liable in civil and penal courts", and to distribute to obtain two defaults; then we have to give priority to the "18-rule". In general this will be done by adding a control predicate to the antecedent of the less reliable rules, that will appear in the consequent of the more reliable ones. Here, for simplicity, we will just translate the "18-rule" by a material implication.
$liableC(x, d) \wedge liableP(x, d) \wedge minor(x) \wedge child(x, y) : M\ liableP(y, d) \vdash liableP(y, d)$
$liableC(x, d) \wedge liableP(x, d) \wedge minor(x) \wedge child(x, y) : M\ liableC(y, d) \vdash liableC(y, d)$
$liableC(x, d) \wedge liableP(x, d) \wedge above18(x) \wedge child(x, y) \Rightarrow \neg liableP(y, d)$
This representation works as expected on cases. However, we would like also to derive universal rules like: *Parents are always liable in civil court for their minor children.* This is clearly impossible using default logic.

In our approach, the corresponding expression is:
$[(\forall x, y, d. liableC(x, d) \wedge liableP(x, d) \wedge minor(x) \wedge child(x, y) \Rightarrow liableP(y, d)$
$\| \forall x, y, d. liableC(x, d) \wedge liableP(x, d) \wedge minor(x) \wedge child(x, y) \Rightarrow liableC(y, d))$
$/ \forall x, y, d. liableC(x, d) \wedge liableP(x, d) \wedge above18(x) \wedge child(x, y) \Rightarrow \neg liableP(y, d)]$
Using the rule $2E$, we simplify this expression to
$\forall x, y, d. (liableC(x, d) \wedge liableP(x, d) \wedge child(x, y) \Rightarrow ((above18(x) \Rightarrow \neg liableP(y, d))$
$\wedge (minor(x) \Rightarrow liableC(y, d))$
$\wedge (minor(x) \wedge \neg above18(x) \Rightarrow \neg liableP(y, d)))$.

For practicality, the logic should allow several levels of exceptions. In the absence of this feature, it is still possible to obtain the desired results, but the interaction between various exceptions has to be controlled manually. In our experience, this control code can be larger than the formalisation of the norms themselves. In some approaches, any error in this control code causes an inconsistency (e.g. both P and $notP$ are considered true, in the notation of [39]).

A proposal found in [5, 32] is to associate an integer (the priority level) to each rule. This approach has an important danger built-in: it tends to order artificially norms. For instance, if two legal knowledge engineers build separate formalisations of legal bodies, that are later merged, exceptions from both formalisations will be interspersed without rational justification, depending on the particular numbers that each engineer chose. Our approach uses instead a *partial* order on exceptions: when no precedence is given between two rules, both conclusions are deemed acceptable.

Entire legal hierarchies are often taking precedence over other ones (see example 3.5). Our syntax expresses this easily, in contrast to e.g. the level numbering system.

In some (rare) cases the ordering of norms has to be dynamic.

Example 4.3 [9] The British law states that the marriage law applicable to an individual is the marriage law of the country where he resides. However, some other British Acts are still

applicable; assuming than some of them have lower precedence than marriage law (call them A), and others higher precedence (call them C), we would represent this hierarchy by e.g.: $reside(Italy) \Rightarrow [A/M_I/C] \wedge reside(France) \Rightarrow [A/M_F/C]$
where M_I designates the marriage law of Italy.

In summary, our logic combines the following features:

- partial ordering of laws (like [1, 14], unlike [5, 32]);
- algebraization of this partial order (new);
- model-theoretic definition (like [21, 19], unlike [1, 28, 24]), leading to a better treatment of disjunction and quantification.

We have argued that each of these features is desirable. Actually, they are independent: we can construct a number of logics, each with a different subset of these features.

5 Further research

In the first-order case, the set of theorems is not recursively enumerable, making automated deduction non-terminating. We are currently looking at an implementation based on well-founded orderings [2].

The base logic should be extended to handle the many modalities needed in legal reasoning. We have tried the temporal and deontic logic of [10], adding a closeness extending ours (A.2).

