Completeness of a First-order Temporal Logic with Time-Gaps*

Matthias Baaz[a,**], Alexander Leitsch[b], Richard Zach[b,c]

[a] *Institut für Algebra und Diskrete Mathematik E118.2, Technische Universität Wien, A-1040 Vienna, Austria*
[b] *Institut für Computersprachen E185.2, Technische Universität Wien, A-1040 Vienna, Austria*
[c] *Group in Logic and the Methodology of Science, University of California, Berkeley, CA 94720-3840, USA*

Abstract. The first-order temporal logics with $\Box$ and $\bigcirc$ of time structures isomorphic to ω (discrete linear time) and trees of ω-segments (linear time with branching gaps) and some of its fragments are compared: The first is not recursively axiomatizable. For the second, a cut-free complete sequent calculus is given, and from this, a resolution system is derived by the method of Maslov.

1 Introduction

In recent years, various temporal logics have been studied and applied to the description and analysis of dynamic properties of programs [7]. The investigations have focussed on discrete, linearly ordered, well-founded temporal structures because temporal states can then be identified with program states. It turns out that the first-order logics corresponding to this semantics are not recursively axiomatizable if $\Box$ (henceforth always) and $\bigcirc$ (nexttime) are present in the language: It is possible to characterize the set of natural numbers by $\neg\Box\neg U(x)$, where $U(x)$ holds for exactly one domain element at each state and is determined by a recursion in $\bigcirc$ (see [8]). This incompleteness result is based on a standard model of linear time; if similarity types are allowed , one can obtain completeness results for first order temporal logic relative to classes of models of linear time (see [1]). With a change in the semantics (branching time gaps), however, a complete first-order logic *can* be obtained; this is the subject of the present paper. Our proof of completeness can be carried over to several types of future-oriented temporal operators (see [8]); there may be problems however if future- and past-oriented operators are present simultaneously.

For simplicity, we consider here only languages with $\Box$ and $\bigcirc$ as the only temporal operators, and constants as the only function symbols. We compare the logic of discrete linear time **TL** to the logic of discrete linear time with *branching time gaps* **TB**. In both logics, the semantics of the temporal operators are as usual: a formula $\Box A$ is true at a time point t, iff A is true at every time point $\geq t$; a formula $\bigcirc A$ is true at t, iff A is true at $t+1$. The difference lies in the admitted time structures: for **TL**, this is the class of structures order isomorphic to ω. We call such a structure an ω-segment. In such a segment, there is always an earliest point, for every point there is a unique next point, and every point can be reached from the earliest point by passing finitely often to the next point. For **TB**, the admitted

* to appear in *Theoretical Computer Science*
** Corresponding author. Email addresses: {baaz,leitsch}@logic.tuwien.ac.at, zach@math.berkeley.edu

structures are isomorphic to (possibly infinitary) well-founded trees of ω-segments. There is always a unique earliest next point in time, but also points "after the gap" (which cannot be reached by successively passing on to the next point) which are initial states in the next ω-segments themselves, etc.

We give a sequent calculus for **TB**, which is shown to be cut-free complete by an extension of Schütte's reduction tree method. The rules of the calculus constructed are not analytic in the sense that the formulas in the premises are not proper subformulas of the conclusion. Therefore, cut-free proofs in general lack the subformula property, a property essential for usual methods of proof search. The completeness proof shows, however, that we can salvage a large part of analyticity, enough to be able to construct a resolution system for the logic: Every valid sequent has a cut-free proof which uses only formulas A and $\bigcirc A$, where A is a subformula of the end-sequent. Exploiting this property, we construct a complete resolution method for **TB** using the method of Maslov [9, 11].

In a sense then, the investigations of **TB** can also be seen as a case study in (a) how far the completeness proof of Schütte can be carried, and (b) how to overcome mild forms of non-analyticity. It also sheds some light on necessary conditions for the resolution calculus to be sound (completeness is not problematic).

The paper is organized as follows: In Section 2, the semantical structures underlying the logics **TL** and **TB** are introduced, and a proof of non-axiomatizability of **TL** is sketched. In Section 3 we present the sequent calculus **LB** for **TB**. The completeness proof for **LB** is presented in Section 4. Section 5 contains some remarks comparing (fragments of) **TL** and **TB**. The resolution system for **TB** is developed in Section 6. Finally, we conclude with a discussion of the significance of the completeness result for future applications.

2 First-order Temporal Logics

We consider the following first-order language: free variables: a, b, c, a_1, ...; bound variables: x, y, z, x_1, ...; constant symbols: f, g, h, f_1, ...; predicate symbols of arbitrary arity: P, Q, R, P_1, ...; propositional connectives: $\wedge$, $\vee$, $\supset$, $\neg$; quantifiers: $\forall$, $\exists$; and the temporal operators: $\Box$ (always), $\bigcirc$ (next time). Formulas are built up from the symbols as usual. The sometime operator $\Diamond$ is introduced by definition: $\Diamond A \equiv \neg\Box\neg A$. If $A \equiv *_1 \cdots *_n B$, where $*_i$ is either $\Box$ or $\bigcirc$, then $*_1 \cdots *_n$ is called the *temporal prefix of* A. The semantics of a first-order temporal logic is defined as follows:

Definition 2.1. Let T be a denumerable partially ordered set. T belongs to the class $\mathcal{L}$ of *linear discrete orders* iff it is order isomorphic to ω; it belongs to the class $\mathcal{B}$ of *linear discrete orders with branching gaps* iff it is order isomorphic to a well-founded tree of ω-segments.

Definition 2.2. Let $\mathcal{T}$ be $\mathcal{L}$ or $\mathcal{B}$, and let $\mathrm{Frm}(L)$ be the set of formulas over some first-order temporal language L. A *structure* $\mathbf{K}$ *for* L is a tuple $\langle T, \{D_i\}_{i\in T}, \{\mathbf{S}_i\}_{i\in T}\rangle$, where $T \in \mathcal{T}$, D_i is a set called the *domain* at state i, $D_i \subseteq D_j$ if $i \leq j$, $\mathbf{S}_i$ is a function mapping free variables and constant symbols to elements of D_i, and n-ary predicate symbols to functions from D_i^n to $\{\top, \bot\}$.

We define the *valuation functions* $\mathbf{K}_i$ from $\mathrm{Frm}(L)$ to $\{\top, \bot\}$ as follows: Let A be a temporal formula, and not, and, or, impl be the truth functions for negation, conjunction, disjunction, and implication, respectively.

(1) $A \equiv P(t_1, \ldots, t_n)$: $\mathbf{K}_i(A) = \mathbf{S}_i(P)\big(\mathbf{S}_i(t_1), \ldots, \mathbf{S}_i(t_n)\big)$
(2) $A \equiv \neg B$: $\mathbf{K}_i(A) = \mathrm{not}(\mathbf{K}_i(B))$
(3) $A \equiv B \wedge C$: $\mathbf{K}_i(A) = \mathrm{and}(\mathbf{K}_i(B), \mathbf{K}_i(C))$

(4) $A \equiv B \vee C$: $\mathbf{K}_i(A) = \mathrm{or}(\mathbf{K}_i(B), \mathbf{K}_i(C))$
(5) $A \equiv B \supset C$: $\mathbf{K}_i(A) = \mathrm{impl}(\mathbf{K}_i(B), \mathbf{K}_i(C))$
(6) $A \equiv (\forall x)B(x)$: $\mathbf{K}_i(A) = \top$ if $\mathbf{K}_i[d/x](A(d)) = \top$ for every $d \in D_i$, and $= \bot$ otherwise
(7) $A \equiv (\exists x)B(x)$: $\mathbf{K}_i(A) = \top$ if $\mathbf{K}_i[d/x](A(d)) = \top$ for some $d \in D_i$ and $= \bot$ otherwise
(8) $A \equiv \Box B$: $\mathbf{K}_i(A) = \top$ if $\mathbf{K}_j(B) = \top$ for every $j \geq i$ and $= \bot$ otherwise
(9) $A \equiv \bigcirc B$: $\mathbf{K}_i(A) = \top$ if $\mathbf{K}_{i+1}(B) = \top$ and $= \bot$ otherwise

A formula A is *satisfied* in a temporal structure $\mathbf{K}$, $\mathbf{K} \models A$, iff $\mathbf{K}_0(A) = \top$. A is *valid* in a class of temporal structures $\mathcal{T}$, $\mathcal{T} \models A$, iff every $\mathbf{K} = \langle T, \{D_i\}_{i \in T}, \{\mathbf{S}_i\}_{i \in T} \rangle$ with $T \in \mathcal{T}$ satisfies A.

Definition 2.3. The logic of linear discrete time **TL** is the set of all formulas $A \in \mathrm{Frm}(L)$ s.t. $\mathcal{L} \models A$. The logic of linear discrete time with branching gaps **TB** is the set of all formulas $A \in \mathrm{Frm}(L)$ s.t. $\mathcal{B} \models A$.

Example 2.4. In **TL**, the formula $\Box\bigcirc A \equiv \bigcirc\Box A$ is valid. In **TB**, however, only $\bigcirc\Box A \supset \Box\bigcirc A$ holds. The other direction $\Box\bigcirc A \supset \bigcirc\Box A$ does not hold in general, as can be seen by evaluating the formula on the countermodel $\mathbf{K} = \langle \omega + \omega, \{D_i\}_{i \in \omega+\omega}, \{\mathbf{S}_i\}_{i \in \omega+\omega} \rangle$, where $\mathbf{S}_\omega(A) = \bot$ and $\mathbf{S}_i(A) = \top$ for $i < \omega$, $i > \omega$.

The semantics considered here is usually called *initial semantics*. Normal semantics is defined via truth in all states, not only in $\mathbf{K}_0$. We will need the following lemma later on:

Lemma 2.5. *Let A be a formula.*

(1) $\models A$ iff A is true in every world in every temporal structure.
(2) $\models A$ iff $\models \Box A$
(3) $\models A$ iff $\models \bigcirc A$

Proof. (1) If: trivial. Only if: Let T be a temporal structure in which A is not true at a state i. Consider $T' = \{j \in T \mid j \geq i\}$: T' is also a temporal structure, and, since our logics contain no operators acting backwards in time, A is true at state i in T' if it was true in state i in T. But i is the initial state in T'.

(2) If: by the truth definition of $\Box$. Only if: immediate by (1)

(3) If: Let T be a structure where A is false in the initial world. Consider $T' = T \cup 0'$ with $0' < 0$, and $\mathbf{S}_{0'} = \mathbf{S}_0$. The addition of a state before the initial state does not change the truth of formulas in T. But in T', $\bigcirc A$ is false in the initial world. Only if: immediate by (1). $\Box$

Remark 2.6. The logics we consider differ from the ones in the literature in that we do not use global and local variables, but the interpretation of predicate symbols can vary over the states. This is more in keeping with the tradition in quantificational modal logics. However, by using the Barcan formulas for $\Box$ and $\bigcirc$, definable two-sortedness and other expressible concepts, most effects of global and local variables can be simulated. Another minor difference is in the definition of $\Box$: Kröger's $\Box$ is defined via truth in all *later* worlds; in Kröger's logic, our $\Box$ can be defined by $\Box A \wedge A$, his $\Box$ can be expressed by $\bigcirc\Box A$ in **TL**.

As indicated in the introduction, the logic **TL** is not axiomatizable. This was shown for the original formulation of Kröger by Szalas [12] and Kröger [8]. Two binary function symbols have to be present for the results to hold. If the operator **until** is also present, or if quantification over local variables is allowed, then the empty

signature suffices, as was shown by Szalas and Holenderski [13] and Kröger [8], respectively.

Following Szalas [12] and Kröger [8] we sketch a proof for the incompleteness result for **TL** with equality, where the signature contains two binary function symbols (equivalently, two ternary predicate symbols):

Let $'$ designate the successor function, and the constant 0 the number zero. Consider the formula axiomatizing the predicate U,

$$U(0) \wedge \Box(\forall x)\Big(U(x) \equiv (\exists y)\big(y = x' \wedge \bigcirc U(y)\big)\Big) \wedge \Box(\forall x)(\forall y)\big(U(x) \wedge U(y) \supset x = y\big).$$

In every model, $\Diamond U(x)$ represents exactly the set of natural numbers. If the language is expressive enough, we can write down the usual axioms for addition and multiplication (e.g., Robinson's Q). A sentence of arithmetic is true in the natural numbers iff its relativization to $\Diamond U(x)$ follows in **TL** from these axioms. The non-axiomatizability of **TL** thus follows from Gödel's Incompleteness Theorems.

3 A Sequent Calculus for TB

In the standard definition a sequent is an expression of the form

$$A_1, \ldots, A_k \rightarrow B_1, \ldots, B_\ell$$

where the A_i and B_j are first-order formulas. For the purpose of completeness proofs it is more convenient to use instead *infinite sequents* (see, e.g., Takeuti's book [14, Ch. 1.8]). More precisely, the completeness theorem requires a generalization of finite sequences of formulas to countably infinite well-ordered sequences. We will use this more general notion of sequents and indicate the use of finite sequents explicitly.

