Kragujevac J. Math. 23 (2001) 107–117.

THE RESOLUTION TABLEAU FOR LOGICS OF LIKELIHOOD

Zoran Ognjanović

Matematički institut, Kneza Mihaila 35, 11000 Beograd, Yugoslavia

(Received May 15, 2001)

Abstract. In this article we consider a proof procedure for logics of likelihood. The procedure relies on tableau-like and resolution-like inferences and is suitable for execution on parallel computers.

1. INTRODUCTION

The tableau was emphasized as a useful basis for theorem proving [5], especially in the field of non-classical logics [1]. On the other side, the resolution method [6] is the best known procedure in the automated theorem proving community, but it is not so widely utilized when the modal logics are discussed. The resolution tableau for modal logics was presented in [4]. It combines tableau-like approach and resolution-like inference.

In this article we consider an application of the resolution tableau to the logics of likelihood (LL-logics) [2]. LL-logics formalize qualitative reasoning about belief and decision making in situations when the knowledge is not crisp, but is not clear that probability theory is applicable. A modal operator, signed L, is used to cover the notion to being likely.

We exploit a Kripke-like possible world approach [3] to the semantics of LL-logic and extend our original method for modal theorem proving. The aim is twofold: to develop a tool for automated reasoning about likelihood, and to examine the generality of the method.

This paper is organized as follows: We begin with an introduction to the so-called weaker logic of likelihood (signed LL^-). In Section 3 the unifying notation is introduced. Section 4 contains a description of our system and the completeness theorem for LL^-. In Section 5 the "real" logic of likelihood (LL) is considered. Conclusions are summarized in Section 6.

2. THE SYNTAX AND THE SEMANTICS OF LL^-

In this section we follow the notion from [2], except that we use $\Box$ (it is necessarily true) and $\Diamond$ (it is possibly true) instead of G and F.

The set ϕ of LL^--formulas are built over the set of logical operators ($\neg$, $\wedge$, $\Box$ and L) and the set $\phi_0 = \{p, q, r, \ldots\}$ of propositional letters. A formula $L\ P$ can be read as "P is reasonably likely to be a consistent hypothesis". The usual abbreviations ($\vee$, $\rightarrow$, $\leftrightarrow$ and $\Diamond$) are also used. For example, $\neg\Diamond\neg P$ means $\Box P$.

The semantics of LL^--logic is given by means of Kripke models. An LL^--model is a quadruple $M = \langle S, L, C, \pi\rangle$, where S is a set of states, L and C are binary relations on S with L reflexive. If $(s,t) \in L \cup C$ (resp. $(s,t) \in L$, $(s,t) \in C$), then t is a successor (L-successor, C-successor) of s. A state t is reachable (L-reachable, C-reachable) from s, if for some finite sequence $s_0 = s, \ldots, s_n = t$, and $(s_i, s_{i+1}) \in L \cup C$ (resp. L, C) for $i < n$. The valuation π is a mapping from $\phi \rightarrow 2^S$ (the power set of S) that satisfies the following:

1. $\pi(\neg P) = S - \pi(P)$,
2. $\pi(P \wedge Q) = \pi(P) \cap \pi(Q)$,

3. $\pi(\Box P) = \{s : \text{for all } t \text{ reachable from } s,\ t \in \pi(P)\}$,

4. $\pi(L\ P) = \{s : \text{for some } t \text{ such that } (s,t) \in L, t \in \pi(P)\}$.

$M, s \models P$ (or $s \models P$ can be written instead of $s \in \pi(P)$.

If $M = \langle S, L, C, \pi\rangle$ is an LL^--model, the triple $\langle S, L, C\rangle$ is an LL^--frame. A formula P is satisfiable if for some LL^--model $M = \langle S, L, C, \pi\rangle$ and some $s \in S$, $s \models P$. A formula P is valid if for all LL^--models $M = \langle S, L, C, \pi\rangle$ and all $s \in S$, $s \models P$. It is easy to see that $s \models \Diamond P$ iff $t \models P$, for some t reachable from s. A decreasing sequence of strength of modal propositions is given by: $\Box P, \ldots, \neg L \ldots L \neg P, \ldots, \neg L \neg P, P, L\ P, \ldots, L \ldots L\ P, \ldots, \Diamond P$.

Informally, a state s is introduced as a set of hypotheses that are "true for now". An L-successor of s describes a set of hypotheses which is reasonably likely given current hypotheses. A C-successor describes a set of hypotheses which is conceivable, but not necessarily reasonably likely. Further discussion about L and C and their relationship can be found in [2].

