How to optimize proof-search in modal logics: new methods of proving redundancy criteria for sequent calculi

ANDREI VORONKOV
The University of Manchester

We present a bottom-up decision procedure for propositional modal logic K based on the inverse method. The procedure is based on the "inverted" version of a sequent calculus. To restrict the search space, we prove a number of redundancy criteria for derivations in the sequent calculus. We introduce a *new technique* of proving redundancy criteria, based on the analysis of tableau-based derivations in K. Moreover, another new technique is used to prove completeness of proof-search with a strong notion of subsumption. This technique is based on so-called *traces*. A new formalization of the inverse method in the form of a *path calculus* considerably simplifies all proofs as compared to the previously published presentations of the inverse method. Experimental results demonstrate that our method is competitive with many state-of-the-art implementations of K.

Categories and Subject Descriptors: F.4.1 [**Mathematical Logic**]: Theorem Proving; I.2.4 [**Artificial Intelligence**]: Knowledge Representation Formalisms and Methods—*Modal Logic*

General Terms: Experimentation, Theory

Additional Key Words and Phrases: Description Logics, Theorem Proving, Modal Logic, Inverse Method, Proof-Search

1. INTRODUCTION

Nonclassical (propositional) logics play an increasingly important role in computer science. They are used in model checking, verification, and knowledge representation. Traditional decision procedures for these logics use semantic tableaux [Baader and Hollunder 1991; Catach 1991; Haarslev et al. 1998; Balsiger et al. 1998], optimized tableaux [Horrocks 1997; 1998; Patel-Schneider 1998], SAT-based methods [Giunchiglia and Sebastiani 1996a; 1996b], or highly optimized translations into first-order classical logic [Hustadt and Schmidt 1997; Schmidt 1998].

In this article we present a bottom-up decision procedure for propositional modal logic K based on the inverse method. All our results can be easily modified to its multimodal variant K(m) with m modalities, so we restrict ourselves to K for simplicity. Automatic theorem proving in K(m) is important for several reasons, mainly because this logic is a syntactic variant of the knowledge representation formalism $\mathcal{ALC}$.

This work is supported by a grant from EPSRC. Author's address: Department of Computer Science, University of Manchester, Manchester M13 9PL, UK.

Our experiments demonstrate that our method is competitive with the best state-of-the-art methods for K. We believe that the technique described in our article opens a new interesting research path in the implementation of decision procedures for other important nonclassical logics, such as temporal, description logics, or quantified boolean formulas.

This article is organized as follows. In Sections 2, 3, and 4 we define main notions and introduce several sequent calculi for K, including an original calculus based on the notion of path in the goal formula. In Section 5 we prove several properties of derivations in the path calculus. In Section 6 we introduce the inverse path calculus and use these properties of derivations to prove that some sequents and derivations in the inverse path calculus are redundant. In Section 7 we introduce new redundancy criteria based on two different versions of subsumption. Section 8 describes an implementation of our method in the system Kꓘ and experimental results. Section 9 briefly mentions related work on the inverse method and implementations of nonclassical logics. Section 10 discusses new research directions which we find interesting for efficient automation of nonclassical logics.

2. PRELIMINARIES

In this article we consider propositional modal logic K. K is the minimal normal modal logic, i.e. the set of modal formulas valid in all Kripke models. For the discussion of modal logics and Kripke models see any standard references, e.g. Fitting [1983], and Wallen [1990]. Our technique of proving redundancies is applicable to other modal logics, at least including T, K4, S4, D, and D4 and their versions with global assumptions,[1] but the notions of redundancy for these logics are different.

A *literal* is a propositional variable or its negation. We only consider *modal formulas in negation normal form*, i.e., formulas built from literals using $\wedge$, $\vee$, $\Box$, and $\Diamond$. A formula is called *satisfiable* if it is true in at least one Kripke model. In this article, we will speak about (un)satisfiability of formulas instead of the dual notion of *validity*. The satisfiability problem for K is PSPACE-complete. Satisfiability in K can be characterized using a *sequent calculus* introduced below.

We denote propositional variables by lowercase letters, and formulas by uppercase letters. We will use the formula

$$E = \Box a \wedge \Box(\neg a \vee b) \wedge \Diamond\neg b \wedge \Diamond b \tag{1}$$

for nearly all examples of this article. We consider conjunction and disjunction right-associative, so, for example, formula (1) denotes $\Box a \wedge (\Box(\neg a \vee b) \wedge (\Diamond\neg b \wedge \Diamond b))$.

A *sequent* is a finite multiset of formulas. We denote sequents by Greek letters Γ and Δ. We denote the multiset union of two multisets Γ and Δ by Γ, Δ, and the addition of an element A to a multiset Γ by Γ, A. The *multiset difference* of multisets Γ and Δ is denoted $\Gamma \dot{-} \Delta$. If an element occurs in Γ k times and in Δ l times and $k > l$, then it occurs in $\Gamma \dot{-} \Delta$ $k - l$ times; otherwise it does not

[1] All we need from a logic to apply the inverse method to it is a sequent calculus with a suitable subformula property. Logics K, T, K4, S4, D, and D4 admit such calculi. A more detailed discussion on how to design the inverse method calculi for various logics can be found in Mints et al. [2000].

$$\text{axioms: } \Gamma, p, \neg p \qquad \frac{\Gamma, A, B}{\Gamma, A \wedge B}\ (\wedge) \qquad \frac{\Gamma, A \quad \Gamma, B}{\Gamma, A \vee B}\ (\vee) \qquad \frac{\Delta, A}{\Gamma, \Box\Delta, \Diamond A}\ (\Diamond)$$

$$\cfrac{\cfrac{\cfrac{a, \neg a, \neg b \quad a, b, \neg b}{a, \neg a \vee b, \neg b}\ (\vee)}{\Box a, \Box(\neg a \vee b), \Diamond\neg b, \Diamond b}\ (\Diamond)}{\Box a \wedge \Box(\neg a \vee b) \wedge \Diamond\neg b \wedge \Diamond b}\ (\wedge), (\wedge), (\wedge)$$

A sequence of inferences $(\wedge)$ is shown as one inference.

Fig. 1. Sequent calculus Seq and a derivation in it.

occur in $\Gamma \dot{-} \Delta$ at all. If A is a member of a multiset Γ, we will write $A \dot{\in} \Gamma$. For a sequent $\Gamma = A_1, \ldots, A_n$, we denote by $\Box\Gamma$ the sequent $\Box A_1, \ldots, \Box A_n$. A sequent $A_1, \ldots, A_n$ is called *satisfiable* if there exists a Kripke model in which all formulas $A_1, \ldots, A_n$ are true.

A sequent Γ is a called a *subsequent* of a sequent Δ if Γ is a submultiset of Δ. For example, the sequent a is a subsequent of the sequent a, a, b, but the sequent a, a is not a subsequent of the sequent a.

The *sequent calculus* Seq and an example derivation in this calculus are given in Figure 1. We call an *inference* in Seq any instance of an inference rule. A *derivation of a sequent* Γ is any tree formed by inferences and having Γ as the root, and a *refutation* Γ is any derivation of Γ whose leaves are axioms.

We will use the notions of inference, derivation, and refutation for all other calculi introduced in this article. We use the term "refutation" because the calculi introduced here are intended at establishing unsatisfiability of formulas and sequents.

THEOREM 2.1 *(Completeness of* Seq*). A sequent S is unsatisfiable if and only if it has a refutation in* Seq.

The proof of this theorem may be found in standard textbooks, e.g., Fitting [1983], or in Goble [1974].

Let us formulate some properties of Seq that easily follow from the analysis of inference rules of Seq or from the completeness theorem. We call an inference rule *invertible* if for every instance of this inference rule, if its conclusion has a refutation, then all its premises have refutations too.

The following properties of Seq are easy to prove:

LEMMA 2.2. *(i)* Seq *has the* subformula property*: every formula occurring in a derivation of a sequent Γ consists of subformulas of formulas of Γ. (ii) If a sequent Γ has a refutation, then any sequent Γ, Δ has a refutation too. (iii) The rules $(\wedge)$ and $(\vee)$ are invertible.*

3. THE INVERSE CALCULUS

The calculus Seq is well-known. It is straightforward to design satisfiability-checking procedures that apply the rules of Seq in the top-down way, from the goal to axioms (the tableau method). Much less known is the inverse method, based on a bottom-up version of Seq. In the inverse method, the sequent calculus depends on the goal formula G. Such a calculus, denoted by Inv^G, and an example refutation in this calculus are shown in Figure 2. Compare this refutation with that of Figure 1. Note that all formulas occurring in sequents of Inv^G are subformulas of G *by definition*.

The completeness of the inverse method is formulated as follows:

axioms: $p, \neg p$ $\qquad \dfrac{\Gamma, A, A}{\Gamma, A}\ (C)$

$$\frac{\Gamma, A}{\Gamma, A \wedge B}\ (\wedge_l) \qquad \frac{\Gamma, B}{\Gamma, A \wedge B}\ (\wedge_r)$$

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

$$\frac{\Gamma, A}{\Box\Gamma, \Diamond A}\ (\Diamond) \qquad \frac{\Gamma}{\Box\Gamma, \Diamond A}\ (\Diamond^+)$$

In the rules, all formulas occurring in the rules are subformulas of G. The rule (C) is called the *contraction rule*.

$$\cfrac{\cfrac{\cfrac{\cfrac{\cfrac{a, \neg a \quad b, \neg b}{a, \neg a \vee b, \neg b}\ (\vee)}{\Box a, \Box(\neg a \vee b), \Diamond\neg b}\ (\Diamond)}{\begin{array}{c}\Box a,\ \Box(\neg a \vee b) \wedge \Diamond\neg b \wedge \Diamond b,\\ \Box(\neg a \vee b) \wedge \Diamond\neg b \wedge \Diamond b\end{array}}\ (\wedge), (\wedge), (\wedge)}{\Box a, \Box(\neg a \vee b) \wedge \Diamond\neg b \wedge \Diamond b}\ (C)}{\Box a \wedge \Box(\neg a \vee b) \wedge \Diamond\neg b \wedge \Diamond b}\ (\wedge), (\wedge), (C)$$

Some sequences of inferences are shown as one inference.

Fig. 2. Sequent calculus Inv^G and a refutation in it.

THEOREM 3.1 *(Completeness of* Inv^G*). A formula G is unsatisfiable if and only if it has a refutation in* Inv^G.

The main technique used in the proof of this theorem is the so-called *Subsequent Lemma*.

LEMMA 3.2 *(Subsequent Lemma). Let G be a formula and Γ be a sequent consisting of subformulas of G and having a refutation in* Seq*. Then there exists a subsequent Δ of Γ that has a refutation in* Inv^G.

Completeness of Inv^G follows from this lemma immediately: indeed, if G is unsatisfiable, then by Completeness Theorem 2.1 for Seq, the sequent G has refutation in Seq, so by Subsequent Lemma, a subsequent of G has a refutation in Inv^G. The sequent Γ has only two subsequents: G itself, and the empty sequent, however the latter has obviously no refutation, so G has a refutation in Inv^G.

The Subsequent Lemma can be proved by a straightforward induction on derivations. We do not give a full proof here, because below in Section 6 we will prove a much stronger form of completeness (Theorem 6.2) in presence of a number of redundancies. However, we will illustrate an interesting point in the proof that gives rise to two rules for handling $\Diamond$ in Inv^G out of one rule in Seq.

Suppose a sequent $\Gamma, \Box\Delta, \Diamond A$ has refutation in Seq and that the last inference of this refutation is

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

Since the premise Δ, A has a shorter refutation in Seq, we may assume that a subsequent Δ' of Δ, A has a refutation in Inv^G. We have to find a refutation in Inv^G of a subsequent of $\Gamma, \Box\Delta, \Diamond A$. There are two possible cases, depending on whether A occurs in Δ' or not. If it does, then Δ' has the form Δ'', A where Δ'' is a subsequent of Δ; then we use the following inference in Inv^G to obtain a refutation of a subsequent of $\Gamma, \Box\Delta, \Diamond A$:

$$\frac{\Delta'', A}{\Box\Delta'', \Diamond A}\ (\Diamond).$$

If A does not occur in Δ', then Δ' is a subsequent of Δ, and we use the following inference in Inv^G to obtain a refutation of a subsequent of $\Gamma, \Box\Delta, \Diamond A$:

$$\frac{\Delta'}{\Box\Delta', \Diamond A}\ (\Diamond^+).$$

It is not hard to find an example showing that the rule $(\Diamond^+)$ is necessary for completeness. Consider the formula $G = \Box a \wedge \Box\neg a \wedge \Diamond b$. Its has a refutation in Inv^G:

$$\cfrac{\cfrac{\cfrac{\cfrac{\cfrac{\cfrac{\cfrac{a, \neg a}{\Box a, \Box\neg a, \Diamond b}\ (\Diamond^+)}{\Box a, \Box\neg a, \Box\neg a \wedge \Diamond b}\ (\wedge_r)}{\Box a, \Box\neg a \wedge \Diamond b, \Box\neg a \wedge \Diamond b}\ (\wedge_l)}{\Box a, \Box\neg a \wedge \Diamond b}\ (C)}{\Box a, \Box a \wedge \Box\neg a \wedge \Diamond b}\ (\wedge_r)}{\Box a \wedge \Box\neg a \wedge \Diamond b, \Box a \wedge \Box\neg a \wedge \Diamond b}\ (\wedge_l)}{\Box a \wedge \Box\neg a \wedge \Diamond b}\ (C)$$

The only axiom of Inv^G is $a, \neg a$, and the only inference applicable to this axiom is the inference $(\Diamond^+)$ used in this refutation (remember that the formula $\Diamond A$ in $(\Diamond^+)$ must be a subformula of G). Therefore, Inv^G is incomplete without the $(\Diamond^+)$ rule.

This example also shows necessity in the contraction rule (C). We could avoid the contraction rule by defining sequents as *sets* of formulas rather than multisets, but this would create some technical problems when we discuss prefix-subsumption in Section 7. Multisets are also more useful for designing inverse method-based procedures for first-order modal logics, but first-order logics are outside the scope of this article.

Note that the calculus Inv^G has the following properties:

1. Every formula occurring in a derivation is a subformula of G.
2. There exists a finite number of axioms.
3. For every sequent Γ, there exists only a finite number of ways of applying $(\wedge_l)$, $(\wedge_r)$, $(\Diamond)$, $(\Diamond^+)$, or (C) to Γ.
4. For every pair of sequents Γ and Δ, there exists only a finite number of ways of applying $(\vee)$ to Γ and Δ.

A typical proof procedure based on the inverse method uses these properties and works roughly as follows.

