Toshimasa Matsumoto

TIME COMPLEXITY OF A PROOF–SEARCH PROCEDURE FOR K4

Abstract

In this paper, we will give the time complexity of the proof–search procedure given by a sequent system for K4 introduced in [9]. Here, the time complexity means the total number of applications of rules in the proof–search in the worst case for a given formula. We estimate the time complexity by using a certain order which guarantees termination of the proof–search procedure.

1. Introduction

We will discuss the time complexity of a proof–search procedure determined by Mouri’s sequent system for K4 introduced in [9]. Here, proof–search procedure is one of decision procedures which decides first whether a given formula is provable in K4 or not, and then gives us a proof–figure of it when it is provable. In [5,3,4], the space complexity of decision procedures for modal logics are discussed. On the other hand, in [11,1,6], the time complexity of decision procedures has been discussed. In [11], it is shown that checking provability in K is in EXPTIME–complete. In [1], it is shown that Fitting’s translation of S4 to T can be constructed in deterministic polynomial time. From the result, a polynomial bound to the length of branches in both tableau and sequent proof search for S4 and K4 is established. In [6], NPTIME–proof–search strategies for K45 and S5 are discussed.

An early version of the paper was presented at the 36th MLG meeting at Kinosaki, December, 2002.
In the present paper, the time complexity is regarded as the total number of applications of rules in all proof–figures of a given formula in the worst case. In estimating the time complexity of the proof–search procedure determined by Mouri’s sequent system, we use a lexicographic order, which guarantees termination of the proof–search procedure (without redundant loop-checking), to measure the time complexity. Unlike [11,6,1], we estimate the strict value of the time complexity from our proof of termination.

The present paper consists of four sections. In Section 2, we will give a brief explanation of Mouri’s sequent system for $\mathbf{K4}$ and show that the proof–search procedure given by it always terminates for any given formula. The time complexity of the proof–search procedure will be discussed in Section 3. The detailed proof will be discussed somewhere. In the last section, some concluding remarks are given.

2. Mouri’s sequent system for $\mathbf{K4}$

In this section, we will first explain Mouri’s sequent system for $\mathbf{K4}$ and then show that the proof–search procedure given by it always terminates for any given formula. The sequent system for $\mathbf{K4}$ gives a proof–search procedure for $\mathbf{K4}$, in the sense that it checks whether a given formula is provable in $\mathbf{K4}$ or not and moreover gives us a proof–figure of it if it is provable in $\mathbf{K4}$. In addition, a counter–model is constructed by using information on failed proof–figures when the proof–search procedure fails to find a proof–figure of a given formula (see [9] for the details).

In the following, capital letters $A$, $B$, $C$ etc. denote formulas. We take $\land$, $\neg$ and $\Box$ for logical connectives. As usual, formulas of the form $A \lor B$ and $A \supset B$ are taken as abbreviations of $\neg(\neg A \land \neg B)$ and $\neg(A \land \neg B)$, respectively. Greek capital letters $\Gamma$, $\Sigma$, $\Pi$ etc. denote (finite, possibly empty) multisets of formulas. The notation $\square \Gamma$ denotes the multiset $\{\Box A | A \in \Gamma\}$. For a formula $A$, $\text{Sub}(A)$ and $\text{Sub}_{\text{mul}}(A)$ denote the set of all subformulas of $A$ and the multiset consisting of all occurrences of subformulas in $A$, respectively.

