

*Liet. mat. rink.*, **49**, 2008, 1–4

## Proof-search in hybrid logic

Daiva ALEKNAVIČIŪTĖ, Stanislovas NORGĖLA (VU)

e-mail: daiva.aleknaviciute@mif.vu.lt, stasys.norgela@mif.vu.lt

**Abstract.** This paper describes a new tactic for proof-search in Hybrid logic  $\mathcal{H}(@)$ , which always terminates.

*Keywords:* hybrid logic, sequent calculus.

### 1. Introduction

Hybrid logic  $\mathcal{H}(@)$  is decidable. However, a tableau method for Hybrid logic  $\mathcal{H}(@, \downarrow)$  described in [2] does not always terminate even for formulae belonging to  $\mathcal{H}(@)$ . Recently substantial interest has been shown in terminating proof-search methods for decidable classes of Hybrid logic. T. Bolander and T. Braüner [3] give a tableau method with loop checking. S. Cerrito and M. Cialdea Mayer [1] describe a tableau method without loop-checking, which always terminates for formulae from  $\mathcal{H}(@)$ .

This paper proves that a derivation tree in sequent calculus for every  $\mathcal{H}(@)$  formula will be finite if we use  $(\diamond)$  rule as late as possible. The proof refers to the paper [1], which provides a similar tableau method. The main difference is that if we use  $(Sub)$  rule in [1] we might need to delete some formulae from sequent and later to create them again. Our proposed method does not have this restriction. This is because if we use  $(\diamond)$  rule as late as possible we will not create new unnecessary nominals that  $(Sub)$  rule would need to remove.

### 2. Sequent calculus

**DEFINITION 1.** Let  $S$  be the initial sequent of derivation tree. Let  $C_T$  be a set of nominals in the initial sequent. Let  $NOM_\Gamma$  be a set of nominals in sequent  $\Gamma$ . Let  $S_T^*$  be a set of all formulae which we can get from subformulae of the initial sequent if we substitute nominals with different nominals from set  $C_T$  (we might leave the same nominals as well).

Notice that  $S_T^*$  is a finite set, since  $S$  and  $C_T$  are finite sets.

**LEMMA 1.** *Let sequent  $\Gamma$  be stable if it contains only formulae of form  $@_s \diamond t$  or  $@_s F$ , where  $s, t$  are nominals (not necessary belonging to  $C_T$ ) and  $F \in S_T^*$ . If we use any rule on stable sequent we will also get a stable sequent(s).*

*Proof.* This can be easily proven by analysing all rules.

COROLLARY. *Since the initial sequent of a derivation tree contains only formulae of form  $@_s F$ , where  $F \in S_T^*$ , we get that all sequents in the derivation tree are stable.*

Let us choose a tactic that  $(\diamond)$  rule must be used only if no other rule can be used. Further we will analyse some particular branch in the derivation tree. A part of the branch from the initial sequent to the first usage of  $(\diamond)$  rule will be called *phase 1*, a part between the first and the second usage of  $(\diamond)$  – *phase 2*, etc. At the end of each phase we will only have formulae that belong to one of the following sets:

$$\begin{aligned} S_{\diamond} &= \{ @_s \diamond t : s, t \in Nom \}, & S_{\square} &= \{ @_s \square F : s \in Nom, F \in S_T^* \}, \\ S_{\neg} &= \{ @_s \neg t : s \in Nom, t \in C_T \}, & S_{\diamond F} &= \{ @_s \diamond F : s \in Nom, F \in S_T^* \}. \end{aligned}$$

We will not get any formulae of other forms, because if the top operator (excluding the first  $@_s$ ) in the formula is either  $\&$ ,  $\vee$ ,  $@_t$  or the formula has a form  $@_s t$ , we could apply a rule other than  $(\diamond)$ . This would contradict our chosen tactic.