1. Let S be a set of sequents, initially $\{p, \neg p \mid p, \neg p \text{ are subformulas of } G\}$. Note that p must be an atomic formula in $\neg p$, since we use negation normal forms.[2]
2. Repeatedly apply all possible inference rules to sequents in S, adding their conclusions to S until no new sequents can be obtained or until a refutation of G is found.

[2]The restriction on p to be in negation normal form can be relaxed; in fact we can use more general axioms A, B, where B is equivalent to the negation of A. There are simple sufficient criteria to check that B is a equivalent to a negation of A. Such nonatomic axioms are used in the systems **DLP** and **FaCT** and in our current implementation of KK.

Evidently, this procedure terminates when the sequents are represented as sets of formulas, since the sequents consist of subformulas of G, and since there is only a finite number of such sequents. When the sequents are multisets, one can use subsumption to ensure termination. Even without subsumption, we can prove termination by using multiset orderings based on a relation $\succ$ defined below; see end of Section 6.

However, such naive procedures are hopelessly inefficient, unless they are augmented with powerful *redundancy criteria*. There are two kinds of redundancy criteria, allowing us to get rid of *redundant sequents* and *redundant inferences*. Intuitively, a sequent is redundant if it can be removed from the search space without losing completeness. An inference is redundant if the calculus obtained by dropping this inference is still complete. Very powerful techniques for proving redundancies have been developed in resolution-based calculi for classical logic, e.g., see Bachmair and Ganzinger [2000; Niewenhuis and Rubio [2000]. The use of such powerful criteria is one of the reasons for the efficiency of the system **TA** [Hustadt and Schmidt 1997; Schmidt 1998] that translates the goal formula into a fragment of first-order logic and uses a first-order theorem prover SPASS that implements many redundancy criteria.

Redundancy criteria can be incompatible. For example, it is possible that some sequents Γ and Δ are redundant, but every refutation contains at least one of these sequents. Therefore, one usually proves completeness for a collection of redundancy criteria. If we prove several redundancy criteria for a calculus, the proof-search by the inverse method works as follows.

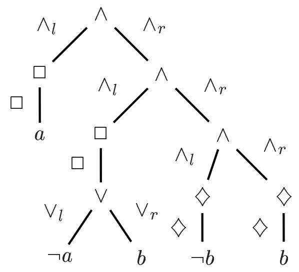
1. Let S be a set of sequents; initially $\{p, \neg p \mid p, \neg p$ are subformulas of $G\}$ *without the redundant sequents of this form*.
2. Repeatedly apply all *nonredundant* inferences to sequents in S, adding to S those conclusions of these inferences *that are nonredundant*.
3. Remove all sequents that *become redundant* due to the addition of these conclusions of inferences.

A number of redundancy criteria for various nonclassical logics are proved using a rather general technique in Voronkov [1992] (though many proofs are absent). There is a paper on the inverse method for S4 [Mints et al. 1996]. This paper is interesting because it discusses many redundancy criteria, but again the proofs of many statements are missing. In particular, Mints et al. [1996] take it for granted that some strategies *deleting* clauses from the search space (like subsumption) are complete, though no technique for proving this has been developed.

Here we present a different technique based on proving properties of refutations in Seq and transforming them into redundancy criteria for the inverse method using the Subsequent Lemma. In order to capture redundancies, we give a suitable reformulation of the sequent calculus as a *path calculus*.

4. THE PATH CALCULUS

For a number of reasons, we want now to speak about *occurrences* of subformulas in a goal formula G. Together with each occurrence of a subformula, we also want to keep information about Inv^G-inference rules that can be applied to the subformula



π	$E\vert_\pi$
ϵ	$\Box a \wedge \Box(\neg a \vee b) \wedge \Diamond\neg b \wedge \Diamond b$
$\wedge_l$	$\Box a$
$\wedge_l\Box$	a
$\wedge_r$	$\Box(\neg a \vee b) \wedge \Diamond\neg b \wedge \Diamond b$
$\wedge_r\wedge_l$	$\Box(\neg a \vee b)$
$\wedge_r\wedge_l\Box$	$\neg a \vee b$
$\wedge_r\wedge_l\Box\vee_l$	$\neg a$
$\wedge_r\wedge_l\Box\vee_r$	b
$\wedge_r\wedge_r$	$\Diamond\neg b \wedge \Diamond b$
$\wedge_r\wedge_r\wedge_l$	$\Diamond\neg b$
$\wedge_r\wedge_r\wedge_l\Diamond$	$\neg b$
$\wedge_r\wedge_r\wedge_r$	$\Diamond b$
$\wedge_r\wedge_r\wedge_r\Diamond$	b

Fig. 3. Paths and subformulas in the formula E.

corresponding to this occurrence. To achieve both aims, we introduce the notion of path in G.

A *path* is any finite sequence of symbols $\wedge_l$, $\wedge_r$, $\vee_l$, $\vee_r$, $\Box$, $\Diamond$. An example of a path is $\wedge_l\Diamond\Diamond\vee_r$. The empty path is denoted ϵ. Let G be a formula. The notion of *path in* G, or G-*path*, and the *subformula of* G *on a path* π, denoted $G|_\pi$, are defined inductively as follows.

1. The empty path ϵ is a G-path, and $G|_\epsilon$ is G.
2. Suppose that π is a G-path and $G|_\pi = F$. Then
   (a) If F has the form $F_1 \wedge F_2$, then $\pi\wedge_l$ and $\pi\wedge_r$ are G-paths, and $G|_{\pi\wedge_l} = F_1$, $G|_{\pi\wedge_r} = F_2$. In this case we call π a $\wedge$-*path*.
   (b) If F has the form $F_1 \vee F_2$, then $\pi\vee_l$ and $\pi\vee_r$ are G-paths, and $G|_{\pi\vee_l} = F_1$, $G|_{\pi\vee_r} = F_2$. In this case we call π a $\vee$-*path*.
   (c) If F has the form $\Box F_1$, then $\pi\Box$ is a G-path and $G|_{\pi\Box} = F_1$. In this case we call π a $\Box$-*path*.
   (d) If F has the form $\Diamond F_1$, then $\pi\Diamond$ is a G-path and $G|_{\pi\Diamond} = F_1$. In this case we call π a $\Diamond$-*path*.

We call a path π a *literal path* if $G|_\pi$ is a literal. Note that the notions like $\wedge$-path or literal path make sense only when we speak about G-paths for a fixed formula G, and make no sense for paths in general.

The notion of subformula on a path is illustrated in Figure 3.

Evidently, for each subformula F of G, there is a G-path π such that $G|_\pi = F$. The tableau calculus Seq can be turned into a calculus where occurrences of subformulas of G are replaced by the corresponding paths. We call a *path sequent* any finite multiset of paths. When no ambiguity arises, we will refer to path sequents simply as sequents. For a path sequent $\Pi = \pi_1, \ldots, \pi_n$ we denote by $\Pi\Box$ the path sequent $\pi_1\Box, \ldots, \pi_n\Box$. The *path calculus* Pat^G and a refutation in it are given in Figure 4. For any inference of Pat^G of Figure 4, the path π is called the *principal path* of this inference. The paths $\pi\wedge_l$, $\pi\wedge_r$, $\pi\vee_l$, $\pi\vee_r$, and $\pi\Diamond$ are called the *side paths* of the respective inference. We will use the notions of principal and side paths for some other calculi in this article.

The following *bisimulation lemma* is the key to proving completeness of the path axiom: Γ, π_1, π_2 $\qquad \dfrac{\Gamma, \pi\wedge_l, \pi\wedge_r}{\Gamma, \pi}\ (\wedge)$

$$\frac{\Gamma, \pi\vee_l \quad \Gamma, \pi\vee_r}{\Gamma, \pi}\ (\vee) \qquad \frac{\Pi\Box, \pi\Diamond}{\Gamma, \Pi, \pi}\ (\Diamond)$$

All paths occurring in path sequents are G-paths. In the axioms $G|_{\pi_1} = p$ and $G|_{\pi_2} = \neg p$ for some propositional variable p.

$$\cfrac{\begin{array}{c}\wedge_l\Box,\ \wedge_r\wedge_l\Box\vee_l, \\ \wedge_r\wedge_r\wedge_l\Diamond\end{array} \qquad \begin{array}{c}\wedge_l\Box,\ \wedge_r\wedge_l\Box\vee_r, \\ \wedge_r\wedge_r\wedge_l\Diamond\end{array}}{\cfrac{\wedge_l\Box,\ \wedge_r\wedge_l\Box,\ \wedge_r\wedge_r\wedge_l\Diamond}{\cfrac{\wedge_l,\ \wedge_r\wedge_l,\ \wedge_r\wedge_r\wedge_l,\ \wedge_r\wedge_r\wedge_r}{\epsilon}\ (\wedge),(\wedge),(\wedge)}\ (\Diamond)}\ (\vee)$$

A sequence of rules $(\wedge)$ is shown as one inference.

Fig. 4. Path sequent calculus Pat^G and a refutation in it.

calculus and various refinements thereof. Let us call the *formula image* of a path sequent or a derivation in Pat^G the figure obtained from this path sequent or derivation by replacing every path π by the formula $G|_\pi$.

LEMMA 4.1 *(Bisimulation Lemma for* Pat^G*). (i) Let D be a derivation in* Pat^G*. Then the formula image of D is a derivation of G in* Seq*. (ii) Let D' be a derivation of a sequent $A_1, \ldots, A_n$ in* Seq*, and $\pi_1, \ldots, \pi_n$ be paths such that $G|_{\pi_i} = A_i$, for all $i = 1, \ldots, n$. Then there exists a derivation D of $\pi_1, \ldots, \pi_n$ in* Pat^G *such that D' is the formula image of D. (iii) Both (i) and (ii) also hold if "derivation" is replaced by "refutation" everywhere.*

PROOF. (i) is proved by the straightforward analysis of inferences in Pat^G. (iii) easily follows from (i) and (ii). We will only show how to prove (ii). The proof is by induction on the number of inferences in D'. We consider only one case, when the last inference is $(\vee)$; other cases are similar. In this case without loss of generality we assume that $A_n = B \vee C$ and that A_n is the principal formula of the last inference. Then the derivation D' has the form

$$\frac{\begin{array}{c}\vdots\ D'_1 \\ A_1, \ldots, A_{n-1}, B\end{array} \quad \begin{array}{c}\vdots\ D'_2 \\ A_1, \ldots, A_{n-1}, C\end{array}}{A_1, \ldots, A_{n-1}, B \vee C}\ (\vee).$$

Evidently, $\pi_n\vee_l$ and $\pi_n\vee_r$ are paths in G. Applying the induction hypothesis to the derivations D'_1 and D'_2, we obtain derivations D_1 and D_2 such that D'_1 and D'_2 are their formula images. Define D as the derivation

$$\frac{\begin{array}{c}\vdots\ D_1 \\ \pi_1, \ldots, \pi_{n-1}, \pi_n\vee_l\end{array} \quad \begin{array}{c}\vdots\ D_2 \\ \pi_1, \ldots, \pi_{n-1}, \pi_n\vee_r\end{array}}{\pi_1, \ldots, \pi_{n-1}, \pi_n}\ (\vee).$$

It is not hard to argue that D' is the formula image of D. $\Box$

As a corollary of the Bisimulation Lemma and Completeness Theorem 2.1, we obtain

THEOREM 4.2 *(Completeness Theorem for* Pat^G*). A formula G is unsatisfiable if and only if ϵ has a refutation in* Pat^G*.*

PROOF. By the Completeness theorem for Seq, unsatisfiability of G is equivalent to the existence of a refutation of G in Seq.

$(\Rightarrow)$ Let ϵ has a refutation Pat^G. By the Bisimulation Lemma there exists a refutation in Seq of the formula $G|_\epsilon$, i.e., of G.

($\Leftarrow$) Let D' be a refutation of G in Seq. By the Bisimulation Lemma there exists a refutation D in Pat^G whose formula image is D'. Evidently, D is a refutation of ϵ. $\square$

5. REDUNDANCIES

Resolution-based calculi for classical logic become efficient only when they are equipped with powerful redundancy criteria, e.g., see Bachmair and Ganzinger [2000] and Niewenhuis and Rubio [2000]. Likewise, a naive implementation of the inverse calculus does not give us an efficient decision procedure. In order to optimize the inverse calculus, we will prove several statements about redundant sequents and derivations. To prove these statements, we will first establish several properties of derivations in the path calculus.

Let us call the *modal length* of a path the number of occurrences of $\Diamond$ and $\Box$ in the path. For example, the modal length of $\Box \wedge_l \Diamond\Diamond\vee_r$ is 3.

LEMMA 5.1. *Let Π be a path sequent occurring in a derivation of ϵ in Pat^G. Then all paths in Π have the same modal length.*

PROOF. First, observe the following fact. Let an inference in Pat^G have a conclusion Π satisfying the claim. Then all premises of this inferences satisfy the claim. Second, apply a straightforward induction on the number of inferences. $\square$

A path π' is called a *prefix* of a path π, denoted $\pi' \sqsubseteq \pi$, if $\pi = \pi'\pi''$ for some path π''. Evidently, if $\pi' \sqsubseteq \pi$ and π is a G-path, then π' is a G-path and $G|_\pi$ is a subformula of $G|_{\pi'}$. A prefix π' of π is called *proper* denoted $\pi' \sqsubset \pi$, if $\pi' \neq \pi$. A path sequent Π is called *prefix-free* if for every pair π_1, π_2 of different occurrences of paths in Π, π_1 is not a prefix of π_2. Note that a prefix-free sequent cannot contain two occurrences of the same path.

LEMMA 5.2 *(Prefix-Freedom). Any path sequent occurring in a derivation of ϵ is prefix-free.*

PROOF. As in the proof of Lemma 5.1, observe that this property holds for all premises of an inference, whenever it holds for its conclusion. $\square$

We say that two paths π_1 and π_2 form a $\vee$-*fork* if one of the following conditions is satisfied:

1. $\pi_1 = \pi \vee_l \pi_1'$, $\pi_2 = \pi \vee_r \pi_2'$ for some paths π, π_1', π_2'; or, symmetrically,
2. $\pi_1 = \pi \vee_r \pi_1'$, $\pi_2 = \pi \vee_l \pi_2'$ for some paths π, π_1', π_2'.

The paths π_1 and π_2 are called *diamond-separated* if $\pi_1 = \pi_1'\Diamond\pi_1''$ and $\pi_2 = \pi_2'\Diamond\pi_2''$ for some paths $\pi_1', \pi_2', \pi_1'', \pi_2''$ such that the paths π_1' and π_2' have the same modal length and such that $\pi_1' \neq \pi_2'$.

For example, consider the paths of Figure 3. The paths $\wedge_r \wedge_l \Box\vee_l$ and $\wedge_r \wedge_l \Box\vee_r$ form a $\vee$-fork. The paths $\wedge_r \wedge_r \wedge_l\Diamond$ and $\wedge_r \wedge_r \wedge_r\Diamond$ are diamond-separated.