Let Δ be a countable (possibly finite) well-ordered sequence. If Δ is order isomorphic to the well-ordered set of numbers α via a mapping ϕ s.t. $\phi(i) = A_i$ for $i \in \alpha$ then we write $\Delta = (A_i)_{i \in \alpha}$.

Definition 3.1. A *sequent* is an expression of the form $\Gamma \rightarrow \Delta$, where Γ and Δ are countable well-ordered sequences of first-order temporal formulas.

Definition 3.2. The sequence $(A_i)_{i \in \alpha}$ is called a *subsequence* of $(A_i)_{i \in \beta}$ if $\alpha \subseteq \beta$ and there exists an order-preserving 1-1 mapping $\psi : \alpha \rightarrow \beta$. If the sequences are finite and $\beta = \{1, \ldots, n\}$ then α is of the form $\{i_1, \ldots, i_k\} \subseteq \{1, \ldots, n\}$. A sequent $\Gamma' \rightarrow \Delta'$ is called a *subsequent* of $\Gamma \rightarrow \Delta$ if Γ' and Δ' are subsequences of Γ and Δ, respectively.

Definition 3.3. Let $(S_i)_{i \in \omega}$ be a sequence of sequents s.t. $S_i = \Pi_i \rightarrow \Lambda_i$ for $i \in \omega$. Then the sequent $S = (\Pi_i)_{i \in \omega} \rightarrow (\Lambda_i)_{i \in \omega}$ is called the *union sequent* of $(S_i)_{i \in \omega}$

Note that the order type of $(\Pi_i)_{i \in \omega}$ is characterized by the property: if $i < j$ and α_i, α_j are the well-ordered sets of numbers corresponding to Π_i and Π_j respectively then all elements of α_i are smaller than all elements of α_j.

The validity of *finite* sequents is defined as usual: $A_1, \ldots, A_k \rightarrow B_1, \ldots, B_k$ is valid in **TL** (**TB**) iff $(A_1 \wedge \ldots \wedge A_k) \supset (B_1 \vee \ldots \vee B_l)$ is valid in **TL** (**TB**). A finite sequent is provable if it has a derivation in a suitable calculus.

The concepts of provability (defined for finite sequents originally) can be extended to the infinite case via the usual compactness condition:

Definition 3.4. A (not necessarily finite!) sequent S is called *provable* if there exists a finite subsequent of S which is provable.

It is only a matter of convention that we use the term "provable" for infinite sequents, as **LB** works only on finite sequents. This convention is, however, of essential advantage in completeness proofs. In our completeness proof we do not need the *semantics* of infinite sequents; particularly we do not speak about (semantic) compactness (i.e. about the property that an infinite sequent is valid iff there exists a finite subsequent which is valid).

As basis for the sequent calculus **LB** for **TB** we take a variant of Gentzen's calculus **LK** for classical predicate logic. The rules of **LK** are well-known and can be found in, e.g., [14]. We use a *weakening friendly* formulation of the rules: The side formulas in the premises of the rules ($\wedge$:right), ($\vee$:left), and ($\supset$:left) are not required to be identical, e.g.,

$$\frac{\Gamma \to \Delta, A \quad \Gamma' \to \Delta', B}{\Gamma, \Gamma' \to \Delta, \Delta', A \wedge B} \quad \text{instead of} \quad \frac{\Gamma \to \Delta, A \quad \Gamma \to \Delta, B}{\Gamma \to \Delta, A \wedge B}$$

LB consists of the rules of **LK** plus the following rules for $\bigcirc$ and $\Box$:

$$\frac{A, \bigcirc\Box A, \Gamma \to \Delta}{\Box A, \Gamma \to \Delta}\ \Box\text{:left} \qquad \frac{\Gamma \to \Delta, A \quad \Gamma' \to \Delta', \bigcirc\Box A}{\Gamma, \Gamma' \to \Delta, \Delta', \Box A}\ \Box\text{:right}$$

$$\frac{\Gamma \to \Delta}{\bigcirc\Gamma \to \bigcirc\Delta}\ \text{nex} \qquad \frac{\Box\Gamma \to A}{\Box\Gamma \to \Box A}\ \text{nec}$$

Note that **LB** (like **LK**) is defined for finite sequents only. If Γ is $A_1, \ldots, A_n$, then $\bigcirc\Gamma$ denotes the sequence $\bigcirc A_1, \ldots, \bigcirc A_n$ (similarly for $\Box\Gamma$). The notations $\bigcirc\Gamma$ and $\Box\Gamma$ can be extended to infinite sequents in a straightforward way (e.g., $\bigcirc(A_i)_{i\in\alpha} = (\bigcirc A_i)_{i\in\alpha}$). Note that, unlike the rules of **LK**, the rules ($\Box$:left) and ($\Box$:right) are not analytic (i.e., the subformula property does not hold). The rule (nex) works on the left and right sides of the sequent simultaneously (but is analytic) and (nec) is "context dependent." It is clear that (nec) corresponds to the necessitation rule common in Hilbert-style modal calculi. When using rules with two auxiliary formulas in one premise (i.e., ($\supset$:right) or ($\Box$:left)), the inference is admitted even if only one formula is actually present (implicit weakening). Alternatively, we could have split the rule into two, in a similar way as the ($\vee$:right) and ($\wedge$:left) rules. Otherwise the notion of proof is the standard one (cf. [14, Ch.1, § 2]). In particular, recall that *initial sequents* are of the form $A \to A$ (A any formula) and cut-free provable means having a proof not containing an application of the cut rule. The sequent appearing at the root of the proof tree is called *end-sequent*.

Proposition 3.5. *If a sequent is* **LB***-provable, then a (non-empty) subsequent is provable without weakenings.*

Proof. This is easily seen by induction on the length of the proof, and is due to the special formulation of the rules. □

Example 3.6. We give an **LB**-proof of the formula $\bigcirc\Box A \supset \Box\bigcirc A$.

$$\dfrac{\dfrac{\dfrac{\dfrac{A \to A}{\Box A \to A}\ \Box\text{:left}}{\bigcirc\Box A \to \bigcirc A}\ \text{nex} \qquad \dfrac{\dfrac{\dfrac{\dfrac{A \to A}{\Box A \to A}\ \Box\text{:left}}{\bigcirc\Box A \to \bigcirc A}\ \text{nex}}{\Box A \to \bigcirc A}\ \Box\text{:left}}{\Box A \to \Box\bigcirc A}\ \text{nec}}{\bigcirc\Box A \to \bigcirc\Box\bigcirc A}\ \text{nex}}{\bigcirc\Box A, \bigcirc\Box A \to \Box\bigcirc A}\ \Box\text{:right}$$

$$\dfrac{\bigcirc\Box A, \bigcirc\Box A \to \Box\bigcirc A}{\bigcirc\Box A \to \Box\bigcirc A}\ \text{contr:left}$$

$$\dfrac{\bigcirc\Box A \to \Box\bigcirc A}{\to \bigcirc\Box A \supset \Box\bigcirc A}\ \supset\text{:right}$$

Note that, on the right branch of the proof, we introduced $\Box A$ twice on the left-hand side of a sequent. This is necessary because of the way (nex) introduces $\bigcirc$ in all formulas of the sequent.

Theorem 3.7. **LB** *is sound for* **TB**, *i.e., every finite* **LB***-provable sequent is valid in* **TB**.

Proof. It is sufficient to prove the soundness of the **LB**-rules. The soundness of the **LK**-part is proved as usual. The soundness of the rules $\Box$:left and $\Box$:right follows from the "recursion" equivalence of $\Box A$ and $\bigcirc\Box A \wedge A$ in the **TB**-semantics. The soundness of (nex) follows from Lemma 2.5(3) and from the fact that $\bigcirc$ distributes over the propositional connectives (e.g., $\bigcirc(A \wedge B)$ is equivalent to $\bigcirc A \wedge \bigcirc B$). The soundness of (nec) follows from Lemma 2.5(2), from the **TB**-equivalence of $\Box A$ and $\Box\Box A$, from the distributivity of $\Box$ over $\wedge$, and from the fact that $\Box(A \supset B)$ implies $\Box A \supset \Box B$. $\Box$

If we look closely at the rules of **LB** we notice that ($\Box$:left) and ($\Box$:right) are not strictly analytical. Therefore it is convenient to extend the usual notion of subformula. Note that we have disjoint sets of free and bound variables. A *term* is defined as usual but subject to the restriction that it may only contain free variables; if also bound variables are allowed to occur we speak about *semi-terms*. Similarly we distinguish between *formulas* and *semi-formulas*. The concept of strict sub-semi-formula represents the intuitive notion of subformula, while the definition of semi-formulas takes care about the nonanalytic behaviour of $\bigcirc$ and $\Box$.

Definition 3.8. Let F be a formula. The set $\mathrm{ssf}(F)$ of *strict sub-semi-formulas* of F is defined as $\mathrm{ssf}(F) = \{F\} \cup \Sigma(F)$, where

$$\Sigma(F) = \begin{cases} \{F\} & \text{if } F \text{ is atomic} \\ \mathrm{ssf}(A) & \text{if } F \equiv *A \text{ for } * \in \{\neg, \Box, \bigcirc\} \\ \mathrm{ssf}(A) \cup \mathrm{ssf}(B) & \text{if } F \equiv A * B \text{ for } * \in \{\wedge, \vee, \supset\} \\ \mathrm{ssf}(A(x)) & \text{if } F \equiv (\mathsf{Q}x)A(x) \text{ for } \mathsf{Q} \in \{\forall, \exists\} \end{cases}$$

The set $\mathrm{sub}(F)$ of *sub-semi-formulas* of F is defined by

$$\mathrm{sub}(F) = \mathrm{ssf}(F) \cup \{\bigcirc\Box A \mid \Box A \in \mathrm{ssf}(F)\}$$

By $\mathrm{sub}^*(F)$ we denote the set of formulas obtained from $\mathrm{sub}(F)$ by replacing bound variables without matching quantifier in each member of $\mathrm{sub}(F)$ by free variables or constant symbols (i.e., we obtain actual subformulas corresponding to the semi-formulas).

4 Completeness of LB

The main result of this paper is the following theorem.

Theorem 4.1. **LB** *is complete for* **TB***: Every finite* **TB***-valid sequent S has a cut-free* **LB***-proof from atomic axioms.*

The proof requires some additional definitions and technical lemmata. In order to emphasize the main lines of the argument we give a rough sketch of the proof in advance:

The proof uses a variant of Schütte's method of reduction trees as modified for intuitionistic logic with Kripke semantics by [14, Ch. 1, § 8]. It proceeds by exhibiting a countermodel for any given unprovable sequent in the following way: Let us assume that $S\colon \Gamma \rightarrow \Delta$ is unprovable. We first generate a reduction tree

by reverse application of all the rules of **LB** *except (nex) and (nec)*. This tree contains a branch $B(S)$ consisting of unprovable sequents only. We form the union sequent of $B(S)$ and extract from it the subsequent $\bigcirc\Gamma_B \rightarrow \bigcirc\Delta_B$ consisting of all formulas of the form $\bigcirc A$. By reverse application of (nex) we arrive at the sequent $\Gamma_B \rightarrow \Delta_B$, which is unprovable as well. For this sequent, we repeat the construction of a reduction tree. By iterating this procedure we obtain an infinite sequence N of reduction branches, all of them containing unprovable sequents only. Now we take the union sequent of the sequence of all sequents contained in these branches. In turn, we extract a subsequent $\Box\Gamma_N \rightarrow \Box\Delta_N$ consisting of all formulas of the form $\Box A$, but with the following restriction: $\Box A$ is in $\Box\Delta_N$ only if it occurs in infinitely many reduction branches of the sequence N. If $\Box\Delta_N$ is the empty sequence we have completed our construction and obtain a countermodel, otherwise we continue as follows: By construction, $\Box\Gamma_N \rightarrow \Box\Delta_N$ is unprovable, and so is any subsequent of the form $\Box\Gamma_N \rightarrow \Box A$, for any formula $\Box A$ occurring in $\Box\Delta_N$. We then repeat the whole construction for all sequents $\Box\Gamma_N \rightarrow A$ (note that these are unprovable too). This gives us a possibly infinite and possibly infinitary tree of infinite chains of reduction branches containing unprovable sequents only. This tree is contained in $\mathcal{B}$ and we obtain from it a countermodel for the original sequent $S{:}\,\Gamma \rightarrow \Delta$.

Definition 4.2. The *reduction tree* $R(S)$ of a sequent $S{:}\,\Gamma \rightarrow \Delta$ is an infinite, infinitary tree (i.e., the nodes may be of infinite degree) s.t. the set of nodes is a set of (occurrences of) sequents. $R(S)$ is defined in stages as follows:

Stage 0: R_0 consists of S alone (S is the root node of $R(S)$).