Now, rules of Mouri’s sequent system for $\mathbf{K4}$ are shown in Figure 1. It is slightly different but not in an essential way from the standard system for $\mathbf{K4}$. Sequents of the sequent system are expressions of the form $\Gamma \rightarrow \Delta$ with $\langle \Box \Pi | \Box \Sigma \rangle$. where $\langle \Box \Pi | \Box \Sigma \rangle$ is a pair of finite sets of formulas. (Note
that both $\Gamma$ and $\Delta$ are multisets of formulas.) The pair $\langle \Box \Pi | \Box \Sigma \rangle$ is called a history. It plays an important role in order to avoid any redundant loop-checking in proof-search. The technique of using histories was previously used in the work by Heuerding, Seyfried and Zimmermann [2]. The rules $(\text{init})_s$ and $(\text{init})_t$ substantially denote initial sequents. Each upper sequent of $(\Box)_s$ and $(\Box)_t$ is taken 'or'-branch, though it should be understood as 'and'-branch in other rules. It means that if one of the upper sequents of $(\Box)_s$ and $(\Box)_t$ is provable then the lower sequent of them is provable. In order to emphasize 'or'-branch, double lines are used in $(\Box)_s$ and $(\Box)_t$. The idea of using 'or'-branch was introduced in Pinto and Dyckhoff [10].

$$\Box \Gamma, p_1, \ldots, p_n \rightarrow q_1, \ldots, q_m, \Box \Delta \langle \Box \Pi | \Box \Sigma \rangle \quad (\text{init})_s$$

(\Box \Delta \subseteq \Box \Sigma, \Box \Delta \neq \emptyset)

$$\Box \Gamma, p_1, \ldots, p_n \rightarrow q_1, \ldots, q_m \langle \Box \Pi | \Box \Sigma \rangle \quad (\text{init})_t$$

$\Gamma, A, B \rightarrow \Delta \langle \Box \Pi | \Box \Sigma \rangle$ (L)

$\Gamma, A \land B \rightarrow \Delta \langle \Box \Pi | \Box \Sigma \rangle$ (\land R)

$\Gamma \rightarrow A, \Delta \langle \Box \Pi | \Box \Sigma \rangle$ (L)

$\Gamma \rightarrow A, \Delta \langle \Box \Pi | \Box \Sigma \rangle$ (\land R)

$\Box \Gamma, \Gamma \rightarrow A_1 \langle \Box \Pi | \Box \Sigma, \Box \Theta \rangle \ldots \Box \Gamma, \Gamma \rightarrow A_m \langle \Box \Pi | \Box \Sigma, \Box \Theta \rangle$ (\Box)_s

(\Box \Theta \equiv \Box A_1, \ldots, \Box A_m, \Box \Theta \cap \Box \Sigma = \emptyset, \Box \Delta \subseteq \Box \Sigma)

$\Box \Gamma, \Gamma \rightarrow A_1 \langle \Box \Pi | \Box \Sigma, \Box \Theta \rangle \ldots \Box \Gamma, \Gamma \rightarrow A_m \langle \Box \Pi | \Box \Sigma, \Box \Theta \rangle$ (\Box)_t

(\Box \Theta \equiv \Box A_1, \ldots, \Box A_m, \Box \Theta \subseteq \Box \Pi)

Figure 1: Mouri’s sequent system for K4

Although both sides of a sequent in each rule are multisets of formulas, we note that duplications of same formulas will be eliminated in an application of $(\Box)_s$ or $(\Box)_t$ as follows:

$$\Box A, \Box B \rightarrow C < \Box A, \Box B | \Box C, \Box D >$$

As we can see, the subformula property holds in this system because it is a cut–free system.

Mouri proved that the sequent system is sound and complete with respect to K4-frame in [9]. Therefore, the sequent system gives us a proof–
search procedure for K4. We say that a formula $A$ is provable in K4 when
the proof–search procedure find a proof–figure of $A$, in which every initial
sequent contains a formula which appears in both sides of the sequent.
Otherwise, we say that $A$ is unprovable.

The proof–search procedure determined by Mouri’s sequent system
will proceed by generating one or two instances of the upper sequent of
each rule from a given instance of the lower sequent of the rule. In other
words, proof–search proceeds from bottom to top. Hereafter, application
of rules is done from bottom to top.

Here, we remark the order of applying rules of Mouri’s sequent system
in proof–search. The rules (init)$_s$ and (init)$_t$ have priority over other rules.
That is, we must check first whether each of topmost sequents is an initial
one or not. As we can see from the form of each lower sequent of (☐)$_s$ and
(☐)$_t$, it is possible to apply each of (☐)$_s$ and (☐)$_t$ when we cannot apply
any of (∧L), (∧R), (¬L) and (¬R) any more. On the other hand, The
order of applications of (∧L), (∧R), (¬L) and (¬R) are inessential.