The applicability of our logic to large legal systems has not yet been tried out. Structuring concepts from the school of algebraic specifications [12] can be useful here: our logic has been designed to accommodate them easily.

The syntax of our logic is not fully satisfactory. We would prefer to have a single syntactic category, allowing to mix freely $||$, / and $\wedge, \neg$.

Our logic does not deal with the *source* of precedence among laws. This not a problem for two classical principles of precedence, *lex posterior* (newest laws take precedence) and *lex superior* (laws issued by a higher authority take precedence). The third one, *lex specialis* (laws more specific (to the case at hand?) take precedence), has to be coded as in example 4.3. A system where the content of laws is examined to determine automatically their specificity relative to the current case, like [7, 24] is superior here. Note that a precise definition of *specificity* is currently a debated issue.

The interaction among principles of precedence is also debated. We are currently exploring a multi-level version of our logic, where the precedence is itself the result of a non-monotonic, paraconsistent reasoning (useful to express e.g. the principles of [4]; this has also applications in the theory of inheritance hierarchies [8]).

6 Conclusion

From a legal point of view, our logic proposes a practical and theoretically well-grounded approach to the reasoning from hierarchies of laws, testimonies, presumptions, and some forms of common sense. The need for such a logic has been identified by theorists of law [27] for long; several such logics are now developing, based on various non-monotonic foundations.

From a logical point of view, our logic proposed has has a simple model-theoretic foundation. Its proof theory, presented in 2.2, is sound and complete. Its algebraic character allows short proofs.

A Definitions

A.1 Default institutions

Our definition is parameterized by a *default institution*, which is given by:

- a category $Sign$ of signatures;
- a functor $\mathcal{L} : Sign \rightarrow Sets$, giving languages $\mathcal{L}(\Sigma)$ linked by translations Tr_i;
- a contravariant functor $Int : Sign \rightarrow Cat^{op}$, giving interpretations $Int(\Sigma)$ and their morphisms $Mor(\Sigma)$, linked by forgetful functors noted $|_i$.
- a family of satisfaction relations $\models_\Sigma$ between the interpretations of Σ and its formulae.
- a functor $Comp : Sign \rightarrow Cat^{op}$, such that, for each $Sigma$:
  - the objects of $Comp$ are Mor, the morphisms of Int;
  - $Comp$ is a preorder, that is, there is at most one morphism of $Comp$ between two objects of $Comp$;
  - the identities of Mor are initial (minima) in $Comp$;
  - the morphisms of Mor that are minima in $Comp$ are called *agreements*; they must form a subcategory Int_0;
  - Int_0 is weakly abstract: $\exists\, h : M \rightarrow N \in Int_0 \Rightarrow \forall\, \phi \in \mathcal{L}(\Sigma), M \models \phi \Rightarrow N \models \phi$;
  - 0-symmetry: each agreement $h : M \rightarrow N$ has a reverse agreement $h^R : N \rightarrow M$ such that $(h^R)^R = h$;
  - 0-equivalence: for any morphism $h : B \rightarrow C$ and agreements $a : A \rightarrow B, c : C \rightarrow D$, $a;\ h \equiv h \equiv h;\ c$.

Example A.1 Our propositional default institution.
A *signature* is here a set of *propositional symbols*. A morphism of signatures is a function mapping the propositional symbols of the source signature to some propositional symbols of the target signature. Given a signature Σ, the language $\mathcal{L}(\Sigma)$ is given by the usual syntax:

$$f ::= f_1 \wedge f_2 \mid \neg f_1 \mid p$$

, where f, f_1, f_2 are propositional formulae, and $p \in \Sigma$ is a propositional symbol. The translation Tr_i corresponding to a morphism of signatures i simply replaces all occurrences of a propositional symbol by its image under i. $Int(\Sigma)$ is the usual class of propositional interpretations, i.e. functions from Σ to

$\{T, F\}$. The morphisms of interpretations are simply pairs of interpretations. The forgetful functor $|_i$ constructs the inverse image of an interpretation: it forgets the propositional symbols that are not in the range of i, takes the value of $i(p)$ for p (thus renaming and possibly duplicating propositional symbols). The satisfaction relation is as usual: we define $\bar{M}(\phi)$ by recursion on formulae, and pose $M \models \phi$ iff $\bar{M}(\phi) = T$. Up to here, all is standard.