LEMMA 5.3. *Let Π be a path sequent occurring in a derivation of ϵ in Pat^G. Then no pair of paths π_1, π_2 in Π forms a $\vee$-fork.*

PROOF. The proof is by contradiction. We assume that two paths π_1, π_2 occurring in Π form an $\vee$-fork. Without loss of generality we assume that $\pi_1 = \pi \vee_l \pi_1'$ and $\pi_2 = \pi \vee_r \pi_2'$ for some paths π, π_1', π_2'.

Consider the branch $\mathcal{B}$ of the derivation on which Π lies and the topmost sequents on $\mathcal{B}$ where $\pi\vee_l$ and $\pi\vee_r$ first appeared. Without loss of generality assume that $\pi\vee_l$ first appeared above $\pi\vee_r$ (the other case is symmetric). Then the derivation contains the subderivation

$$\dfrac{\dfrac{\Gamma_2, \pi\vee_l \qquad \dfrac{\dfrac{\Gamma_1, \pi\vee_l \quad \Gamma_1, \pi\vee_r}{\Gamma_1, \pi}\ (\vee)}{\vdots\atop \Gamma_2, \pi\vee_r}}{\Gamma_2, \pi}}{}\ (\vee).$$

Since Γ_1, π is above $\Gamma_2, \pi\vee_r$ in the derivation, Γ_2 must contain a prefix of π. This contradicts Lemma 5.2, since the path sequent $\Gamma_2, \pi\vee_r$ should be prefix-free. □

Lemma 5.4. *Let Π be a path sequent occurring in a derivation of ϵ in* Pat^G. *Then no pair of paths π_1, π_2 in Π is diamond-separated.*

Proof. The proof is by contradiction. We assume that a pair of paths π_1, π_2 occurring in Π is diamond-separated. Then $\pi_1 = \pi_1'\Diamond\pi_1''$, $\pi_2 = \pi_2'\Diamond\pi_2''$, $\pi_1' \neq \pi_2'$, and paths π_1' and π_2' have the same modal length L.

Consider the branch $\mathcal{B}$ of derivation on which Π lies and the topmost sequents on $\mathcal{B}$ where $\pi_1'\Diamond$ and $\pi_2'\Diamond$ first appeared. Without loss of generality assume that $\pi_1'\Diamond$ first appeared above $\pi_2'\Diamond$ (the other case is symmetric). Then the derivation contains a subderivation

$$\dfrac{\dfrac{\Gamma_1\Box, \pi_1'\Diamond}{\Gamma_1, \pi_1', \Delta_1}\ (\Diamond)}{}\qquad \begin{array}{c}\vdots\end{array}\qquad \dfrac{\Gamma_2\Box, \pi_2'\Diamond}{\Gamma_2, \pi_2', \Delta_2}\ (\Diamond).$$

Since π_2' has the modal length L, it is not hard to argue that all path sequents between $\Gamma_2\Box, \pi_2'\Diamond$ and $\Gamma_1, \pi_1', \Delta_1$ contain a formula with the modal length at least $L + 1$. Therefore, the sequent $\Gamma_1, \pi_1', \Delta_1$ contains a path of the modal length at least $L+1$ and the path π_1' with the modal length L. This contradicts Lemma 5.1, since all paths in a path sequent must have the same modal length. □

In view of Lemmas 5.3 and 5.4 we say that a pair of paths π_1, π_2 are *in conflict* if they either form a $\vee$-fork or are diamond-separated.

The lemmas above assert some properties of all possible derivations in the path calculus. Now we want to restrict search for refutation to only a subset of derivations by introducing an ordering on the set of all G-paths.

In ordered resolution for classical logic, one can order literals in clauses and require that a resolution inference be only applied when the greatest literals in both clauses are resolved. We want to introduce a similar restriction in the modal case, by modifying classical literal orderings to their modal equivalent, called a *G-ordering*. In the classical case, we can use *any* ordering on subformulas of G (or on G-paths) that respects the prefix relation. In the modal case, the situation is more complex, as we will now see: not every ordering on paths preserves completeness. For example, consider the top inference of the derivation of Figure 4:

$$\dfrac{\begin{matrix}\wedge_l \Box,\ \wedge_r \wedge_l \Box\vee_l, \\ \wedge_r \wedge_r \wedge_l \Diamond\end{matrix} \qquad \begin{matrix}\wedge_l \Box,\ \wedge_r \wedge_l \Box\vee_r, \\ \wedge_r \wedge_r \wedge_l \Diamond\end{matrix}}{\wedge_l \Box,\ \wedge_r \wedge_l \Box,\ \wedge_r \wedge_r \wedge_l \Diamond}\ (\vee)$$

It is essential in that derivation that an $(\vee)$-inference is applied above the rule $(\Diamond)$, because the rule $(\Diamond)$ is not applicable to the top sequents. However, if we impose an ordering on paths in which $\wedge_r \wedge_l \Box\vee_l$ is the smallest path in the first premise, we will rule out the possibility of applying $(\vee)$ first, and no proof will be found. So, the definition of G-ordering is more complex than in the classical case.

We call two paths *brothers* if one of them has the form $\pi\wedge_l$ and the other $\pi\wedge_r$, or likewise, if one of them has the form $\pi\vee_l$ and the other $\pi\vee_r$. For example, the following two paths in our example formula E are brothers: $\wedge_r \wedge_l \Box\vee_l$ and $\wedge_r \wedge_l \Box\vee_r$, the formulas on this path are the two immediate subformulas of the disjunction $\neg a \vee b$. In general, every conjunction and disjunction in a formula gives rise to a pair of brothers.

We denote by the generic symbol $\mathbin{\wedge\!\!\!\!\vee}$ any of the symbols $\wedge$, $\vee$. Likewise, we denote by $*$ any of the symbols l, r and use notation like $\wedge_*$ or $\vee_*$. Thus $\mathbin{\wedge\!\!\!\!\vee}_*$ denotes any of the symbols $\wedge_l$, $\wedge_r$, $\vee_l$, $\vee_r$. We also denote by the generic symbol $\Diamond\!\!\!\!\Box$ any of the symbols $\Diamond$, $\Box$.

Let us fix a formula G. We call a *G-ordering* any total order $\succ$ on the set of all G-paths satisfying the following conditions:

1. $\pi_1 \succ \pi_2$, whenever
   (a) the modal length of π_1 is strictly greater than the modal length of π_2; or
   (b) π_1, π_2 have the same modal length, the last symbol of π_1 is $\mathbin{\wedge\!\!\!\!\vee}_*$, and the last symbol of π_2 is $\Diamond\!\!\!\!\Box$; or
   (c) π_1, π_2 have the same modal length and $\pi_2 \sqsubset \pi_1$. (Note that this case and the previous one are not exclusive, but they do not contradict each other.)

2. There is no path between two brothers, i.e., there exists no G-paths π_1, π_2, π_3 such that $\pi_1 \succ \pi_2 \succ \pi_3$ and π_1, π_3 are brothers.

The intuitive meaning of $\succ$ is that we should first apply rules to the formulas greater with respect to $\succ$. The relation $\succ$ is also defined in such a way that the conclusion of any rule is less than both its premises in the multiset ordering on sequents induced by $\succ$. Condition (1a) ensures that the conclusion is less than the premise when the rule $(\Diamond)$ (or the rule $(\Diamond^+)$ defined later) are applied. Condition (1b) is due to the fact that application of $(\Diamond)$ or $(\Diamond^+)$ to a sequent containing a path $\pi\mathbin{\wedge\!\!\!\!\vee}_*$ may give an incomplete calculus. Conditions (1c) and (2) are more technical and are used in some proofs of this article.

It will be clear why these conditions are required of G-orderings when we prove Lemma 5.8 below.

To illustrate G-orderings, we will show one possible E-ordering (recall that E is our example goal formula). The paths in E were illustrated in Figure 3. In Figure 5, we show these paths again; this time each path is labeled as π_i, where i is a natural number. We claim that the ordering given by $\pi_i \succ \pi_j$ if $i > j$ is an

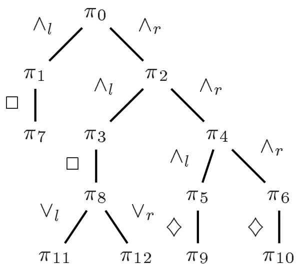


Fig. 5. A E-ordering

E-ordering. Indeed, the conditions (1a)–(1c) of G-orderings are easy to check. We leave it to the reader to verify that condition (2) holds, too.

We write $\pi_1 \succeq \pi_2$ to express that $\pi_1 \succ \pi_2$ or $\pi_1 = \pi_2$.

In addition to having more complex conditions on orderings for the modal case, another problem will appear in the proof of completeness. For classical logic, one can prove completeness of ordered resolution by purely semantical considerations, e.g., see Bachmair and Ganzinger [2000]. In the modal case, we want to also prove that the strategy of selecting the greatest formula (or path) in a clause is compatible with the redundancies considered so far, so we will follow the same pattern of proof: first prove a property of derivations in the tableau calculus, and then transfer it to the inverse calculus using a variant of the Subsequent Lemma. However, unlike the redundancies considered so far, such a proof of completeness will meet a technical obstacle whose essence can be expressed shortly as follows: *the ordering requirement is formulated in terms of premises of an inference, while the completeness proofs for the tableau calculus expand proofs from conclusions.* This obstacle leads to a slightly complex definition of the ordering on paths.

We will now prove that G-orderings exist for every G. To this end we will describe an *ordering procedure* that orders G-paths and note that the resulting ordering is a G-ordering.

The procedure works with a sequence of sets of paths; this sequence is denoted $S_n \succ S_{n-1} \succ \ldots \succ S_0$. The intuitive meaning of the sequence is that for all $\pi \in S_i$ and $\pi' \in S_{i-1}$ we have $\pi \succ \pi'$. The paths belonging to the same S_i are yet unordered (but will be ordered later). At each step we will pick up one set S_i in the sequence containing one or more members and replace S_i by two or more sets $S_{i1} \succ \ldots \succ S_{ik}$ such that $S_{i1} \cup \ldots \cup S_{ik} = S_i$. The procedure terminates when each set contains exactly one member. An ordering procedure is shown in Figure 6. Note that some sets obtained by the procedure (for example L or R) may be empty; in this case we do not include them in the sequence. When the procedure terminates, the sequence consists of singleton sets. We then let $\pi \succ \pi'$ if the sequence has the form $\ldots\{\pi\} \succ \ldots \succ \{\pi'\}\ldots$.

We show how the E-ordering of Figure 5 can be obtained by this procedure. The splitting steps will be denoted by $\rightarrow$ superscribed by the line number in the algorithm. Instead of singleton sets $\{\pi_i\}$ we write π_i.

Initially, define S_i to be the set of paths in G of modal length i;	/* 1 */
for all S_i except the last one **do**	/* 2 */
let $\pi_1, \ldots, \pi_n$ be all paths in S_i ending at $\davidsstar$;	/* 3 */
split S_i into $S_i - \{\pi_1, \ldots, \pi_n\} \succ \{\pi_1\} \succ \ldots \succ \{\pi_n\}$;	/* 4 */
split S_0 into $S_0 - \{\epsilon\} \succ \{\epsilon\}$;	/* 5 */
while there exists S_i with more than one member **do**	/* 6 */
let $\pi\mathbb{X}_l, \pi\mathbb{X}_r$ be two brothers in S_i such that $\pi \notin S_i$;	/* 7 */
let L and R be the sets of all prefixes of respectively $\pi\mathbb{X}_l$ and $\pi\mathbb{X}_r$ in S_i ;	/* 8 */
split S_i into $S_i - (L \cup R) \succ R \succ L \succ \{\pi\mathbb{X}_r\} \succ \{\pi\mathbb{X}_l\}$;	/* 9 */

Fig. 6. An ordering procedure.

$$
\begin{array}{l}
\{\pi_7, \pi_8, \pi_9, \pi_{10}, \pi_{11}, \pi_{12}\} \succ \{\pi_0, \pi_1, \pi_2, \pi_3, \pi_4, \pi_5, \pi_6\} \rightarrow^4 \\
\{\pi_{11}, \pi_{12}\} \succ \pi_{10} \succ \pi_9 \succ \pi_8 \succ \pi_7 \succ \{\pi_0, \pi_1, \pi_2, \pi_3, \pi_4, \pi_5, \pi_6\} \rightarrow^5 \\
\{\pi_{11}, \pi_{12}\} \succ \pi_{10} \succ \pi_9 \succ \pi_8 \succ \pi_7 \succ \{\pi_1, \pi_2, \pi_3, \pi_4, \pi_5, \pi_6\} \succ \pi_0 \rightarrow^9 \\
\pi_{12} \succ \pi_{11} \succ \pi_{10} \succ \pi_9 \succ \pi_8 \succ \pi_7 \succ \{\pi_1, \pi_2, \pi_3, \pi_4, \pi_5, \pi_6\} \succ \pi_0 \rightarrow^9 \\
\pi_{12} \succ \pi_{11} \succ \pi_{10} \succ \pi_9 \succ \pi_8 \succ \pi_7 \succ \{\pi_3, \pi_4, \pi_5, \pi_6\} \succ \pi_2 \succ \pi_1 \succ \pi_0 \rightarrow^9 \\
\pi_{12} \succ \pi_{11} \succ \pi_{10} \succ \pi_9 \succ \pi_8 \succ \pi_7 \succ \{\pi_5, \pi_6\} \succ \pi_4 \succ \pi_3 \succ \pi_2 \succ \pi_1 \succ \pi_0 \rightarrow^9 \\
\pi_{12} \succ \pi_{11} \succ \pi_{10} \succ \pi_9 \succ \pi_8 \succ \pi_7 \succ \pi_6 \succ \pi_5 \succ \pi_4 \succ \pi_3 \succ \pi_2 \succ \pi_1 \succ \pi_0.
\end{array}
$$

LEMMA 5.5. *Every ordering produced by an ordering procedure on G is a G-ordering.*

PROOF. First, we have to ensure that the procedure indeed terminates when all sets are singleton. To this end we have to show that at line 7 we can always choose two brothers $\pi\mathbb{X}_l, \pi\mathbb{X}_r \in S_i$ such that $\pi \notin S_i$. This follows from the following observation.

(2) *Every nonsingleton set S_i obtained after line 5 is made in the following way: there exists a set of paths $\pi'_1, \ldots, \pi'_n$ of the same modal length such that π'_i is not a prefix of π'_j for $i \neq j$ and*

$$S_i = \{\pi'' \mid \textit{for some } k,\ \pi'_k \sqsubset \pi'' \textit{ and } \pi'', \pi'_k \textit{ have the same modal length}\}.$$

Such a set always contains the required pair of brothers, namely those of the form $\pi'_k\mathbb{X}_l$ and $\pi'_k\mathbb{X}_r$.

