AUTOMATED DEDUCTION FOR DEONTIC LOGICS: LAYING THE FOUNDATIONS OF LEGAL EXPERT SYSTEMS

Simon Thornton [1]

[1] Dept. of Computer Science, The Australian National University

ABSTRACT: Deontic logic can be used to formalise legal rules. The automation of reasoning in deontic logic can therefore be used as the basis of a legal expert system. In this paper we investigate the automation of deontic logic using tableau calculi for modal logics. The correspondance between the deontic and modal logics is outlined and the tableau for the modal logics used in the prover are given. We also use some of the well known paradoxes of deontic logic to demonstrate the operation of an automated theorem prover for deontic logic.

INTRODUCTION

The law can be thought of as a collection of rules, or norms, backed by the threat of sanction. Citizens seeking to obey the law need to use these rules to avoid becoming liable to punishment. This can be seen as reasoning 'with' the law at an object level. At the same time, authorities enacting laws need to reason 'about' rules at a meta level to ensure that laws are consistent. A legal expert system, a program that contains a legal knowledge base and some algorithms for inferring new facts can help with both these types of reasoning. Such a system can decide if a citizen's acts have broken the law or whether a new law is consistent with those already in existence.

In any expert system the method chosen for both knowledge representation and for inferring new facts is important. The representation must allow the knowledge to be rigorously captured and expressed and the inference rules must ensure that only valid conclusions are drawn. Formal reasoning with and about legal rules has often been done with deontic logic, a branch of philosophical logic involving the concepts of obligation, permission and prohibition. This makes deontic logic a suitable choice for knowledge representation in a legal expert system.

To allow a computer to perform the kind of automated deduction needed in an expert system we need an algorithm that the computer can apply to draw conclusions. This algorithm can be purely syntactic so that the computer does not have to understand the meaning of what it is manipulating. Tableau calculi, which provide rules for breaking a formula down in order to find truth valuations for its individual parts, are one method of allowing the automation of deduction in formal logics like deontic logic. By implementing a computerised tableau calculus for deontic logic we can create the foundations of a legal expert system.

STANDARD DEONTIC LOGIC

[Roy98], [FH71] and [Duc95] all refer to a system of 'Standard Deontic Logic'. This system is derived from one described by von Wright in [vW51]. The logic itself contains the propositional calculus along with three inter-definable deontic operators; O, P and F, representing obligation, permission and prohibition respectively. In describing the relationship between these operators we choose O to be the basic symbol, where OA reads "it is obligatory that A". We can then define P and F as follows, where $\neg A$ can be read as "it is not the case that A" or "A was not done":

(1) $$PA \equiv \neg O \neg A$$

(2) $$FA \equiv O \neg A$$

That is, A is permitted if $\neg A$ is not obligatory and A is forbidden if $\neg A$ is obligatory.

The semantics of deontic logic are described in [FH71] but are usually taken to be the standard Kripke-style semantics for modal logics. Kripke semantics can be represented as a triple $\mathcal{M} = \langle \mathrm{W}, \mathrm{R}, \vartheta \rangle$

where W is the set of all possible worlds, R defines a binary relationship over W ($\mathrm{R} \subseteq \mathrm{W} \times \mathrm{W}$) and ϑ is an assignment of a truth value to each propositional letter on each world in W. On each world $\omega \in \mathrm{W}$ the propositional calculus operates as normal. The deontic operator *O* is a 'transitional' operator which moves along a relationship defined by R to another world. In this world, everything that is obligatory is now true. Alternative worlds are therefore considered to be 'deontically perfect' worlds, where all obligations have been fulfilled. The model-structure semantics just outlined provide the basis for determining the consistency of deontic formula using a tableau calculus.

AUTOMATING STANDARD DEONTIC LOGIC

Modal Tableau Calculi

A number of tableau calculi already exist for standard modal logics, see [Gor99] for examples. We therefore opted to use a tableau for modal logics as the basis of our deontic logic prover. Modal logics have two non-propositional operators, one of necessity usually represented as $\Box$ and one of possibility represented as $\Diamond$. The relationship between these operators is shown in formula 3.

(3) $\Box\varphi \equiv \neg\Diamond\neg\varphi$, where φ is a formula of modal logic.

