Efficient loop-check for backward proof search in some non-classical propositional logics

Alain Heuerding*, Michael Seyfried, Heinrich Zimmermann

Institut für Informatik und angewandte Mathematik,
Universität Bern, Switzerland

Abstract. We consider the modal logics KT and $\mathsf{S4}$, the tense logic $\mathsf{K_t}$, and the fragment $\mathsf{IPC}_{(\wedge,\rightarrow)}$ of intuitionistic logic.
For these logics backward proof search in the standard sequent or tableau calculi does not terminate in general. In terms of the respective Kripke semantics, there are several kinds of non-termination: loops inside a world (KT), infinite resp. looping branches ($\mathsf{S4}$, $\mathsf{IPC}_{(\wedge,\rightarrow)}$), and infinite branching degree ($\mathsf{K_t}$).
We give uniform sequent-based calculi that contain specifically tailored loop-checks such that the efficiency of proof search is not deteriorated. Moreover all these loop-checks are easy to implement and can be combined with optimizations.
Note that our calculus for $\mathsf{S4}$ is not a pure contraction-free sequent calculus, but this (theoretical) defect does not hinder its application in practice.

1 Introduction

For many non-classical propositional logics, backward proof search in the usual sequent calculi does not terminate in general.
For all the logics we consider in the following backward proof search corresponds to the search for a Kripke counter-model. The non-termination can have several reasons:
• Loop inside a world: This is the case for KT. The problem is easily remedied, but the method we use to solve the problem proves to be useful for other calculi as well.
• Infinite resp. looping branch: This is probably the most interesting case. It is easy to solve this problem if only a finite number of sequents can occur during the search: One just checks whether a new sequent occurred already before on the branch. However, this method leads to a very inefficient implementation. Therefore loop-checks are often considered as completely useless.
In the case of IPC the problem has been solved by the discovery of a contraction-free sequent calculus ([2], [9]). In [10], the same has been achieved for $\mathsf{S4}$, however only for formulas in a certain normal form.

* Work supported by the Swiss National Science Foundation, SPP 5003-34279.

We show that in the case of $\mathsf{S4}$ it is possible to add a loop-check to the standard sequent calculus that is both very efficient and easy to implement. Moreover it can be combined with optimizations. The same idea also provides an efficient loop-check for the fragment $\mathsf{IPC}_{(\wedge,\to)}$ of intuitionistic logic.
• Infinite branching degree: $\mathsf{K_t}$ is the modal logic K with two additional modal operators. Whereas the backward proof search in K always terminates, this is not the case for $\mathsf{K_t}$, although all the branches have finite length. In the Kripke structure the non-termination corresponds to an infinite branching degree.

Note that this paper is mainly concerned with efficient loop-check methods. It is obvious that in an implementation of backward proof search many improvements are possible.

2 KT: loops inside a world

Definition 1. $Var_{\mathrm{KT}} := \{p_0, p_1, p_2, \ldots\}$ is the set of the variables of KT.
The set Fml_{KT} of the formulas of KT is inductively defined as follows: if $P \in Var_{\mathrm{KT}}$, then $P \in Fml_{\mathrm{KT}}$ and $\neg P \in Fml_{\mathrm{KT}}$; if $A, B \in Fml_{\mathrm{KT}}$, then $(\Box A)$, $(\Diamond A)$, $(A \wedge B)$, and $(A \vee B) \in Fml_{\mathrm{KT}}$.

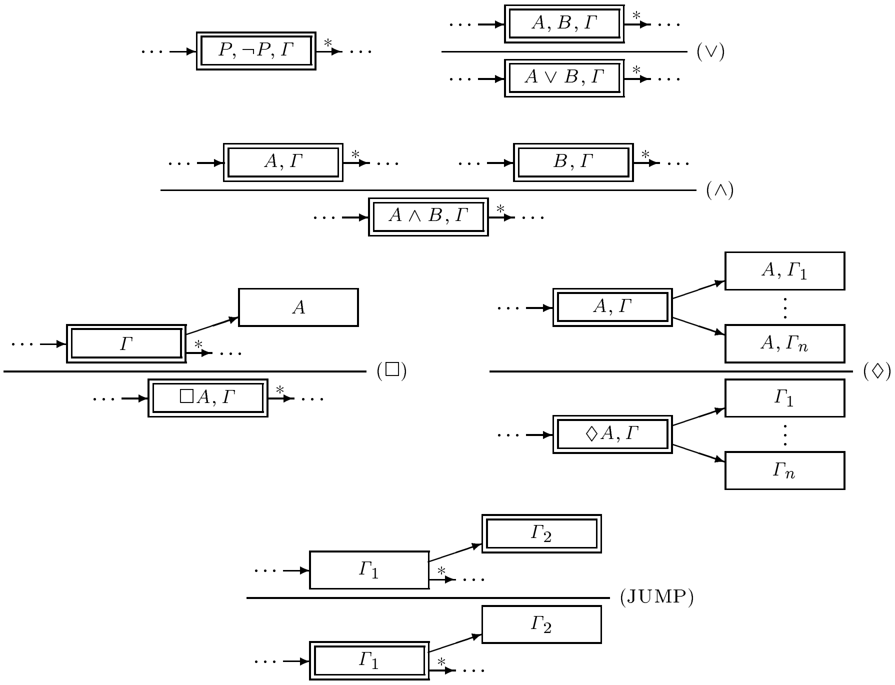
Note 2. In order to obtain less cumbersome calculi we only consider formulas in negation normal form. However, this is no restriction since it is easy to extend our methods to formulas that can contain $\to$, $\leftrightarrow$ and where the negation symbol can occur in front of any subformula. (We do this in our implementation, where we use two-sided sequent calculi.)

Usually we omit the outermost parentheses. The unary connectives have the highest priority.
We use meta-variables as follows: P, Q for variables, A, B, C for formulas, Γ, Δ, Σ, Π for (perhaps empty) multisets of formulas, n for natural numbers (including 0).
ε denotes the empty multiset.
If $\Gamma = A_1, \ldots, A_n$ and $\circ$ is a unary operator, then $\circ\Gamma = \circ A_1, \ldots, \circ A_n$.

Definition 3. The length of a formula A, $l(A)$, is defined as follows: $l(P) := 1, l(\neg P) := 2$, $l(\Box A) := l(A) + 1$, $l(\Diamond A) := l(A) + 1$, $l(A \wedge B) := l(A) + l(B) + 1$, $l(A \vee B) := l(A) + l(B) + 1$.
If $\Gamma = A_1, \ldots, A_n$, then $l(\Gamma) := \sum_{i=1}^{n} l(A_i)$.

$\mathsf{KT} \models A$ means that A is true in all the worlds of all the KT Kripke structures. (See e.g. [1] or [13] for more information on the basics of KT.)
For a given formula A, we can now try to construct a counter-model. If we fail after trying all the possibilities, then we know that $\mathsf{KT} \models A$. This method leads to the calculus $\mathsf{KT}^{\mathcal{G}}$ (See e.g. [4] for more information.)