Second, we have to show that the ordering satisfies all conditions of the definition of G-orderings. Condition 1a is satisfied at line 1 of the algorithm. Condition 1b is satisfied at lines 2–4.

Let us check condition (1c). It is obviously satisfied at line 1 when π_1, π_2 have different modal length and at line 5 when $\pi_2 = \epsilon$. So we assume π_1, π_2 have the same modal length. Then this condition can only be violated at line 9. We show that it is not the case. Indeed, from (2) it follows that (i) no path in $S_i - (L \cup R)$ is a prefix of a path in $L \cup R \cup \{\pi\mathbb{X}_r, \pi\mathbb{X}_l\}$; (ii) no path in R is a prefix of a path in $L \cup \{\pi\mathbb{X}_r, \pi\mathbb{X}_l\}$; (iii) no path in L is a prefix of either $\pi\mathbb{X}_r$ or $\pi\mathbb{X}_l$; (iv) $\pi\mathbb{X}_r$ is not a prefix of $\pi\mathbb{X}_l$.

Condition (2) is obviously satisfied at line 9. □

Since any run of the ordering procedure results in a G-ordering, we obtain the following corollary.

COROLLARY 5.6. *G-orderings exist.*

In practice, the time needed to run the ordering procedure is negligible compared to the time needed for satisfiability checking (and even compared to the time consumed by other preprocessing steps in KƎ).

The notion of G-ordering is introduced in order to prove the existence of refutations of a special form related to these orderings and therefore to restrict the search space to such refutations only.

For a path π and a path sequent Γ we write $\pi \succ \Gamma$ to denote $\pi \succ \pi'$ for every π' in Γ.

A ($\vee$)-inference in Pat^G

$$\frac{\Gamma, \pi\vee_l \quad \Gamma, \pi\vee_r}{\Gamma, \pi}\ (\vee)$$

is said to *respect a G-ordering* $\succ$ if $\pi\vee_l \succ \Gamma$ and $\pi\vee_r \succ \Gamma$. Likewise, a ($\wedge$)-inference in Pat^G

$$\frac{\Gamma, \pi\wedge_l, \pi\wedge_r}{\Gamma, \pi}\ (\wedge)$$

is said to respect $\succ$ if $\pi\wedge_l \succ \Gamma$ and $\pi\wedge_r \succ \Gamma$. A derivation in Pat^G is said to respect $\succ$ if every ($\vee$) and every ($\wedge$)-inference of this derivation respect $\succ$.

LEMMA 5.7. *Let G be a unsatisfiable formula and $\succ$ be a G-ordering. Then there exists a refutation of ϵ in* Pat^G *that respects $\succ$.*

It is not immediately obvious how to prove this lemma. If we try to prove this lemma by induction, starting with the empty derivation of the empty path and trying to extend the derivation from conclusion to premises, we may be in a "deadlock" situation: i.e., obtain a path sequent Γ such that *every* ($\wedge$)- and ($\vee$)-inference having Γ as the conclusion does not respect $\succ$. Let us illustrate such a deadlock situation. Suppose that $\pi_1\Box$ and $\pi_2\vee_l$ are G-paths such that π_1, π_2 have the same modal length and $\pi_1 \succ \pi_2\vee_l$. Consider the sequent π_1, π_2. We can apply a ($\vee$)-inference in the tableau path calculus to obtain this sequent:

$$\frac{\pi_1, \pi_2\vee_l \quad \pi_1, \pi_2\vee_r}{\pi_1, \pi_2}\ (\vee).$$

In fact, this inference is *the only* way to derive π_1, π_2. However, this inference does not respect $\succ$.

The conditions on an inference to respect $\succ$ are formulated in terms of the *premises* of the inference, but in such a inductive proof we can only extend the *conclusion* by a new inference. To overcome this problem, we will try to avoid path sequents that bring us into a deadlock, by restricting ourselves to the so-called *$\succ$-compact path sequents* defined below.

Let $\pi_1, \ldots, \pi_n$ be paths in G and $\succ$ be a G-ordering. The path sequent $\Gamma = \pi_1, \ldots, \pi_n$ is called *$\succ$-compact* if for every $i = 1, \ldots, n$,

1. if π_i is a $\wedge$-path, then $\pi_i\wedge_l \succ \Gamma$ and $\pi_i\wedge_r \succ \Gamma$;

2. likewise, if π_i is a $\vee$-path, then $\pi_i \vee_l \succ \Gamma$ and $\pi_i \vee_r \succ \Gamma$.

Using our generic notation, we can reformulate the definition of $\succ$-compactness as "for every $\bowtie$-path π_i in Γ we have $\pi_i \bowtie_* \succ \Gamma$." Note that $\pi_i \bowtie_* \succ \pi_i$ is guaranteed by the conditions on G-orderings; therefore the requirement $\pi_i \bowtie_* \succ \Gamma$ can be replaced by $\pi_i \bowtie_* \succ \Gamma \setminus \{\pi_i\}$.

The next lemma asserts that a compact sequent Γ cannot lead to a deadlock.

LEMMA 5.8. *Let $\succ$ be a G-ordering. Then*

1. *the premise of every $(\Diamond)$-inference is $\succ$-compact.*

   Furthermore, let Γ be a $\succ$-compact sequent occurring in a derivation of ϵ. Then

2. *every $(\bowtie)$-inference having Γ as the conclusion, respects $\succ$;*
3. *if Γ contains at least one $\wedge$-path or $\vee$-path, then there exists a $(\bowtie)$-inference all whose premises are $\succ$-compact.*

PROOF. 1. We prove: *the premise of every $(\Diamond)$-inference*

$$\frac{\Pi\Box, \pi\Diamond}{\Gamma, \Pi, \pi}\ (\Diamond)$$

is $\succ$-compact, i.e., for every π' in the premise, if $\pi' \bowtie_*$ is a G-path, then $\pi' \bowtie_* \succ \Pi\Box, \pi\Diamond - \{\pi'\}$. Let us prove $\pi\Diamond\bowtie_* \succ \Pi\Box$, the case when $\pi' \in \Pi\Box$ is similar. By Lemma 5.1, all paths in $\Pi\Box, \pi\Diamond$ are of the same modal length, so $\pi\Diamond\bowtie_*$ is of the same modal length as any path in $\Pi\Box$. By condition (1b) of the definition of G-ordering we have $\pi\Diamond\bowtie_* \succ \Pi\Box$.

2. We prove: *every $(\bowtie)$-inference having Γ as the conclusion, respects $\succ$.* Consider the case of $(\wedge)$-inferences, the case of $(\vee)$ is similar. Then Γ is of the form Π, π and the inference is

$$\frac{\Pi, \pi\wedge_l, \pi\wedge_r}{\Pi, \pi}\ (\wedge).$$

Since Π, π is $\succ$-compact, $\pi\wedge_l \succ \Pi, \pi$ and $\pi\wedge_r \succ \Pi, \pi$, therefore the inference respects $\succ$.

3. We assume that Γ contains at least one $\bowtie$-path and prove: *there exists a $(\bowtie)$-inference all whose premises are $\succ$-compact.* Consider the set of paths

$$S = \{\pi\bowtie_* \mid \pi \in \Gamma \text{ and } \pi\bowtie_* \text{ is a } G\text{-path}\}.$$

By the assumption, this set is nonempty. Since $\succ$ is a linear ordering, this set has the least element with respect to $\succ$. Without loss of generality we assume that this path is $\pi\wedge_l$. Then Γ has the form Π, π for some path sequent Π. Consider the inference

$$\frac{\Pi, \pi\wedge_l, \pi\wedge_r}{\Pi, \pi}\ (\wedge).$$

We claim that the premise of this inference is $\succ$-compact. This amounts to proving that (3a) $\pi \wedge_* \bowtie_* \succ \Pi$, whenever $\pi \wedge_* \bowtie_*$ is a G-path; (3b) for every path $\pi' \in \Pi$ we have $\pi' \bowtie_* \succ \Pi, \pi\wedge_*$, whenever $\pi'\bowtie_*$ is a G-path; and (3c) $\pi \wedge_l \bowtie_* \succ \pi\wedge_r$ and $\pi \wedge_l \bowtie_* \succ \pi\wedge_r$. Let us prove this.

(a) Since Π, π is $\succ$-compact, $\pi\wedge_* \succ \Pi$. Evidently, $\pi\wedge_*$ is a prefix of $\pi \wedge_* ⩙_*$; thus by the definition of G-orderings (condition (1c)) we have $\pi \wedge_* ⩙_* \succ \pi\wedge_*$. This implies $\pi \wedge_* ⩙_* \succ \Pi$.

(b) Since Π, π is compact, $\pi' ⩙_* \succ \Pi$, so it remains to prove $\pi' ⩙_* \succ \pi\wedge_*$, i.e., $\pi' ⩙_* \succ \pi\wedge_l$ and $\pi' ⩙_* \succ \pi\wedge_r$. Note that $\pi' ⩙_* \in S$ and that $\pi\wedge_l$ is the least element of S, therefore $\pi' ⩙_* \succeq \pi\wedge_l$. By Prefix-Free Lemma 5.2, $\pi' ⩙_*$ and $\pi\wedge_l$ cannot coincide; thus $\pi' ⩙_* \succ \pi\wedge_l$. Again by this lemma, $\pi' ⩙_*$ cannot be a prefix of $\pi\wedge_l$. In addition, we have $\pi' ⩙_* \succ \pi\wedge_l$, and $\pi\wedge_l$ is a brother to $\pi\wedge_r$; then by condition (2) of the definition of G-ordering, $\pi' ⩙_* \succ \pi\wedge_r$.

(c) We will only prove $\pi \wedge_l ⩙_* \succ \pi\wedge_r$, the second statement is symmetric. Suppose, by contradiction, $\pi\wedge_r \succ \pi \wedge_l ⩙_*$. Since $\succ$ respects the prefix relation, we also have $\pi\wedge_r \succ \pi \wedge_l ⩙_* \succ \pi\wedge_l$. This contradicts condition (2) of the definition of G-orderings, since the path $\pi \wedge_l ⩙_*$ is between two brothers. $\square$

This proof explains the conditions used in the definition of G-orderings. The only condition that has not been used is 1a, and indeed this condition can be omitted. This condition ensures that in the multiset ordering on sequents induced by $\succ$ the premise of any rule of Seq is greater than its conclusion.

Now we can prove Lemma 5.7: if G is unsatisfiable, then there exists a refutation of ϵ in Pat^G that respects $\succ$.

Proof of Lemma 5.7. The lemma is proved by induction. Starting with the path sequent ϵ, we will construct, step by step, a derivation D of ϵ such that (i) D consists only of $\succ$-compact path sequents, (ii) the formula images of all top sequents of D are unsatisfiable, until we obtain a refutation. Lemma 5.8 guarantees that the obtained refutation respects $\succ$. At every step we choose any top path sequent Π of D that is not an axiom and extend the derivation by adding an inference having Π as its conclusion.

1. (Induction base.) *The sequent ϵ is $\succ$-compact.* This is obvious.
2. (Induction step). Denote by Γ the formula image of Π. We have that Γ is unsatisfiable and not an axiom, and Π is $\succ$-compact. We show how to find an inference with the conclusion Π and premises satisfying the same properties.
   (a) If Γ contains any conjunction or disjunction, then by Lemma 5.8 there exists an ($⩙$)-inference having Π as the conclusion and $\succ$-compact premises. Since ($⩙$)-rules are invertible (Lemma 2.2), the formula images of these premises are also unsatisfiable.
   (b) If Γ contains neither conjunction nor disjunction and Γ is unsatisfiable, then there is a ($\Diamond$)-inference in Seq with the conclusion Γ and unsatisfiable premise Γ'. It is not hard to argue that there exists an ($\Diamond$)-inference in Pat^G with a premise Π' such that Γ' is the formula image of Π'. By Lemma 5.8, Π' is $\succ$-compact.

The termination of this construction follows from the following observation: every infinite construction would yield an infinite derivation in Seq which does not exist. $\square$

axioms: π_1, π_2 $\qquad \dfrac{\Gamma, \pi, \pi}{\Gamma, \pi}\ (C)$

$\dfrac{\Gamma, \pi\wedge_l}{\Gamma, \pi}\ (\wedge_l) \qquad \dfrac{\Gamma, \pi\wedge_r}{\Gamma, \pi}\ (\wedge_r)$

$\dfrac{\Gamma, \pi\vee_l \quad \Delta, \pi\vee_r}{\Gamma, \Delta, \pi}\ (\vee)$

$\dfrac{\Gamma\Box, \pi\Diamond}{\Gamma, \pi}\ (\Diamond) \qquad \dfrac{\Gamma\Box}{\Gamma, \pi}\ (\Diamond^+)$

All paths occurring in path sequents are G-paths. In the rule (ax) we have $G|_{\pi_1} = p$ and $G|_{\pi_2} = \neg p$ for some propositional variable p. In the rule $(\Diamond^+)$, π is an $\Diamond$-path.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\wedge_l\Box,\ \wedge_r\wedge_l\Box\vee_l \qquad \wedge_r\wedge_l\Box\vee_r,\ \wedge_r\wedge_r\wedge_l\Diamond}{\wedge_l\Box,\ \wedge_r\wedge_l\Box,\ \wedge_r\wedge_r\wedge_l\Diamond}\ (\vee)}{\wedge_l,\ \wedge_r\wedge_l,\ \wedge_r\wedge_r\wedge_l}\ (\Diamond)}{\wedge_l,\ \wedge_r,\ \wedge_r}\ (\wedge),(\wedge),(\wedge)}{\wedge_l,\ \wedge_r}\ (C)}{\epsilon}\ (\wedge),(\wedge),(C)$$

Some sequences of inferences are shown as one inference.

Fig. 7. Sequent calculus IP^G and a refutation in it

6. INVERSE PATH CALCULUS

In this section we define the inverse version of the path calculus, obtained by turning Inv^G into a path calculus.

The *inverse path calculus* IP^G and a refutation in it are given in Figure 7. For any inference of IP^G of Figure 7, the path π is called the *principal path* of this inference.

The same arguments as in Bisimulation Lemma 4.1 prove the following lemma.

LEMMA 6.1 *(Bisimulation Lemma for* IP^G*).*
(i) Let D be a derivation in IP^G*. Then the formula image of D is a derivation of G in* Inv^G*. (ii) Let D' be a derivation of a sequent $A_1, \ldots, A_n$ in* Inv^G*, and $\pi_1, \ldots, \pi_n$ be paths such that $G|_{\pi_i} = A_i$, for all $i = 1, \ldots, n$. Then there exists a derivation D of $\pi_1, \ldots, \pi_n$ in* IP^G *such that D' is the formula image of D. (iii) Both (i) and (ii) also hold if "derivation" is replaced by "refutation" everywhere.*

The calculus IP^G has different rules for handling $\wedge$, so we change the definition of derivations respecting a G-ordering in the following way.