The new part is the functor $Comp$ expressing closeness between interpretations. Intuitively, if there is a morphism of $Comp$ from (M_1, N_1) to (M_2, N_2), it means that M_1 is less different from N_1 than M_2 is from N_2. We define a morphism of $Comp$ as a pair $((M_1, N_1) \leq (M_2, N_2))$ such that
$\{p \in \Sigma \mid M_1 \models p \wedge N_1 \not\models p\} \subseteq \{p \in \Sigma \mid M_2 \models p \wedge N_2 \not\models p\}$
$\{p \in \Sigma \mid M_1 \not\models p \wedge N_1 \models p\} \subseteq \{p \in \Sigma \mid M_2 \not\models p \wedge N_2 \models p\}$.

It is easy to check that this defines indeed a default institution.

Example A.2 Our first-order default institution
A *signature* Σ is a triple:

- S, a set of *sorts*;
- O, a set of *operators* with functions $\alpha : O \to S^*$ giving the sort of their arguments and $\sigma : O \to S$, the sort of their result;
- P, a set of *predicates* with $\alpha : P \to S^*$ as above.

A morphism of signatures $\Sigma_1 \to \Sigma_2$ is a triple of functions (i_S, i_O, i_P), respecting sorting, i.e. $\sigma_2(i_O(f)) = i_S(\sigma_1(f))$ for $f \in O_1$, and $\alpha_2(i_O(f)) = i_S(\alpha_1(f))$ for $f \in O_1$ or P_1. We also assume to have a set of *variables* X with $\sigma : X \to S$ giving their sort. A *term* of sort s is either:

- a variable of X of sort s, or
- an operator of result sort s applied to terms of its argument sorts.

$\mathcal{L}(\Sigma)$ contains *formulae* ϕ, that may be:

- *conjunctions*: $\phi_1 \wedge \phi_2$
- *negations*: $\neg\phi_1$
- *universal quantifications*: $\forall\, x : s, \phi_1$
- literals: $p(t_1, \ldots, t_n)$

where

- ϕ_1, ϕ_2 are formulae;
- $p \in P_{s_1, \ldots, s_n}$;
- $t_1, \ldots, t_n$ are terms of sorts $s_1, \ldots, s_n$.

An *algebra* A gives

- for each sort s, a set s_A (called the *carrier* of the sort);
- for each operator f, a function f_A from the carriers of the argument sorts to the carrier of the result sort;
- for each predicate p, a relation p_A between the carriers of the argument sorts.

Here, interpretations have an *internal signature* ΣA containing Σ; an interpretation is thus a pair $(\Sigma A, A)$, where A is a surjective ΣA-algebra (see below). A *valuation* V is a function that for each variable yields its value, i.e. a member of the carrier of its sort. We say that $V' \approx_v V$, if $\forall\, x \in X_s \setminus \{v\}$, $V'(x) = V(x)$.

The *evaluation* V_A is the function that extends V by assigning to each term of $T_{\Sigma A}(X)$ a value so that $V_A(f(t_1, \ldots, t_n)) = f_A(V_A(t_1), \ldots, V_A(t_n))$. There is a single *ground evaluation*, noted e_A, that gives a value to each (internal) ground term. If e_A is surjective, the algebra is called *surjective*.

A morphism of interpretations can only exist between interpretations having the same internal signature. It is called a *correspondence* between $(\Sigma A, A$ and $(\Sigma A, B)$, defined as a family of relations $\sim_s$ between the carriers of A and B, with the following properties :

1. *compatible with internal operators*: $\forall f \in OA : s_1, \ldots, s_n \to s \in O;\ a_1 \sim_{s_1} b_1, \ldots, a_n \sim_{s_n} b_n \Rightarrow f_A(a_1, \ldots, a_n) \sim_s f_B(b_1, \ldots, b_n)$,
2. *total*: $\forall\, a \in s_A, \exists\, b \in s_B, a \sim_s b$,
3. *surjective*: $\forall\, b \in s_B, \exists\, a \in s_A, a \sim_s b$