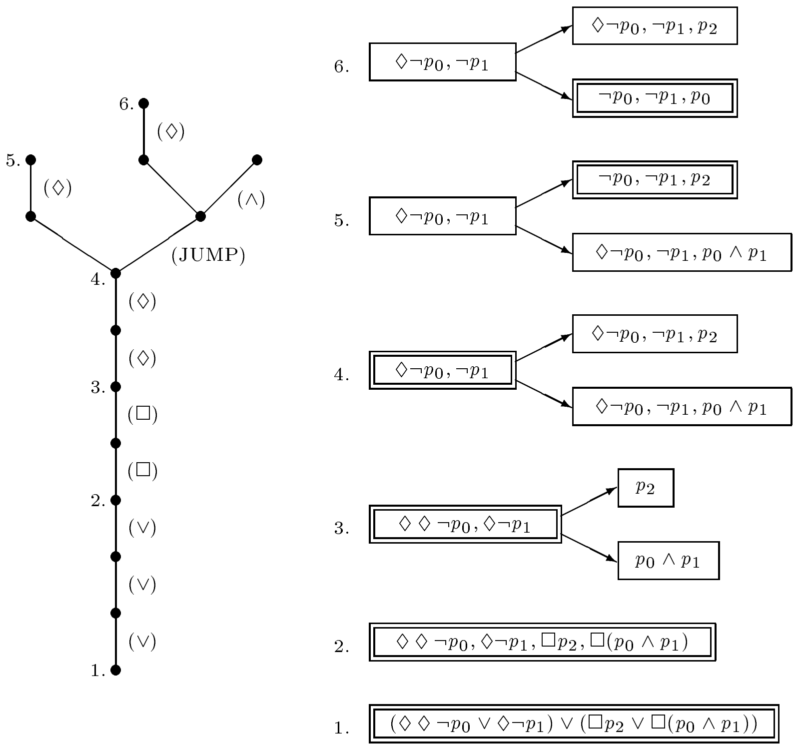
Definition 4. $\mathsf{KT}^{\mathcal{G}}$ is the following graph calculus:



All these rules convert graphs (the premises) into a graph (the conclusion). Boxes represent the worlds, the arrows represent the accessibility relation between these worlds. The sequent in two boxes is the current world.
$\cdots\rightarrow$ on the left hand side of a world means that this world can have a father.
$\xrightarrow{*}\cdots$ on the right hand side of a world means that this world can have any number of children in addition to those we show explicitly.
If we start with $\boxed{\boxed{A}}$, i.e. with one world that contains just the formula A, and do backward proof search, then we obtain a search tree with graphs in the nodes. (Here we obtain not arbitrary graphs, but only trees. This will be different in the case of $\mathsf{K_t}$.)

Note 5. The rules ($\Diamond$) and (JUMP) are the non-invertible rules of $\mathsf{KT}^{\mathcal{G}}$. (A rule is invertible if its premises are provable if and only if its conclusion is provable.)

Example 1. We check whether $\mathsf{KT}^{\mathcal{G}} \vdash (\Diamond \Diamond \neg p_0 \vee \Diamond \neg p_1) \vee (\Box p_2 \vee \Box(p_0 \wedge p_1))$.
The search tree and some of its nodes are shown in the picture on the next page. The search starts at 1. with three ($\vee$) backward applications, resulting in 2. . Then two successor worlds are created (3.) and the $\Diamond$ formulas are handled (4.). Then we have to choose between two jumps. One jumps ends with a fail (5.). We backtrack and try the other jump that ends in two axioms (6., 7.). Thus the formula $(\Diamond \Diamond \neg p_0 \vee \Diamond \neg p_1) \vee (\Box p_2 \vee \Box(p_0 \wedge p_1))$ is provable.



If we do backward proof search in $\mathsf{KT}^{\mathcal{G}}$, then we can forget the worlds in the past (i.e. on the left hand side) of the current formula. Therefore it is possible to convert the calculus $\mathsf{KT}^{\mathcal{G}}$ into a sequent calculus $\mathsf{KT}^{\mathcal{S}}$. The (JUMP) and the ($\Box$) rule of $\mathsf{KT}^{\mathcal{G}}$ are combined in the ($\Box$, JUMP) rule of $\mathsf{KT}^{\mathcal{S}}$. ($\mathsf{KT}^{\mathcal{S}}$ is the usual sequent calculus for KT, cp. e.g. [3] or [6].)
In the case of the tense logic $\mathsf{K_t}$ that we investigate in the last section it is not possible to obtain a sequent calculus in such an easy way because of the rule (JUMP2).

Definition 6. $\mathsf{KT}^{\mathcal{S}}$ is the following one-sided sequent calculus:

$$P, \neg P, \Gamma$$

$$\frac{A, B, \Gamma}{A \vee B, \Gamma}\ (\vee) \qquad\qquad \frac{A, \Gamma \quad B, \Gamma}{A \wedge B, \Gamma}\ (\wedge)$$

$$\frac{A, \Delta}{\Box A, \Diamond\Delta, \Gamma}\ (\Box, \mathrm{JUMP}) \qquad\qquad \frac{A, \Diamond A, \Gamma}{\Diamond A, \Gamma}\ (\Diamond)$$

Example 2. In general backward proof search in $\mathsf{KT}^{\mathcal{S}}$ (and also in $\mathsf{KT}^{\mathcal{G}}$) does not terminate.

$$
\dfrac{\dfrac{\text{axiom}}{p_0, \Diamond(p_0 \wedge p_2), \neg p_0} \qquad \dfrac{\dfrac{\dfrac{\text{axiom}}{p_2, p_0, \Diamond(p_0 \wedge p_2), \neg p_0} \qquad \dfrac{\vdots}{p_2, p_2, \Diamond(p_0 \wedge p_2), \neg p_0}\,(\Diamond)}{p_2, p_0 \wedge p_2, \Diamond(p_0 \wedge p_2), \neg p_0}\,(\wedge)}{p_2, \Diamond(p_0 \wedge p_2), \neg p_0}\,(\Diamond)}{\dfrac{p_0 \wedge p_2, \Diamond(p_0 \wedge p_2), \neg p_0}{\dfrac{\Diamond(p_0 \wedge p_2), \neg p_0}{\Diamond(p_0 \wedge p_2) \vee \neg p_0}\,(\vee)}\,(\Diamond)}\,(\wedge)
$$

Note 7. We cannot replace the rule $(\Diamond)$ of $\mathsf{KT}^{\mathcal{S}}$ by the rule

$$\frac{A, \Gamma}{\Diamond A, \Gamma}\ (\Diamond)$$

since otherwise e.g. the formula $\Diamond \Diamond \Box(p \vee \Box \Diamond \neg p)$ is no longer provable.

Of course such loops are easily recognized and avoided during proof search 'by hand': We apply the $(\Diamond)$ rule only once in each world on a certain main formula. We formalize this idea by splitting the sequents into two halves. In a sequent $\Sigma \mid \Gamma$, the multiset Γ contains the formulas not yet handled in this world and the multiset Σ contains the formulas already handled.