A derivation in IP^G is said to *respect a G-ordering* $\succ$ if

1. for every $(\vee)$ inference of this derivation
$$\dfrac{\Gamma, \pi\vee_l \quad \Delta, \pi\vee_r}{\Gamma, \Delta, \pi}\ (\vee)$$
we have $\pi\vee_l \succ \Gamma$ and $\pi\vee_r \succ \Delta$;
2. likewise, for every $(\wedge_l)$ inference (respectively $(\wedge_r)$ inference) of this derivation
$$\dfrac{\Gamma, \pi\wedge_l}{\Gamma, \pi}\ (\wedge_l) \qquad \left(\quad \text{respectively} \quad \dfrac{\Gamma, \pi\wedge_r}{\Gamma, \pi}\ (\wedge_r) \quad\right)$$
we have $\pi\wedge_l \succ \Gamma$ (respectively $\pi\wedge_r \succ \Gamma$).

The next theorem is the main theorem of this article. It asserts that a formula is unsatisfiable if and only if it has a refutation satisfying several redundancy criteria. Namely, we prove that any sequent containing a conflicting pair is redundant, and that any inference not respecting $\succ$ is redundant as well. Later, we will give an even stronger version of this statement related to the subsumption rule, but we will need to introduce a new notion of derivation first.

We denote by $\mathrm{IP}^{G,\succ}$ the calculus obtained from IP^G by (i) removal of all path sequents containing paths of different modal lengths, (ii) removal of all path sequents containing a conflicting pair of paths, (iii) removal of all inferences that do not respect $\succ$.

THEOREM 6.2 *(Completeness of* $\mathrm{IP}^{G,\succ}$*). A formula G is unsatisfiable if, and only if, ϵ has a refutation in the calculus* $\mathrm{IP}^{G,\succ}$.

The key lemma in the proof of Theorem 6.2 is as follows.

LEMMA 6.3 *(Subsequent Lemma for* $\mathrm{IP}^{G,\succ}$*). Suppose that D is a refutation of ϵ in* Pat^G *that respects $\succ$ and I is an inference in D of the form*

$$\frac{\Gamma_1 \quad \cdots \quad \Gamma_n}{\Gamma},$$

and path sequents $\Delta_1, \ldots, \Delta_n$ are such that each Δ_i is a subsequent of Γ_i. Then there exists a derivation D' of Δ in $\mathrm{IP}^{G,\succ}$ *from $\Delta_1, \ldots, \Delta_n$ such that Δ is a subsequent of Γ.*

PROOF. We will give a proof by case analysis on the inference I, after having noted one general property of $\Delta, \Delta_1, \ldots, \Delta_n$.

By Lemmas 5.3 and 5.4 each path sequent among $\Gamma_1, \ldots, \Gamma_n$ contains no conflicting pair of paths. Since $\Delta_1, \ldots, \Delta_n$ are submultisets of $\Gamma_1, \ldots, \Gamma_n$, respectively, we obtain that none of $\Delta_1, \ldots, \Delta_n$ contains a conflicting pair of paths. Likewise, if we find a derivation of Δ from $\Delta_1, \ldots, \Delta_n$, since Δ is a submultiset of Γ, we obtain that Δ contains no conflicting pair of paths. In a similar way we can check that it contains no path sequent with a pair of paths of different modal length. Therefore, it is enough to check that the intermediate sequents of D' do not violate the condition, which will be obvious in all cases.

Therefore, we will only show how to find the derivation D' and prove that it respects $\succ$. Consider all possible cases of inferences in Pat^G.

1. The inference I is an axiom Γ, π_1, π_2 such that $G|_{\pi_1} = \neg G|_{\pi_2}$. Then π_1, π_2 is a subsequent of Γ, π_1, π_2, and is an axiom of $\mathrm{IP}^{G,\succ}$. Therefore, we can let D' be the derivation consisting of one axiom π_1, π_2. Evidently, it respects $\succ$.
2. The inference I has the form

$$\frac{\Pi, \pi\wedge_l, \pi\wedge_r}{\Pi, \pi}\ (\wedge).$$

Note that $\pi\wedge_l, \pi\wedge_r \succ \Pi$, because D respects $\succ$. We assume that some sequent Δ_1 is a subsequent of $\Pi, \pi\wedge_l, \pi\wedge_r$ and find a derivation in $\mathrm{IP}^{G,\succ}$ of a subsequent of Π, π from Δ_1.

Consider the following possible cases:

(a) *Case: Δ_1 contains neither $\pi\wedge_l$, nor $\pi\wedge_r$.* In this case Δ_1 is a subsequent of Π. Then Δ_1 is also a subsequent of Π, π, and the empty derivation of Δ_1 from Δ_1 proves the case.

(b) *Case:* Δ_1 *contains* $\pi\wedge_l$*, but not* $\pi\wedge_r$*.* In this case $\Delta_1 = \Delta', \pi\wedge_l$ for some subsequent Δ' of Π. The following derivation proves the case:

$$\frac{\Delta', \pi\wedge_l}{\Delta', \pi}\ (\wedge_l).$$

Since $\pi\wedge_l, \pi\wedge_r \succ \Pi$ and Δ' is a submultiset of Π, we have $\pi\wedge_l \succ \Delta'$. The ordering condition can be checked in the similar way for all other cases; we do not consider it any more.

(c) *Case:* Δ_1 *contains* $\pi\wedge_r$*, but not* $\pi\wedge_l$*.* This case is symmetric to the previous one.

(d) *Case:* Δ_1 *contains both* $\pi\wedge_l$ *and* $\pi\wedge_r$*.* In this case $\Delta_1 = \Delta', \pi\wedge_l, \pi\wedge_r$ for some submultiset Δ' of Π. Without loss of generality assume $\pi\wedge_l \succ \pi\wedge_r$ (and hence $\pi\wedge_l \succ \Delta', \pi\wedge_r$). The following derivation proves the case:

$$\cfrac{\cfrac{\cfrac{\Delta', \pi\wedge_l, \pi\wedge_r}{\Delta', \pi, \pi\wedge_r}\ (\wedge_l)}{\Delta', \pi, \pi}\ (\wedge_r)}{\Delta', \pi}\ (C).$$

3. The inference I has the form

$$\frac{\Pi, \pi\vee_l \quad \Pi, \pi\vee_r}{\Pi, \pi}\ (\vee).$$

Note that $\pi\vee_l, \pi\vee_r \succ \Pi$, because D respects $\succ$. We assume that some sequents Δ_1, Δ_2 are subsequents of $\Pi, \pi\vee_l$ and $\Pi, \pi\vee_r$, respectively, and find a derivation in $\mathrm{IP}^{G,\succ}$ of a subsequent of Π, π from Δ_1, Δ_2.

Consider the following possible cases:

(a) *Case:* Δ_1 *does not contain* $\pi\vee_l$*.* In this case Δ_1 is a subsequent of Π, and hence a subsequent of Π, π. Then we can take the empty derivation of Δ_1 from Δ_1 as D'.

(b) *Case:* Δ_2 *does not contain* $\pi\vee_r$*.* This case is symmetric to the previous one.

(c) *Case:* Δ_1 *contains* $\pi\vee_l$ *and* Δ_2 *contains* $\pi\vee_r$*.* In this case $\Delta_1 = \Delta_1', \pi\vee_l$ and $\Delta_2 = \Delta_2', \pi\vee_r$ for some subsequents Δ_1', Δ_2' of Π. Evidently, some subsequent Π' of Π can be derived from the sequent Δ_1, Δ_2 by a sequence of inferences by the contraction rule (C). The following derivation proves the case:

$$\begin{array}{c} \dfrac{\Delta_1', \pi\vee_l \quad \Delta_2', \pi\vee_r}{\Delta_1', \Delta_2', \pi}\ (\vee). \\ \vdots \quad \text{series of contractions} \\ \Pi', \pi \end{array}$$

4. The inference I has the form

$$\frac{\Pi\Box, \pi\Diamond}{\Gamma, \Pi, \pi}\ (\Diamond).$$

We assume that some sequent Δ is a subsequent of $\Pi\Box, \pi\Diamond$ and find a derivation in $\mathrm{IP}^{G,\succ}$ of a subsequent of Π, π, and hence also a subsequent of the conclusion Γ, Π, π.

Consider the following possible cases:

(a) *Case: Δ does not contain $\pi\Diamond$.* In this case Δ is a subsequent of $\Pi\Box$, so Δ has a form $\Delta'\Box$ for a subsequent Δ' of Π. The following derivation in $\mathrm{IP}^{G,\succ}$ proves the case:

$$\frac{\Delta'\Box}{\Delta', \pi}\ (\Diamond^{+}).$$

(b) *Case: Δ contains $\pi\Diamond$.* In this case Δ has the form $\Delta'\Box, \pi\Diamond$ for a subsequent Δ' of Π. The following derivation in $\mathrm{IP}^{G,\succ}$ proves the case:

$$\frac{\Delta'\Box, \pi\Diamond}{\Delta', \pi}\ (\Diamond).$$

□

By the induction on the length of derivations one can easily generalize Lemma 6.3 from inferences to arbitrary derivations:

LEMMA 6.4. *Let D be a refutation of ϵ in Pat^G that respects $\succ$, D'' be a subderivation in D of a sequent Γ from sequents $\Gamma_1, \ldots, \Gamma_n$, and $\Delta_1, \ldots, \Delta_n$ be subsequents of $\Gamma_1, \ldots, \Gamma_n$, respectively. Then there exists a derivation D' of Δ in $\mathrm{IP}^{G,\succ}$ from $\Delta_1, \ldots, \Delta_n$ such that Δ is a subsequent of Γ.*

Now we can prove Theorem 6.2.

PROOF OF THEOREM 6.2. ($\Rightarrow$) Suppose that a formula G is unsatisfiable. By Completeness Theorem 4.2 for Pat^G, the sequent ϵ has a refutation in Pat^G. By Lemma 6.4, there exists a subsequent of ϵ that has a refutation in $\mathrm{IP}^{G,\succ}$. Evidently, the empty sequent has no refutation in IP^G; therefore ϵ has a refutation in $\mathrm{IP}^{G,\succ}$.

($\Leftarrow$) Suppose that ϵ has a refutation in $\mathrm{IP}^{G,\succ}$. By Bisimulation Lemma 6.1 for IP^G, the sequent G has refutation in Inv^G. It is not hard to argue that Inv^G is sound, i.e., every sequent that has refutation is unsatisfiable. Therefore, G is unsatisfiable. □

Let $\succ$ be a G-ordering. Consider the standard *multiset extension* of $\succ$, denoted $\succ^{mul}$, defined as follows: $\Pi \succ^{mul} \Pi'$ if $\Pi \neq \Pi'$ and for every $\pi' \mathrel{\dot\in} \Pi' \mathbin{\dot-} \Pi$ there exists a $\pi \mathrel{\dot\in} \Pi \mathbin{\dot-} \Pi'$ such that $\pi \succ \pi'$. It is well-known that $\succ^{mul}$ is a well-founded order. Note that all inference rules of $\mathrm{IP}^{G,\succ}$ have the following property: the conclusion of any rule is strictly less that every premise of this rule with respect to $\succ^{mul}$. For ($\between$) rules, this follows from the fact that the conclusion is obtained from the multiset union of the premises by replacing the greatest path by one or more smaller paths. For the contraction rule this is obvious. For ($\Diamond$) and ($\Diamond^{+}$)-rules, this follows from the first condition of G-orderings. This implies that any derivation process from a finite set of path sequents in $\mathrm{IP}^{G,\succ}$ will terminate (so we do not even need a subsumption rule that is often essential for termination of bottom-up derivation processes).

7. MORE REDUNDANCIES: SUBSUMPTION

We say that a path sequent Π *subsumes* a path sequent Π' if Π is a subsequent of Π'[3]. Saturation-based theorem provers for first-order logic often use subsumption as a redundancy criterion: clauses subsumed by other clauses are removed from the search space.

In order to formalize subsumption as a redundancy criterion, we need a new notion of derivation, since subsumed clauses are not redundant per se, but only redundant with respect to other clauses in the search space. So to prove completeness of subsumption, we need to formalize the notion of search space. In this section we give such a formalization and prove that subsumption can be used to remove clauses from the search space. The formalization here uses two main ideas: (i) inference rules are reformulated as behaving on sets of clauses [Bachmair and Ganzinger 2000]; (ii) a very general property of derivations on such sets is proved to guarantee completeness in the style of Voronkov [1992]. Then we prove that the system with subsumption satisfies these properties.

As before, we assume to deal with *inferences*, consisting of zero or more premises and one conclusion, we assume that all premises and the conclusion belong to a set of derivable objects, or simply *objects*. Examples of objects used in this article are sequents and path sequents. An *inference rule* is a collection of inferences. An *inference system* or a *calculus* is a collection of inference rules. Alternatively, we will consider an inference system as consisting of inferences that are instances of inference rules of this system. Suppose that $\mathcal{I}$ is an inference system on objects. We turn it into an inference system $\{\mathcal{I}\}$ on *sets of objects*, called the *set calculus* as follows. Take any inference

$$\frac{\Pi_1 \cdots \Pi_n}{\Pi} \tag{3}$$

of $\mathcal{I}$. Then for every set S of objects the following is an inference rule of $\{\mathcal{I}\}$:

$$\frac{S \cup \{\Pi_1, \ldots, \Pi_n\}}{S \cup \{\Pi_1, \ldots, \Pi_n, \Pi\}} .$$

The intuitive meaning of $\{\mathcal{I}\}$ is the following: sets of objects formalize search space; the inference rule above means: if $\Pi_1, \ldots, \Pi_n$ are already proved (belong to the search space), we can add Π to the search space, provided that (3) is an inference of $\mathcal{I}$. Inferences in $\{\mathcal{I}\}$ will be denoted using the $\triangleright$ sign:

$$S \cup \{\Pi_1, \ldots, \Pi_n\} \triangleright S \cup \{\Pi_1, \ldots, \Pi_n, \Pi\}.$$

Sometimes we will superscribe $\triangleright$ with the name of the inference rule.

We call the *initial set of objects* of $\{\mathcal{I}\}$ the set of *all* axioms of $\mathcal{I}$. We also assume that with $\mathcal{I}$ associated is a set of objects, called the *goal object*. For example, in all our path calculi the goal object is the sequent ϵ. We call a *derivation* in the set calculus any sequence of inferences:

$$S_0 \triangleright S_1 \triangleright S_2 \triangleright \ldots \ ,$$

[3]This notion of subsumption should not be mixed with the subsumption of concepts in description logics.

possibly infinite, where S_0 is the initial set of objects. We say the derivation *succeeds* if some S_i contains a goal object, and *fails* otherwise.