From this equation the similarities between the *O* and *P* operators and the $\Box$ and $\Diamond$ operators should be clear (see formula 1 above). These similarities allow the *O* operator to be defined as $O \equiv \Box$ and the *P* operator as $P \equiv \Diamond$. This reduction allowed us to use established modal tableaux as the basis for an automated theorem prover for deontic logic. All that is required is a simple rewriting rule that transforms deontic formulae into modal formulae using the above equivalence relationships. The simple reduction also means that the following descriptions of the tableau calculi that were automated all use the modal operators rather than the deontic ones. This reflects the fact that modal logics have interpretations other than deontic ones and that the deontic prover is actually very general.

Once the reduction of the deontic operators to their modal counterparts has been defined it can be seen that the axioms of standard deontic logic (SDL) correspond to the axioms of the modal logic $\mathbf{KD}$:

Axioms of SDL		**Axioms of KD**	
	All propositional tautologies		All propositional tautologies
	$O(A \rightarrow B) \rightarrow (OA \rightarrow OB)$	K:	$\Box(\varphi \rightarrow \psi) \rightarrow (\Box\varphi \rightarrow \Box\psi)$
	$OA \rightarrow PA$	D:	$\Box\varphi \rightarrow \Diamond\varphi$
O rule:	If *A* is a theorem, then so is OA	$\Box$ rule:	If φ is a theorem, then so is $\Box\varphi$

A Tableau for Standard Deontic Logic

A tableau calculus is a set of syntactic rules that can be applied to a formula based on the logical operators it contains.[1] Each step of a tableau proof involves matching a set of formula to the numerator of a rule in order to produce a set of formula like the ones in the rule's denominator. Usually, each modified formula is less complicated than the formula the rule was applied to. The rules may contain branches where two different sets of formula are produced. In this way a given formula produces a tableau tree with the original formula at the root and the sets of formulae produced by rule applications at nodes progressing toward the leaves. If no truth valuation can be found for a given formula, that is sub-formula of the shape $\neg A$ and *A* appear on the end of every branch, then the negation of that formula must always be true. Theorems of a logic can therefore be proven by a tableau calculus using a method of proof by contradiction.

The correspondence outlined above means that a tableau calculus for $\mathbf{KD}$ can be used to prove the theorems of SDL. Such an implementation would also allow sets of legal rules, formalised in deontic

[1] For a good introduction to tableau see [DGHP99].

formula, to be checked for consistency. An example of this kind of application is given below. The tableau rules for $\mathbf{KD}$ are shown in figure 1.

$$(ax)\ \frac{\Gamma, \varphi, \neg\varphi}{} \qquad (\wedge)\ \frac{\Gamma, \varphi \wedge \psi}{\Gamma, \varphi, \psi} \qquad (\vee)\ \frac{\Gamma, (\varphi \vee \psi)}{\Gamma, \varphi \mid \Gamma, \psi}$$

$$(K\theta)\ \frac{\Diamond\varphi, \Box\Gamma, \Diamond\Delta, \Sigma}{\varphi, \Gamma, \Delta \,||\, \Box\Gamma, \Diamond\Delta, \Diamond\varphi} \qquad (D)\ \frac{\Box\varphi, \Box\Gamma, \Sigma}{\Diamond\varphi, \Box\varphi, \Box\Gamma, \Sigma}$$

Σ contains no $\Box$-formulae and no $\Diamond$-formulae

Figure 1: List of tableau rules for modal logic $\mathbf{KD}$. φ and ψ represent individual formula while Γ, Δ and Σ represent sets of formulae.

The rules assume that the given formula is in negated normal form (innermost negations and only $\wedge$ and $\vee$ propositional connectives). Both the $(\vee)$ and (K) rules are examples of branching rules. The difference between the branching in the $(\vee)$ rule (denoted by the bar in the denominator) and the branching in the (K) rule (denoted by the double bar) is that for the $(\vee)$ rule both branches must close, that is end in a contradiction detected by the (ax) rule, for the tableau to be a closed tableau (one with no possible truth valuations), while for the (K) rule only one branch needs to close for the tableau to be closed.

The $\mathbf{KD}$ tableau was implemented in the Tableau Workbench, a program that provides an easy to use syntax to allow the implementation of many different tableau for different logics. A more detailed description of the TWB can be found in [AG03]. The TWB automates a proof in $\mathbf{KD}$ by using pattern matching to match the numerator of one of the tableau rules to the formulae at the current node. When a rule is applicable the TWB changes the formulae to match the denominator of the rule. If no rule is applicable then the branch is open. The TWB decides which rule to pattern match first based on a strategy defined by the user. In the case of the KD calculus the strategy looks like this:

$$[[[(\wedge);(\vee)]^*;(K);(D)]^+]^*$$

The * operator causes rules to be repeatedly applied while the + operator causes them to be applied only once. Therefore, this strategy attempts to apply the propositional rules until no more can be applied. It then tries to apply the K or the D rule once, in that order, before repeating the entire process. Using the KD tableau rules, the above strategy and a rewriting function to transform deontic operators into modal ones, we are able to prove deontic theorems by negating them and giving them to the TWB to prove by contradiction. A list of some of these theorems can be found in [Roy98] on pages 37 and 38.