Definition 8. $\mathsf{KT}^{\mathcal{SU}}$ is defined as follows:

$$\Sigma \mid P, \neg P, \Gamma$$

$$\frac{\Sigma \mid A, B, \Gamma}{\Sigma \mid A \vee B, \Gamma}\ (\vee) \qquad\qquad \frac{\Sigma \mid A, \Gamma \quad \Sigma \mid B, \Gamma}{\Sigma \mid A \wedge B, \Gamma}\ (\wedge)$$

$$\frac{\Diamond A, \Sigma \mid A, \Gamma}{\Sigma \mid \Diamond A, \Gamma}\ (\Diamond) \qquad\qquad \frac{\varepsilon \mid A, \Sigma}{\Diamond \Sigma \mid \Box A, \Gamma}\ (\Box, \mathrm{JUMP})$$

A similar calculus was used in [12] to prove complexity results about KT.

Note 9. It would be more uniform to put every handled formula on the left hand side of the sequents (cp. $\mathsf{K_t}^{\mathcal{GU}}$). We refrained from doing so in order to make the calculus more readable.

Note 10. $(\Box, \mathrm{JUMP})$ is the only non-invertible rule of $\mathsf{KT}^{\mathcal{SU}}$.

Theorem 11.
$\mathsf{KT}^{\mathcal{G}} \vdash \boxed{\boxed{A}} \quad \Leftrightarrow \quad \mathsf{KT}^{\mathcal{S}} \vdash A \quad \Leftrightarrow \quad \mathsf{KT}^{\mathcal{SU}} \vdash \varepsilon \mid A$

Proof. The left equivalence follows immediately.
In order to prove the right equivalence, we show that the rules $(\vee)$, $(\wedge)$, $(\Diamond)$ of $\mathsf{KT}^{\mathcal{S}}$ are invertible and that $\mathsf{KT}^{\mathcal{S}} \vdash A, \Gamma \Leftrightarrow \mathsf{KT}^{\mathcal{S}} \vdash A, A, \Gamma$. It follows that we

can assume that the $(\Diamond)$ rule is not applied on the same $\Diamond$ formula without a $(\Box, \text{JUMP})$ application in between. Thus $KT^{\mathcal{S}} \vdash A$ implies $\mathsf{KT}^{\mathcal{SU}} \vdash \varepsilon \mid A$.
Since $\mathsf{KT}^{\mathcal{SU}}$ is just a restriction of $\mathsf{KT}^{\mathcal{S}}$, the other direction of the right equivalence follows immediately. $\Box$

Example 3. In $\mathsf{KT}^{\mathcal{SU}}$ backward proof search for $\Diamond(p_0 \wedge p_2) \vee \neg p_0$ fails without looping, in contrast to example 2, since no rule is applicable on the sequent $\Diamond(p_0 \wedge p_2) \mid p_2, \neg p_0$.

$$\dfrac{\dfrac{\dfrac{\text{axiom}}{\Diamond(p_0 \wedge p_2) \mid p_0, \neg p_0} \quad \Diamond(p_0 \wedge p_2) \mid p_2, \neg p_0}{\Diamond(p_0 \wedge p_2) \mid p_0 \wedge p_2, \neg p_0}\,(\wedge)}{\varepsilon \mid \Diamond(p_0 \wedge p_2), \neg p_0}\,(\Diamond)$$

Theorem 12. *Backward proof search in* $\mathsf{KT}^{\mathcal{SU}}$ *for sequents of the form* $\varepsilon \mid A$ *always terminates.*

Proof. We assume that we do proof search for the sequent $\varepsilon \mid A$.
$c := l(A)$. For all multisets Σ and Γ let $g(\Sigma \mid \Gamma)$ be the number of $\Box$ symbols in the multiset Σ, Γ. We define $f(\Sigma \mid \Gamma) := c \cdot g(\Sigma \mid \Gamma) + l(\Gamma)$.
For every sequent $\Sigma \mid \Gamma$ that can occur during the proof search for $\varepsilon \mid A$, we have $f(\Sigma \mid \Gamma) \in I\!N$, and $f(\Sigma \mid \Gamma)$ decreases with each backward application of a rule of $\mathsf{KT}^{\mathcal{SU}}$. $\Box$

3 S4: looping branches

We use the same language and the same conventions as for KT.

Definition 13. $\mathsf{S4}^{\mathcal{S}}$ is the calculus $\mathsf{KT}^{\mathcal{S}}$ with the rule $(\Box, \text{JUMP})$ replaced by the rule

$$\frac{A, \Diamond\Delta}{\Box A, \Diamond\Delta, \Gamma}\,(\Box, \text{JUMP})$$

Example 4. In general backward proof search in $\mathsf{S4}^{\mathcal{S}}$ does not terminate. The simplest example is:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\vdots}{p_0, \Diamond\Box p_0}\,(\Diamond)}{p_0, \Box p_0, \Diamond\Box p_0}\,(\Box, \text{JUMP})}{p_0, \Diamond\Box p_0}\,(\Diamond)}{\Box p_0, \Diamond\Box p_0}\,(\Box, \text{JUMP})}{\Diamond\Box p_0}\,(\Diamond)$$

In the following we again use sequents with two halves. However, in contrast to KT, only the simplest loops are avoided with this splitting, as the example 4 shows. Therefore the calculus $\mathsf{S4}^{\mathcal{SU}}$ contains a supplemental loop-check.

We add a so-called history H as a third element to the sequents of a $\mathcal{SU}$ calculus for S4. This history contains the required information about the previous worlds. In the simplest solution, the history contains all the sequents that occurred on this branch in the past. However, with this method one obtains very large histories and thus an extremely inefficient loop-check. A slightly improved solution was used in [12] to prove complexity results about S4.
In the following calculus, the loop-check is considerably optimized. A sequent has the form $H \,||\, \Sigma \mid \Gamma$, where the multiset Γ contains the formulas we did not yet handle in this world, the multiset Σ contains the formulas we already handled in this world, and the multiset H contains the required information about the previous worlds.

Note 14. Before defining the improved calculus $\mathsf{S4}^{\mathcal{SU}}$ we explain the motivation behind it in terms of backward proof search in $\mathsf{S4}^{\mathcal{S}}$.

- A loop during backward proof search in $\mathsf{S4}^{\mathcal{S}}$ must contain a $(\Box, \mathrm{JUMP})$ application. Therefore it is sufficient to save the sequents we obtain after a backwards application of $(\Box, \mathrm{JUMP})$ on some formula $\Box A$. Thus the history has the form $\langle A_1, \Sigma_1\rangle, \langle A_2, \Sigma_2\rangle, \ldots, \langle A_m, \Sigma_m\rangle$, where $\forall i, j \in \{1, \ldots, m\}$: $(i \neq j \Rightarrow \langle A_i, \Sigma_i\rangle \neq \langle A_j, \Sigma_j\rangle)$.
- A $\Diamond$-formula cannot disappear during the backward proof search, i.e. in the history $\langle A_1, \Sigma_1\rangle, \langle A_2, \Sigma_2\rangle, \ldots, \langle A_n, \Sigma_n\rangle$, we have $\forall i \in \{1, \ldots, n\} : \Sigma \subseteq \Sigma_{i+1}$. Therefore we can empty the history if a new $\Diamond$-formula appears. Thus the improved history has the form $\langle B_1, \Sigma\rangle, \langle B_2, \Sigma\rangle, \ldots, \langle B_n, \Sigma\rangle$, where $\forall i, j \in \{1, \ldots, n\} : (i \neq j \Rightarrow B_i \neq B_j)$.
  But if every element of the history has the same second element, it is useless to store this second element. Thus the further improved history is a simple list $B_1, \ldots, B_n$, where $\forall i, j \in \{1, \ldots, n\} : (i \neq j \Rightarrow B_i \neq B_j)$.
- Assume we apply $(\Box, \mathrm{JUMP})$ on the sequent $\Box A, \Box B, \Gamma, \Diamond\Delta$, where Γ contains no $\Diamond$-formulas, and obtain $A, \Diamond\Delta$. Assume further that after several backwards applications of rules we obtain the sequent $\Box B, \Gamma', \Diamond\Delta$, where Γ' contains no $\Diamond$-formulas.
  Then we do not have to apply $(\Box, \mathrm{JUMP})$ on $\Box B, \Gamma', \Diamond\Delta$ with B as the main formula, because the resulting sequence $B, \Diamond\Delta$ can be obtained directly from the original sequent $\Box A, \Box B, \Gamma, \Diamond\Delta$.