The subsumption strategy that deletes a clause from the search space can now be formalized as the inference rule in the set calculus

$$S \cup \{\Pi_1, \Pi_2\} \rhd S \cup \{\Pi_1\},$$

provided that Π_1 subsumes Π_2.

When we have a notion of derivation that deletes objects from the search space, we can have the following problem. Suppose that an inference rule adds an object Π to the search space, and then such a deletion rule removes it. Then we may obtain an infinite failing derivation:

$$S \rhd S \cup \{\Pi\} \rhd S \rhd S \cup \{\Pi\} \rhd \dots ,$$

even when a successful derivation exists. Such a derivation is *unfair*: there have been inferences applicable to S but that were never applied. Let us formalize the notion of fairness.

Consider a derivation $S_0 \rhd S_1 \rhd \dots$ in $\{\mathcal{I}\}$. The set $\bigcup_i \bigcap_{j \geq i} S_j$ is called the *limit* of this derivation. It is easy to see that the limit consists of the sequents that remain in the derivation forever after some step. The derivation is called *fair* if, for every inference of $\mathcal{I}$,

$$\frac{\Pi_1 \cdots \Pi_n}{\Pi} ,$$

if $\Pi_1, \dots, \Pi_n$ belong to the limit of the derivation, then an inference of $\{\mathcal{I}\}$

$$S \cup \{\Pi_1, \dots, \Pi_n\} \rhd S \cup \{\Pi_1, \dots, \Pi_n, \Pi\}$$

has been applied in the derivation. This means that for some i and S, the set S_i has the form $S \cup \{\Pi_1, \dots, \Pi_n\}$, and S_{i+1} has the form $S \cup \{\Pi_1, \dots, \Pi_n, \Pi\}$.

We will now prove a statement about $\{\mathrm{IP}^{G,\succ}\}$, i.e., the set inference system obtained from $\mathrm{IP}^{G,\succ}$. It will be clear from the proof that the statement is very general and applicable to any inference system without the deletion rule.

THEOREM 7.1 *(Completeness of* $\{\mathrm{IP}^{G,\succ}\}$*). Let G be a formula and $\succ$ be a G-ordering. (i) If any derivation in* $\{\mathrm{IP}^{G,\succ}\}$ *succeeds, then G is unsatisfiable. (ii) If G is unsatisfiable, then every fair derivation in* $\{\mathrm{IP}^{G,\succ}\}$ *succeeds.*

PROOF. (i) Let a derivation $S_0 \rhd \dots \rhd S_n$ succeed. It is easy to see that every $\Pi \in \bigcup_i S_i$ has a refutation in $\mathrm{IP}^{G,\succ}$. Therefore ϵ has a refutation, and by completeness of $\mathrm{IP}^{G,\succ}$ (Theorem 6.2), G is unsatisfiable.

(ii) Let $\mathcal{S} = S_0 \rhd S_1 \rhd \dots$ be a fair derivation. We prove that every path sequent Π that has a refutation in $\mathrm{IP}^{G,\succ}$ occurs in some S_i. The proof is by induction on the length of the refutation of Π in $\mathrm{IP}^{G,\succ}$. If the refutation is of length 0, i.e., Π is an axiom, then Π is already a member of S_0. For refutations of longer lengths, consider the last inference of this refutation:

$$\frac{\Pi_1 \quad \dots \quad \Pi_n}{\Pi} .$$

By the induction hypothesis, every Π_i is a member of some S_{k_i}. Since $\{\mathrm{IP}^{G,\succ}\}$ does not delete any path sequents, all Π_i belong to the limit of $\mathcal{S}$. Then by fairness of $\mathcal{S}$, Π is a member of some S_m, too. $\square$

Now we are ready to prove that $\{\mathrm{IP}^{G,\succ}\}$ with subsumption is complete, i.e., every fair derivation in this system contains ϵ, whenever G is unsatisfiable. In fact, we will define an even stronger notion of subsumption first, and then prove that $\{\mathrm{IP}^{G,\succ}\}$ is complete with this stronger notion.

We say that a path sequent Π *prefix-subsumes* a path sequent Π' if for every $\pi \mathrel{\dot\in} \Pi \mathrel{\dot-} \Pi'$ there exists a $\pi' \mathrel{\dot\in} \Pi' \mathrel{\dot-} \Pi$ such that $\pi \sqsubset \pi'$. It is easy to see that prefix-subsumption is the standard multiset extension of the ordering $\sqsubset$, e.g., see Niewenhuis and Rubio [2000].

The intuition behind the prefix-subsumption is the following. Suppose that a path $\pi_1\pi_2$ occurs in a path sequent Π, occurring in a derivation of ϵ, and π_2 contains no modal operators. If we consider the branch of this derivation between Π and ϵ, there will be a rule that adds π_1 on the branch. Now, if we replace $\pi_1\pi_2$ in Π by π_1, obtaining a path sequent Π', one can see that the derivation can be made into a (maybe shorter) derivation of ϵ from Π'. Therefore, replacement of a path by its subpath of the same modal length preserves derivability of ϵ, and sometimes make derivations shorter.

We call $\{\mathrm{IP}^{G,\succ}\}$ *with prefix-subsumption* the inference system obtained from $\{\mathrm{IP}^{G,\succ}\}$ by adding the *prefix-subsumption rule*:

$$S \cup \{\Pi, \Pi'\} \triangleright S \cup \{\Pi\},$$

where Π prefix-subsumes Π' and $\Pi \neq \Pi'$.

Note that the notion of fair derivation introduced above only required inferences of the original calculus (*cumulative* inferences) be applied at some step. Now we have new kinds of steps that *delete* sequents from the search space. We will use the old notion of fairness for $\{\mathrm{IP}^{G,\succ}\}$ with prefix-subsumption, i.e., we do *not* require that prefix-subsumption inferences applicable to sequents in the limit be applied at some step. Thus, any fair derivation in $\{\mathrm{IP}^{G,\succ}\}$ without prefix-subsumption is also a fair derivation in $\{\mathrm{IP}^{G,\succ}\}$ with prefix-subsumption.

Let us note some obvious properties of prefix-subsumption:

LEMMA 7.2.

1. *Note that* Π *subsumes* Π' *if and only if* $\Pi \mathrel{\dot-} \Pi'$ *is empty. Therefore, if* Π *subsumes* Π'*, then* Π *also prefix-subsumes* Π'.
2. *If* Π_1 *prefix-subsumes* Π_2 *and* Π_2 *prefix-subsumes* Π_1*, then* $\Pi_1 = \Pi_2$.[4]
3. *If* Π *prefix-subsumes* Π' *and* Π' *prefix-subsumes* Π''*, then* Π *prefix-subsumes* Π''.

[4]Using sets instead of multisets in the formulation of the inverse method creates a subtle problem with prefix-subsumption. The path sequents $\pi\pi', \pi$ and $\pi\pi'$ would prefix-subsume each other if we used sets instead of multisets (because $\pi\pi'$ prefix-subsumes $\pi\pi', \pi$, which, in turn, prefix-subsumes $\pi\pi', \pi\pi'$). In fact, we use sets in our implementation, but the correctness of using sets instead of multisets can be established from the proof of completeness of the calculus with subsumption introduced below.

4. *If* $\pi_1 \sqsubset \pi', \ldots, \pi_n \sqsubset \pi'$, *and* Π *prefix-subsumes* Π', *then* $\Pi, \pi_1, \ldots, \pi_n$ *prefix-subsumes* Π', π'.
5. *If* $\pi \sqsubseteq \pi'$ *and* Π *prefix-subsumes* Π', *then* Π, π *prefix-subsumes* Π', π'.
6. *There is no infinite sequence* $\Pi_0, \Pi_1, \ldots$ *such that* Π_{i+1} *prefix-subsumes* Π_i *and* $\Pi_{i+1} \neq \Pi_i$.

Many of these properties immediately follow from the standard facts about the multiset extensions of well-founded orders which can be found in e.g. Niewenhuis and Rubio [2000]: for example, the prefix-subsumption relation is a well-founded order because so is $\sqsubset$.

We will note two further properties of $\sqsubseteq$ related to proof-search optimization after proving completeness of $\{\mathrm{IP}^{G,\succ}\}$ with prefix-subsumption.

THEOREM 7.3 *(Completeness of* $\{\mathrm{IP}^{G,\succ}\}$ *With Prefix-Subsumption). (i) If any derivation in the system* $\{\mathrm{IP}^{G,\succ}\}$ *with prefix-subsumption succeeds, then* G *is unsatisfiable. (ii) If* G *is unsatisfiable, then every fair derivation* $\mathcal{S}$ *in* $\{\mathrm{IP}^{G,\succ}\}$ *with prefix-subsumption succeeds.*

PROOF. (i) is proved as in Theorem 7.1 (completeness of $\{\mathrm{IP}^{G,\succ}\}$). For (ii), we will use Theorem 7.1. Let us show that (ii) immediately follows from

(4) Let $\mathcal{T}$ be a fair derivation in $\{\mathrm{IP}^{G,\succ}\}$ and a path sequent Π occurs in $\mathcal{T}$. Then there exists a path sequent Γ that prefix-subsumes Π and belongs to the limit of $\mathcal{S}$.

Suppose G is unsatisfiable; then by Theorem 7.1 $\mathcal{T}$ contains the path sequent ϵ. Then by (4), $\mathcal{S}$ must contain a path sequent that prefix-subsumes ϵ, but there is only one such path sequent: ϵ itself. So, it remains to prove (4).

Suppose $\mathcal{S} = S_0 \rhd S_1 \rhd \ldots$ and denote by S_∞ the limit of $\mathcal{S}$. We say that a path sequent Π has been *removed* in $\mathcal{S}$ by Π' if some S_{i+1} is $S_i \setminus \{\Pi\}$, and $\Pi' \in S_{i+1}$ prefix-subsumes Π. If Π has been removed by Π_1, it is possible that Π_1 is later removed by Π_2 etc., but item (6) of Lemma 7.2 guarantees that this cannot continue infinitely often, so, starting from some j, all S_j will contain the same path sequent Π' that prefix-subsumes Π. Thus, we proved:

(5) *for every path sequent* Π *in any* S_i *there exists a* $\Pi' \in S_\infty$ *such that* Π' *prefix-subsumes* Π.

Moreover, using similar considerations one can strengthen this result to

(6) *for every path sequent* Π *in any* S_i *there exists a* $\Pi' \in S_\infty$ *such that* Π' *prefix-subsumes* Π, *and if* $\Pi'' \in S_\infty$ *prefix-subsumes* Π', *then* $\Pi'' = \Pi'$.

For any path sequent Π (not necessarily in some S_i) we call a *trace* of Π in $\mathcal{S}$ any $\Pi' \in S_\infty$ such that Π' prefix-subsumes Π and no $\Pi'' \in S_\infty$ prefix-subsumes Π'. Now (6) can be reformulated: every $\Pi \in S_i$ has a trace in $\mathcal{S}$.

Note that (4) obviously follows from a slightly more general statement

(7) *Every path sequent* Π *that appears in a fair derivation* $\mathcal{T}$ *in* $\{\mathrm{IP}^{G,\succ}\}$ *has a trace in* $\mathcal{S}$.

We will now prove (7), and then we are done. Let $\mathcal{T} = T_0 \triangleright T_1 \triangleright \dots$. Take the least i such that $\Pi \in T_i$. (7) will be proved by induction on i. If $i = 0$, then Π is an axiom; therefore it belongs to S_0, and by (6) has a trace in $\mathcal{S}$. If $i > 0$, then Π is obtained by an inference I of $\mathrm{IP}^{G,\succ}$ from path sequents in T_{i-1}. By the induction hypothesis, these path sequents have traces in $\mathcal{S}$. We use the case analysis on the inference I, but before we will note some general properties related to redundancies and prefix-subsumption. Namely, we prove the following two statements.

(8) *Suppose* Π *is a path sequent that contains a pair of conflicting paths* π_1, π_2 *and* Π *prefix-subsumes* Π'. *Then* Π' *contains a pair of conflicting paths, too.*

(9) *Suppose* $\pi \succ \Pi'$ *and* Π *prefix-subsumes* Π'. *Then* $\pi \succ \Pi$.

(8) follows from the following observation: if $\pi_1, \pi_2 \mathrel{\dot\in} \Pi$ and Π prefix-subsumes Π', then there exist $\pi'_1, \pi'_2 \mathrel{\dot\in} \Pi'$ such that $\pi_1 \sqsubseteq \pi'_1$ and $\pi_2 \sqsubseteq \pi'_2$. So if π_1, π_2 are conflicting then π'_1, π'_2 are conflicting, too. To prove (9), note the following: Π consists of prefixes of Π', and every path is greater than any its prefix with respect to $\succ$, so π is greater than any path sequent in Π.

Let us now return to the proof of (7).

1. The inference I has the form

$$\frac{\Pi, \pi\wedge_l}{\Pi, \pi}\ (\wedge_l).$$

Let Γ be a trace of $\Pi, \pi\wedge_l$ in $\mathcal{S}$. Consider three possible cases:

(a) *Case* (Γ *prefix-subsumes* Π). Then Γ also prefix-subsumes Π, π, and is a trace of Π, π.

(b) *Case* (Γ *has the form* $\Gamma', \pi\wedge_l$, *where* Γ' *prefix-subsumes* Π). Consider the inference

$$\frac{\Gamma', \pi\wedge_l}{\Gamma', \pi}\ (\wedge_l). \tag{10}$$

Let us show that this inference is not redundant in $\mathrm{IP}^{G,\succ}$. Evidently, Γ', π contains no pair of paths of different model lengths, since so is $\Gamma', \pi\wedge_l$. (8) guarantees that Γ', π contains no conflicting pairs of paths, because so is Π, π, and finally, (9) guarantees that $\pi \succ \Gamma'$, because $\pi \succ \Pi$.
Since inference (10) is nonredundant and $\mathcal{S}$ is fair, $\mathcal{S}$ contains this inference. By (6), $\mathcal{S}$ contains a trace of Γ', π; evidently, this trace is also a trace of Π, π.
Below we will no more prove any more that the inferences in $\mathcal{S}$ are nonredundant, since the proof is similar to the one in this case.