Now, we show termination of the proof–search procedure determined
by Mouri’s sequent system. Although its termination is discussed already
in his paper [9], we will give here an alternative proof of termination, which
leads us to an estimation of the time complexity. The point is that any
automatic application of rules eventually terminates without redundant
loop-checking, and shows us whether a given formula is provable or not.

For a formula $A$, we define the length $\ell(A)$ of $A$ inductively as follows:

$$
\ell(p) = 1 \text{ for any propositional variable } p,
\ell(\neg A) = 1 + \ell(A),
\ell(A \land B) = 1 + \ell(A) + \ell(B),
\ell(\Box A) = 1 + \ell(A).
$$

For a multiset $\Delta$ of formulas and for a sequent $\Gamma \rightarrow \Delta \langle \Box \Pi | \Box \Sigma \rangle$, their
lengths are defined as follows:

$$
\ell(\Delta) = \sum_{A \in \Delta} \ell(A),
\ell(\Gamma \rightarrow \Delta \langle \Box \Pi | \Box \Sigma \rangle) = \ell(\Gamma) + \ell(\Delta).
$$

**Theorem 1.** The proof–search procedure determined by Mouri’s sequent
system for K4 always terminates for any given formula, no matter how
rules are applied.
PROOF. Let $A$ be any given formula. We show that the proof-search for proof-figures of $A$ always terminates. Since the subformula property holds in the sequent system, any formula occurring in a proof-figure of $A$, if exists, belongs to $\text{Sub}(A)$. Let $c = |\text{Sub}(A)|$. For any sequent $\Gamma \rightarrow \Delta(\square \Pi \square \Sigma)$ occurring in a proof-figure of $A$, we define the degree $D(\Gamma \rightarrow \Delta(\square \Pi \square \Sigma))$ of $\Gamma \rightarrow \Delta(\square \Pi \square \Sigma)$, which is a triple of natural numbers, as follows:

$$D(\Gamma \rightarrow \Delta(\square \Pi \square \Sigma)) = (c - |\square \Pi|, c - |\square \Sigma|, \ell(\Gamma \rightarrow \Delta(\square \Pi \square \Sigma))).$$

We note that both $c - |\square \Pi|$ and $c - |\square \Sigma|$ are positive. We define the lexicographic order $\ll$ over triples of natural numbers as follows:

$$(x_1, x_2, x_3) \ll (y_1, y_2, y_3) \iff x_1 < y_1 \text{ or } (x_1 = y_1 \text{ and } x_2 < y_2) \text{ or } (x_1 = y_1, x_2 = y_2 \text{ and } x_3 < y_3).$$

Now, we can show that every application of rules decreases strictly the value of the degree $D$. Since the lexicographic order $\ll$ is well-order, we have the termination of the proof-search procedure.

The following table shows that each application of the rules decreases the triples with respect to the lexicographic order:

<table>
<thead>
<tr>
<th>(\square \Pi)</th>
<th>(\square \Sigma)</th>
<th>$\ell(\Gamma \rightarrow \Delta(\square \Pi \square \Sigma))$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\wedge L$</td>
<td>$\wedge R$</td>
<td>$\wedge$</td>
</tr>
<tr>
<td>$\neg L$</td>
<td>$\neg R$</td>
<td>$\neg$</td>
</tr>
<tr>
<td>$\square s$</td>
<td>$\square t$</td>
<td>$&lt;$</td>
</tr>
</tbody>
</table>

The interpretation is obvious: $(\square)_{t}$ and $(\square)_{s}$ strictly decrease $c - |\square \Pi|$ and $c - |\square \Sigma|$, respectively. Others leave $c - |\square \Pi|$ and $c - |\square \Sigma|$ unchanged but strictly decrease $\ell(\Gamma \rightarrow \Delta)$. Thus, we can obtain our theorem.