Definition 15. $\mathsf{S4}^{\mathcal{SU}}$ is defined as follows:

$$H \,||\, \Sigma \mid P, \neg P, \Gamma$$

$$\frac{H \,||\, \Sigma \mid A, B, \Gamma}{H \,||\, \Sigma \mid A \vee B, \Gamma}\ (\vee) \qquad \frac{H \,||\, \Sigma \mid A, \Gamma \quad H \,||\, \Sigma \mid B, \Gamma}{H \,||\, \Sigma \mid A \wedge B, \Gamma}\ (\wedge)$$

$$\frac{\varepsilon \,||\, \Diamond A, \Sigma \mid A, \Gamma}{H \,||\, \Sigma \mid \Diamond A, \Gamma}\ \Diamond A \notin \Sigma \quad (\Diamond, \mathrm{NEW}) \qquad \frac{H \,||\, \Sigma \mid A, \Gamma}{H \,||\, \Sigma \mid \Diamond A, \Gamma}\ \Diamond A \in \Sigma \quad (\Diamond)$$

$$\frac{A, \Delta, H \,||\, \Sigma \mid A, \Sigma}{H \,||\, \Sigma \mid \Box A, \Box \Delta, \Gamma}\; A \notin H \quad (\Box, \mathrm{JUMP})$$

Note 16. The rule $(\Box, \mathrm{JUMP})$ is the only non-invertible rule of $\mathsf{S4}^{\mathcal{SU}}$. For an efficient search this rule should only be applied such that Γ contains no $\Box$-formulas.

Theorem 17. $\mathsf{S4}^{\mathcal{S}} \vdash A \quad \Leftrightarrow \quad \mathsf{S4}^{\mathcal{SU}} \vdash \varepsilon \,||\, \varepsilon \mid A$

Proof.
'$\Leftarrow$':
Let $\mathcal{P}$ be a proof of $\varepsilon \,||\, \varepsilon \mid A$ in $\mathsf{S4}^{\mathcal{SU}}$. We replace in $\mathcal{P}$ each sequent $H \,||\, \Sigma \mid \Gamma$ by Σ, Γ, obtaining $\mathcal{P}'$. Since contraction is admissible in $\mathsf{S4}^{\mathcal{S}}$ we can convert $\mathcal{P}'$ into a proof of A in $\mathsf{S4}^{\mathcal{S}}$. ($\mathsf{S4}^{\mathcal{SU}}$ is just a restriction of $\mathsf{S4}^{\mathcal{S}}$.)

'$\Rightarrow$':
$\mathsf{S4}_1^{\mathcal{SU}}$ is the calculus $\mathsf{S4}^{\mathcal{SU}}$ with the $(\Box, \mathrm{JUMP})$ rule replaced by the rule

$$\frac{A, H \,||\, \Sigma \mid A, \Sigma}{H \,||\, \Sigma \mid \Box A, \Gamma}\; A \notin H \quad (\Box, \mathrm{JUMP})$$

$\mathsf{S4}_2^{\mathcal{SU}}$ is the calculus $\mathsf{S4}^{\mathcal{SU}}$ with the $(\Box, \mathrm{JUMP})$ rule replaced by the rule

$$\frac{A, H \,||\, \Sigma \mid A, \Sigma}{H \,||\, \Sigma \mid \Box A, \Gamma}\; (\Box, \mathrm{JUMP})$$

We use these two calculi as intermediate calculi and prove in the following:

$$\mathsf{S4}^{\mathcal{S}} \vdash A \Rightarrow \mathsf{S4}_2^{\mathcal{SU}} \vdash \varepsilon \,||\, \varepsilon \mid A \Rightarrow \mathsf{S4}_1^{\mathcal{SU}} \vdash \varepsilon \,||\, \varepsilon \mid A \Rightarrow \mathsf{S4}^{\mathcal{SU}} \vdash \varepsilon \,||\, \varepsilon \mid A$$

1. $\mathsf{S4}^{\mathcal{S}} \vdash A \;\Rightarrow\; \mathsf{S4}_2^{\mathcal{SU}} \vdash \varepsilon \,||\, \varepsilon \mid A$:
Note that in $\mathsf{S4}_2^{\mathcal{SU}}$ the history has absolutely no influence on the proof. Therefore we can use the same method as in the proof of $\mathsf{KT}^{\mathcal{S}} \vdash A \Rightarrow \mathsf{KT}^{\mathcal{SU}} \vdash \varepsilon \mid A$.

2. $\mathsf{S4}_2^{\mathcal{SU}} \vdash \varepsilon \,||\, \varepsilon \mid A \;\Rightarrow\; \mathsf{S4}_1^{\mathcal{SU}} \vdash \varepsilon \,||\, \varepsilon \mid A$:
The only difference between the calculi $\mathsf{S4}_2^{\mathcal{SU}}$ and $\mathsf{S4}_1^{\mathcal{SU}}$ is the condition $A \notin H$ in the $(\Box, \mathrm{JUMP})$ rule of $\mathsf{S4}_1^{\mathcal{SU}}$. (This condition is the loop-check of $\mathsf{S4}_1^{\mathcal{SU}}$.)
Let $\mathcal{P}$ be a proof of $\varepsilon \,||\, \varepsilon \mid A$ in $\mathsf{S4}_2^{\mathcal{SU}}$. We show that we can convert $\mathcal{P}$ into a proof of $\varepsilon \,||\, \varepsilon \mid A$ in $\mathsf{S4}_1^{\mathcal{SU}}$. If $\mathcal{P}$ is not already a proof of $\varepsilon \,||\, \varepsilon \mid A$ in $\mathsf{S4}_1^{\mathcal{SU}}$, then the following structure must occur as a part of a branch of $\mathcal{P}$:

$$\left.\begin{array}{c}
\dfrac{A, A, H' \,||\, \Sigma' \mid A, \Sigma'}{A, H' \,||\, \Sigma' \mid \Box A, \Gamma'}\,(\Box, \mathrm{JUMP}) \\
\hline
\vdots \\
\hline
\dfrac{A, H \,||\, \Sigma \mid A, \Sigma}{H \,||\, \Sigma \mid \Box A, \Gamma}\,(\Box, \mathrm{JUMP})
\end{array}\right\} (*)$$

where $A \notin H$ and no $(\Diamond, \mathrm{NEW})$ application occurs in $(*)$.
But then $\Sigma' = \Sigma$ and $H' = \Pi, H$ for some multiset Π, i.e. the following structure must occur in $\mathcal{P}$:

$$\cfrac{\cfrac{A, A, \Pi, H \,||\, \Sigma \mid A, \Sigma}{A, \Pi, H \,||\, \Sigma \mid \Box A, \Gamma'}\,(\Box, \mathrm{JUMP})}{\cfrac{\vdots}{\cfrac{A, H \,||\, \Sigma \mid A, \Sigma}{H \,||\, \Sigma \mid \Box A, \Gamma}\,(\Box, \mathrm{JUMP})}}$$