Stage $k+1$: Suppose that the reduction tree R_k has already been constructed. In order to construct R_{k+1} we need some additional terminology. Let B be a branch (i.e., a maximal path starting from the root) in R_k. We call B *closed* if it is finite and its end sequent $\Pi \rightarrow \Lambda$ contains an atomic formula which is contained in both Π and Λ; otherwise B is called *open*. The free variables occurring in the sequents of a branch B are called the *available variables* of B; if there are none, pick any free variable and call it available. Note that our sequents may be infinite and thus there may be infinitely many free variables even on a finite branch. Since in the definition of R_{k+1} there may be nodes of uncountable degree we need an uncountable supply of free variables (note that this poses no problem, as $R(S)$ is a semantic structure and not an actual proof tree). Constants occurring in S (by construction no new constants are generated) are treated like available variables. The reduction applies to any top sequent (i.e., leaf sequent) of R_k. The method is a generalization of the first-order case (which applies to $\neg$, $\wedge$, $\vee$, $\supset$, $\forall$, $\exists$) by extending it to the case of $\Box$. For the time being, we postpone treatment of $\bigcirc$. Concerning formulas with outermost logical symbols among $\neg$, $\wedge$, $\vee$, $\supset$, $\forall$, $\exists$ we proceed as in [14]. We present only some typical cases and omit most of the details. The principle is that of decomposing formulas according to their outermost logical symbol. In order to avoid reducing formulas more often than needed, we mark formulas as "treated" once the reduction has been applied to them.

In the first step the root sequent contains only unmarked formulas. So let us assume that $S'{:}\,\Pi \rightarrow \Lambda$ is a leaf node of a branch B in R_k.

(a1) Outermost logical symbol $\wedge$ (left reduction)
Let $(A_i \wedge B_i)_{i\in\alpha}$ be the subsequence of Π consisting of unmarked formulas with outermost logical symbol $\wedge$. Then we define $S''{:}\,(A_i, B_i)_{i\in\alpha}, \Pi \rightarrow \Lambda$ and add the edge (S', S'') to R_k. Mark the thus reduced formulas $(A_i \wedge B_i)_{i\in\alpha}$ in S''.

(a2) Outermost logical symbol $\wedge$ (right reduction):
Here let $(A_i \wedge B_i)_{i\in\alpha}$ be the subsequence of Λ consisting of all unmarked formulas with outermost logical symbol $\wedge$. Let $\lambda(S') = \{\Pi \rightarrow \Lambda, (C_i)_{i\in\alpha} \mid C_i = A_i \text{ or } C_i = B_i\}$. For every $S'' \in \lambda(S')$ add S'' and the edge (S', S'') to R_k and

mark the formulas $(A_i \wedge B_i)_{i\in\alpha}$ therein. Note that the node S' has an uncountable degree in the new tree R_{k+1} if α is an infinite ordinal.

We skip the definition for the other propositional connectives and refer the reader to [14].

(b1) Outermost logical symbol $\forall$ (left reduction):
Let $\big((\forall x_i)A_i(x_i)\big)_{i\in\alpha}$ be the subsequence of Π consisting of all unmarked formulas with outermost logical symbol $\forall$. Let $(a_i)_{i\in\beta}$ be a sequence consisting of all free variables on the branch B from S to S'. Note that all sequents are countable and the length of B is finite; thus β is a countable ordinal again. We define $S'' : \big(\big(A_i(a_j)\big)_{j\in\beta}\big)_{i\in\alpha}, \Pi \rightarrow \Lambda$ and add S'' and the edge (S', S'') to R_k.

(b2) Outermost logical symbol $\forall$ (right reduction):
Let $\big((\forall x_i)A_i(x_i)\big)_{i\in\alpha}$ be the subsequence of Λ consisting of all unmarked formulas with outermost logical symbol $\forall$. Create a sequence $(b_i)_{i\in\alpha}$ of free variables which do not occur in any sequent constructed so far. We define $S'' : \Pi \rightarrow \Lambda, \big(A_i(b_i)\big)_{i\in\alpha}$ and add S'' and the edge (S', S'') to R_k. Mark the formulas $(\forall x_i)A(x_i)$ for $i \in \alpha$ in the consequent of the new sequent S''.

The construction for $\exists$ is completely symmetric to the case of $\forall$.

(c1) Outermost logical symbol $\Box$ (left reduction):
Let $(\Box A_i)_{i\in\alpha}$ be the subsequence of all formulas in Π which are unmarked and have $\Box$ as outermost symbol. Let $S'' : (A_i, \bigcirc\Box A_i)_{i\in\alpha}, \Pi \rightarrow \Lambda$ and add S'' and the the edge (S', S'') to R_k. Mark all formulas $\Box A_i$ for $i \in \alpha$ in Π of S''. Note that, like in the other cases, the form of S'' is obtained by applying $\Box$:left "backwards."

(c2) Outermost logical symbol $\Box$ (right reduction):
Let $(\Box A_i)_{i\in\alpha}$ be the subsequence of all formulas in Λ which are unmarked and have $\Box$ as outermost logical symbol. Let $\mu(S') = \{\Pi \rightarrow \Lambda, (C_i)_{i\in\alpha} \mid C_i = A_i \text{ or } C_i = \bigcirc\Box A_i\}$ and add S'' and the edge (S', S'') to R_k for every $S'' \in \mu(S')$. Note that, like in case (a2) above, the degree of the node S' in R_{k+1} is uncountable provided α is infinite. Finally, mark the formulas $\Box A_i$ for $i \in \alpha$ in Λ of S''.

As already indicated we do not introduce reduction rules for $\bigcirc$ here. Suppose none of the reduction rules for $\neg, \wedge, \vee, \supset, \forall, \exists$ or $\Box$ apply and the branch B (from S to S') is open. Then we simply add a copy S'_0 of S' and the edge (S', S'_0) to R_k. (Note that we work with *occurrences* of sequents, not merely sequents. The reduction therefore indeed produces a tree, and not a cyclic graph.)

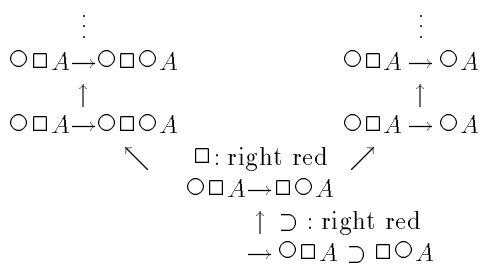
In order to guarantee that all formulas in the sequents are eventually processed, we postulate a "clockwise" order in reducing $\neg$, $\wedge$, $\vee$, $\supset$, $\forall$, $\exists$, $\Box$. If we take the order as given, we reduce $\neg$ first, then $\wedge$, etc. After having reduced $\Box$ on all sequents we start with $\neg$ again. Since reduced formulas in ($\forall$:right) (and ($\exists$:left) reductions are *not* marked, these formulas can be reduced infinitely often. Without postulating such a clockwise order, open branches would not define countermodels in general.

By the above construction we obtain an (infinite) sequence of trees which is monotonic. Thus, by taking the union over the sets of vertices and edges, we obtain the limit tree R_ω. R_ω is precisely the tree $R(S)$ we intended to construct.

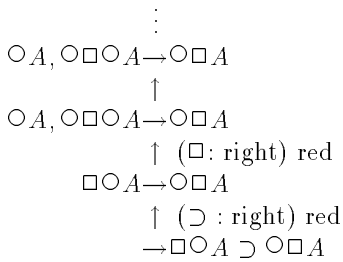
Note that our construction, if applied to formulas neither containing $\bigcirc$ nor $\Box$, yields the familiar construction of a counterexample in classical predicate logic. Indeed, if A is such a formula which is not valid (in the standard first-order semantics) we obtain an infinite open branch B representing a counterexample. Our construction, however, is not completed so far. In fact, we may obtain open branches

in $R(S)$ even for sequents valid in **TB**. Note that in the construction of $R(S)$ itself we cannot obtain infinite sequents provided the root sequent is finite. But in some further constructions we will obtain infinite sequents out of infinite branches and apply the method of reduction trees to these sequents as well. Let us illustrate the construction of $R(S)$ by a simple example (cf. also Example 3.6):

Example 4.3. Let S be $\rightarrow \bigcirc\Box A \supset \Box\bigcirc A$. The tree $R(S)$ is given below:



$R(S)$ possesses two open infinite branches. As $\bigcirc\Box A \supset \Box\bigcirc A$ is **TB**-valid, these open branches do not represent counterexamples. On the other hand we will prove that for unprovable sequents there are always branches in the reduction tree containing unprovable sequents only. Take for example $S' : \rightarrow \Box\bigcirc A \supset \bigcirc\Box A$. We already know that S' is not **TB**-valid. $R(S')$ is the following tree consisting of one infinite branch only:



It is easy to verify that the branch contains only sequents which are not valid in **TB**. Clearly, by soundness of **LB**, these sequents are all unprovable.

In the case of **LK**, finite sequents, and an unprovable end-sequent S we obtain a tree $R(S)$ with the following property: If S' is an unprovable sequent in $R(S)$, then there is a successor of S' in $R(S)$ which is also unprovable. As $R(S)$ must be infinite and its node degree finite, there is an infinite branch by König's Lemma. This infinite branch consist of unprovable sequents only and represents a counterexample. This argument obviously yields the completeness of **LK**.

In the case of infinite sequents S there may be nodes in $R(S)$ of uncountable degree. This phenomenon occurs if, in a sequent S' occurring in $R(S)$, we have infinitely many formulas containing an outermost logical operator with a binary reduction rule (e.g., ($\wedge$:right) or ($\Box$:right)). It is, however, still possible to prove the existence of an infinite branch containing unprovable sequents. For this purpose we will use a generalization of König's Lemma due to Takeuti [14].

Definition 4.4. Let α be a set and $\{W_i\}_{i\in\alpha}$ be a family of sets indexed by α. If $f \in \prod_{i\in\alpha_1} W_i$ and $\alpha_1 \subseteq \alpha$ then f is called a *partial function (over α)* with domain $\mathrm{dom}\, f = \alpha_1$. If $\mathrm{dom}\, f = \alpha$ then f is called *total*. If f and g are partial functions s.t. $\mathrm{dom}\, f = D_0 \subseteq \mathrm{dom}\, g$ and $f(x) = g(x)$ for all $x \in D_0$, then we call g an *extension of f* and write $f \prec g$ and $f = g \restriction D_0$.

Theorem 4.5. (Takeuti [14], p. 51f) *Let α be a set and $\{W_i\}_{i\in\alpha}$ be a family of finite sets. Let P be a property of partial functions over α s.t.*

(1) $P(f)$ holds iff there exists a finite subset $N \subseteq \alpha$, s.t. $P(f \restriction N)$ holds.
(2) $P(f)$ holds for every total f.

Then there exists a finite subset $N_0 \subseteq \alpha$ s.t. $P(f)$ holds for every f with $N_0 \subseteq \operatorname{dom} f$.

Lemma 4.6. *Let $R(S)$ be the reduction tree of a (possibly infinite) unprovable sequent S. Then $R(S)$ has a branch $B(S)$ containing unprovable sequents only. Such a branch is called a reduction branch of $R(S)$.*

Proof. We have to show that, in $R(S)$, a sequent S' is unprovable iff there exists a successor S'' of S' s.t. S'' is unprovable. Equivalently:

(*) If all successors of a sequent node S' are provable then S' itself is provable.

Using transfinite induction on trees (by ordering trees according to the standard subset relation) we derive from (*): If S is unprovable, then there exists an infinite reduction branch in $R(S)$ (every maximal finite branch must end in a provable sequent). Thus, by (*), every path leading to an unprovable sequent can be extended). Note again that the degree of some nodes in $R(S)$ may be uncountable, but branches in $R(S)$ are always countable! Thus it remains to prove (*):

Case 1: S' is of degree 1: The rule used for the reduction of S' has only one premise, e.g., ($\vee$:right), ($\exists$:left), ($\Box$:left). Then S' has only one successor S''. Let us assume that S'' is provable. By definition of provability (of infinite sequents) there exists a finite subsequent S_0'' of S'' which is provable too. Now let $B_1, \ldots, B_m$ be the formulas in S_0'' obtained by reduction using some rule (let us call it ρ). Then, by repeated application of ρ on the B_i combined with contractions and exchanges, we obtain a finite subsequent S_0' of S' which is provable too; the proof of S_0'' can be easily extended to a proof of S_0'.

Case 2: S' is of degree > 1 (possibly of uncountable degree): The rule corresponding the reduction of this node must be binary , e.g., ($\vee$:left), ($\Box$:right). By definition of a reduction tree the successors of S' must be of the form

$$\Pi \to \Lambda, (C_{j_i,i})_{i\in\alpha} \quad \text{or} \quad (C_{j_i,i})_{i\in\alpha}, \Pi \to \Lambda$$

where for all $i \in \alpha$ we have $j_i \in \{0,1\}$ depending on which (of the two) subformulas occurs on position i. Moreover, for every sequence $(j_i)_{i\in\alpha}$ there exists a successor corresponding to this sequence. In the argument to follow it does not matter whether the rule under consideration is a left or a right rule. Thus, we restrict attention to the case where ρ is a right rule and the reduced sequent is $\Pi \to \Lambda, (C_{j_i,i})_{i\in\alpha}$.