3. Time complexity

In this section, we give an estimation of the time complexity of proof-search procedure determined by Mouri’s sequent system for $\mathbf{K4}$ in terms of the length of a given formula $A$. Here, the time complexity of the proof-search procedure denotes an upper bound for the number of required steps.
in the proof-search for a given formula $A$, that is, the total number of applications of the rules in all possible proof-figures, in the worst case, in the proof-search for $A$.

Now, as we can see, each application of $(\wedge R)$ generates two upper sequents, that is, two branches. However, for simplicity’s sake, first ignoring the branching by each application of $(\wedge R)$, we will estimate the time complexity of the proof-search procedure. Then, taking the branching by $(\wedge R)$ into account, we will briefly discuss a more exact estimation of the time complexity.

**Simple estimation**

By analyzing the proof of termination shown in the previous section, an upper bound for the number of required steps in the proof-search will be estimated in terms of the length of a given formula $A$. The underlying idea is based on the lexicographic order used in the proof of termination in the previous section. Therefore, the degree of sequents is used to measure time complexity. For a given formula $A$, let $T(A)$ denote time complexity of the proof-search procedure determined by Mouri’s sequent system for $K4$.

**Lemma 1.** For any formula $A$, $|\text{Sub}(A)| \leq \ell(A)$ and $|\text{Sub}_{\text{mul}}(A)| = \ell(A)$.

**Proof.** Trivial.

**Theorem 2.** Let $l = \ell(A)$ for a given formula $A$. Then $T(A)$ is bounded by $2l^2$. In other words, the proof-search procedure determined by Mouri’s sequent system for $K4$ terminates within $2l^2$ steps.

**Proof.** To show this, consider the degree of any possible sequents occurring in all proof-figures of $A$. Recall that the degree of sequent is as follows:

$$D(\Gamma \rightarrow \Delta(\Box \Pi \Box \Sigma)) = (c - |\Box \Pi|, c - |\Box \Sigma|, \ell(\Gamma \rightarrow \Delta(\Box \Pi \Box \Sigma))) ,$$

where $c = |\text{Sub}(A)|$. Here, we ignore the branching by $(\wedge R)$. It enables us to estimate the total number of applications of $(\Box)_s$ and $(\Box)_t$, and that of $(\wedge L)$, $(\wedge R)$, $(\neg L)$ and $(\neg R)$, separately. We can see that $T(A)$ is the summation of them.
First, in order to estimate the number of applications of \((\Box)_s\) and \((\Box)_t\) in all possible proof-figures of \(A\), we fix \(\ell(\Gamma \rightarrow \Delta(\Box\Pi \Box\Sigma))\). Here, see Figure 2. At worst, \((c - |\Box\Pi|, c - |\Box\Sigma|)\) visits not only the point \((c, c)\) but also all points of the shadowed square, in Figure 2, with integer coefficients. Moreover, consider an application of either \((\Box)_s\) or \((\Box)_t\). Since upper sequents of these rules must be interpreted as 'or'-branches, there will be many possibilities of the choice of proof-figures to be searched. The tree shown in Figure 3 is the search tree generated by 'or'-branches. The branches at each node denote 'or'-branches generated by each application of \((\Box)_s\) or \((\Box)_t\). Let \(b\) be the maximal number of possible 'or'-branches generated by each application of \((\Box)_s\) or \((\Box)_t\). The height of the tree shown in Figure 3 is \((c - 1)c\). This means that the total number of applications of \((\Box)_s\) and \((\Box)_t\) on a path in a proof-figure of \(A\) is \((c - 1)c\) at worst. Therefore, the total number of applications of \((\Box)_s\) and \((\Box)_t\) in the proof-search in the worst case can be estimated at:

\[
\sum_{i=0}^{c} b^i = \frac{b(b^{(c-1)c} - 1)}{b - 1}.
\]