THE PARADOXES OF SDL

[Duc95], [FH71] and [Roy98] all provide details of some of the paradoxes of deontic logic. [Duc95] divides the various paradoxes into three categories: disjunctive norms, conjunctive norms and conditional norms. A brief examination of these categories provides direction for the development of the theorem prover.

Paradoxes of Disjunctive and Conjunctive Norms

The paradoxes of disjunctive and conjunctive norms are often demonstrated using formulae 4 and 5 respectively. Both are theorems of SDL and can be proven in our prover.

(4) $$O p \rightarrow O(p \vee q)$$

(5) $$F p \rightarrow F(p \wedge q)$$

If p is taken to mean "you post the letter" and q is taken to mean "you burn the letter" in formula 4 then the formula seems to say that if you ought to post the letter you ought to burn it. Such a result seems very unintuitive. Similarly, formula 5 seems to say that if it is forbidden to insult someone, then it is forbidden to insult someone and apologise. As [Duc95] explains, these paradoxes result from attempting to interpret the formula as expressing concepts of ought-to-do. When they are read as ought-to-be expressions their paradoxical nature disappears. For example, factually it makes sense that if it is the case that the letter has been posted, it is also the case that the letter was posted or burnt (it was posted!). This means that formulae like 4 and 5 are not really paradoxes at all.

The Paradox of Conditional Obligation

In contrast to the paradoxes of disjunctive and conjunctive norms, the paradox of conditional obligation cannot be explained by a different semantic reading of the formula involved.

The Paradox of Conditional Obligation

It ought to be that John does not impregnate Suzy.	(6)	$O\neg$p
Not impregnating Suzy commits John to not marrying her.	(7)	$O(\neg \text{p} \rightarrow \neg \text{q})$
Impregnating Suzy commits John to marrying her.	(8)	$\text{p} \rightarrow O\text{q}$
John impregnates Suzy.	(9)	p

These 4 statements do not seem to be intuitively inconsistent. However, their formalisation in formula 6 to 9 are inconsistent in SDL. This means the conjunction of formulae 6 to 9 has a closed tableau when given to the theorem prover for SDL. This is an example of the kind of consistency checks that can be performed using an automated theorem prover for deontic logic. While the example above is reasonably simple the process of formalising rules in deontic logic, taking the conjunction of the logical formula and checking the resulting formula for consistency could be easily applied to a piece of legislation.

[Duc95] proposes a solution to the paradox of conditional obligation by incorporating a new operator, *L*, into the logic. This operator is defined to be equivalent to the $\Box$ operator in the modal logic $\mathbf{S5}$ and has the temporal meaning "for all time points". The axioms for $\mathbf{S5}$ and the new formalisations of each of the original statements are shown in table 1 below. The logic created by the addition of the *L* operator to SDL is called Temporal Deontic Logic (TDL) in [Duc95].

Table 1: The Solution to the Paradox of Conditional Norms

The Axioms of S5		**The Solution**	
$L(A \rightarrow B) \rightarrow (LA \rightarrow LB)$		(10)	$LO\neg$p
$LA \rightarrow A$		(11)	$L(\neg \text{p} \rightarrow O\neg \text{p})$
$LA \rightarrow LLA$		(12)	$L(\text{p} \rightarrow O\text{q})$
$\neg LA \rightarrow L\neg LA$		(13)	p
L-rule: if *A*is a theorem, so is *LA*			

Formulae 10 to 13 can be shown to be consistent in TDL. This result is not entirely satisfactory as a careful examination of formula 7 and formula 11 reveals that the SDL component of the statement has been rewritten in order to remove the inconsistency. [Duc95] claims this is allowed in TDL because the *L* operator ensures that formula 10 to 13 remain independent. The same is not true of the original SDL formulae as writing formula 7 as $\neg \text{p} \rightarrow O\neg \text{p}$ means that formula 6 to 9 are no longer independent. This is because $\neg \text{p} \rightarrow O\neg \text{p}$ can be deduced from formula 9 using formula 14, shown below, which is a theorem

of SDL.