An algebra A *satisfies* a formula ϕ for a valuation V, noted $A \models_V \phi$, if:

- $A \models_V \phi_1 \wedge \phi_2$ iff $A \models_V \phi_1$ and $A \models_V \phi_2$
- $A \models_V \neg\phi$ iff $A \models_V \phi$ is false
- $A \models_V \forall\, v, \phi$ iff $A \models_{V'} \phi$ for all $V' \approx_v V$

An interpretation $(\Sigma A, A)$ *satisfies* a formula ϕ iff A satisfies ϕ for all valuations.

Although the definitions given here are not exactly those of classical first-order logic, it can be shown that they are equivalent for classical purposes.

The closeness that we will use is a direct extension of the propositional one. Here, we have to know which elements of the carriers correspond to each other; therefore a morphism of $Comp$ is a *double correspondence* from a correspondence $\sim : A \to B$ to another $\sim' : A' \to B'$ with the same internal signature is a family of relations $\approx_s$ indexed by sorts between pairs of corresponding elements, with the same properties as correspondences:

1. *compatible with internal operators*: $\forall f : s_1, \ldots, s_n \to s \in O;\ (a_1, b_1) \approx_{s_1} (a_1', b_1'), \ldots, (a_n, b_n) \approx_{s_n} (a_n', b_n') \Rightarrow (f_A(a_1, \ldots, a_n), f_B(b_1, \ldots, b_n)) \approx_s (f_A(a_1', \ldots, a_n'), f_B(b_1', \ldots, b_n'))$,
2. *total*: $\forall\, s \in S, \forall (a, b) \in \sim;\ \exists (a', b') \in \sim';\ (a, b) \approx_s (a', b')$,
3. *surjective*: $\forall\, s \in S, \forall (a', b') \in \sim';\ \exists (a, b) \in \sim;\ (a, b) \approx_s (a', b')$,

Furthermore a morphism of $Comp$ goes from close interpretations to farther ones, so that it should obey: for all (a', b') such that $(a, b) \approx (a', b')$:

- $p_A(a) \wedge \neg p_B(b) \Rightarrow p_{A'}(a') \wedge \neg p_{B'}(b')$
- $\neg p_A(a) \wedge p_B(b) \Rightarrow \neg p_{A'}(a') \wedge p_{B'}(b')$

Lastly, to ensure that $Comp(\Sigma)$ is a preorder, a morphism of $Comp$ should be minimum (for set inclusion) among double correspondences.

A.2 Reliability posets

A.2.1 Syntax

We define a *poset* for a given signature Σ as:

- a finite set W of *witnesses*.
- a partial preorder $<_W$ on W (Intuitively, $v < w$ means that v is more reliable than w).
- a function Φ from W to formulae, called the *testimonies*.

A *family of morphisms* $\vec{h}$ for a poset G is a set of morphisms indexed by the witnesses of G, originating from a single model: $Mor(G) = \{\vec{h} \mid \exists\, e, d_w;\ h_w : e \rightarrow d_w, d_w \models G_w\}$. The ordering among families is *lexicographic*: $\vec{h} \leq_G \vec{h}' \Leftrightarrow \forall\, w \in W, h_w \leq h'_w \vee \exists\, t <_W w,\ \ h_t < h'_t$.

$Min(G)$ are the minimal morphisms of $Mor(G)$ for this ordering, i.e. $Min(G) = \{\vec{h} \mid \vec{h} \in Mor(G)\ \not\exists\, \vec{h}' \in Mor(G), \vec{h}' <_G \vec{h}\}$. An interpretation e is a model of G (noted $e \in [G]$) if it is the domain of a minimal family of morphisms: $e \in [G]$ iff $\exists\, \vec{h} \in Min(G);\ \forall\, w \in W;\ dom(h_w) = e$.