We note that there is a connection between LL^--logic and well known modal logic $S4$. If $\langle S, L, C, \pi\rangle$ is an LL^--model, then the triple $\langle S, R, \pi'\rangle$ is an $S4$-model, where R is transitive closure of $L \cup C$ and π' is corresponding extensions of π. A sound and complete axiom system for LL^- contains the system for $S4$, as well as the following axioms: $\Box A \to \neg L \neg A$, $L(A \vee B) \to (L\ A \vee L\ B)$, $A \to L\ A$ and $\Box(A \to B) \to (L\ A \to L\ B)$.

3. THE UNIFYING NOTATION

Literals are propositional letters and their negations. Modal formulas (without literals) are grouped in α, β, ν, π, λ and μ formulas and their respective components [5, 1] (Fig. 1). An α-formula is true iff its components are also true. A β-formula is true iff either β_1 or β_2 is true. Let $M = \langle S, L, C, \pi\rangle$ be an LL^--model, and $s \in S$. For a ν-formula ν, $s \models \nu$ iff for every state t reachable from s, $t \models \nu_0$. For a π-formula π, $s \models \pi$ iff there is at least one state t reachable form s, such that $t \models \pi_0$.

α	α_1	α_2
$P \wedge Q$	P	Q
$\neg(P \vee Q)$	$\neg P$	$\neg Q$
$\neg(P \to Q)$	P	$\neg Q$
$\neg\neg P$	P	P

β	β_1	β_2
$\neg(P \wedge Q)$	$\neg P$	$\neg Q$
$P \vee Q$	P	Q
$P \to Q$	$\neg P$	Q

ν	ν_0
$\Box P$	P
$\neg\Diamond P$	$\neg P$

π	π_0
$\neg\Box P$	$\neg P$
$\Diamond P$	P

λ	λ_0
$L\ P$	P

μ	μ_0
$\neg L\ P$	$\neg P$

Figure 1: The unifying notation

For a μ-formula μ and a state $s \in S$, $s \models \mu$ iff for every state t, such that $(s,t) \in L$, $t \models \mu_0$. For a λ-formula λ, $s \models \lambda$ iff there is at least one state t such that $(s,t) \in L$, $t \models \lambda_0$. Obviously, the λ and μ-formulas behave like the π and ν formulas.

Prefix is an integer. If P is a formula and k is a prefix, then $k\ P$ is a prefixed formula. Prefixes are used as names of states of LL^--models.

4. THE RESOLUTION TABLEAU FOR THE LL^--LOGIC

Let we examine an LL^--formula X. Supposing that X is valid, we study its behavior in an arbitrary state of an arbitrary model. The procedure begins with a 1-node tree $T(X)$ that contains the prefixed formula $0\ X$, a set of prefixes $Pref(T(X)) = \{0\}$ and two relations ρ and σ defined on this set of prefixes that are set to be $\rho = \{0,0\}$ and $\sigma = \emptyset$. The tree $T(X)$ called resolution tableau is enlarged during the execution of the proving procedure, as well as the set $Pref(T(X))$ and the relations ρ and σ. Tableau nodes are labeled by prefixed subformulas of the formula X. Prefixes are understood as states of the LL^--frame $M_{T(X)} = \langle Pref(T(X)), \rho, \sigma \rangle$ which corresponds to the tableau. By reduction rules some sets of subformulas of X are generated. We always need that at least one of those sets is satisfied to acknowledge that the examined formula is really valid. To check that a procedure dual to the classical

$$\begin{array}{c} k\ \alpha \\ k\ \alpha_1 \\ k\ \alpha_2 \end{array} \qquad \begin{array}{c} k\ \beta \\ k\ \beta_1 \qquad k\ \beta_2 \end{array} \qquad \begin{array}{c} k\ \pi \\ k_0\ \pi_0 \quad k_1\ \pi_0 \quad \ldots \quad k_n\ \pi_0 \end{array}$$

α-rule $\qquad$ β-rule $\qquad$ π-rule

$$\begin{array}{c} k\ \nu \\ k'\ \nu_0 \\ \\ \rho := \rho + (k', k') \\ \sigma := \sigma + (k, k') \end{array} \qquad \begin{array}{c} l\ \mu \\ l'\ \mu_0 \\ \\ \rho := \rho + (l, l') \\ \rho := \rho + (l', l') \end{array} \qquad \begin{array}{c} l\ \lambda \\ l_0\ \lambda_0 \quad l_1\ \lambda_0 \quad \ldots \quad l_m\ \lambda_0 \end{array}$$

ν-rule $\qquad$ μ-rule $\qquad$ λ-rule

Figure 2: The tableau construction rules

propositional resolution is used.