LEMMA 2. *If we use  $@_s \diamond t$  formula in  $(\diamond)$  rule when we move to the next phase, at the end of the next phase we will have the same sequent as we had before.*

*Proof.* For a proof see “Some Decidable Classes of Formulas of Pure Hybrid Logic” [4] Lemma 3.

COROLLARY. *When making a transition to a new phase we need to use a formula from  $S_{\diamond F}$ .*

If we use  $(\square)$  rule twice on the same pair of formulae we will not get any new formulae. To avoid such repetition we can annotate each  $\square$  operator with a set of nominals  $N$ , which were used in  $(\square)$  rule for that operator. To add this annotation we need to adjust  $(\square)$  and  $(Sub)$  rules:

$$\frac{\Gamma, @_t F, @_s \square^{N \cup \{t\}} F, @_s \diamond t}{\Gamma, @_s \square^N F, @_s \diamond t} (\square),$$

where  $t \notin N$ ,

$$\frac{\Gamma[t/s]}{\Gamma, @_s t} (Sub).$$

$\Gamma[t/s]$  also replaces  $s$  to  $t$  inside annotation sets  $N$ . In the initial sequent we annotate all  $\square$  operators with an empty set –  $\square^{\emptyset}$ .

It is also useless to start two phases by using  $(\diamond)$  for the same formula, since same formulae will hold for a new nominal as for an old nominal. Consequently for each branch in the derivation tree we remember formulae used to move from one phase to another.

LEMMA 3. *We can use only a finite number of rules inside each phase.*

*Proof.* Since we do not create new nominals inside a phase  $i$  then  $NOM_\Gamma$  is a finite set and  $NOM_\Gamma \subseteq NOM_{\Gamma_0}$  for all sequents  $\Gamma$  in the phase  $i$ , where  $\Gamma_0$  is the first sequent of the phase  $i$ . Let  $W = \{ @_t \Box F : t \in C_T \cup NOM_{\Gamma_0}, F \in S_T^* \}$ . From Lemma 1 we get that  $S_{i\Box} \subseteq W$ , where  $S_{i\Box}$  is the set of formulae belonging to  $S_\Box$  in the phase  $i$ . Since  $C_T, NOM_{\Gamma_0}$  and  $S_T^*$  are finite sets then  $S_{i\Box}$  is also finite. Similarly  $S_{i\Diamond}$  is finite because a set of formulae belonging to  $S_\Diamond$  at the beginning of the phase  $i$  is finite and we can only add new formulae to  $S_{i\Diamond}$  of form  $@_s \Diamond t$ , where  $s, t \in NOM_{\Gamma_0}$ . Consequently since  $S_{i\Box}$  and  $S_{i\Diamond}$  are finite sets then we can use ( $\Box$ ) rule only finitely many times inside the phase  $i$ .

Notice that all rules except ( $\Diamond$ ) and ( $\Box$ ) reduce the total number of operators inside a sequent at least by one. Since we can use ( $\Box$ ) and ( $\Diamond$ ) rules finitely many times and we can not use other rules successively infinitely many times, we get that the total number of rules used in the phase  $i$  is finite. This holds for all  $i$  in all branches of a derivation tree.

DEFINITION 2. A *degree of modality* of a formula  $F$  (denoted by  $mod(F)$ ) will be the number of modal operators in  $F$ .

DEFINITION 3. In a particular branch of a derivation tree we denote  $maxmod(s)$  to be the *maximal degree of modality* of formulae which have a form  $@_s F$ , but not  $@_s \Diamond t$ , where  $F \in S_T^*, t \in Nom$ . If no such formula exists in the branch then  $maxmod(s) = 0$ .

DEFINITION 4. As in [1] we will give definitions for *child* and *parent* nominals. If we get a formula  $@_s \Diamond t$  using ( $\Diamond$ ) rule ( $t$  is a new nominal), we will call  $s$  – “a parent of  $t$ ” and  $t$  – “a child of  $s$ ”. Let this relation be denoted by  $s \rightsquigarrow t$ .