We replace this structure in $\mathcal{P}$ by $\dfrac{A, H \,||\, \Sigma \mid A, \Sigma}{H \,||\, \Sigma \mid \Box A, \Gamma}$ $(\Box, \mathrm{JUMP})$ and remove Π and one A from the history in the sequents above $H \,||\, \Sigma \mid A, \Sigma$ (up to the next $(\Diamond, \mathrm{NEW})$ application). The result is still a proof of $\varepsilon \,||\, \varepsilon \mid A$ in $\mathsf{S4}_2^{\mathcal{SU}}$.
In this way we eliminate all the $(\Box, \mathrm{JUMP})$ application in $\mathcal{P}$ that are not allowed as $(\Box, \mathrm{JUMP})$ applications in $\mathsf{S4}_1^{\mathcal{SU}}$.

3. $\mathsf{S4}_1^{\mathcal{SU}} \vdash \varepsilon \,||\, \varepsilon \mid A \;\Rightarrow\; \mathsf{S4}^{\mathcal{SU}} \vdash \varepsilon \,||\, \varepsilon \mid A$:
Using the motivation from note 14 it is possible to convert a proof of $\varepsilon \,||\, \varepsilon \mid A$ in $\mathsf{S4}_1^{\mathcal{SU}}$ into a proof of $\varepsilon \,||\, \varepsilon \mid A$ in $\mathsf{S4}^{\mathcal{SU}}$, using similar methods as in 2. $\Box$

Note 18. Already in the calculus $\mathsf{S4}_1^{\mathcal{SU}}$ (defined in the proof of theorem 17) no loops can occur. However, consider e.g. the formula $\Diamond \Box p_1 \vee (\Diamond \Box p_2 \vee (\ldots (\Diamond \Box p_{n-1} \vee \Diamond \Box p_n) \ldots))$ for some $n > 2$. For this formula we obtain an exponential speed-up in $\mathsf{S4}^{\mathcal{SU}}$ compared to $\mathsf{S4}_1^{\mathcal{SU}}$. We discuss this optimization because it is directly related to the loop-check.

Example 5. In $\mathsf{S4}^{\mathcal{SU}}$, backward proof search for $\Diamond \Box p_0$ fails without looping, in contrast to example 4. (No rule is applicable on the sequent $p_0 \,||\, \Diamond \Box p_0 \mid p_0, \Box p_0$.)

$$\cfrac{\cfrac{\cfrac{p_0 \,||\, \Diamond \Box p_0 \mid p_0, \Box p_0}{p_0 \,||\, \Diamond \Box p_0 \mid p_0, \Diamond \Box p_0}\,(\Diamond)}{\varepsilon \,||\, \Diamond \Box p_0 \mid \Box p_0}\,(\Box, \mathrm{JUMP})}{\varepsilon \,||\, \varepsilon \mid \Diamond \Box p_0}\,(\Diamond, \mathrm{NEW})$$

The example becomes more interesting if we replace p_0 by $p_0 \vee \Diamond p_2$, since then the history is removed twice by the $(\Diamond, \mathrm{NEW})$ rule before a loop is detected. To obtain harder formulas, add a $\wedge$ somewhere inside a $\Diamond$. (If we consider only formulas in negation normal form without $\wedge$, then the satisfiability problem for S4 is in coNP.)

Note 19. The formula $\Diamond \Box (p_0 \vee (\Box \Diamond p_0 \vee \Box \Diamond \neg p_0))$ shows that we cannot replace ε by H in the premise of the $(\Diamond, \mathrm{NEW})$ rule, i.e. we have to remove the history in such situations.

Theorem 20. *Backward proof search in* $\mathsf{S4}^{\mathcal{SU}}$ *for sequents of the form* $\varepsilon \,||\, \varepsilon \mid A$ *always terminates.*

Proof. We assume that we do proof search for the sequent $\varepsilon \,||\, \varepsilon \mid A$.
$c := l(A)$. We define $f(H \,||\, \Sigma \mid \Gamma) := c^2 \cdot (c - l(\Sigma)) + c \cdot (c - l(H)) + l(\Gamma)$.
For every sequent $H \,||\, \Sigma \mid \Gamma$ that can occur during the proof search for $\varepsilon \,||\, \varepsilon \mid A$, we have $f(H \,||\, \Sigma \mid \Gamma) \in I\!N$, and $f(H \,||\, \Sigma \mid \Gamma)$ decreases with each backward application of a rule of $\mathsf{S4}^{\mathcal{SU}}$. $\square$

4 $\mathsf{IPC}_{(\wedge,\rightarrow)}$: looping branches again

In this section we will briefly discuss the fragment $\mathsf{IPC}_{(\wedge,\rightarrow)}$ of intuitionistic propositional logic, proving the method for recognizing loops in $\mathsf{S4}$ to be applicable in a non-modal environment as well.

Definition 21. $Var_{\mathrm{IPC}_{(\wedge,\rightarrow)}} := \{p_0, p_1, p_2, \ldots\}$ is the set of variables of $\mathsf{IPC}_{(\wedge,\rightarrow)}$.
The set $Fml_{\mathrm{IPC}_{(\wedge,\rightarrow)}}$ of the formulas of $\mathsf{IPC}_{(\wedge,\rightarrow)}$ is inductively defined as follows: $\bot \in Fml_{\mathrm{IPC}_{(\wedge,\rightarrow)}}$; all variables are formulas, i.e. $Var_{\mathrm{IPC}_{(\wedge,\rightarrow)}} \subset Fml_{\mathrm{IPC}_{(\wedge,\rightarrow)}}$; if $A, B \in Fml_{\mathrm{IPC}_{(\wedge,\rightarrow)}}$, then both $(A \wedge B), (A \rightarrow B) \in Fml_{\mathrm{IPC}_{(\wedge,\rightarrow)}}$.
We use meta-variables as follows: A, B, C, D for formulas, P for variables and R for variables or $\bot$.

Note 22. Our interest in this particular fragment of intuitionistic propositional logic is due to its close connection to λ-calculus, $\mathsf{IPC}_{(\wedge,\rightarrow)}$ is the core logic of the Curry-Howard isomorphism.