In the system of resolution tableau two types of rules are used: the tableau construction rules (Fig. 2) and the resolution rule which is given later on.

The α and β rules are classical rules [5]. The ν and π rules are modal rules [4]. The μ and λ rules are developed to capture L-operator. The rules are read as follows. If an α-formula occurs on a branch, its α_1 and α_2 components are added to the end of the branch. If a β-formula occurs on a branch, the end of the branch is split, and the corresponding components are added to the end of the left (β_1) and the right fork (β_2). If a ν-formula with prefix k occurs on a branch, its prefixed component $k'\ \nu 0$ is added to the end of the branch. The prefix k' is a new one. The pair (k, k') is added to σ, and the pair (k', k') is added to ρ. A branch that contains a π-formula with prefix k branches whenever a new prefix k_i reachable from k (in $M_{T(X)}$) is introduced. New extension contains $k_i\ \pi_0$. The μ-rule looks like the ν-rule, but the μ-rule changes only ρ by adding pairs (l, l') and (l', l'). If a λ-formula with prefix l occurs on a branch, the branch branches whenever a new prefix l_i, a ρ-successor of l, is introduced.

A node is reduced by the rules at most once on a particular branch. After that the node is finished. π and λ-nodes are exceptions. They are finished if there is no (and there will be no) reachable prefixes not yet used in their reduction. A branch is finished if it cannot be extended by the reduction rules. A tableau is finished if every branch is finished.

Definition 1. *An LL^--interpretation I is a mapping from set $Pref$ of prefixes into a set S of states of an LL^--model $M = \langle S, L, C, \pi \rangle$, when the following is satisfied:*

- $(\forall k \in Pref)(\forall k' \in Pref)(k\rho k' \Rightarrow I(k)\ L\ I(k'))$*, and*
- $(\forall k \in Pref)(\forall k' \in Pref)(k\sigma k' \Rightarrow I(k)\ C\ I(k'))$*.*

Definition 2. *A formula X with prefix k is satisfied under an LL^--interpretation I if $I(k) \models X$. A set S of formulas is satisfied under an LL^--interpretation I if each formula from S is satisfied under I.*

The next lemma is the main one about the resolution tableau system.

Lemma 1. *Let $T(X)$ be a finished tableau whose root contains formula $0\ X$. Formula X is LL^--valid iff for each LL^--interpretation I there is at least one branch of the tableau whose set of all prefixed literals is satisfied under the interpretation I.*

Proof. ($\Leftarrow$) Suppose that X is not LL^--valid. Then, there is a model $M = \langle S, L, C, \pi \rangle$ and a state $s \in S$ such that $s \not\models X$. We consider the following interpretation I: $I(0) = s$, if prefix k' is introduced from prefix k by the ν-rule, and $I(k) = s_1$, then

$$I(k') = \begin{cases} \text{any } s_2 \text{ reachable from } s_1 & \text{, if } I(k) = s_1 \models \nu \\ \text{some } s_2 \text{ reachable from } s_1, \text{ such that } s_2 \not\models \nu_0 & \text{, otherwise,} \end{cases}$$

and if prefix l' is introduced from prefix l by the μ-rule, and $I(l) = s_1$, then

$$I(l') = \begin{cases} \text{any } s_2 \text{ such that } s_1\ L\ s_2 & \text{, if } I(l) = s_1 \models \mu \\ \text{some } s_2 \text{ such that } s_1\ L\ s_2, s_2 \not\models \mu_0 & \text{, otherwise.} \end{cases}$$

Next, suppose that there is at least one finished branch whose set of all prefixed literals is satisfied under the interpretation I. Using induction it is easy to show that the set B of all prefixed formulas from the branch is satisfied. For example, if $k\ \alpha \in B$, then by construction rules $\{k\ \alpha_1, k\ \alpha_2\} \subset B$, and by induction hypothesis $I(k) \models \alpha_1$ and $I(k) \models \alpha_2$. It is easy to see that, $I(k) \models \alpha$. If $l\ \mu \in B$, then for some prefix l', $l'\ \mu_0 \in B$, and $I(l') \models \mu_0$. Using the definition of the interpretation I, $I(l) \models \mu$. If $l\ \lambda \in B$, then for a prefix l' formula $l'\ \lambda_0 \in B$, and $I(l') \models \lambda_0$. Since $I(l)\ L\ I(l')$, by the definition of LL^--models $I(l) \models \lambda$. The other cases follow similarly. Since $0\ X \in B$ and $I(0) = s \models X$ a contradiction follows.