A.2.2 Operators

Operators allow to create new graphs by combining existing ones. A n-ary *operator* is a poset, but it can be labelled either by formulae or by names. These names represent the arguments of the operator. When there are no formulae, it is *pure*. When there are no names, it is a poset. A *substitution* or an *environment* is a function from names to operators. The application of a substitution σ to an operator G, noted $G\sigma$, where $\sigma = [n_1 := G_1, \ldots, n_m := G_m]$, gives a new operator G' defined by:

- W' is $\{(w \in W, w' \in \sigma(\Phi(w)))\}$
-
  - the order between nodes originating from the same witness is the order of the argument: $(w, w_1) \leq_{W'} (w, w_2)$ iff $w_1 \leq_{\sigma\Phi(w)} w_2$.
  - the order between nodes from different witnesses is the order of the operator G: $(w, w_1) \leq_{W'} (w', w_2)$ iff $w \leq_W w'$ where $w \neq w'$.
-
  - $\Phi'((w, w_1)) = \Phi(w_1)$ if $w \notin dom\sigma$;
  - $\Phi'((w, w_1)) = \Phi(w)$ if $\Phi(w)$ is not a name.
- G/G' (G but G'), deems G less reliable than G'. This operator can be used to integrate new information taking precedence.
- $G||G'$ (G; on the other hand G'?) gives no precedence to G nor G'. This operator can be used to integrate new information in a sceptical way.
- the *single node* operators takes a formula and makes a poset of it. This operator will not be noted.
- the *empty poset* operator, $\varnothing$.

Two posets G, G' are deemed *equivalent* ($G \equiv G'$), if there is a relation $\sim$ between $Mor(G)$ and $Mor(G')$, such that $h \sim h' \Rightarrow dom(h) = dom(h')$, and whenever $h_1 \sim h'_1, h_2 \sim h'_2$, $h_1 \leq_G h_2$ iff $h'_1 \leq_{G'} h'_2$. Two operators are *equivalent* if any substitution that yields a poset for one of them, yields an equivalent poset for the other.

B Rule for our first-order institution

To present this second-order rule, we need abbreviations:

- As a preliminary step, we need to replace the many-sorted formulae by the single-sorted ones allowed by second-order logic. This is a simple and standard technique called *relativization* [34], that replaces sorts by predicates.
- For each witness $w \in W$, we introduce a copy of the set of predicates P_w, that will give the value of each predicate in the closest model of the testimony.
- For each sort $s \in S$, we introduce a predicate s' containing elements that are not generated by the external signature.
- $(P', P'_w) \leq (P, P_w)$ abbreviates $\bigwedge_{p \in P}(\forall\, \vec{x}, (p'(\vec{x}) \wedge \neg p'_w(\vec{x})) \Rightarrow (p(\vec{x}) \wedge \neg p_w(\vec{x})))$ $\wedge\, (\forall\, \vec{x}, (\neg p'(\vec{x}) \wedge p'_w(\vec{x})) \Rightarrow (\neg p(\vec{x}) \wedge p_w(\vec{x})))$, expressing the order on our "distance" between models.
- we use the usual abbreviation $(P', P'_w) < (P, P_w)$ for $(P', P'_w) \leq (P, P_w) \wedge \neg(P, P_w) \leq (P', P'_w)$.
- $(P', \vec{P}') \leq_G (P, \vec{P})$ abbreviates the lexicographic ordering: $\bigwedge_{w \in W}(P', P'_w) \leq (P, P_w) \vee \bigvee_{t < w}(P', P'_t) < (P, P_t)$
- $TERM(S')$ is the conjunction of:
  - the algebra is well-typed: $\bigwedge_{f : s_1 \ldots s_n \rightarrow s \in O} \forall\, \vec{x};\ [\bigwedge_{i \leq n} s_i(x_i)] \Rightarrow s(f(\vec{x}))$
  - variables are terms: $\bigwedge_{s \in S} \forall\, x . s'(x) \Rightarrow s(x)$
  - functions yield different results: $\bigwedge_{f, g : \in O, f \neq g} \forall\, \vec{x}, \vec{y} . \vec{s}(\vec{x}) \wedge \vec{s}'(\vec{y}) \Rightarrow \neg(f(\vec{x}) = g(\vec{y}))$
  - each function is injective: $\bigwedge_{f \in O} \forall\, \vec{x}, \vec{y} . \vec{s}(\vec{x}) \wedge \vec{s}(\vec{y}) \wedge f(\vec{x}) = f(\vec{y}) \Rightarrow \bigwedge_{i \leq n} x_i = y_i$
  - functions do not yield variables: $\bigwedge_{f : \vec{s} \rightarrow r \in O} \forall\, \vec{x}, \vec{s}(\vec{x}) \Rightarrow \neg r'(f(\vec{x}))$
  - the algebra is surjective: $\forall (I_s)_{s \in S} . [\bigwedge_{f : \vec{s} \rightarrow r \in O} \forall\, \vec{x}, \vec{s}(\vec{x}) \wedge I_{\vec{s}}(\vec{x}) \Rightarrow I_r(f(\vec{x})) \wedge \bigwedge_{s \in S} \forall\, x . s'(x) \Rightarrow I_s(x)] \Rightarrow [\bigwedge_{s \in S} \forall\, x . s(x) \Rightarrow I_s(x)]$