- The contraction-free sequent calculi for full IPC, given by Dyckhoff (see [2]) and Hudelmaier (see [9], [10]), avoid loop checks completely, but in the context of the 'Formulas-as-Types'-paradigm they expose a defect. The proofs which are found by the respective algorithms, are not necessarily normal, i.e. the corresponding λ-terms can contain redexes, even if the calculi are restricted to $\mathsf{IPC}_{(\wedge,\rightarrow)}$.
- The calculus $\mathsf{IPC}^{RP}_{(\wedge,\rightarrow)}$ allows to avoid some of the non-determinism which is inherent to proof-search in intuitionistic propositional logic. $\mathsf{IPC}^{RP}_{(\wedge,\rightarrow)}$ is a syntactic modification of the respective fragment of Kleenes calculus $\mathsf{G3}$ (see [11]) and is considered in [15] (similar ideas can be found in [5]), where its completeness and correctness are proved. Furthermore it is shown there that the basic idea used for limiting non-determinism in proof search in $\mathsf{IPC}_{(\wedge,\rightarrow)}$ can not be successfully applied to full intuitionistic propositional logic IPC.

In the following we will present the calculus ${\mathsf{IPC}^{RP}_{(\wedge,\rightarrow)}}^{\mathcal{SU}}$, which incorporates efficient loop-checking on the syntactic level, in the same sense as in the modal logics considered before.

Definition 23. $\mathsf{IPC}^{RP}_{(\wedge,\to)}{}^{\mathcal{SU}}$ is a modified two-sided sequent calculus. Sequents are of the form $H \,||\, \Gamma \supset A$, where the history H is a multiset, Γ is a multiset, and A is a formula. $\mathsf{IPC}^{RP}_{(\wedge,\to)}{}^{\mathcal{SU}}$ is defined as follows:

$$H \,||\, \bot, \Gamma \supset R \qquad\qquad H \,||\, P, \Gamma \supset P$$

$$\frac{H \,||\, \Gamma \supset B}{H \,||\, \Gamma \supset A \to B}\ A \in \Gamma \quad (\to \mathrm{R}_{\in}) \qquad\qquad \frac{\varepsilon \,||\, A, \Gamma \supset B}{H \,||\, \Gamma \supset A \to B}\ A \notin \Gamma \quad (\to \mathrm{R}_{\notin})$$

$$\frac{H \,||\, \Gamma \supset A \quad H \,||\, \Gamma \supset B}{H \,||\, \Gamma \supset A \wedge B}\ (\wedge \mathrm{R}) \qquad\qquad \frac{R, H \,||\, \Gamma \supset \bot}{H \,||\, \Gamma \supset R}\ R \notin H \quad (\mathrm{EFQ})$$

$$\frac{\varepsilon \,||\, A_i, A_1 \wedge A_2, \Gamma \supset R}{H \,||\, A_1 \wedge A_2, \Gamma \supset R}\ A_i \notin \Gamma, R \in rp(A_i) \quad (\wedge \mathrm{L}_i) \qquad i \in \{1, 2\}$$

$$\frac{R, H \,||\, A \to B, \Gamma \supset A \quad \varepsilon \,||\, B, A \to B, \Gamma \supset R}{H \,||\, A \to B, \Gamma \supset R}\ B \notin \Gamma, R \notin H, R \in rp(B) \quad (\to \mathrm{L})$$

The additional condition in rules $(\wedge \mathrm{L}_1)$, $(\wedge \mathrm{L}_2)$ and $(\to \mathrm{L})$ ensures that the main formula of the respective (backwards) application is 'connected' to succedent atom R of the conclusion, where the set $rp(.)$ is inductively defined as follows: $rp(S) := \{S\}$, $rp(A \wedge B) := rp(A) \cup rp(B)$ and $rp(A \to B) := rp(B)$.

Example 6. We try to prove $(p_0 \to p_1) \to ((p_1 \to p_0) \to p_0)$ as an example of how loops are detected. Using the abbreviations $C := p_0 \to p_1$, $D := p_1 \to p_0$ for space-conservation this formula can be written as $C \to (D \to p_0)$.

$$\dfrac{\dfrac{\dfrac{\dfrac{p_0 \,||\, D, C \supset p_0 \quad \dfrac{\text{axiom}}{\varepsilon \,||\, p_1, D, C \supset p_1}}{p_0 \,||\, D, C \supset p_1}\,(\to \mathrm{L}) \quad \dfrac{\text{axiom}}{\varepsilon \,||\, p_0, D, C \supset p_0}}{\varepsilon \,||\, D, C \supset p_0}\,(\to \mathrm{L})}{\varepsilon \,||\, C \supset D \to p_0}\,(\to \mathrm{R}_{\notin})}{\varepsilon \,||\, \varepsilon \supset C \to (D \to p_0)}\,(\to \mathrm{R}_{\notin})$$

The left-most leaf of the tree is neither an axiom nor does it allow any of the rules to be applied. p_0 being a member of the 'history' multiset prevents all possibilities.

Note 24. In fact, loop checking is considerably easier than in the modal environment treated before, since in order to detect loops, only a list of atomic propositions has to be stored.
Please note that other commonly used optimizations for proof search in IPC are compatible with the loop-checking method here.

5 K_t: infinite branching degree

Definition 25. $Var_{\mathrm{K_t}} := \{p_0, p_1, p_2, \ldots\}$ is the set of the variables of $\mathsf{K_t}$. The set $Fml_{\mathrm{K_t}}$ of the formulas of $\mathsf{K_t}$ is inductively defined as follows: if $P \in Var_{\mathrm{K_t}}$, then $P \in Fml_{\mathrm{K_t}}$ and $\neg P \in Fml_{\mathrm{K_t}}$; if $A, B \in Fml_{\mathrm{K_t}}$, then $(\blacksquare A)$, $(\blacklozenge A)$, $(\Box A)$, $(\Diamond A)$, $(A \wedge B)$, $(A \vee B) \in Fml_{\mathrm{K_t}}$.

Definition 26. $\mathsf{K_t}^{\mathcal{H}}$ is the following Hilbert-style calculus:

$$A\{B_1/P_1, \ldots, B_n/P_n\}, \text{ if } A \text{ is a tautology in classical propositional logic}$$

$$\blacksquare(A \to B) \to (\blacksquare A \to \blacksquare B) \qquad \blacklozenge \Box A \to A$$

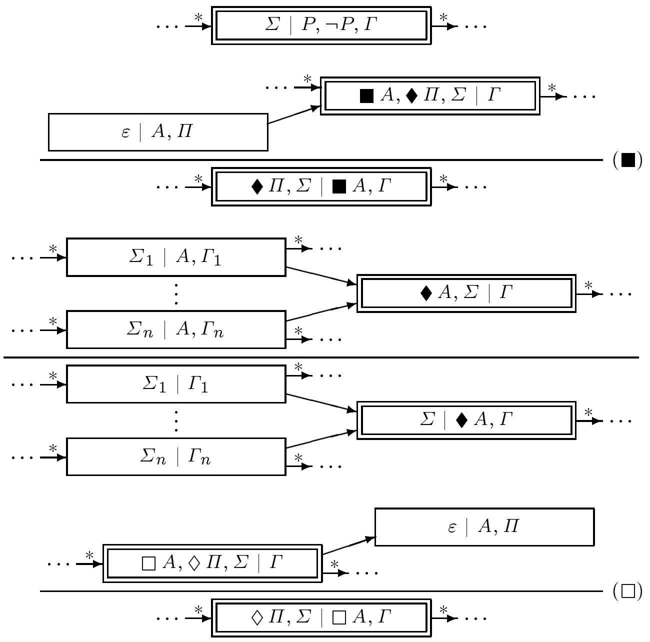
$$\Box(A \to B) \to (\Box A \to \Box B) \qquad \Diamond \blacksquare A \to A$$