Now let $W_i = \{0,1\}$ for every $i \in \alpha$ and f denote functions in $\prod_{i\in\alpha} W_i$ ($= \{0,1\}^\alpha$). Let us assume that all successors of S' are provable. Then to every successor S'' of S' there corresponds exactly one $f \in \{0,1\}^\alpha$. Thus if S'' corresponds to f we write $S'' = S''[f]$. Since $S''[f]$ is provable there exists a finite subsequent $S_0''[f]$ of $S''[f]$ which is provable too. This means, for every total f (see Definition 4.4) there is a finite subsequent $S_0''[f]$ of $S''[f]$ s.t. $S_0''[f]$ is provable. Hence, for $S' = \Pi \to \Lambda, (C_{j_i,i})_{i\in\alpha}$ and every $f \in \{0,1\}^\alpha$ we obtain a finite provable subsequent $S_0''[f]$ of the form

$$\Pi^f \to \Lambda^f, (C_{j_i,i})_{i\in\alpha_1}$$

where α_1 is a finite subset of α.

Let $\alpha_1 = \{i_1, \ldots, i_n\}$ be an arbitrary finite subset of α and let $f \in \{0,1\}^{\alpha_1}$. Then we call the finite sequence of formulas

$$\left(C_{f(i_1),i_1}, \ldots, C_{f(i_n),i_n}\right)$$

selected for f if there are finite subsequences Π^f, Λ^f of Π, Λ, respectively, s.t. $\Pi^f \to \Lambda^f, (C_{f(i),i})_{i\in\alpha_1}$ is provable. By the explications above, there are such subsequences for every f. Hence, there exist selected sequences for every total f.

In order to apply Takeuti's theorem we have to define a property P of partial functions over R. We choose:

$$P(f) \iff (\exists n \in \omega)(\exists i_1, \ldots, i_n \in \operatorname{dom} f)(C_{f(i_1),i_1}, \ldots, C_{f(i_n),i_n}) \text{ is selected}$$

$P(f)$ obviously satisfies both conditions (1) and (2) of Theorem 4.5. Thus, Takeuti's theorem applies and there exists a finite set $\alpha_0 = \{r_1, \ldots, r_\ell\} \subseteq \alpha$ s.t. if $\alpha_0 \subseteq \operatorname{dom} f$ then $P(f)$ holds. We define

$$F = \{f \mid \operatorname{dom} f = \alpha_0\}$$

Then F is a finite set and $P(f)$ holds for all $f \in F$. But this means that for every $f \in F$ there exists $s_1, \ldots, s_k \in R_0$ $(= \operatorname{dom} f)$ s.t. $(C_{f(s_1),s_1}, \ldots, C_{f(s_k),s_k})$ is selected, i.e., there exists a finite subsequence Π^f, Λ^f of Π, Λ s.t.

$$\Pi^f \to \Lambda^f, C_{f(s_1),s_1}, \ldots, C_{f(s_k),s_k}$$

is provable. Now the set $\{0,1\}^{\alpha_0}$ is isomorphic to $\{0,1\}^{\{1,\ldots,\ell\}}$, the set of all binary sequences of length ℓ. Thus for every such binary sequence $\beta = (i_1, \ldots, i_\ell)$ there exist finite subsequences Π^β, Λ^β of Π, Λ s.t.

$$S^\beta\colon \Pi^\beta \to \Lambda^\beta, C_{i_1,r_1}, \ldots, C_{i_\ell,r_\ell}$$

is provable. We see that the $C_{i_1,r_1}, \ldots, C_{i_\ell,r_\ell}$ for $(i_1, \ldots, i_\ell) \in \{0,1\}^{\{1,\ldots,\ell\}}$ $(= B_\ell)$ are exactly the reduction formulas obtained from the reduction of the finite subsequent $S'_0\colon \Pi_0 \to \Lambda_0, C_{r_1}, \ldots, C_{r_\ell}$ where Π_0 is the union sequence of $(\Pi^\beta)_{\beta\in B_\ell}$ and Λ_0 is the union sequence of $(\Lambda^\beta)_{\beta\in B_\ell}$. By repeated application of the binary rule ρ under consideration we can derive S'_0 from the sequents S^β. Together with the respective **LB**-proofs of the S^β we obtain a proof of S'_0. But S'_0 is a finite subsequence of S' and thus S' is provable. □

Note that in order to prove lemma 4.6 we made use of the compactness of the provability concept (which holds by definition). We did not use (semantic) compactness of the logic **TB** and do not even claim that **TB** is indeed compact.

So far we know that for unprovable sequents S, there must be an infinite branch containing only unprovable sequents (i.e., a reduction branch) in $R(S)$. In our next step we "pass" the ordinal ω in our construction and obtain infinite sequents out of finite ones (note that, if S is finite, then $R(S)$ contains only finite sequents). The basic idea is to construct (infinite) unprovable sequents out of reduction branches and iterate this procedure infinitely often.

Definition 4.7. Let S be an unprovable sequent and B be a reduction branch in $R(S)$. Let S' be the union sequent of B (see Definition 3.3) and $S'_0\colon (\bigcirc A_i)_{i\in\alpha} \to (\bigcirc B_j)_{j\in\beta}$ be the subsequent of S' consisting of all formulas in S' with outermost logical symbol $\bigcirc$. Let S''_0 be $(A_i)_{i\in\alpha} \to (B_j)_{j\in\beta}$ (This is the sequent S'_0 "stripped" of its outermost $\bigcirc$'s). Then S''_0 is called the *successor of* S *w.r.t.* B.

Lemma 4.8. *Let S be an unprovable sequent and B be a reduction branch in $R(S)$ and let S' be the successor of S w.r.t. B. Then S' is unprovable.*

Remark 4.9. By lemma 4.6 we know that $R(S)$ must have a reduction branch; thus the assumption of the lemma can always be fulfilled and S' exists.

Proof. Let S' be $(A_i)_{i\in\alpha} \rightarrow (B_j)_{j\in\beta}$. Assume, by way of contradiction, that S' is provable. By definition of provability, there is a finite subsequent $S'': A_{i_1}, \ldots, A_{i_k} \rightarrow B_{j_1}, \ldots, B_{j_\ell}$ of S' which is **LB**-provable. But from S'' we can derive (in one step), using (nex), the sequent $S_1'': \bigcirc A_{i_1}, \ldots, \bigcirc A_{i_k} \rightarrow \bigcirc B_{j_1}, \ldots, \bigcirc B_{j_\ell}$. Since S' is the successor of S w.r.t. B, by Definition 4.7, S_1'' is a finite subsequent of the union sequent $U(B)$ of B. Thus if $B = (S_i)_{i\in\omega}$ there exists a finite initial segment $B' = (S_1, \ldots, S_n)$ of B, with $S_1 = S$ and so that the union sequent $U(B')$ of B' contains S_1''. Let $\mathrm{left}(\Pi \rightarrow \Lambda)$ denote the set of all formulas in Π, and $\mathrm{right}(\Pi \rightarrow \Lambda)$ denote the set of all formulas in Λ. By construction of $R(S)$ we have that $\mathrm{left}(S_i) \subseteq \mathrm{left}(S_j)$ and $\mathrm{right}(S_i) \subseteq \mathrm{right}(S_j)$ for $1 \leq i \leq j \leq n$. Hence, $\mathrm{left}(U(B')) = \mathrm{left}(S_n)$ and $\mathrm{right}(U(B')) = \mathrm{right}(S_n)$. In other words, S_1'' is a finite subsequent of S_n. S_1'' is provable and thus S_n is provable, too. But this is impossible because S is a reduction branch. Hence, S' must be unprovable. $\square$

Definition 4.10. Let S_1 be an unprovable sequent. A *next-time sequence* is an infinite sequence of reduction branches $(B_i)_{i\in\omega}$ s.t. B_1 is a reduction branch of S_1, and for every $i \geq 2$, B_i is a reduction branch of a successor S_i of S_{i-1} w.r.t. B_{i-1}. All variables occurring in B_i are available for the construction of B_{i+1} (i.e., for the reductions $\forall$:left and $\exists$:right).

Note that, by Lemma 4.8, next-time sequences exist for all unprovable sequents. This is easily seen by induction.

Example 4.11. We construct a next-time sequence $N(S_1)$ corresponding to the sequent $S: \Box\bigcirc A \rightarrow \bigcirc\Box A$. The following sequence is a reduction branch in $R(S_1)$:

$$B_1: S_1; \bigcirc A, \bigcirc\Box\bigcirc A, \Box\bigcirc A \rightarrow \bigcirc\Box A; \bigcirc A, \bigcirc\Box\bigcirc A, \Box\bigcirc A \rightarrow \bigcirc\Box A; \ldots$$

The union sequent of B_1 is $\bigcirc A, \bigcirc\Box\bigcirc A, \Box\bigcirc A \rightarrow \bigcirc\Box A$. Therefore the successor of S_1 w.r.t. B_1 is

$$S_2: A, \Box\bigcirc A \rightarrow \Box A$$

For S_2 we obtain a reduction branch of the form

$$B_2: S_2; A, \Box\bigcirc A \rightarrow \bigcirc\Box A; \bigcirc A, \bigcirc\Box\bigcirc A, A, \Box\bigcirc A \rightarrow \bigcirc\Box A; \ldots$$

with the union sequent $\bigcirc A, \bigcirc\Box\bigcirc A, A, \Box\bigcirc A \rightarrow \bigcirc\Box A$. The successor of S_2 w.r.t. B_2 is

$$S_3 = S_2 = A, \Box\bigcirc A \rightarrow \Box A$$

In general, $S_i = S_2 = A, \Box\bigcirc A \rightarrow \Box A$ and $B_i = B_2$ for $i \geq 3$.

The sequents in a next-time sequence represent necessary conditions for a sequent S to be true: If S is true at time point 0, then S_1 is true at point 1, S_2 at 2, etc. But these conditions are not sufficient. Let us look at the sequent $S_1: \Box\bigcirc A \rightarrow \bigcirc\Box A$. We know that S_1 is not **TB**-valid. Let us assume that S_1 is true at time point 1. Then the sequent $S_2: A, \Box\bigcirc A \rightarrow \Box A$ is true at time point 2 and at time k we would have $A, \Box\bigcirc A \rightarrow \Box A$ being true. According to our semantics there is a counterexample to the sequent S_2 at every time point $k \in \omega$. But recall that at time point ω we may set A to false. Note that ω is not a successor ordinal. Thus in order to construct counterexamples to sequents we have to "jump" across time gaps; this jump will be performed via reverse application of the necessitation rule.

Definition 4.12. Let S be an unprovable sequent. A *gapjump tree* $G(S)$ for S is a tree with nodes consisting of next-time sequences satisfying the following conditions:

(1) The root of $G(S)$ is a next-time sequence of S.

(2) Let N be a next-time sequence in $G(S)$ corresponding to a sequent S'. Then (N, N_i), $i \in \alpha$, are edges in $G(S)$ if the N_i are constructed as follows:
Let $N = (B_i)_{i\in\beta}$ and $\Pi' \to \Lambda'$ be the union sequent of N (i.e. $\Pi' \to \Lambda'$ is the union sequent of the union sequents of the B_i). Let $\Box\Pi_N$ be the subsequence of all formulas in Π' with outermost logical symbol $\Box$. $\Box\Lambda_N$ is a subsequence of Λ' obtained in the following way: delete all formulas in Λ' except formulas of the form $\Box A$, where $\Box A$ occurs in the right hand sides of infinitely many successor sequents in N. Thus we obtain a sequent of the form

$$\Box\Pi_N \to \Box\Lambda_N = \Box\Pi_N \to (\Box A_i)_{i\in\alpha}$$

and define a next-time sequence N_i for every $\Box\Pi_N \to A_i$ $(i \in \alpha)$, provided $\Box\Pi_N \to A_i$ is unprovable.
If $\Box\Lambda_N$ is empty then N is a leaf in $G(S)$.

In the definition of the next-time sequence N_i all free variables available for the construction of N are available for the construction of N_i too (for the $\forall$:left and $\exists$:right reduction in the reduction branches).

If $\Box\Lambda_N$ is empty then, according to definition 4.12, the node corresponding to N must be a leaf. But even if $\Box\Lambda_N$ is nonempty it might be the case that the sequents $\Box\Pi_N \to A_i$ are provable and thus do not define new next-time sequences. We will see in Lemma 4.14 that such a case cannot occur.