(14) $\neg p \rightarrow (p \rightarrow O q)$

AUTOMATING TEMPORAL DEONTIC LOGIC

Implementing $\mathbf{S5}$ in the TWB was more complicated than implementing $\mathbf{KD}$. There are two reasons for this. The first is that a tableau calculus for $\mathbf{S5}$ requires the use of a cut rule for both the $\Diamond$ and $\vee$ operators. The second is that it is possible for the tableau for an $\mathbf{S5}$ formula to loop. To ensure termination of the tableau a history mechanism is required so that if a node that is identical to one seen before appears the same series of rules is not applied so as to end up back at the same spot. The TWB supports the use of such a history mechanism. The use of the loop check is reflected by the condition on the numerator in the $(S5)$ rules shown in figure 2 below.

$$(S5\theta)\ \frac{\Diamond\varphi, \Box\Gamma, \Diamond\Delta, \Sigma \quad \Diamond\varphi \notin H}{\varphi, \Diamond\varphi, \Box\Gamma, \Diamond\Delta \mid\mid\ \Box\Gamma, \Diamond\Delta, \Diamond\varphi} \qquad (T)\ \frac{\Box\varphi, \Gamma}{\varphi, \Box\varphi, \Gamma} \qquad (\wedge)\ \frac{\Gamma, \varphi \wedge \psi}{\Gamma, \varphi, \psi}$$

$$(sfc\Diamond)\ \frac{\Diamond\varphi, \Gamma}{\varphi, \Diamond\varphi, \Gamma \mid\ \neg\varphi, \Diamond\varphi, \Gamma} \qquad (sfc\vee)\ \frac{\varphi \vee \psi, \Gamma}{\neg\varphi, \neg\psi, \Gamma \mid\ \neg\varphi, \psi, \Gamma \mid\ \varphi, \psi, \Gamma}$$

Σ contains no $\Box$-formulae and no $\Diamond$-formulae

Figure 2: The tableau rules for modal logic $\mathbf{S5}$.

When the conjunction of formulae 10 to 13 is entered into the TWB when it is running the tableau for $\mathbf{S5}$ combined with $\mathbf{KD}$ they are found to be consistent.

CONCLUSION

By defining the relationship between deontic and modal logics we have been able to use the automation of a tableau calculus for $\mathbf{KD}$ in the Tableau Workbench to investigate the potential of an automated theorem prover for standard deontic logic. Because SDL is capable of formalising legal rules we can consider this automated theorem prover to be a very primitive legal expert system. Using the paradox of conditional obligation we demonstrated the way in which sets of rules can be checked for consistency using such a system. We then extended the basic logic of SDL to temporal deontic logic in order to overcome the paradox. TDL was automated in the TWB using the tableau rules for the modal logic $\mathbf{S5}$.

ACKNOWLEDGMENTS

I am grateful to Raj Goré for his help and suggestions throughout the process of implementing the tableau calculi in the TWB. I would also like to thank Pietro Abate, the author of the TWB, for his help and his willingness to constantly modify the program at my request.

REFERENCES

[AG03] P. Abate and R. Goré. System Description: The Tableaux Work Bench. Can be found at: http://csl.anu.edu.au/ abate/twb, 2003.

[DGHP99] M D'Agostino, D. Gabbay, R. Hänle, and J. Posegga, editors. *Handbook of Tableau Methods*. Kluwer Academic Publishers, 1999.

[Duc95] H. N. Duc. Semantical Investigations in the Logic of Actions and Norms. Master's thesis, Faculty of Sociology and Philosophy, Leipzig University, 1995. Can be found at: http://dol.uni-leipzig.de/pub/1995-20/en.

[FH71] D. Føllesdal and R. Hilpinen. Deontic Logic: An Introduction. In R. Hilpinen, editor, *Deontic Logic: Introductory and Sstematic Readings*, pages 1–35. D.Reidel Publishing, 1971.

[Gor99] R. Goré. Tableau methods for modal and temporal logics. In M D'Agostino, D. Gabbay, R. Hänle, and J. Posegga, editors, *Handbook of Tableau Methods*, pages 297–396. Kluwer Academic Publishers, 1999.

[Roy98] L. M. M. Royakkers. *Extending Deontic Logic for the Formalisation of Legal Rules*. Kluwer Academic Publishers, 1998.

[vW51] G. H. von Wright. Deontic Logic. *Mind*, 60:1–15, 1951.