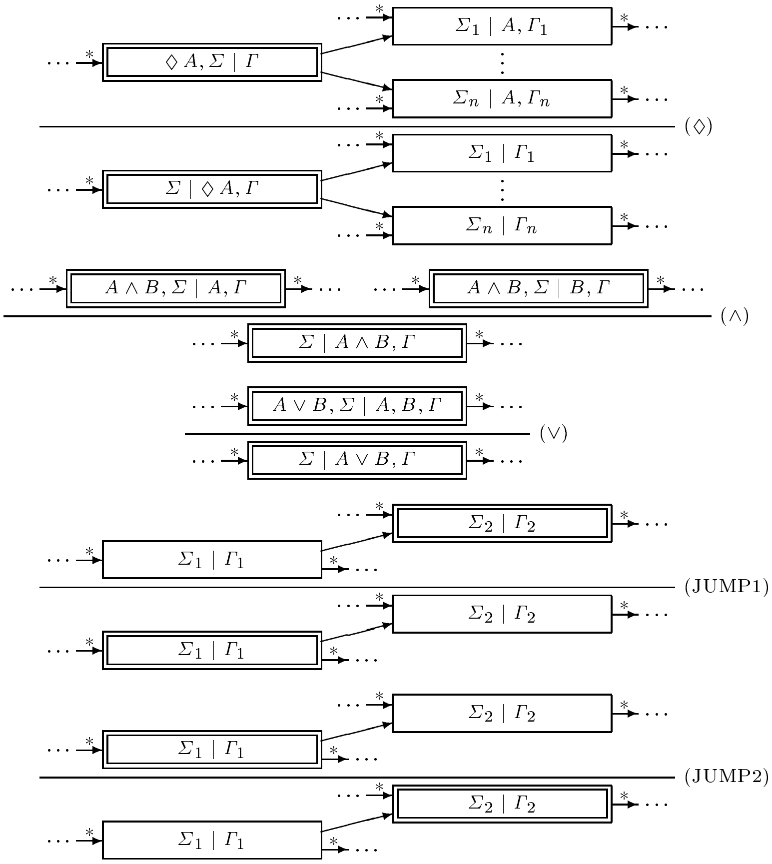
$$\frac{A \quad A \to B}{B}\ (\mathrm{MP}) \qquad \frac{A}{\blacksquare A}\ (\blacksquare) \qquad \frac{A}{\Box A}\ (\Box)$$

where $A \to B := \neg A \vee B$ and if A is not a variable, $\neg A$ is defined as usual.

$\blacksquare$ can be interpreted as 'for all worlds in the past', $\blacklozenge$ as 'exists a world in the past', $\Box$ as 'for all worlds in the future', and $\Diamond$ as 'exists a world in the future'. More information on the logic K_t can be found e.g. in [14].

Definition 27. $\mathsf{K_t}^{\mathcal{G}}$ is the following graph calculus:





Note 28. Because the possibility of jumping back and forth between different worlds is essential, there is no easy way of converting this calculus into a pure sequent calculus. Consider e.g. the provable formula $\Diamond P \rightarrow \Box \blacklozenge \Diamond P$.

Definition 29. $\mathsf{K_t}^{\mathcal{GU}}$ consists of the same axiom and rules as $\mathsf{K_t}^{\mathcal{G}}$ but there are conditions for the application of rules:

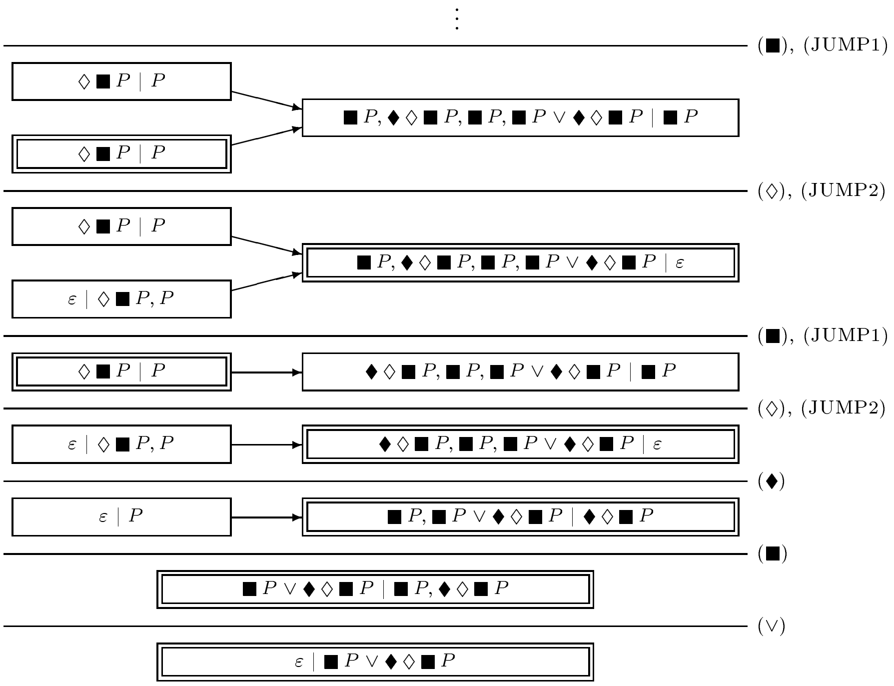
Rule	Condition
$(\blacksquare)$	$\blacksquare A \notin \Sigma$
$(\blacklozenge)$	$\blacklozenge A \notin \Sigma$
$(\Box)$	$\Box A \notin \Sigma$
$(\Diamond)$	$\Diamond A \notin \Sigma$
$(\wedge)$	$A \wedge B \notin \Sigma$
$(\vee)$	$A \vee B \notin \Sigma$

(JUMP1) and (JUMP2) may only be used in the finite search for a world, for which one of the other rules or the axiom schema matches. If no such world exists, then proof search ends with fail.