Example 4.13. We construct a gapjump tree with root $N(S_1)$, where $N(S_1)$ is the next-time sequence of Example 4.11. For $N = (B_i)_{i\geq 1}$ we had $B_i = B_2$ for $i \geq 2$ and

$$B_2\colon A, \Box\bigcirc A \to \Box A; A, \Box\bigcirc A \to \bigcirc\Box A; \bigcirc A, \bigcirc\Box\bigcirc A, A, \Box\bigcirc A \to \bigcirc\Box A; \ldots$$

B_1 starts with $\Box\bigcirc A \to \bigcirc\Box A$, so we obtain as union sequent of N:

$$\Box\bigcirc A, A, \bigcirc A, \bigcirc\Box\bigcirc A, \Box\bigcirc A, \ldots \to \Box A, \bigcirc\Box A, \ldots$$

where formulas (if we do not use contraction) may be repeated infinitely many times. Note that in all the successors S_{i+1} w.r.t. B_i we have $S_{i+1} = A, \Box\bigcirc A \to \Box A$ for $i \in \omega$ and thus $\Box A$ occurs infinitely often on the right side. Hence, $\Box\Pi_N \to \Box\Lambda_N = \Box\bigcirc A \to \Box A$. We have to consider only the single sequent $S'\colon \Box\bigcirc A \to A$. The only edge leaving $N(S_1)$ in $G(S_1)$ is $(N(S_1), N(S'))$. A next-time sequence $N(S')$ for S' is easily obtained. We construct the reduction tree for S' and find a reduction branch B_0 with successor sequent $S''\colon A, \Box\bigcirc A \to$. It is immediately clear that the successor sequents will be repeated infinitely often. A next-time sequence for S' is $N(S') = (B_i)_{i\in\omega}$ where

$$\begin{aligned} B_0 &= \Box\bigcirc A \to A; \bigcirc A, \bigcirc\Box\bigcirc A, \Box\bigcirc A \to A; \ldots \\ B_1 &= A, \Box\bigcirc A \to; \bigcirc A, \bigcirc\Box A, A, \Box\bigcirc A \to; \ldots \\ B_i &= B_1 \quad \text{for all } i \geq 1 \end{aligned}$$

$N(S')$ is a leaf node of the gapjump tree, since there is no formula of the form $\Box A$ occurring in infinitely many successor sequents on the right hand side (in fact, there are no such formulas at all).

In defining the gapjump tree we have constructed a sequent of the form $S'\colon \Box\Pi_N \to \Box\Lambda_N$ where $\Box\Lambda_N$ contains only formulas appearing infinitely many times. S' can be "extracted" from the next-time sequence N, which is also a node in the tree. If the consequent of S' is empty then clearly N is a leaf in the gapjump tree. Otherwise we obtain sequents of the form $\Box\Pi_N \to A_i$, where $\Box A_i$ occurs in $\Box\Lambda_N$. We call S' the $\Box$-*extract* of N and every sequent $S''\colon \Box\Pi_N \to A$ for $\Box A$ in $\Box\Lambda_N$ a *right reduct* of S'. (The term "right reduct" should not be confused with the ($\Box$:right) reduction of S', which has a different form.)

Lemma 4.14. *Let N be a node in a gapjump tree $G(S')$ for unprovable S' and S be the $\Box$-extract of N. If the consequent of S is not empty then every right reduct of S is unprovable.*

Remark 4.15. A consequence of this lemma is that every right reduct defines a next-time sequence and thus a successor node of N.

Proof. Let $S\colon \Box\Pi_N \rightarrow \Box\Lambda_N$ be the $\Box$-extract of N s.t. $\Box\Lambda_N$ is not empty, and let $\Box A$ be a formula in $\Box\Lambda_N$.

Assume, by way of contradiction, that $S'\colon \Box\Pi_N \rightarrow A$ is provable. By definition of provability there is a finite subsequent $S''\colon \Box\Pi'_N \rightarrow \Lambda'_N$ of S' s.t. S'' is provable. Λ'_N is either empty or A alone.

If Λ'_N is empty then S'' is a subsequent of the union sequent of N (recall that N is a next-time sequence). We show that there exists a successor sequent S_i of a branch B_{i-1} in N s.t. $\Box\Pi'_N$ is a subsequence of the antecedent of S_i: Let $\Box C$ be a formula in $\Box\Pi'_N$. $\Box C$ occurs in some sequent S_C in a reduction branch B in N. By definition of a reduction branch, $\bigcirc\Box C$ must occur in almost all descendents of S_C and thus also in the union sequent (($\Box$:left) reduction). By definition of a successor sequent, the successor w.r.t. B must contain $\Box C$ in the antecedent. Moreover, $\Box C$ must occur in all further successor sequents in N. As $\Box\Pi'_N$ is a finite sequence there must be a successor sequent S_j of a reduction branch B_{j-1} s.t. $\Box\Pi'_N$ is subsequence of the antecedent of S_j. But then S_j would be provable, which contradicts Lemma 4.8.

If $\Lambda'_N = A$ then, like in the case where Λ'_N is empty above, we obtain a successor sequent S_i in N s.t. $\Box\Pi'_N$ is a subsequence of the antecedent of S_i. By definition of a $\Box$-extract, the formula $\Box A$ must occur in infinitely many S_j's. Observe that $\Box\Pi'_N$ is a subsequent of the antecedents of all S_j for $j \geq i$. Therefore, there must be a k s.t. $\Box\Pi'_N \rightarrow \Box A$ is a subsequent of S_k. If $S''\colon \Box\Pi'_N \rightarrow A$ is, as we assumed, provable, then so is $S'''\colon \Box\Pi'_N \rightarrow \Box A$ by application of the rule (nec). Since S''' is a subsequent of S_k, S_k were provable too, again contradicting Lemma 4.8. $\Box$

Corollary 4.16. *If N is a leaf in a gapjump tree then the consequent of the $\Box$-extract of N is empty.*

Proof. Assume that the $\Box$-extract S of N were not empty. Then S would have right reducts. Any such right reduct S' is unprovable by Lemma 4.14. But then there would be a next-time sequence $N(S')$ for S' and and an edge $(N, N(S'))$. $\Box$

Proof of the Completeness Theorem 4.1.

We have to show that finite unprovable sequents are not valid. More precisely, if S is a finite sequent which is unprovable in **LB**, then there exists a **TB**-interpretation K for S which falsifies S.

Let $G(S)$ be a gapjump tree for S. We define the following **TB**-interpretation $\mathbf{K} = \langle T, \{D_B\}_{B\in T}, \{\mathbf{S}_B\}_{B\in T}\rangle$ where

(1) T is the set of all occurrences of reduction branches (in the next-time sequences) in $G(S)$. For the remaining part of this proof we use the letter B for occurrences of branches and $\bar{B}$ for the branch corresponding to B. Moreover we introduce the following partial order:
If B and B' are two occurrences within the same next-time sequence $(\beta_i)_{i\in\omega}$ then there are $i, j \in \omega$ s.t. $\bar{B} = \beta_i$ and $\bar{B'} = \beta_j$. We set $B < B'$ if $i < j$ and $B' < B$ if $j < i$. Clearly $B = B'$ for $i = j$.
If B, B' are occurrences in different next-time sequences N, N' (which are nodes of $G(S)$) then $B < B'$ iff there is a path from N to N' in $G(S)$. Evidently, the order type of $<$ is in $\mathcal{T}$.

(2) For every $B \in T$, D_B is the set of all free variables $V(B)$ occurring in $\bar{B}$. Note that by the definitions of a next-time sequence (4.10) and of a gapjump tree (4.12), $V(B) \subseteq V(B')$ if $B < B'$. Thus we obtain $D_B \subseteq D_{B'}$ for $B < B'$ (second condition of Definition 2.2)

(3) Definition of the evaluation function $\mathbf{S}_B$ for $B \in T$:
Set $\mathbf{S}_B(a) = a$ for $a \in D_B$ (note that elements of D_B are available as constant symbols in the extended language). If A is an atomic formula, we define $\mathbf{S}_B(A) = \top$ if A occurs in the antecedent of a sequent occurring in B and $= \bot$ otherwise.

We have to show that this truth assignment is consistent, i.e., that it is impossible that an atomic formula A occurs in an antecedent *and* in a consequent of a sequent in B. Thus let $B = (S_i)_{i \in \omega}$. By construction of a reduction tree we have $\mathrm{sub}^*(S_i) \subseteq \mathrm{sub}^*(S_j)$ for $i < j$ (see definition 3.8). In particular, all atomic formulas occurring in the antecedent (consequent) of S_i also occur in the antecedent (consequent) of S_j. Thus if A occurs in the antecedent of S_i and in the consequent of S_j it must occur in both sides in S_k for $k \geq \max(i, j)$. This, however, contradicts the definition of reduction branches in a reduction tree (Definition 4.2), as B would be closed. So $\mathbf{S}_B$ is consistently defined.

It remains to show that $\mathbf{K}$, as defined above, is indeed a countermodel to S. It suffices to show the following: $(*)$ If F is a formula occurring in the antecedent (consequent) of a sequent in a reduction branch $\bar{B}$ (in a next-time sequence N occurring as a node in $G(S)$) then $\mathbf{K}_B(F) = \top$ ($\mathbf{K}_B(F) = \bot$). Then, by Definition 2.2 and by the finiteness of S, (the implication corresponding to) S is falsified in $\mathbf{K}$, as $\mathbf{K}_{B_0}(S) = \bot$ (B_0 being the first reduction branch in the root of $G(S)$). Note that F cannot occur in an antecedent of S and in a consequent of S' for two sequents S, S' in B. The reasons are the same as for atomic formulas described above. We prove $(*)$ by induction on the logical complexity of F.

If F is an atomic formula (logical complexity 0), then $(*)$ follows from $\mathbf{K}_B(F) = \mathbf{S}_B(F)$ and the definition of $\mathbf{S}_B$. Suppose that (*) has been shown for all formulas of logical complexity $\leq n$. Let F be a formula of logical complexity $n + 1$. If the outermost logical symbol is not a temporal operator ($\Box$, $\bigcirc$) then the reduction to the case $\leq n$ follows exactly the classical first-order case (see [14], Ch. 1, § 8). It remains to handle the cases $F \equiv \Box F'$ and $F \equiv \bigcirc F'$ for some formula F'.

(1) $F \equiv \bigcirc F'$:
Let us assume that F occurs in the antecedent (consequent) of a reduction branch B. Because B is a branch in a next-time sequence there is a successor S' w.r.t. B (see Definition 4.7) on which the next reduction branch starts (see Definition 4.10). By definition of successor, F' occurs in the antecedent (consequent) of S', which is the first sequent of the successor branch B'. By the induction hypothesis, $\mathbf{K}_{B'}(F') = \top$ ($\mathbf{K}_{B'}(F') = \bot$). By Definition 2.2, $\mathbf{K}_B(F) = \mathbf{K}_B(\bigcirc F') = \mathbf{K}_{B'}(F')$. As F must occur on the same side as F' we conclude that $(*)$ holds for F.

(2) $F \equiv \Box F'$:
(a) F occurs in the antecedent of a sequent in a reduction branch B in the next-time sequence N:
By the semantics of $\Box$ we have $\mathbf{K}_B(F) = \top$ iff for all B' s.t. $B \leq B'$ it holds that $\mathbf{K}_{B'}(F') = \top$. So let us assume that F occurs in the antecedent of the sequent S_i in B. Then $\bigcirc\Box F'$ must occur in (the antecedent of) a sequent S_j for some $j > i$. By definition of a next-time sequence N, $\Box F'$ must occur in (the antecedent of) the successor of B. By induction, $\Box F'$ occurs in the antecedent of *every* sequent in every reduction branch in this next-time sequence. Hence, $\Box F'$ occurs in the union $\Box\Pi'$ and in the antecedent of the $\Box$-extract of N. By definition of right reducts and the gapjump tree then,

$\Box F'$ also occurs in the antecedents of all (initial) reduction branches in the successor nodes N' of N in $G(S)$. By the same arguments as before, we have that $\Box F'$ occurs in the antecedent of every sequent in every reduction branch $B' > B$. Every reduction branch containing $\Box F'$ in the antecedent of some sequent also contains F' in the antecedent of some sequent (by ($\Box$:left) reduction). Hence, by the induction hypothesis, $\mathbf{K}_{B'}(F') = \top$ for all $B' \geq B$ and therefore $\mathbf{K}_B(F) = \top$.

(b) F occurs in the consequent of a sequent in a reduction branch B in the next-time sequence N:
By definition of ($\Box$:right) reduction , which is binary, there is a sequent in B which either contains (in the consequent) F' or $\bigcirc\Box F'$. In the former case we have immediately, by the induction hypothesis, that $\mathbf{K}_B(F') = \bot$ and hence $\mathbf{K}_B(F) = \bot$. Otherwise, observe that the successor of B contains $\Box F'$ in the consequent. We have two cases: (i) either all reduction branches $> B$ in N contain $\bigcirc\Box F'$, or (ii) some branch B' contains F' in the consequent of some sequent. The former holds if at every ($\Box$:right) reduction of F in N the right premise lies on the reduction branch, the latter if in some reduction the left premise does. Case (ii) is handled as above. For case (i), observe that $\Box F'$ occurs (in the consequent of) every successor sequent of branches $B' > B$ in N. Thus, by definition of the $\Box$-extract $\Box\Pi_N \rightarrow \Box\Lambda_N$ of N, $\Box F'$ belongs to $\Box\Lambda_N$. Then there is some right reduct of N of the form $\Box\Pi_N \rightarrow F'$. By Lemma 4.14 this right reduct is unprovable and thus is the initial sequent of the first reduction branch B' of some successor node N' of N in $G(S)$. By the induction hypothesis, $\mathbf{K}_{B'}(F') = \bot$. Since $B < B'$ the semantics of $\Box$ gives us $\mathbf{K}_B(\Box F') = \bot$.