($\Rightarrow$) Suppose that X is an LL^--valid formula and that $T(X)$ is a tableau that contains $0\ X$ in the root. Let $M = \langle S, L, C, \pi \rangle$ be an LL^--model, and I an LL^--interpretation, such that $I : \mathit{Pref}(T(X)) \to S$. We consider the LL^--model $M_0 = \langle \mathit{Pref}(T(X)), \rho, \sigma, V \rangle$, where $\mathit{Pref}(T(X))$, ρ and σ are as above, and V is a valuation such that, for any literal Z, $j \in V(Z)$ iff $I(j) \in \pi(Z)$. Since X is LL^--valid, $M_0, 0 \models X$. Every prefixed literal is satisfied under the interpretation I iff it is satisfied under the LL^--interpretation $I_0 : \mathit{Pref}(T(X)) \to \mathit{Pref}(T(X))$, such that for every $j \in \mathit{Pref}(T(X))$, $I_0(j) = j$. We can, step by step, choose a branch b whose set B of all prefixed literals is satisfied under I_0. At the beginning $b = \{0\ X\}$. After i steps we have chosen an initial segment of b that is satisfied under I_0. Consider a formula $Y \in B$. If Y is not a prefixed literal, we extend b by component(s) of Y. If Y is an μ-formula, by the definitions of the valuation and the interpretation, its μ_0-component is also satisfied, and it is added to b. Similar holds for ν and α-formulas. If Y is a β-formula, at least one of its component is satisfied under I_0. We add that one to B. If Y is a formula $l\ \lambda$, for every prefix l', such that $(l, l') \in \rho$, there is a continuation of the branch that contains $l'\ \lambda_0$. But, only those prefixes (states from M_0) are in the relation ρ with l, and by the definition of LL^--models there is at least one l' between them such that $l' \models \lambda_0$. We add the node containing formula $l'\ \lambda_0$ to b. Similar holds for π-formulas. In such a way the branch b is chosen. Obviously, the set B of all prefixed literals from b is satisfied under I_0, and consequently under I.

The term dual clause will be used for a set of prefixed literals. The empty clause is denoted by $\emptyset$. Note that a prefixed literal $k\ A$ (a dual clause C) can be seen as a propositional literal A_k (a set of propositional literals). We will use the term dual clause with the both meanings and say that a dual clause corresponds to a branch of a tableau if it contains all prefixed literals from the branch.

To obtain a complete proof procedure we include the so called dual resolution rule:

$$(\mathrm{DR}) = \begin{array}{l} \text{if } C_1 \text{ and } C_2 \text{ are dual clauses, } k\ A \in C_1 \text{ and } k\ \neg A \in C_2, \\ \text{by resolving these clauses we get their resolvent} \\ R(C_1, C_2, A) = (C1 \setminus \{k\ A\}) \cup (C2 \setminus \{k\ \neg A\}). \end{array}$$

The connection of satisfaction under LL^--interpretations and the resolution procedure is explained by the lemmas 2 and 3. Lemma 2 follows from the duality principle and the completeness of the classical propositional resolution [6], while Lemma 3 is a trivial consequence of the lemmas 1 and 2.

Lemma 2. *A set S of dual clauses is valid iff $\emptyset$ could be inferred from S by (DR).*

Lemma 3. *Let $T(X)$ be a finished tableau for a formula X. X is LL^--valid iff $\emptyset$ can be inferred from the set of all dual clauses that correspond to branches of $T(X)$.*

In the resolution tableau procedure we allow in applications of (DR) only those dual clauses that correspond to finished tableau branches and their resolvents.

By an appropriately chosen order in which the tableau construction rules should be applied, one can guarantee that for every formula there is a finished tableau. Since all the rules introduce only a finite number of nodes and prefixes, after a finite number of reduction rules were applied, there would be only finite numbers of nodes and prefixes suitable for reduction. We will not reduce anything else until we finish with those nodes and prefixes. It is enough that for every stage of the reduction we can guarantee that all reducible nodes will be processed and that a resolution tableau will be finished. If a formula is not valid, this order of execution of rules may produce infinite tableau. A modification of the order that always produces finite tableaus is given in [1, 4].