LEMMA 4. A set of children  $V_s = \{ t : s \rightsquigarrow t \}$  for a particular nominal  $s$  is finite.

*Proof.* When entering a new phase we might use a formula of form  $@_s \Diamond F$ , where  $F \in S_T^*$ . Each such formula can be used only once. Since  $S_T^*$  is a finite set,  $V_s$  is also finite.

LEMMA 5. Each branch of a derivation tree contains a finite number of phases.

*Proof.* We will prove this by showing that for all  $s \rightsquigarrow t$  in the branch we have  $maxmod(t) < maxmod(s)$ .

If  $s \rightsquigarrow t$  then some phase begins by using ( $\Diamond$ ) rule for a formula  $@_s \Diamond F$  and gives formulae  $@_s \Diamond t, @_t F$ . Obviously,

$$mod(@_t F) < mod(@_s \Diamond F),$$

and the first sequent of this phase does not contain a formula  $@_t G$  such that

$$mod(@_t G) \geq maxmod(s).$$

( $\&$ ), ( $\vee$ ), (*Simp*), (*Sub*), ( $\Diamond$ ) rules preserve this property in further sequents. If we use ( $\Box$ ) rule we get a new formula of form  $@_t G$ . But we also must have a formula  $@_s \Box G$

for  $(\Box)$  rule. Consequently  $(\Box)$  rule also preserves that  $\text{mod}(@_t G) < \text{maxmod}(s)$ . By definition of  $\text{maxmod}(t)$  we get  $\text{maxmod}(t) < \text{maxmod}(s)$ .

If  $\text{maxmod}(s) = 0$  then  $s$  can not have any children, since we would need a formula of form  $@_s \diamond F$ , which gives  $\text{maxmod}(s) > 0$ . Consider a sequence  $s_0 \rightsquigarrow s_1 \rightsquigarrow s_2 \rightsquigarrow \dots$ . Since  $\text{maxmod}(s_i) > \text{maxmod}(s_{i+1})$  and  $\text{maxmod}(s_k) = 0$  means  $s_k$  does not have any children, we get that the sequence is finite. Using Lemma 4 we get that a set of all new nominals

$$\{s_1, s_2, \dots : s_0 \rightsquigarrow s_1 \rightsquigarrow s_2 \rightsquigarrow \dots, s_0 \in C_T\}$$

is finite. Since every phase creates a new nominal, the total number of phases in a branch is finite.

**THEOREM 1.** *Annotation of  $\Box$ , remembering of used  $@_s \diamond F$  formulae and using  $(\diamond)$  rule as late as possible gives us a finite derivation tree.*

*Proof.* From Lemma 3 and Lemma 5 we get that every branch in the derivation tree uses a finite number of rules. Consequently the whole derivation tree is finite.

## References

1. S. Cerrito and M. Cialdea Mayer, Terminating tableaux for  $\mathcal{HL}(@)$  without loop-checking. *Technical Report IBISC-RR2007-07*, Université d'Évry Val d'Essonne (2007).
2. P. Blackburn, M. Marx, Tableaux for quantified hybrid logic, *LNAI*, **2381**, 38–52 (2002).
3. T. Bolander and T. Braüner, Tableau-based decision procedures for hybrid logic, *Journal of Logic and Computation*, **16**(6), 737–763 (2006).
4. S. Norgėla, A. Šalaviejiėnė, Some decidable classes of formulas of pure hybrid logic, *Lith. Math. J.*, **44**(4), 462–469 (2007).

## REZIUOMĖ

### *D. Aleknavičiūtė, S. Norgėla. Išvedimo paieška Hibridinėje logikoje*

Aprašoma sekvencinio skaičiavimo taktika hibridinei logikai  $H(@)$ , visada užbaigianti darbą hibridinės logikos  $H(@)$  formulėms.

*Raktiniai žodžiai:* ???