This concludes the proof of $(*)$ and we have shown that **K** falsifies S. $\Box$

Remark 4.17. If the original sequents may be infinite, in particular, of unbounded logical complexity, then we no longer have a well-founded ordering on the sequents. On the other hand, the reduction steps which yield infinite sequents in the proof keep the logical complexity of formulas occurring in the sequents bounded. Hence, if the starting sequent is of bounded logical complexity (in particular, if it is finite), we have a well-founded order. Otherwise, the induction proof is problematic.

5 TL versus TB

It should be interesting to compare the two logics **TL** and **TB**. A comparison from the viewpoint of expressibility would clarify the possible application of **TB** in a program specification and verification environment. Such an analysis, however, would go beyond the scope of the present article. An analysis from a logical point of view can be given more easily. Here the comparison centers around the induction rule in propositional **TL** (see [7]),

$$\frac{A \rightarrow B \quad A \rightarrow \bigcirc A}{A \rightarrow \Box B}\ \text{ind}$$

and the weaker necessitation rule of **TB**.

Proposition 5.1. *(1) The propositional fragment of* **TB** *is decidable.*
(2) The fragment of **TB** *without* $\bigcirc$ *is equal to* **S4**.
(3) The monadic fragments of **TL** *and* **TB** *are undecidable.*
(4) The fragment of **TB** *without* $\Box$ *is axiomatizable by* **LK** *plus (nex).*
(5) The fragment of **TB** *without* $\Box$ *is equal to the fragment of* **TL** *without* $\Box$.

Proof. (1) $\mathrm{sub}^*(\Gamma \rightarrow \Delta)$ is finite. (2) **LB** without $\bigcirc$ collapses to the sequent calculus for **S4** given in [4]. (3) Follows from the undecidability of monadic modal predicate logic; see below. (4) A cut-free proof can contain ($\Box$:left), ($\Box$:right), or (nec) only if $\Box$ occurs in the end-sequent. (5) By (4), a proof has to be found *before* jumping over the first gap iff one exists. $\Box$

In contrast to (3) above, the monadic fragment of **TB** without $\Box$ (and hence, by (5), the fragment of **TL** without $\Box$) is decidable:

Proposition 5.2. *It is decidable if a monadic temporal formula containing no $\Box$'s is satisfiable.*

Proof. Note that $\bigcirc$ distributes over all propositional connectives. Hence, any formula F containing no $\Box$'s is equivalent to a formula of the form $\bigvee_j K_j$ where

$$K_j = \bigwedge_k E^j_k \wedge \bigwedge_l A^j_l$$
$$E^j_k = \bigcirc^{e_k}(\exists x) \bigwedge_i \bigcirc^{e'_{i,k}} L_{i,k}(x)$$
$$A^j_l = \bigcirc^{a_l}(\forall x) \bigvee_i \bigcirc^{a'_{i,l}} L_{i,l}(x)$$

where $L^i_{j,k}$ is a negated or unnegated atomic formula. F is satisfiable iff K_j is satisfiable for some j. Consider the set $\Gamma(K) = \Gamma_1(K) \cup \Gamma_2(K)$ with

$$\Gamma_1(K) = \{\bigwedge_i \bigcirc^{e_k + e'_{i,k}} L_{i,k}(t_k) \mid k\}$$
$$\Gamma_2(K) = \{\bigvee_i \bigcirc^{a_l + a'_{i,l}} L_{i,l}(t_k) \mid l, k, e_k \leq a_l\}$$

where t_k are constant symbols, and $\bigcirc^v L_{i,k}(t_l)$ is considered as a propositional literal $L^v_{i,k,l}$: K is satisfiable iff $\Gamma(K)$ is satisfiable in classical propositional logic. $\Box$

So already the monadic fragments containing $\Box$ but not $\bigcirc$ are undecidable. It is worth to recapitulate the construction of the proof of Kripke [6]: A binary predicate $P(x, y)$ can be encoded in monadic temporal logic as $P'(x, y) = \Diamond(P_1(x) \wedge P_2(y))$. Let F be a formula in the language of predicate logic, and F' be obtained from it by replacing n-ary predicates $P(x_1, \ldots, x_n)$ by $\Diamond(P_1(x_1) \wedge \ldots \wedge P_n(x_n))$. If F is valid, then F' is too, it being a substitution instance of F. If F is not valid, then we construct a temporal countermodel for F': Let $\mathbf{M}$ be a (first-order) structure in which F is not satisfied. By the Löwenheim-Skolem Theorem, we can assume $\mathbf{M}$ to be countable. We can enumerate all n-tuples of elements of the domain M using a function e. Let T be ω, and $\mathbf{S}_j(P_i) = \{a\}$ iff a is the i-th component of the j-th (in e) n-tuple of M. So $\Diamond(P_1(a_1) \wedge \ldots \wedge P_n(a_n))$ is true in $\langle \omega, \{D_i = M\}_{i \in \omega}, \{\mathbf{S}_i\}_{i \in \omega} \rangle$ iff $\mathbf{M} \models P(a_1, \ldots, a_n)$.

As remarked above, the undecidability of the monadic fragments of **TL** and **TB** follows from the undecidability of dyadic predicate logic and the above construction. We have two immediate consequences: First, the monadic fragment of **TL** (with $\bigcirc$) is not even *axiomatizable*, since we can replace the function symbols 0, $'$, $+$, $\cdot$ by (a unary, a binary, and two ternary) predicate symbols. These predicate symbols can in turn be replaced by temporal constructions of the kind used above; so non-axiomatizability follows from the non-axiomatizability of the full logic (see Section 2). A second interesting consequence is that already the fragment with only one monadic predicate symbol (but including $\bigcirc$) is undecidable: With some adjustment to the construction of the countermodel in the proof above, a binary predicate

can also be encoded by $\Diamond(P(x) \wedge \bigcirc P(y))$. We do not know, however, whether the corresponding fragment of **TL** is still not axiomatizable.

Even without a deep analysis it is obvious that propositional **TL** is decidable by embedding it into the the monadic second order logic of one successor of Büchi [3]. (A decision method based on a similar reduction method as the one used here for **TB** can be found in [2].) For the same reason, the *quantified* propositional variant of **TL** is decidable. We do not know whether quantified propositional **TB** is decidable. Note that even though propositional $\mathbf{TB} - \bigcirc$ equals **S4**, the propositionally quantified logics differ. Hence, the result of Kremer [5, III.1], i.e., that propositional **S4** is recursively isomorphic to second-order logic, is of no help here. We conjecture, however, that quantified propositional **TB** is not axiomatizable as well. In summary, we have the following situation:

	TL	**TB**
propositional	decidable	decidable
monadic w/o $\Box$	equal and decidable	
monadic	not axiomatizable	undecidable
quantified propositional	decidable	not axiomatizable?
full first-order	not axiomatizable	axiomatizable

6 Resolution for TB

A practical consequence of the *cut-free* completeness of **LB** is the ability to construct a resolution calculus. The exact relationship between cut-free proofs in sequent calculus and resolution proofs has been investigated at length by Mints [10, 11]. This relationship is also the starting point for very fruitful investigations into resolution systems and strategies for other non-classical logics, e.g., linear logic (see [15]).

The resolution procedure for **TB** works as follows: The formula F to be proved ($\neg F$ to be refuted) is translated to clause form via translation rules based on the calculus **LB**. The translation is structure preserving, and the literals have the form $(\neg)[\![A]\!](a_1, \ldots, a_n)$, where A is the sub-semi-formula corresponding to this literal, and $a_1, \ldots, a_n$ are free variables or constant symbols. A clause is an expression of the form C, where C is a set of literals. A clause may carry a *variable restriction*, denoted C^a, meaning that a resolution involving C is only allowed if a does not occur in the resulting clause and if a is not substituted into. The rules are the resolution and factoring rules, plus two rules corresponding to the (nec) and (nex) rules. By Lemma 2.5 and replacement of free variables with constant symbols, we can assume that F is closed and does not start with $\Box$ or $\bigcirc$.

Definition 6.1. Let F be a semi-formula, and let $\alpha_1, \ldots, \alpha_n$ be all the constant symbols and bound variables without matching quantifier in order of occurrence. Then the *code of* F is defined as $[\![F]\!](\alpha_1\sigma, \ldots, \alpha_n\sigma)$, where $[\![F]\!]$ is an n-ary predicate symbol, and σ is a canonical renaming, mapping $\alpha_1, \ldots, \alpha_n$ to new free variables.

The *axiom set* $Ax(F)$ is defined as the smallest set satisfying the following: Let $P(\alpha_1, \ldots, \alpha_n)$ and $P(\beta_1, \ldots, \beta_n)$ be two atomic sub-semi-formulas of F with the same predicate symbol. Then the clause $\{\neg[\![P(\bar{\alpha})]\!](\gamma_1, \ldots, \gamma_n), [\![P(\bar{\beta})]\!](\gamma_1, \ldots, \gamma_n)\} \in Ax(F)$, where $\gamma_i = \alpha_i\sigma\vartheta$, with σ the renaming as above and ϑ a most general unifier of $(\alpha_1\sigma, \ldots, \alpha_n\sigma)$ and $(\beta_1\sigma, \ldots, \beta_n\sigma)$.

The *clause translation* $\mathrm{Cl}(F)$ is the following set of clauses: $\mathrm{Cl}(F) = \bigcup\{C_F(A) \mid A \in \mathrm{sub}(F)\} \cup Ax(F) \cup \big\{\{\neg[\![F]\!](\alpha_1\sigma, \ldots, \alpha_n\sigma\}\big\}$, where $C_F(A)$ is given by the following table:

A	occurrence	$C_F(A)$
$\neg B$	pos	$\big\{\{[\![B]\!],[\![\neg B]\!]\}\big\}$
$\neg B$	neg	$\big\{\{\neg[\![B]\!],\neg[\![\neg B]\!]\}\big\}$
$B \wedge C$	pos	$\big\{\{\neg[\![B]\!],\neg[\![C]\!],[\![B \wedge C]\!]\}\big\}$
$B \wedge C$	neg	$\big\{\{[\![B]\!],\neg[\![B \wedge C]\!]\},\{[\![C]\!],\neg[\![B \wedge C]\!]\}\big\}$
$B \vee C$	pos	$\big\{\{\neg[\![B]\!],[\![B \vee C]\!]\},\{\neg[\![C]\!],[\![B \vee C]\!]\}\big\}$
$B \vee C$	neg	$\big\{\{[\![B]\!],[\![C]\!],\neg[\![B \vee C]\!]\}\big\}$
$B \supset C$	pos	$\big\{\{[\![B]\!],[\![B \supset C]\!]\},\{\neg[\![C]\!],[\![B \supset C]\!]\}\big\}$
$B \supset C$	neg	$\big\{\{\neg[\![B]\!],[\![C]\!],\neg[\![B \supset C]\!]\}\big\}$
$(\forall x)B$	pos	$\big\{\{\neg[\![B(x)]\!](a),[\![(\forall x)B(x)]\!]\}^a\big\}$
$(\forall x)B$	neg	$\big\{\{[\![B(x)]\!](a),\neg[\![(\forall x)B(x)]\!]\}\big\}$
$(\exists x)B$	pos	$\big\{\{\neg[\![B]\!](a),[\![(\exists x)B(x)]\!]\}\big\}$
$(\exists x)B$	neg	$\big\{\{[\![B]\!](a),\neg[\![(\exists x)B(x)]\!]\}^a)\big\}$
$\Box B$	pos	$\big\{\{\neg[\![B]\!],\neg[\![\bigcirc\Box B]\!],[\![\Box B]\!]\}\big\}$
$\Box B$	neg	$\big\{\{[\![B]\!],\neg[\![\Box B]\!]\},\{\{[\![\bigcirc\Box B]\!],\neg[\![\Box B]\!]\}\big\}$

Here, a stands for σx in the code for $A(x)$, σ is the same for all literals in a clause in $C_F(A)$, and positive and negative occurrences are defined as usual.

Note that there are no translation rules for formulas with outermost symbol $\bigcirc$, just as there are no introduction rules (without restrictions) for $\bigcirc$ in **LB**. This is clear, since there is no relation between A and $\bigcirc A$ which depends only on A.

Definition 6.2. The *degree* $\deg(A)$ of a semi-formula A is the number of occurrences of logical symbols except $\Box$ and $\bigcirc$ in A. The degree of a clause is

$$\deg(C) = \begin{cases} \infty & \text{if } C = \emptyset \\ \max\{\deg(A) \mid [\![A]\!] \in C \text{ or } \neg[\![A]\!] \in C\} & \text{otherwise} \end{cases}$$

Note that $\max\{\deg(A) \mid A \in (\mathrm{sub}(F) \setminus \{F\})\} < \deg(F)$, since F is assumed to be prefix-free (see the comments above).