Definition 3. *A proof for the formula X in the resolution tableau system is a*

$$\begin{array}{ccc}
 & 0\ \Box P \to \neg L \neg P\ (1) & \\
0\ \neg \Box P\ (2) & & 0\ \neg L \neg P\ (3) \\
0\ \neg P \qquad 1\ \neg P & & 1 \neg\neg P\ (4) \\
(6) \qquad\quad (7) & & 1\ P\ (5)
\end{array}$$

Figure 3: Dual tableau for $\Box P \to \neg L \neg P$

sequence of applications of the tableau construction rules and the dual resolution rule. The last step in a proof is the derivation of the empty clause.

Completeness theorem for LL^{-}. *A formula X is LL^{-}-valid iff it has a finite proof in the resolution tableau system.*

Proof. ($\Leftarrow$) If X has a finite proof, according to the lemmas 1, 2 and 3, it is LL^{-}-valid.

($\Rightarrow$) Suppose that formula X is LL^{-}-valid and that $T(X)$ is the corresponding tableau. By the ($\Rightarrow$) of Lemma 1, under arbitrary LL^{-}-interpretation I there is a tableau's branch whose set of all prefixed literals is satisfied under I. According to the lemmas 2 and 3, $\emptyset$ can be inferred from the set S of all dual clauses from tableau branches. Because of compactness of the propositional calculus, the same holds for a finite subset S' of S. Since the part of $T(X)$ that corresponds to S' could be constructed, and the empty clause could be derived form S in finite number of steps, the finite proof is obtained.

Example 1. Fig. 3 shows tableau for formula $\Box P \to \neg L \neg P$, an axiom of LL^{-}-logic. The nodes (2) and (3) are obtained from (1) by the β-rule. (4) is from (3) by the μ-rule. (5) is from (4) by the α-rule. The relation $\rho = \{(0,0),(0,1),(1,1)\}$. (6) and (7) are obtained from (2) by the π-rule. The set of dual clauses is: $\{\{0\ \neg P\}$, $\{1\ P\}$, $\{1\ \neg P\}\}$. An application of the rule (DR) on the second and third clauses infers the empty clause. It follows that the formula $\Box P \to \neg L \neg P$ is LL^{-}-valid.

5. LL-LOGIC, THE FULL LOGIC OF LIKELIHOOD

The language of LL is produced by adding the L^* operator to the LL^--language. An LL^--model $M = \langle S, L, C, \pi \rangle$ is an LL-model if $\pi(L^*P) = \{s$: for some t L-reachable from $s, t \in \pi(P)\}$.

It is straightforward to extend the resolution tableau system to cover the full LL-logic. The rule that solve formulas of the form L^*P would be similar to the π-rule, except that branching would be done only for ρ-reachable prefixes. On the other hand, the rule for $\neg L^*P$ would be similar to the μ-rule. Finally, it is easy to extend the above given proofs for the corresponding statements about LL-logic.

6. CONCLUSION

The resolution tableau system can be used as a tool for automatic reasoning about likelihood. It is straightforward to implement the tableau construction rules. They reflect semantic laws of valuation and in a natural way reduce a modal formula to a set of propositional clauses suitable for a classical resolution procedure.

We note that our proof procedure is suitable for a parallel execution. We have found two main possibilities for parallelization: tableau construction is independent of the dual resolution procedure up to the generation of dual clauses and dual clauses are ground and can be resolved in a fully distributed manner, without any additional communication overhead. Thus, a parallel MIMD computer, like a transputer based system or a local network, can be used in an implementation of the resolution tableau system.

Acknowledgments: This work was supported by the Ministarstvo za nauku, tehnologiju i razvoj Republike Srbije, through Matematički institut, Belgrade.

References

[1] M.Fitting, *Proof Methods for Modal and Intuitionistic Logic*, D.Reidel Publishing Co., Dordecht, 1983.

[2] J.Y. Halpern, M.O. Rabin, *A logic to reason about likelihood*, Artificial Intelligence, **32**, (1987) 379–405.

[3] S. Kripke, *Semantical analysis of modal logics I, normal propositional calculi*, Zeitschrift für mathematische Logik und Grundlagen der Mathematik, **9**, (1963) 67–96.

[4] Z. Ognjanović, *A tableau-like proof procedure for normal modal logics*, Theoretical Computer Science, **129** (1994), 167–186.

[5] Smullyan, R., *First-Order Logic*, Springer-Verlag, Berlin (1968).

[6] J.A. Robinson, *A Machine-oriented logic based on the resolution principle*, Journal of the ACM, **12** (1965) 23–41.