The formula $R(G)$ is defined as:

$$\exists\, S' . TERM(S') \wedge \bigwedge_{w \in W} \Phi(w)[P_w/P] \wedge / \exists\, P', \vec{P}' . \bigwedge_{w \in W} \Phi(w)[P'_w/P] \wedge (P', \vec{P}') <_G (P, \vec{P})$$

It characterizes exactly the models of $[G]$, so that:

$2E$ $\quad [G_1] \Leftrightarrow [G_2]$ iff $R(G_1) \Leftrightarrow R(G_2)$.

References

[1] C. Alchourron and D. Makinson. Hierachies of regulations and their logic. In R. Hilpinen, editor, *New Studies in Deontic Logic*, pages 123–148. Reidel, 1981.

[2] L. Bachmair and H. Ganzinger. Perfect model semantics for logic programs with equality. In *Proc. 8th Intl. Conf. on Logic Programming*. MIT Press, 1991.

[3] M. Bidoit and G. Bernot. Proving correctness of algebraically specified software: Modularity and observability issues. In M. Nivat, C. Rattray, T. Rus, and G. Scollo, editors, *AMAST'91*, Worskshops in Computing, pages 139–161. Springer-Verlag, 1992.

[4] N. Bobbio. Des critères pour résoudre des antinomies. In C. Perelman, editor, *Les Antinomies en Droit*, Travaux du Centre National de Recherches de Logique. Bruylant, 1965.

[5] G. Brewka. *Nonmonotonic reasoning*. Cambridge Univ. Press, 1991.

[6] M. Dalal. Investigations into a theory of knowledge base revision: preliminary report. In *Proc. of the 7th Nat. Conf. on Art. Int. (AAAI-88)*, pages 475–479, 1988.

[7] J. Delgrande. An approach to default reasoning based on a first-order conditional logic. *Artificial Intelligence*, 36:63–90, 1988.

[8] R. Durcournau and M. Habib. Masking and conflicts. In M. Lenzerini, D. Nardi, and M. Simi, editors, *Inheritance Hierarchies in Knowledge Representation and Programming Languages*, pages 223–244. Wiley, 1991.

[9] R. V. Elst. Antinomies en droit international privé. In C. Perelman, editor, *Les Antinomies en Droit*, Travaux du Centre National de Recherches de Logique. Bruylant, 1965.

[10] J. Fiadeiro and T. Maibaum. Temporal reasoning over deontic specifications. *J. Logic Computat.*, 1(3):357–395, 1991.

[11] P. Gardenfors. *Knowledge in Flux : Modeling the Dynamics of Epistemic States*. MIT press, 1988.

[12] J. Goguen and R. Burstall. Institutions: Abstract model theory for specification and programming. *J. ACM*, 39(1):95–146, Jan. 1992.

[13] T. Gordon. An abductive theory of legal issues. *Intl. Journal of Man-Machine Studies*, 35:95–118, 1991.

[14] B. Grosof. Generalizing prioritization. In *Proc. of the 2nd Intl. Conf. on Knowledge Representation and Reasonig (KR'91)*, pages 289–300, 1991.