(c) *Case* (Γ *has the form* Γ', π', *where* Γ' *prefix-subsumes* Π *and* $\pi' \sqsubset \pi\wedge_l$). Note that if $\pi' \sqsubset \pi$, then Γ', π' also prefix-subsumes Π, π. So we only consider the case $\pi' = \pi$. Using the definition of prefix-subsumption, one can show that either Γ', π prefix-subsumes Π, in which case we are done, or $\pi \mathrel{\dot\in} \Gamma'$. Let us show that $\pi \mathrel{\dot\in} \Gamma'$ is impossible. Indeed, $\pi \mathrel{\dot\in} \Gamma'$ implies that Γ, π has a form Γ'', π, π for some Γ''. Since $\mathcal{S}$ is fair, the contraction rule should be applied in $\mathcal{S}$ to Γ'', π, π, obtaining Γ'', π which prefix-subsumes Γ'', π, π. By (6) Γ'', π has a trace in $\mathcal{S}$, then this trace prefix-subsumes Γ'', π, π. But Γ'', π, π is a trace itself, and cannot be prefix-subsumed by any other path sequent in S_∞. Contradiction.

2. The inference I has the form

$$\frac{\Pi_1, \pi\vee_l \quad \Pi_2, \pi\vee_r}{\Pi_1, \Pi_2, \pi}\ (\vee).$$

Take traces Γ_1 and Γ_2 of $\Pi_1, \pi\vee_l$ and $\Pi_2, \pi\vee_r$, respectively, in $\mathcal{S}$. Similar to the previous case, we consider the following subcases:

(a) *Case (either Γ_1 prefix-subsumes Π_1 or Γ_2 prefix-subsumes Π_2).* Then either Γ_1 or Γ_2 prefix-subsumes Π_1, Π_2, π.

(b) *Case (Γ_1 contains $\pi\vee_l$ and Γ_2 contains $\pi\vee_r$).* Then Γ_1 has the form $\Gamma_1', \pi\vee_l$ and Γ_2 has the form $\Gamma_2', \pi\vee_r$ such that Γ_1 and Γ_2 prefix-subsume Π_1 and Π_2, respectively. Since $\mathcal{S}$ is fair, it contains an inference

$$\frac{\Gamma_1', \pi\vee_l \quad \Gamma_2', \pi\vee_r}{\Gamma_1', \Gamma_2', \pi}\ (\vee).$$

By (6), $\mathcal{S}$ contains a trace of $\Gamma_1', \Gamma_2', \pi$; evidently, this trace is also a trace of Π_1, Π_2, π.

(c) *Case (Γ_1 has the form Γ_1', π', where Γ_1' prefix-subsumes Π_1 and $\pi' \sqsubset \pi\vee_l$).* This case is similar to the respective case for the conjunction above.

(d) *Case (Γ_2 has the form Γ_2', π', where Γ_2' prefix-subsumes Π_2 and $\pi' \sqsubset \pi\vee_r$).* This case is symmetric to the previous one.

3. The inference I has the form

$$\frac{\Pi\Box, \pi\Diamond}{\Pi, \pi}\ (\Diamond).$$

Take a trace Γ of $\Pi\Box, \pi\Diamond$. All paths in $\Pi\Box, \pi\Diamond$ have the same modal length, so every proper prefix of any path in $\Pi\Box, \pi\Diamond$ has the modal length less than that of $\pi\Diamond$. This implies that Γ either only consists of paths in $\Pi\Box, \pi\Diamond$ or prefixes of Π, π. In the latter case, using that Γ is a trace, one can prove as in item 1c above that Γ prefix-subsumes Π, π. So it remains to consider the case when Γ consists of paths in $\Pi\Box, \pi\Diamond$. Note that Γ cannot contain two copies of $\pi\Diamond$, since Γ is a trace, so it contains either one copy of $\pi\Diamond$, or none at all. So we consider two cases:

(a) *Case (Γ does not contain $\pi\Diamond$).* In this case Γ is a subsequent of $\Pi\Box$, so Γ has a form $\Gamma'\Box$ for a subsequent Γ' of Π. Consider the following inference:

$$\frac{\Gamma'\Box}{\Gamma', \pi}\ (\Diamond^+).$$

Since $\mathcal{S}$ is fair, it contains this inference. It is easy to see that Γ', π subsumes Π, π, and thus also prefix-subsumes Π, π. Take a trace of Γ', π in $\mathcal{S}$; evidently it is also a trace of Π, π.

(b) *Case* (Γ *contains* $\pi\Diamond$). In this case Γ has the form $\Gamma'\Box, \pi\Diamond$ for some subsequent Γ' of Π. Consider the following inference:

$$\frac{\Gamma'\Box, \pi\Diamond}{\Gamma', \pi}\ (\Diamond).$$

Since $\mathcal{S}$ is fair, it contains this inference. It is easy to see that Γ', π subsumes Π, π, and thus also prefix-subsumes Π, π. Take a trace of Γ', π in $\mathcal{S}$; evidently it is also a trace of Π, π.

4. The inference I is $(\Diamond^+)$. This case is nearly identical to the previous.

The proof is complete. $\Box$

We did not implement prefix-subsumption, since we could not yet find efficient data structures that make prefix-subsumption checks fast, but we plan such an implementation. The current version of our system KΞ [Voronkov 2000] uses the ordinary subsumption. Note that the completeness of $\mathrm{IP}^{G,\succ}$ with subsumption immediately follows from completeness of $\mathrm{IP}^{G,\succ}$ with prefix-subsumption.

However, the completeness result for prefix-subsumption has some applications besides the possibility of removing subsumed sequents. Let us look at some interesting observations that come out of this result. Consider any $(\wedge_*)$-inference:

$$\frac{\Pi, \pi\wedge_*}{\Pi, \pi}\ (\wedge_*).$$

Note that the conclusion of this inference prefix-subsumes the premise. Thus, the premise can be discarded as soon as the rule is applied. Even more, when $\pi\wedge_*$ is not maximal in $\Pi, \pi\wedge_*$, so that the inference is redundant, it makes sense to apply this inference, because the premise can be discarded after. If we apply $(\wedge_*)$-rules in an eager way, as soon as they are applicable, any path $\pi\wedge_*$ becomes "short-lived." This observation has a far-reaching consequence: one can modify the calculus $\mathrm{IP}^{G,\succ}$ so that such paths do not occur in sequents at all. Such an optimization first appeared for classical logic in Voronkov [1985] published in Russian, and then in a more accessible Voronkov [1990]. It has also been used in tableau calculi in Degtyarev and Voronkov [1996] as a *least conjunctive superformula* optimization. It is implemented in our system KΞ [Voronkov 2000].

Another interesting consequence of the use of prefix-subsumption is the following. Suppose that during search for a refutation we obtained a path sequent $\Delta = \Pi, \pi, \pi'$ such that $\pi \sqsubseteq \pi'$. Consider the path sequent $\Gamma = \Pi, \pi', \pi'$. This sequent is prefix-subsumed by Δ, so it can be added to the search space without compromising soundness or completeness. But if we add Γ to the search space, by contraction we can derive Π, π' that subsumes Δ. Therefore Π, π, π' can be replaced by a simpler sequent Π, π', as soon as π is a prefix of π'.

Many redundancies considered in this article (except for prefix-subsumption) are obtained by studying the redundancies in a tableau calculus. The reader may have an impression that the inverse method is inherently weaker than tableaux. Let us give an example which shows that in general it is not so. Consider any formula in which every atom occurs at most once. Then the inverse method will terminate immediately, since no axiom can be created, while there maybe an exponential number of failing tableau derivation.

8. EXPERIMENTS

The calculus Inv^G for K with the redundancy criteria mentioned above is implemented in version 1.0 of our system Kꓘ (read "K-inverse K"). A preliminary much slower version of Kꓘ was implemented in ML and described in Voronkov [1999]. The current version 1.0 of Kꓘ used for benchmarking is implemented in C++ and comprises about 13,000 lines of C++ code, of which 5,000 lines implement a parser generated automatically using the ANTLR parser generator and PCCTS tool.

In this section we consider the results of running Kꓘ on the collection of tests used for the Tableau'98 competition. We give a comparison with other systems for K. The results for other systems are taken from Giunchiglia et al. [1999] and Horrocks et al. [2000]. The results for other systems were obtained on a computer with the Pentium 350MHz processor using the time limit of 100 seconds. We ran our experiments on a Pentium 450MHz processor and set the limit to 77 seconds, in order to compensate the difference in performance. Some of the results for other systems are from the Tableaux'98 comparison, but some are more recent (see for details Horrocks et al. [2000]).

The experiments were performed on 9 collections of tests. Each collection consists of 21 provable and 21 unprovable formulas of presumably growing difficulty. The results are given in Table I. p in every column means provable formulas, while n means unprovable. The table contains the number of problems solved by systems within the time limit of 100 seconds. If all 21 problems can be solved, we put ">" in the table. Table II shows the time spent by some systems for the most difficult problem solved by this system in each category.

One can see that Kꓘ 1.0 is weaker than the two best systems **∗SAT** and **DLP**. However, we would like to note that our current implementation is completely nonoptimized as to proof-search strategies and nonlogical optimizations. For example, both **∗SAT** and **DLP** have strategies for intelligent backtracking and **DLP** implements tabulation. In addition, all top systems have powerful preprocessors that simplify the input formula, while the current version of Kꓘ has no preprocessor. We did not try to optimize G-orderings, but simply used the first left-to-right ordering generated by the ordering procedure.

One can see that Kꓘ is stronger on this collection of benchmarks than the less optimized implementations **FaCT** of the tableau method, **KSAT** of the SAT-based method, and the earlier versions of **TA** of the first-order translation-based methods. This shows the high potential of future, more optimized versions of Kꓘ.

Our system is based on a systematic proof procedure without any ad hoc optimizations. We expect a considerable speedup by implementing such optimizations. Our expectations are confirmed by the experience of other provers, whose first versions were very considerably slower than the current versions. Among such future optimizations we consider the following.

1. Implementing a formula preprocessor that makes general simplifications of the goal formula and some specific simplifications that make the inverse proof-search easier.
2. Implementing a greedy search for redundancies, which is a standard technique in first-order theorem provers. Essentially, most of the current first-order provers always check whether a simplifying inference is applicable (i.e., an in-

Table I. Results for K

	branch		*d4*		*dum*		*grz*		*lin*		*path*		*ph*		*poly*		*t4p*	
K	*p*	*n*	*p*	*n*	*p*	*n*	*p*	*n*	*p*	*n*	*p*	*n*	*p*	*n*	*p*	*n*	*p*	*n*
leanK 2.0	1	0	1	1	0	0	0	>	>	4	2	0	3	1	2	0	0	0
Crack 1.0	2	1	2	3	3	>	1	>	5	2	2	6	2	3	>	>	1	1
□**KE**	13	3	13	3	4	4	3	1	>	2	17	5	4	3	17	0	0	3
LWB 1.0	6	7	8	6	13	19	7	13	11	8	12	10	4	8	8	11	8	7
Kris	3	3	8	6	15	>	13	>	6	9	3	11	4	5	11	>	7	5
KSAT	8	8	8	5	11	>	17	>	>	3	4	8	5	5	13	12	10	18
FaCT 1.2	6	4	>	8	>	>	>	>	>	>	7	6	6	7	>	>	>	>
Kꓘ	6	7	>	16	>	>	>	>	>	>	15	14	2	2	>	>	>	>
TA	9	9	>	18	>	>	>	>	>	>	20	20	6	9	16	17	>	19
DLP 3.1	19	13	>	>	>	>	>	>	>	>	>	>	7	9	>	>	>	>
∗**SAT** 1.2	>	12	>	>	>	>	>	>	>	>	>	>	8	12	>	>	>	>

Table II. Timing Results for ∗**SAT** (options `-k1 -e -m6`), **DLP**, **TA** and Kꓘ.

	∗**SAT** 1.2		**DLP** 3.2		**TA** 1.4		Kꓘ	
Test	Size	Time	Size	Time	Size	Time	Size	Time
branch_p	>	0.21	19	46.06	6	65.76	6	29.08
branch_n	12	94.49	13	53.63	6	96.44	7	65.33
d4_p	>	0.06	>	0.05	15	71.11	>	0.09
d4_n	>	2.87	>	1.12	14	44.06	16	69. 17
dum_p	>	0.04	>	0.02	17	64.99	>	0.01
dum_n	>	0.12	>	0.02	16	65.82	>	0
grz_p	>	0.04	>	0.04	>	0.51	>	0.11
grz_n	>	0.01	>	0.05	>	0.33	>	0.08
lin_p	>	0.01	>	0.03	>	9.24	>	0.03
lin_n	>	47.80	>	0.13	>	80.01	>	0.15
path_p	>	0.72	>	0.32	5	25.03	15	36.43
path_n	>	0.96	>	0.36	4	60.84	14	35.92
ph_p	8	48.54	7	10.23	6	43.16	2	0.08
ph_n	12	0.60	>	2.69	9	55.13	2	0.02
poly_p	>	1.73	>	0.11	5	53.48	>	0.54
poly_n	>	2.25	>	0.18	4	9.09	>	0.49
t4p_p	>	0.29	>	0.06	16	88.66	>	0.01
t4p_n	>	1.28	>	0.13	9	87.72	>	0.04

ference that results in removal of redundant clauses), and only when no such inference is possible apply a nonsimplifying inference. This may give a speedup of several orders of magnitude (as witnessed by our unpublished experiments with propositional resolution for classical logic). The current version of Kꓘ behaves like a very naive resolution procedure, which applies the first available inference.

3. Optimizing the system for the cases when propositional reasoning dominates the modal reasoning (cf. the weak performance of Kꓘ on the *ph* benchmarks in which essentially no modal reasoning is involved).

9. RELATED WORKS

The inverse method for nonclassical logics. Gentzen [1934] was the first to formulate the inverse method: he has shown how to prove the decidability of intuitionistic propositional logic using the bottom-up search in a sequent calculus based on the subformula property. The remark of Gentzen [1934] remained unnoticed for quite some time. The general interest to the inverse method appeared due to a series of Maslov's works, e.g., Maslov [1971].

Voronkov [1992] sketches a general scheme of the inverse method for nonclassical logics. Some redundancy criteria considered in this article are similar to strategies introduced in Voronkov [1992]. However, Voronkov [1992] contains no complete proofs.

Mints [1993] and Tammet [1994] consider the inverse method for linear logic. Mints [1994] and Tammet [1996] discuss the inverse method for intuitionistic logic; however the presentation of Tammet [1996] contains several inaccuracies that are difficult to validate in the absence of proofs. For example, his general scheme of constructing inverse method calculi may result in incomplete calculi, and the intuitionistic calculus of Tammet [1996] is indeed incomplete. The ordering strategy of our article is similar to a strategy based on the permutability of rules introduced by Tammet [1996], with a difference that our ordering is computed in compile-time, while Tammet [1996] has to store some runtime information in derived sequents.

Mints et al. [1996] discuss the inverse method for S4 with several redundancies; but many proofs are absent, so it is difficult to compare their technique with ours.