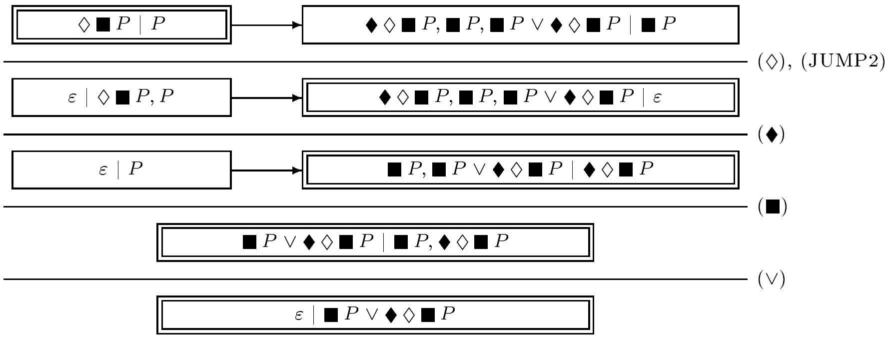
Theorem 30. $\mathsf{K_t}^{\mathcal{H}} \vdash A \quad \Leftrightarrow \quad \mathsf{K_t}^{\mathcal{G}} \vdash \boxed{\boxed{\varepsilon \mid A}} \quad \Leftrightarrow \quad \mathsf{K_t}^{\mathcal{GU}} \vdash \boxed{\boxed{\varepsilon \mid A}}$

A syntactical proof of the equivalence of $\mathsf{K_t}^{\mathcal{H}}$ and $\mathsf{K_t}^{\mathcal{GU}}$ is given in [16].

Example 7. In a backward proof search in $\mathsf{K_t}^{\mathcal{G}}$ for the formula $\blacksquare P \vee \blacklozenge \lozenge \blacksquare P$ we run into an endless loop.



Example 8. A backward proof search in $\mathsf{K_t}^{\mathcal{GU}}$ for the formula from example 7 terminates with fail: appart from (JUMP1) no rule is applicable any more.



Theorem 31. *Backward proof search in* $\mathsf{K_t}^{\mathcal{GU}}$ *always terminates.*

Proof. We define $f(\mathcal{G})$, where $\mathcal{G}$ is a graph with the nodes $\Sigma_1 \mid \Gamma_1, \ldots, \Sigma_n \mid \Gamma_n$, as the sum of the cardinalities of $\Sigma_1, \ldots, \Sigma_n$.
Let $\mathcal{G}$ be the root of a search tree and $\mathcal{G}'$ an arbitrary graph in this search tree. Since for every rule of $\mathsf{K_t}^{\mathcal{GU}}$ f of any of its premises is greater than f of its conclusion we have $f(\mathcal{G}') > f(\mathcal{G})$.
The number s of different subformulas in $\mathcal{G}'$ is bound by the sum of the lengths of all formulas in $\mathcal{G}$, because all rules of $\mathsf{K_t}^{\mathcal{GU}}$ have the subformula property. If d is the maximal nesting depth of $\blacksquare, \blacklozenge, \Box$, and $\Diamond$ in a formula in $\mathcal{G}$, then the depth of $\mathcal{G}'$ is bound by d. And if b is the number of occurrencies of $\blacksquare$ and $\Box$ in $\mathcal{G}$, then the branching degree of $\mathcal{G}'$ is bound by b. Therefore the number of worlds in $\mathcal{G}'$ is bound by $b^0 + b^1 + \cdots + b^d$, and we have $f(\mathcal{G}') \leq s \cdot (b^0 + b^1 + \cdots + b^d)$. $\Box$

6 Implementation

Optimized versions of the discussed calculi are implemented in the Logics Workbench LWB ([8]). You can use them via the World Wide Web. Start a WWW browser, open `http://lwbwww.unibe.ch:8080/LWBinfo.html`, choose the item `run a session`, type e.g. `load(s4); provable(box p -> box box p);` and push the `send` button. After a while you obtain the result `true` .
You can use the commands `timestart` and `timestop` to obtain the required execution time. Sometimes it helps if you apply `arrange` (with the second argument `"sort"`) on the formula before checking its provability. A good source for relatively easy, provable formulas is [7].

7 Conclusion

We have shown for several important non-classical propositional logics how the termination of backward proof search in sequent calculi can be ensured without slowing down the search. The main point in all cases is the storing of the already handled formulas.
In the case of $\mathsf{S4}$, our loop-check method is much more efficient than the standard loop-check. From a practical point of view the problem caused by the hidden contraction in the usual sequent calculus is solved. However, the main problem, namely the non-invertible rules, remains. Since this fact is strongly related to the PSPACE completeness it cannot be easy to avoid this problem.
There are other propositional non-classical logics where a loop-check is required during backward proof search. In the case of the hypersequent calculus for $\mathsf{S5}$, there is the same problem as for $\mathsf{K_t}$. The multimodal variant of KT offers no new problems, but we do not know whether there is a loop-check for the multimodal variant of $\mathsf{S4}$ that is as efficient as the one for $\mathsf{S4}$. Also in the case of the modal logic K we have not been able to find a calculus where the history is but a multiset of formulas (cp. [8]).

References

1. Brian F. Chellas. *Modal logic: an introduction.* Cambridge University Press, 1980.
2. Roy Dyckhoff. Contraction-free sequent calculi for intuitionistic logic. *The Journal of Symbolic Logic*, 57(3):795–807, 1992.
3. Melvin Fitting. *Proof Methods for Modal and Intuitionistic Logics.* Reidel, Dordrecht, 1983.
4. Melvin Fitting. Basic modal logic. In Dov M. Gabbay, C. J. Hogger, and J. A. Robinson, editors, *Handbook of Logic in Artificial Intelligence and Logic Programming*, volume 1, pages 368–448. Clarendon Press, Oxford, 1993.
5. D M Gabbay. Algorithmic Proof with Diminishing Resources Part 1. In E. Börger, editor, *CSL 90*, LNCS 533, pages 156–173, 1991.
6. Rajeev Goré. Tableau methods for modal and temporal logics. Technical report, TR-15-95, Automated Reasoning Project, Australian National University, Canberra, Australia, 1995. To appear in Handbook of Tableau Methods, Kluwer, 199?.
7. Rajeev Goré, Wolfgang Heinle, and Alain Heuerding. Relations between propositional normal modal logics: an overview. Technical report, TR-16-95, Automated Reasoning Project, Australian National University, Canberra, Australia, 1995.
8. Alain Heuerding, Gerhard Jäger, Stefan Schwendimann, and Michael Seyfried. Propositional logics on the computer. In Peter Baumgartner, Reiner Hähnle, and Joachim Posegga, editors, *Theorem Proving with Analytic Tableaux and Related Methods*, LNCS 918, pages 310–323, 1995.
9. Jörg Hudelmaier. An $O(n \log n)$-space decision procedure for intuitionistic propositional logic. *Journal of Logic and Computation*, 3(1):63–75, 1993.
10. Jörg Hudelmaier. On a contraction free sequent calculus for the modal logic S4. In *TABLEAUX 94. 3rd Workshop on Theorem Proving with Analytic Tableaux and Related Methods*, 1994.
11. S.C. Kleene. *Introduction to Metamathematics.* North-Holland, Amsterdam, 1952.
12. Richard E. Ladner. The computational complexity of provability in systems of modal propositional logic. *SIAM Journal on Computing*, 6(3):467–480, 1977.
13. LWBtheory. `http://lwbwww.unibe.ch:8080/LWBtheory.html`, 1995.
14. Nicholas Rescher and Alasdair Urquhart. *Temporal Logic.* Springer, 1971.
15. Michael Seyfried. A sequent calculus for proof search in the $(\rightarrow, \wedge)$-fragment of intuitionistic logic. Unpublished, 1994.
16. Heinrich Zimmermann. A directed tree calculus for minimal tense logic. Master's thesis, IAM, University of Berne, Switzerland, 1995.