[15] S. Hansson. New operators for theory change. *Theoria*, 55(2), 1989.

[16] A. Jones and M. Sergot. Deontic logic in the representation of law: towards a methodology. *Artificial Intelligence and Law*, 1(1), 1992.

[17] S. Kaplan. Positive/negative conditional rewriting. In Jouannaud and Kaplan, editors, *Conditional Term Rewriting*, volume 308 of *Lecture Notes in Computer Science*. Springer, 1988.

[18] H. Kelsen. The pure theory of law. *Law Quarterly Review*, 51:517–535, 1935.

[19] S. Kraus, D. Lehmann, and M. Magidor. Nonmonotonic reasoning, preferential entailment and cumulative logics. *Artificial Intelligence*, 44:167–207, 1990.

[20] D. Makinson. General patterns in non-monotonic reasoning. In D. Gabbay, C. Hogger, and J. Robinson, editors, *Handbook of Logic in Artificial Intelligence*. Oxford Univ. Press, 1993.

[21] J. McCarthy. Applications of circumscription to formalizing common-sense knowledge. *Artificial Intelligence*, 28:89–116, 1986.

[22] R. C. Moore. Semantical considerations on nonmonotonic logic. *Artificial Intelligence*, 25:75–94, 1985.

[23] J. Pollock. A refined theory of counterfactuals. *Journal of Philosophical Logic*, 10:239–266, 1981.

[24] D. Poole. A logical framework for default reasoning. *Artificial Intelligence*, 36:27–47, 1988.

[25] H. Prakken. *Logical Tools for Modelling Legal Argument*. PhD thesis, Vrije Univ. Amsterdam, 1993.

[26] P. Rathmann and M. Winslett. Circumscribing equality. In *Proc. of the 8th Nat. Conf. on Art. Int. (AAAI-89)*, pages 468–473, 1989.

[27] J. Ray. *Essai sur la Structure Logique du Code Civil Francais*. F. Alcan, 1926.

[28] R. Reiter. A logic for default reasoning. *Artificial Intelligence*, 13:81–132, 1980.

[29] T. Routen. Hierarchically organised formalisations. In *The Second International Conference On Artificial Intelligence and Law*. Acm Press, 1989.

[30] M. Ryan. Defaults and revision in structured theories. In *Proceedings, Sixth Annual IEEE Symposium on Logic in Computer Science*, pages 362–373. IEEE Computer Society Press, 1990.

[31] M. Ryan and P.-Y. Schobbens. Laws of generalised prioritization. Technical report, FUNDP, Namur, 1993.

[32] G. Sartor. The structure of norm conditions and nonmonotonic reasoning in law. In *The Third International Conference On Artificial Intelligence and Law*. Acm Press, 1991.

[33] G. Sartor. *Artificial Intelligence and Law: Legal Philosophy and Legal Theory*. Tano, 1993.

[34] A. Schmidt. Die zulässigkeit der behandlung mehrsortiger theorien mittels der üblichen einsortigen prädikatenlogik. *Math. Ann.*, 123:187–200, 1951.

[35] P.-Y. Schobbens. Inheritance with exceptions for software specification: On the meaning of but. Technical report, RR-89-08, Université Catholique de Louvain, Unité d'Informatique, Feb. 1989.

[36] P.-Y. Schobbens. Equality circumscription revisited: surjective circumscription. Technical report, CRIN, Nancy, 1992.

[37] P.-Y. Schobbens. *Exceptions in Algebraic Specifications*. PhD thesis, Univ. Cath. de Louvain, 1992.

[38] P.-Y. Schobbens. On the meaning of "but". *Science of Computer Programming*, March/April 1993.

[39] M. Sergot, F. Sadri, R. Kowalski, F. Kriwaczeck, P. Hammond, and H. Cory. The british nationality act as a logic program. *CACM*, 29:370–386, 1986.

[40] Y. Shoham. *Reasoning about Change*. Electrical Engineering and Computer Science Serie. The MIT Press, 1988.

[41] M. Winslett. *Updating Logical Databases*. Cambridge University Press, 1990.