The resolution calculus for **TB** consists of the the following rules:

$$\frac{C \cup \{[\![A]\!](a_1,\ldots,a_n),[\![A]\!](b_1,\ldots,b_n)\}}{C\sigma \cup \{[\![A]\!](a_1,\ldots,a_n)\sigma\}}\ \text{fact}$$

$$\frac{C \cup \{\neg[\![A]\!](a_1,\ldots,a_n)\} \quad C' \cup \{[\![A]\!](b_1,\ldots,b_n)\}}{\big(C \setminus \{\neg[\![A]\!](a_1,\ldots,a_n)\}\big)\sigma \cup \big(C' \setminus \{[\![A]\!](b_1,\ldots,b_n)\}\big)\sigma}\ \text{res}$$

where σ is the most general unifier of $(a_1,\ldots,a_n)$ and $(b_1,\ldots,b_n)$ and it is assumed that the resolved clauses are variable disjoint (i.e., by renaming variables). The resolution rule is subject to the following restrictions:

(1) $\deg(C \cup C') \geq \min\big(\deg(C \cup \{\neg[\![A]\!]\}),\deg(C' \cup \{[\![A]\!]\})\big)$
(2) if one of the two resolved clauses is restricted on the variable a, then $\sigma(a) = a$ and a does not occur in $\big(C \setminus \{\neg[\![A]\!](a_1,\ldots,a_n)\}\big)\sigma \cup \big(C' \setminus \{[\![A]\!](b_1,\ldots,b_n)\}\big)\sigma$

$$\frac{\{\neg[\![A_1]\!],\ldots,\neg[\![A_1]\!],[\![B_1]\!],\ldots,[\![B_1]\!]\}^{(a)}}{\{\neg[\![\bigcirc A_1]\!],\ldots,\neg[\![\bigcirc A_1]\!],[\![\bigcirc B_1]\!],\ldots,[\![\bigcirc B_1]\!]\}^{(a)}}\ \text{nex}_r$$

$$\frac{\{\neg[\![\Box A_1]\!],\ldots,\neg[\![\Box A_1]\!],[\![B]\!]\}}{\{\neg[\![\Box A_1]\!],\ldots,\neg[\![\Box A_1]\!],[\![\Box B]\!]\}}\ \text{nec}_r$$

The application of the rules (nex_r) and (nec_r) is restricted so that the resulting literals are still within $\mathrm{sub}(F)$. The calculus, therefore, depends on F; we actually are giving a construction schema for resolution calculi for each F.

The following should be noted about the variable restriction:

Proposition 6.3. *In any resolution inference,*

(a) there is never a restriction on more than one variable in any one clause, and
(b) at most one of the two premises carries a restriction.

Proof. (a) Resolution removes all restricted variables (condition (2) above) from the resolvent, and the property holds of all input clauses.

(b) First of all, restricted clauses are input clauses in $\mathrm{Cl}(F)$ corresponding to positive occurrences of $\forall$ or negative occurrences of $\exists$ (or are derived from them by applications of (nex_r), but not using other rules; cf. (a)). A resolution inference with two premises which both carry restrictions would be (up to leading $\bigcirc$'s) of the form

$$\frac{\{\neg[\![A(x)]\!](a,b,\bar{c}),[\![(\forall x)A(x)]\!]\}^a \quad \{[\![A(x')]\!](b',a',\bar{c}'),\neg[\![(\exists x')A(x')]\!]\}^{a'}}{\{[\![(\forall x)A(x)]\!]\sigma,\neg[\![(\exists x')A(x')]\!]\sigma\}}\ \mathrm{res}$$

Other constellations are ruled out by the degree restriction on resolution. Since neither a nor a' may be substituted into by the unifier σ, they cannot stand opposite each other. Instead, they must unify with two other variables b' and b, respectively, i.e., $\sigma(b') = a$ and $\sigma(b) = a'$. But then the restricted variables would occur in the resulting clause, violating condition (2) of resolution. $\square$

Example 6.4. Consider the formula $\Box\bigcirc(\forall x)A(x) \supset (\forall x)\Box\bigcirc A(x)$. The sub-semi-formulas are:

$$\begin{array}{ll} S_1 = A(x) & S_2 = \bigcirc A(x) \\ S_3 = \bigcirc\Box\bigcirc A(x) & S_4 = \Box\bigcirc A(x) \\ S_5 = (\forall x)\Box\bigcirc A(x) & S_6 = (\forall x)A(x) \\ S_7 = \bigcirc\Box\bigcirc(\forall x)A(x) & S_8 = \bigcirc(\forall x)A(x) \\ S_9 = \Box\bigcirc(\forall x)A(x) & S_{10} = \Box\bigcirc(\forall x)A(x) \supset (\forall x)\Box\bigcirc A(x) \end{array}$$

A resolution proof is given by:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\{[\![S_8]\!],\neg[\![S_9]\!]\}\quad\dfrac{\{\neg[\![S_6]\!],[\![S_1]\!](a)\}}{\{\neg[\![S_8]\!],[\![S_2]\!](a)\}}\,\mathrm{nex}_r}{\{\neg[\![S_9]\!],[\![S_2]\!](a)\}}\,\mathrm{res}}{\{\neg[\![S_9]\!],[\![S_4]\!](a)\}}\,\mathrm{nec}_r \qquad \{\neg[\![S_4]\!](b),[\![S_5]\!]\}^b}{\{\neg[\![S_5]\!],[\![S_{10}]\!]\}\qquad\{\neg[\![S_9]\!],[\![S_5]\!]\}}\,\mathrm{res}}{\{\neg[\![S_9]\!],[\![S_{10}]\!]\}\qquad\{[\![S_9]\!],[\![S_{10}]\!]\}}\,\mathrm{res}}{\{[\![S_{10}]\!]\}\qquad\{\neg[\![S_{10}]\!]\}}\,\mathrm{res}}{\emptyset}\,\mathrm{res}$$

Example 6.5. By contrast, consider the formula $F = P(f) \supset (\forall x)P(x)$, which is not valid. Without the eigenvariable condition, we would have the following derivation of the empty clause:

$$\dfrac{\dfrac{\dfrac{\{\neg[\![P(x)]\!](a),[\![(\forall x)P(x)]\!]\}^a\qquad\dfrac{\{[\![P(x)]\!](b),\neg[\![P(c)]\!](b)\}\quad\{[\![P(c)]\!](b),[\![F]\!](b)\}}{\{[\![P(x)]\!](b),[\![F]\!](b)\}}\,\mathrm{res}}{\{[\![(\forall x)P(x)]\!],[\![F]\!](b)\}}\,\mathrm{res}^*\qquad\{\neg[\![(\forall x)P(x)]\!],[\![F]\!](d)\}}{\{\neg[\![F]\!](e)\}\qquad\{[\![F]\!](b)\}}\,\mathrm{res}}{\emptyset}\,\mathrm{res}$$

For the resolution step (res*) to work, either $\sigma(a) = b$, or $\sigma(b) = a$. The former case is expressly forbidden, in the latter case the restricted variable would appear in the resulting clause.

Theorem 6.6. *The resolution calculus for* **TB** *is sound: If $\emptyset$ is derivable from* $\mathrm{Cl}(F)$*, then $\models F$.*

Proof. We show how a resolution derivation ρ not using the goal clause $\{\neg[\![F]\!]\}$ can be translated to an **LB**-derivation. Associate to each clause C in ρ the substitution $\lambda_C = \sigma\theta$, where σ is the original renaming of the bound variables and constants in subsemiformulas of F whose code occurs in C, and θ is the cumulative substitution of the subderivation in ρ ending in C. In effect, if $[\![A(x)]\!](a)$ is a literal in C, then $A(x)\lambda_C$ is the formula $A(a)$. If ρ ends in a clause $C\colon\{\neg[\![A_1]\!](\bar{a}_1),\ldots,\neg[\![A_n]\!](\bar{a}_n),[\![B_1]\!](\bar{b}_1),\ldots,[\![B_m]\!](\bar{b}_m)\}$ we obtain an **LB**-proof of $S_C\colon A'_1\lambda_C,\ldots,A'_n\lambda_C \to B'_1\lambda_C,\ldots,B'_m\lambda_C$. If C carries a variable restriction, the restricted variable is bound by a weak quantifier in S_C. We argue by induction on the length of ρ:

$h = 1$. ρ consists of a clause C from $\mathrm{Cl}(F)\setminus\{\neg[\![F]\!]\}$ only: If $C \in Ax(F)$, say, $C = \{\neg[\![P(\bar{\alpha})]\!](\bar{a}),[\![P(\bar{\beta})]\!](\bar{a})\}$, the sequent $P(\bar{a}) \to P(\bar{a})$ is the corresponding axiom. If $C = C_F(A)$, where $A \in \mathrm{sub}(F)$, then we construct an **LB**-proof of S_C. We present here only some cases:

(1) $C = \big\{\{\neg[\![A]\!],\neg[\![B]\!],[\![A\wedge B]\!]\}\big\}$. The corresponding proof is:

$$\frac{A\to A \quad B\to B}{A,B\to A\wedge B}\ \wedge\text{:right}$$

(2) $C = \big\{\{\neg[\![A(x)]\!](a),[\![(\forall x)A(x)]\!]\}^a\big\}$. The corresponding proof is:

$$(\forall x)A(x)\to(\forall x)A(x)$$

(3) $C = \big\{\{[\![A(x)]\!](a),\neg[\![(\forall x)A(x)]\!]\}\big\}$. The corresponding proof is:

$$\frac{A(a)\to A(a)}{(\forall x)A(x)\to A(a)}\ \forall\text{:left}$$

(4) $C = \{\{[\![\bigcirc\Box A]\!],\neg[\![\Box A]\!]\}\}$. The corresponding proof is:

$$\frac{\bigcirc\Box A\to\bigcirc\Box A}{\Box A\to\bigcirc\Box A}\ \Box\text{:left}$$

$h > 1$: We distinguish cases according to the last inference in ρ. Let N denote the negative and P the positive set of literals in a clause, and Γ_N and Δ_P its translations, respectively.

(1) The last inference in ρ is a resolution where the premises do not carry a variable restriction:

$$\frac{C\colon\neg N\cup P\cup\{[\![A]\!](\bar{a})\}\quad C'\colon\neg N'\cup P'\cup\{\neg[\![A]\!](\bar{b})\}}{\neg N\sigma\cup\neg N'\sigma\cup P\sigma\cup P'\sigma}\ \text{res}$$

By induction hypothesis, we have **LB**-proofs π, π' of $\Pi_N\to\Lambda_P, A\lambda_C(\bar{a})$ and $A\lambda_{C'}(\bar{b}),\Pi_{N'}\to\Lambda_{P'}$. The unifier σ does not substitute into eigenvariables of π or π'. We obtain a proof:

$$\frac{\begin{array}{c}\vdots\ \pi\sigma\\ \Pi_N\sigma\to\Lambda_P\sigma,A(\bar{a})\sigma\end{array}\quad\begin{array}{c}\vdots\ \pi'\sigma\\ A(\bar{b})\sigma,\Pi_{N'}\sigma\to\Lambda_{P'}\sigma\end{array}}{\Pi_N\sigma,\Pi_{N'}\sigma\to\Lambda_P\sigma,\Lambda_{N'}\sigma}\ \text{cut}$$

(2) The last inference in ρ is a resolution where one premise contains the restricted variable a:

$$\frac{\neg N\cup P\cup\{[\![A]\!](a)\}^a\quad\neg N'\cup P'\cup\{\neg[\![A]\!](b)\}}{\neg N\sigma\cup\neg N'\sigma\cup P\sigma\cup P'\sigma}\ \text{res}$$

By Proposition 6.3, a resolution involving restricted variables can only take this form. By induction hypothesis, we have **LB**-proofs π, π' of $\Pi_N\to\Lambda_P, A(a)$

and $(\forall x)A(x), \Pi_{N'} \to \Lambda_{P'}$. The unifier σ does not substitute into restricted variables (i.e., eigenvariables of π, π'). Since a is restricted, we have $\sigma(a) = a$ and $\sigma(b) = a$, so b cannot occur in the resulting clause. Hence, it satisfies the eigenvariable condition. We obtain a proof:

$$\dfrac{\dfrac{\begin{array}{c}\vdots\ \pi\sigma\\ \Pi_N\sigma \to \Lambda_P\sigma, A(b)\sigma\end{array}}{\Pi_N\sigma \to \Lambda_P\sigma, (\forall x)A(x)\sigma}\ \forall\text{:right} \qquad \begin{array}{c}\vdots\ \pi'\sigma\\ (\forall x)A(x), \Pi_{N'}\sigma \to \Lambda_{P'}\sigma\end{array}}{\Pi_N\sigma, \Pi_{N'}\sigma \to \Lambda_P\sigma, \Lambda_{P'}\sigma}\ \text{cut}$$

(3) The last inference in ρ is (fact):

$$\dfrac{\neg N \cup P \cup \{(\neg)[\![A]\!](\bar{a}), (\neg)[\![A]\!](\bar{b})\}}{\neg N\sigma \cup N\sigma \cup \{(\neg)[\![A]\!](\bar{a})\sigma\}}\ \text{fact}$$

By induction hypothesis, we have a proof π of $\Pi_N \to \Lambda_P, A(\bar{a}), A(\bar{b})$ (or $A(\bar{a}), A(\bar{b}), \Pi_N \to \Lambda_P$). Since there are no restrictions on variables, we can rename $\bar{b}$ via $\sigma(b_i) = a_i$ in π. With contraction, we obtain a proof of $\Pi_N\sigma \to \Lambda_P\sigma, A(\bar{a})$ (or $A(\bar{a}), \Pi_N\sigma \to \Lambda_N\sigma$).