Implementation of the inverse method. Tammet [1996] reported some results on the implementation of the inverse method for intuitionistic predicate logic; however it is difficult to comment on his results, since the calculi used in his implementation [Tammet 1996] are incomplete.

Our implementation seems to be the first to show that calculi based on forward reasoning may compete with calculi based on backward reasoning for *propositional* logics.

Implementations of other methods. Implementations of other state-of-the-art systems are described in the following papers. SAT-based decision procedures are considered in Giunchiglia et al. [1998], Tacchella [1999], and Giunchiglia et al. [1999]. Hustadt and Schmidt [1997] describe an implementation based on a translation to classical predicate logic and the use of the classical logic theorem prover SPASS [Weidenbach et al. 1996]. Two best tableau-based systems are discussed in Horrocks and Patel-Schneider [1998], Horrocks [1998], and Patel-Schneider [1998]. A comparison of several approaches and a general discussion of evaluation of decision procedures for nonclassical logics can be found in Hustadt and Schmidt [1997] and Horrocks et al. [2000].

10. FUTURE RESEARCH

The framework described in this article is applicable to a number of nonclassical logics. It would be interesting to *extend it to more expressive description logics* [Calvanese et al. 2000]. An obstacle to such an extension is the way these logics are usually introduced. The technique for designing calculi for various logics developed in the description logic community is essentially oriented to tableau-based procedures. These calculi are formulated in a top-down way and often have explicit annotation for worlds. This simplifies completeness proofs, but the resulting calculi do not have obvious proof-theoretical properties to be exploited. As a result, only two kinds of satisfiability checking procedures can be developed: tableau-based ones and semantics-based ones.

To design an inverse method-based calculus one needs to *obtain reformulations of these tableau-oriented calculi in a more traditional way*, where the inference rules are formulated solely in terms of formulas, or otherwise find new techniques for designing bottom-up calculi. Such a logical reformulation is an interesting line of research by itself and can lead to better versions of tableau-based procedures that also exploit proof-theoretical properties of calculi, while the current tableau-based systems are mostly based on various nonlogical optimizations, such as tabulation or intelligent backtracking.

It seems that bottom-up calculi can offer considerable advantages for handling some inference rules. For examples, some description logics have a restricted form of the cut rule:

$$\frac{\Gamma, A \quad \Gamma, \overline{A}}{\Gamma}\ (cut),$$

where A and $\overline{A}$ (equivalent to the negation of A) can be arbitrary subformulas of the goal formula. The problem of using this rule in the direction from the conclusion to the premises is high nondeterminism in selecting the cut formula A. The application in the inverse direction does not meet this problem because Γ, A and $\Gamma, \overline{A}$ must be already derived sequents. Therefore, from the viewpoint of bottom-up calculi there is no essential difference between this rule and, say, the $(\vee)$-rule:

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

in which Γ, A and Δ, B should be already proved sequents.

Another interesting line of research is *combination of top-down and bottom-up proof-search*, or *bidirectional search*. In such a combination the inverse method would produce sequents that can be used by the tableau method as additional axioms. This is similar to tabulation, but with an essential difference. The traditional tabulation is used to store *derivable* sequents and checking whether a newly derived sequent is unsatisfiable by verifying whether it is subsumed by a tabulated sequent. If it is, then the newly derived sequent is unsatisfiable; if not, then we have to continue the search for refutation. The inverse method tabulates *all* sequents up to a certain modal length k, which means that a newly derived sequent of a modal length $\geq k$ is satisfiable *if and only if* it is not subsumed by any of the tabulated sequent. Therefore, if the subsumption-check fails one should *not* continue search for the proof of this newly derived clause.

A combination of the tableau and the inverse method can also solve the memory problem for the inverse method. On some benchmarks, our implementation of the inverse method uses more than 400M of memory. If one discovers that the memory consumption is too large, one can terminate the inverse proof-search and continue the tableau search with the tabulated sequents collected so far.

However, such bidirectional search meets essential difficulties on the theoretical level. The inverse method without redundancies is not efficient at all. The naive tableau proof-search is inefficient as well. When we run bidirectional proof-search, one has to be sure that the optimizations used in the tableau proof-search are compatible with the redundancies used in the inverse proof-search. This is a challenging task, since the completeness proofs of the redundancies for the inverse method require a nontrivial technique.

We feel that a new technique can be used to give model-theoretical completeness proofs of the inverse path calculus with redundancies built in directly into the proof system, similar to completeness proofs for resolution with redundancies [Bachmair and Ganzinger 2000].

11. CONCLUSION

The main contributions of this article are the following:

1. We introduce a new technique of proving completeness of calculi with redundancies by analyzing tableau-based derivations. The previous completeness proofs were either based on a specialized technique [Tammet 1996; Mints et al. 1996] or on a different technique based on a suitable notion of subsumption [Voronkov 1992].[5]
2. We give a new formalization of the inverse method based on a path calculus. This formalization simplifies the definition of redundancies and completeness proofs. An example is the definition of conflicting paths and related proofs.
3. A new technique for proving completeness of subsumption-like redundancies is introduced.

Our current implementation of the inverse method for K⅄ is not yet convincing to conclude that the inverse method can compete with the very best state-of-the-art tableau-based or SAT-based systems. Extensions to more expressive logics and further experiments are required.

ACKNOWLEDGMENT

AcknowledgmentAck The remarks of anonymous referees were extremely valuable and pointed out a number of problems in a previous version of this article. I highly appreciate the work of the referees and the editor Maarten de Rijke.

This article benefited from numerous discussions with (in the alphabetical order) Anatoli Degtyarev, Enrico Franconi, Enrico Giunchiglia, Fausto Giunchiglia, Ian Horrocks, Fabio Massacci, Alexander Riazanov, Ulrike Sattler, Renate Schmidt, Stephan Tobies, and Christoph Weidenbach.

REFERENCES

ReferencesRef

Baader, F. and Hollunder, B. 1991. A terminological knowledge representation system with complete inference algorithms. In *PDK'91*, H. Boley and M. Richter, Eds. Lecture Notes in Artificial Intelligence, vol. 567. Springer Verlag, 67–86.

[5]Neither Voronkov [1992] nor Tammet [1996] have complete proofs, which makes comparison of techniques difficult.

BACHMAIR, L. AND GANZINGER, H. 2000. Resolution theorem proving. In *Handbook of Automated Reasoning*, A. Robinson and A. Voronkov, Eds. Elsevier Science and MIT Press. To appear.

BALSIGER, P., HEUERDING, A., AND SCHWENDIMANN, S. 1998. Logics Workbench 1.0. In *Automated Reasoning with Analytic Tableaux and Related Methods, International Conference, TABLEAUX'98*, H. de Swart, Ed. Lecture Notes in Artificial Intelligence, vol. 1397. Springer Verlag, Oisterwijk, The Netherlands, 35.

CALVANESE, D., DE GIACOMO, G., LENZERINI, M., AND NARDI, D. 2000. Reasoning in expressive description logics. In *Handbook of Automated Reasoning*, A. Robinson and A. Voronkov, Eds. Elsevier Science and MIT Press. To appear.

CATACH, L. 1991. Tableaux: a general theorem prover for modal logics. *Journal of Automated Reasoning 7,* 4, 489–510.

DEGTYAREV, A. AND VORONKOV, A. 1996. Equality elimination for the tableau method. In *Design and Implementation of Symbolic Computation Systems. International Symposium, DISCO'96*, J. Calmet and C. Limongelli, Eds. Lecture Notes in Computer Science, vol. 1128. Springer Verlag, Karlsruhe, Germany, 46–60.

FITTING, M. 1983. *Proof methods for modal and intuitionistic logics.* Synthese Library, vol. 169. Reidel Publ. Comp.

GENTZEN, G. 1934. Untersuchungen über das logische Schließen. *Mathematical Zeitschrift 39*, 176–210, 405–431. Translated as [Gentzen 1969].

GENTZEN, G. 1969. Investigations into logical deduction. In *The Collected Papers of Gerhard Gentzen*, M. Szabo, Ed. North Holland, Amsterdam, 68–131. Originally appeared as [Gentzen 1934].

GIUNCHIGLIA, E., GIUNCHIGLIA, F., AND TACCHELLA, A. 1999. SAT-based decision procedures for classical modal logics. *Journal of Automated Reasoning.* To appear in the Special Issues: Satisfiability at the start of the year 2000 (SAT 2000).

GIUNCHIGLIA, E., GIUNCHIGLIA, F., SEBASTIANI, R., AND TACCHELLA, A. 1998. More evaluation of decision procedures for modal logics. In *Principles of Knowledge Representation and Reasoning: Proceedings of the Sixth International Conference (KR'98)*, A. Cohn, L. Schubert, and S. Shapiro, Eds. Morgan Kaufmann, San Francisco, CA, 626–635.

GIUNCHIGLIA, F. AND SEBASTIANI, R. 1996a. Building decision procedures for modal logics from propositional decision procedures: Case study of modal K. In *CADE-13*, M. McRobbie and J. Slaney, Eds. Lecture Notes in Artificial Intelligence, vol. 1104. Springer Verlag, 583–597.

GIUNCHIGLIA, F. AND SEBASTIANI, R. 1996b. A SAT-based decision procedure for $\mathcal{ALC}$. In *KR'96.* 304–314.

GOBLE, L. 1974. Gentzen systems for modal logic. *Notre Dame J. of Formal Logic 15*, 455–461.

HAARSLEV, V., MÖLLER, R., AND TURHAN, A.-Y. 1998. Implementing an ALCRP(D) ABox reasoner—progress report. In *Proc. DL-98 International Description Logic Workshop.* Trento, Italy.

HORROCKS, I. 1997. Optimizing tableaux decision procedures for modal logic. Ph.D. thesis, University of Manchester.

HORROCKS, I. 1998. Using an expressive description logic: FaCT or fiction? In *Principles of Knowledge Representation and Reasoning: Proceedings of the Sixth International Conference (KR'98)*, A. Cohn, L. Schubert, and S. Shapiro, Eds. Morgan Kaufmann, San Francisco, CA, 636–647.

HORROCKS, I. AND PATEL-SCHNEIDER, P. 1998. FaCT and DLP. In *Automated Reasoning with Analytic Tableaux and Related Methods, International Conference, TABLEAUX'98*, H. de Swart, Ed. Lecture Notes in Artificial Intelligence, vol. 1397. Springer Verlag, Oisterwijk, The Netherlands, 27–30.

HORROCKS, I., PATEL-SCHNEIDER, P., AND SEBASTIANI, R. 2000. An analysis of empirical testing for modal decision procedures. *Logic Journal of the IGPL 8,* 3, 293–323.

HUSTADT, U. AND SCHMIDT, R. 1997. On evaluating decision procedures for modal logic. In *IJCAI-97.* Vol. 1. 202–207.

MASLOV, S. 1971. Proof-search strategies for methods of the resolution. In *Machine Intelligence*, B. Meltzer and D. Michie, Eds. Vol. 6. American Elsevier, 77–90.

MINTS, G. 1993. Resolution calculus for the first order linear logic. *Journal of Logic, Language and Information 2*, 58–93.

MINTS, G. 1994. Resolution strategies for the intuitionistic logic. In *Constraint Programming*. NATO ASI Series F. Springer Verlag, 289–311.

MINTS, G., DEGTYAREV, A., AND VORONKOV, A. 2000. The inverse method. In *Handbook of Automated Reasoning*, A. Robinson and A. Voronkov, Eds. Elsevier Science and MIT Press. To appear.

MINTS, G., OREVKOV, V., AND TAMMET, T. 1996. Transfer of sequent calculus strategies to resolution for S4. In *Proof Theory of Modal Logic*. Studies in Pure and Applied Logic. Kluwer Academic Publishers.

NIEWENHUIS, R. AND RUBIO, A. 2000. Paramodulation-based theorem proving. In *Handbook of Automated Reasoning*, A. Robinson and A. Voronkov, Eds. Elsevier Science and MIT Press. To appear.

PATEL-SCHNEIDER, P. 1998. DLP system description. In *Collected Papers from the International Description Logic Workshop (DL'98)*, E. Franconi, G. Giacomo, R. MacGregor, W. Nutt, C. Welty, and F. Sebastiani, Eds. CEUR-WS, vol. 11. 87–89.

SCHMIDT, R. 1998. Resolution is a decision procedure for many propositional modal logics. In *Advances in Modal Logic, I*, M. Kracht, M. de Rijke, H. Wansing, and M. Zakharyaschev, Eds. Lecture Notes, vol. 87. CSLI Publications, Stanford, 189–208.

TACCHELLA, A. 1999. *SAT system description. In *Proceedings of the 1999 International Description Logic Workshop (DL'99)*, P. Lambrix, A. Borgida, M. Lenzerini, R. Möller, and P. Patel-Schneider, Eds. CEUR-WS, vol. 22. 142–144.

TAMMET, T. 1994. Proof strategies in linear logic. *Journal of Automated Reasoning 12,* 3, 273–304.

TAMMET, T. 1996. A resolution theorem prover for intuitionistic logic. In *CADE-13*, M. McRobbie and J. Slaney, Eds. Lecture Notes in Artificial Intelligence, vol. 1104. 2–16.

VORONKOV, A. 1985. A proof search method (in Russian). *Vychislitelnye Sistemy 107*, 109–123.

VORONKOV, A. 1990. A proof-search method for the first order logic. In *COLOG'88*, P. Martin-Löf and G. Mintz, Eds. Lecture Notes in Computer Science, vol. 417. Springer Verlag, 327–340.

VORONKOV, A. 1992. Theorem proving in non-standard logics based on the inverse method. In *11th International Conference on Automated Deduction*, D. Kapur, Ed. Lecture Notes in Artificial Intelligence, vol. 607. Springer Verlag, 648–662.

VORONKOV, A. 1999. KЭ: a theorem prover for k. In *Automated Deduction—CADE-16. 16th International Conference on Automated Deduction*, H. Ganzinger, Ed. Lecture Notes in Artificial Intelligence, vol. 1632. Springer Verlag, Trento, Italy, 383–387.

VORONKOV, A. 2000. How to decide K using Э. In *Principles of Knowledge Representation and Reasoning (KR'2000)*, A. Cohn, F. Giunchiglia, and B. Selman, Eds. 198–209.

WALLEN, L. 1990. *Automated Deduction in Nonclassical Logics.* The MIT Press.

WEIDENBACH, C., GAEDE, B., AND ROCK, G. 1996. SPASS & FLOTTER. version 0.42. In *CADE-13*, M. McRobbie and J. Slaney, Eds. Lecture Notes in Artificial Intelligence, vol. 1104. Springer Verlag, 141–145.

Received April 2000; revised July 2000 and October 2000; accepted October 2000