(4) The last inference in ρ is (nec_r). Add a (nec)-inference to the **LB**-proof.

(5) The last inference in ρ is (nex_r). Add a (nex)-inference to the **LB**-proof. □

If there were a resolution proof of $\emptyset$ which does not use the goal clause $\{\neg[\![F]\!]\}$, then we could translate that into an **LB**-proof of the empty sequent $\to$. Such a proof, of course, is impossible. Hence, any resolution derivation of $\emptyset$ must use the goal clause $\{\neg[\![F]\!]\}$. By the degree restriction, the last inference in such a derivation must be a resolution between $\{[\![F]\!]\}$ and $\{\neg[\![F]\!]\}$. A resolution derivation of $\{[\![F]\!]\}$ can, as above, be translated into an **LB**-proof of $\to F$.

Remark 6.7. Observe that the degree restriction on the resolution rule is necessary for soundness. Otherwise, e.g., $P \supset \Box P$ would have the following proof:

$$\dfrac{\dfrac{\dfrac{\{\neg[\![P \supset \Box P]\!]\} \quad \{[\![P]\!], [\![P \supset \Box P]\!]\}}{\{[\![P]\!]\}}\ \text{res}}{\{[\![\Box P]\!]\}}\ \text{nec} \qquad \dfrac{\{\neg[\![P \supset \Box P]\!]\} \quad \{\neg[\![\Box P]\!], [\![P \supset \Box P]\!]\}}{\{\neg[\![\Box P]\!]\}}\ \text{res}}{\emptyset}\ \text{res}$$

In fact, a formula $\neg F$ has a refutation without degree restriction iff $\models \Diamond F$, but $\models \Diamond F$ is not equivalent to $\models F$ (in contrast to $\Box$ and $\bigcirc$; cf. Lemma 2.5).

Theorem 6.8. *The resolution calculus for* **TB** *is complete: If* $\models F$*, then* $\emptyset$ *is derivable from* $\text{Cl}(F)$*.*

Proof. We give, for each **LB**-proof π of a sequent $\to F$, a resolution proof of $\emptyset$ from $\text{Cl}(F)$. By Theorem 4.1, we can assume that π is cut-free, analytic, that its axioms are atomic, and by Proposition 3.5 that it contains no weakenings. Let $\Pi \to \Lambda$ be a sequent in π. As can easily be seen, a formula A occurs positively (negatively) in $\Pi \to \Lambda$ iff it occurs positively (negatively) in F. Furthermore, every formula A in π corresponds to exactly one sub-semi-formula A' of F, which can be determined by tracing the formula A downwards through π. We translate π to a resolution proof ρ of $\{[\![F]\!]\}$ by induction on its subproofs π': If π' ends in $\Pi \to \Lambda$, then ρ' ends in $\neg N_\Pi \cup P_\Lambda$, where the semi-formulas whose codes occur in $\Pi' \cup \Lambda'$ are those sub-semi-formulas of F corresponding to the formulas in $\Pi \to \Lambda$. There is no variable restriction on the last clause in ρ'. We present here some cases:

(1) π' is an axiom:
Translate $P(\bar{a}) \to P(\bar{a})$ to a clause $\{\neg[\![P(\bar{\alpha})]\!](\bar{a}), [\![P(\bar{\beta})]\!](\bar{a})\}$, where $P(\bar{\alpha})$ ($P(\bar{\beta})$) is the sub-semi-formula of F corresponding to the left (right) $P(\bar{a})$. (This clause is in $Ax(F)$.)

(2) π' ends in a contraction on a formula A:
By induction hypothesis, we have a resolution proof of $\neg N_\Pi \cup P_\Lambda \cup \{[\![A']\!](\bar{a}), [\![A']\!](\bar{b})\}$ without restriction of variables. (A' is the sub-semi-formula of F corresponding to A.) Apply (fact).

(3) π' ends in ($\wedge$:right):
By induction hypothesis, we have resolution proofs ending in $\neg N_\Pi \cup P_\Lambda \cup \{[\![A']\!]\}$ and $\neg N_{\Pi'} \cup P_{\Lambda'} \cup \{[\![B']\!]\}$. The clause $\{\neg[\![A']\!], \neg[\![B']\!], [\![A' \wedge B']\!]\}$ is in $\mathrm{Cl}(F)$. We obtain a resolution proof:

$$\dfrac{\dfrac{\{\neg[\![A']\!], \neg[\![B']\!], [\![A' \wedge B']\!]\} \quad \neg N_\Pi \cup P_\Lambda \cup \{[\![A']\!]\}}{\neg N_\Pi \cup P_\Lambda \cup \{[\![A' \wedge B']\!], \neg[\![B']\!]\}} \qquad \neg N_{\Pi'} \cup P_{\Lambda'} \cup \{[\![B']\!]\}}{\neg N_\Pi \cup \neg N_{\Pi'} \cup P_\Lambda \cup P_{\Lambda'} \cup \{[\![A' \wedge B']\!]\}}$$

(4) π' ends in ($\forall$:left):
By induction hypothesis, we have a resolution proof ending in $\{\neg[\![A'(x)]\!](a)\} \cup \neg N_\Pi \cup P_\Lambda\}$. The clause $\{[\![A'(x)]\!](b), \neg[\![(\forall x)A'(x)]\!]\}$ is in $\mathrm{Cl}(F)$. We obtain a resolution proof:

$$\dfrac{\{\neg[\![A'(x)]\!](a)\} \cup \neg N_\Pi, P_\Lambda \quad \{[\![A'(x)]\!](b), \neg[\![(\forall x)A'(x)]\!]\}}{\{\neg[\![(\forall x)A'(x)]\!]\} \cup \neg N_\Pi \cup P_\Lambda}$$

(5) π' ends in ($\forall$:right):
By induction hypothesis, we have a resolution proof of $\neg \Pi \cup \Lambda \cup \{[\![A(x)]\!](a)\}\}$. The clause $\{\neg[\![A(x)]\!](b), [\![(\forall x)A(x)]\!]\}^b$ is in $\mathrm{Cl}(F)$. We obtain the resolution proof:

$$\dfrac{\neg N_\Pi \cup P_\Lambda \cup \{[\![A(x)]\!](a)\} \quad \{\neg[\![A(x)]\!](b), [\![(\forall x)A(x)]\!]\}^b}{\neg N_\Pi \cup P_\Lambda \cup \{[\![(\forall x)A(x)]\!]\}}$$

Note that the conditions on b in the right premise are met, since a satisfies the eigenvariable condition.

(6) π' ends in ($\Box$:left):
By induction hypothesis, we have a resolution proof of $\{\neg[\![A']\!], \neg[\![\bigcirc\Box A']\!]\} \cup \neg N_\Pi \cup P_\Lambda$. The clauses $\{[\![A']\!], \neg[\![\Box A']\!]\}$ and $\{[\![\bigcirc\Box A']\!], \neg[\![\Box A']\!]\}$ are in $\mathrm{Cl}(F)$. We obtain a resolution proof:

$$\dfrac{\dfrac{\{\neg[\![A']\!], \neg[\![\bigcirc\Box A']\!]\} \cup \neg N_\Pi \cup P_\Lambda \quad \{[\![A']\!], \neg[\![\Box A']\!]\}}{\{\neg[\![\bigcirc\Box A']\!], \neg[\![\Box A']\!]\} \cup \neg N_\Pi \cup P_\Lambda} \qquad \{[\![\bigcirc\Box A']\!], \neg[\![\Box A']\!]\}}{\{\neg[\![\Box A']\!]\} \cup \neg N_\Pi \cup P_\Lambda}$$

(7) π' ends in (nex):
Append a (nex_r)inference to the resolution proof to obtain ρ'.

(8) π' ends in (nec):
Append a (nec_r) inference to the resolution proof to obtain ρ'.

Note that in the translation to resolution, the restriction on the rules are all satisfied. The unifiers can be chosen so that only the variables in the clauses from $\mathrm{Cl}(F)$ are substituted into. Given a proof π of $\rightarrow F$ we thus have a resolution proof ρ of $\{[\![F]\!]\}$ from clauses in $\mathrm{Cl}(F)$. By resolving with $\{\neg[\![F]\!]\} \in \mathrm{Cl}(F)$, we obtain $\emptyset$. $\Box$

The translation above shows actually that a refinement of resolution is complete, namely where every resolution step has to involve at least one input clause, i.e., a clause form $\mathrm{Cl}(F)$. The resolution method developed here differs significantly from the resolution method of Robinson developed for classical clause logic, hence the fact that "input resolution" is complete is not a contradiction to the well-known fact that input resolution in the classical case is *not* complete.

7 Conclusion

We have seen how the passage from a non-axiomatizable temporal semantics to an axiomatizable one is paralleled by an extension of the completeness proof of the propositional logic. The point where the proof fails for **TL** is where a true formula starting with $\Box$ is reduced, even infinitely often, but no derivation can be obtained. The extension of the semantics is prompted by this phenomenon, and makes a complete reduction of the formula possible. The reduction discussed here is very similar to Kröger's completeness proof for propositional **TL**. This prompts the question of how to extend similar propositional completeness proofs to the first-order case by avoiding non-axiomatizability of the standard semantics by extension of the semantics itself. A candidate for such investigations would be, e.g., infinite-valued Lukasiewicz logic. It also prompts the question for a characterization of classes of formulas, where a sequent calculus is complete for the original semantics, say, as those formulas where the reduction *works*.

It is quite natural to ask, whether the predicate logic of linear time with gaps (the structures being sequences of ω-segments) is axiomatizable or not; let us call this logic **TLG**. Indeed even the pure $\Box$-part of **TLG** is not axiomatizable. This result can be obtained by reducing the problem to the nonaxiomatizability of the infinite-valued Gödel logic with truth values from the set $\{\frac{1}{n} | n \in \boldsymbol{N} - \{0\}\} \cup \{0\}$. However the proof of this result is quite involved, placing it outside the scope of this paper. It will be presented elsewhere.

Another problem which has not been addressed in depth so far is the correspondence between temporal logics discussed here, and number theory. The proof of non-axiomatizability of **TL** by reduction to arithmetic, and the "induction" rule of propositional **TL** suggest that there is a close relation. This suggestion is supported by our result: the semantics of **TB** is a "non-standard" semantics, similar to non-standard models of arithmetic. Viewed this way, it is not as surprising that **TB** would have a complete axiomatization.

References

[1] H. Andreká, V. Goranko, S. Mikulás, I. Németi, I. Sain, Effective Temporal Logic of Programs, in: *Time and Logic. A Computational Approach*, L. Bolc and A. Szalas, eds. (UCL Press, London, 1995) 51–129.

[2] M. Baaz, An effective decision algorithm for propositional temporal logic, in: *5. Österreichische Artificial-Intelligence-Tagung*, Informatik Fachberichte, Vol. 208 (Springer, Berlin, 1989).

[3] J. R. Büchi, On a decision method in restricted second order arithmetic, in: *Logic, Methodology and Philosophy of Science. Proceedings of the 1960 Congress* (Stanford University Press, Stanford, 1962) 1–11

[4] M. Fitting, *Proof Methods for Modal and Intuitionistic Logics*, Synthese Library, Vol. 169 (Reidel, Dordrecht, 1983).

[5] P. Kremer, Quantifying over propositions in relevance logic: Non-axiomatisability of primary interpretations of $\forall p$ and $\exists p$, *J. Symbolic Logic* **58** (1993) 334–349.

[6] S. A. Kripke, The undecidability of monadic modal quantification theory, *Z. Math. Logik Grundlag. Math.* **8** (1962) 113–116.

[7] F. Kröger, *Temporal Logic of Programs*, EATCS Monographs in Computer Science, Vol. 8 (Springer, Berlin, 1987).

[8] F. Kröger, On the interpretability of arithmetic in temporal logic, *Theoret. Comput. Sci.* **73** (1990) 47–60.

[9] Y. S. Maslov, Inverse method of establishing deducibility, *Trudy Mat. Inst. Steklov* (1968) 26–87.

[10] G. E. Mints, Gentzen-type systems and resolution rules. Part I: Propositional logic, in: *COLOG-88. International Conference on Computer Logic. Proceedings*, Lecture Notes in Computer Science, Vol. 417 (Springer, Berlin, 1990) 198–231.

[11] G. E. Mints. Gentzen-type systems and resolution rules. Part II: Predicate logic, in: *Logic Colloquium 1990*, Lecture Notes in Logic, Vol. 2 (Springer, Berlin, 1993).

[12] A. Szalas, Concerning the semantic consequence relation in first-order temporal logic, *Theoret. Comput. Sci.* **47** (1986) 329–334.

[13] A. Szalas and L. Holenderski, Incompleteness of first-order temporal logic with until. *Theoret. Comput. Sci.*, **57** (1988) 317–325.

[14] G. Takeuti, *Proof Theory*, Studies in Logic, Vol. 81. (North-Holland, Amsterdam, 1987), 2nd ed.

[15] T. Tammet, Proof strategies in linear logic, *J. Automated Reasoning* **12** (1994) 273–304.