Next, we estimate the total number of \((\land L), (\land R), (\neg L)\) and \((\neg R)\). Let \(s\) be the total number of applications of the rules \((\land L), (\land R), (\neg L)\) and \((\neg R)\) in the worst case which are applied before the next application of either \((\Box)_s\) or \((\Box)_t\). Here, recall the order of the applications of the rules of Mouri’s sequent system in proof-search. Each node of tree shown in Figure 3 can be regarded also as possible timing of applications of \((\land L), (\land R), (\neg L)\) and \((\neg R)\). Since the total number of the nodes of the tree is:

\[
1 + b + b^2 + \cdots + b^{(c-1)c} = 1 + \frac{b(b^{(c-1)c} - 1)}{b - 1},
\]

The total number of applications of \((\land L), (\land R), (\neg L)\) and \((\neg R)\) in the worst case can be estimated at:

\[
s \cdot \left(1 + \frac{b(b^{(c-1)c} - 1)}{b - 1}\right).
\]

Thus, the summation \(T(A)\) of the total number of applications of the rules \((\Box)_s\) and \((\Box)_t\), and that of the rules \((\land L), (\land R), (\neg L)\) and \((\neg R)\) in the worst case can be estimated at:
\[ T(A) \leq \frac{b(b^{(c-1)c} - 1)}{b - 1} + s \left( 1 + \frac{b(b^{(c-1)c} - 1)}{b - 1} \right) \]
\[ = (s + 1) \cdot \frac{b(b^{(c-1)c} - 1)}{b - 1} + s \]
\[ \leq (s + 1) \cdot b(b^{(c-1)c} - 1) + s . \]

Figure 2: The total number of applications of \((\Box)_s\) and \((\Box)_t\).

Figure 3: The search tree generated by 'or'-branches

Now, it remains to estimate \(c\), \(b\) and \(s\) in terms of the length of \(A\). First, we estimate \(b\). Since \(b\) is the maximal number of possible 'or'-branches by each application of \((\Box)_s\) and \((\Box)_t\), \(b\) is estimated to be less than or equal to \(|Sub(A)|\). Next, we estimate \(s\). All principal formulas of \((\land L)\), \((\land R)\), \((\neg L)\) and \((\neg R)\) are in \(Sub_{mul}(A)\). An application of them
strictly decreases the length of sequent by one. Recall that the branches by \((\land R)\) are ignored here. Taking these facts into account, \(s\) is estimated to be less than or equal to \(|Sub_{mul}(A)|\). From Lemma 1, it follows that \(c \leq l, b \leq l\) and \(s \leq l\). Therefore,

\[
T(A) \leq (s + 1) \cdot b\{b^{(c-1)c} - 1\} + s \\
\leq (l + 1) \cdot l\{l^{(l-1)l} - 1\} + l \\
= l^{2l-1} + l^{2l-1} - l^2 \\
\leq 2l^2.
\]

Thus, \(T(A)\) can be bounded by \(2l^2\).

Although this time complexity is estimated by ignoring branching by \((\land R)\), it gives us a rough estimation of the bound of the number of necessary steps in the proof–search.

**Strict estimation**

Next, taking branching by \((\land R)\) into account, we consider the time complexity again. We have to consider not only ‘or–branch’ by \((\lor)_{s}\) and \((\lor)_{t}\) but also branches by \((\land R)\). In this case, the time complexity of the proof–search procedure becomes as follows:

**Theorem 3.** Let \(l = \ell(A)\) for a given formula \(A\). Then \(T(A)\) is strictly bounded by \((2l)^{2l^2}\).

We will omit the proof of Theorem 3, since more complicated arguments are necessary. Finally, we remark time complexity of the proof–search procedure for a given formula \(A\) in which there is no occurrence of \(\Box\).

**Remark 1.** Let \(l = \ell(A)\) for a given formula \(A\) in which there is no occurrence of \(\Box\). Then, taking the branching by \((\land R)\) into account, \(T(A)\) is strictly bounded by \(2^l - 1\).
4. Conclusion

We have given two time complexity of the proof–search procedure determined by Mouri’s sequent system for $\mathbf{K4}$. Our method of estimation of the time complexity will give a new criterion for proof–search procedures. The time complexity of decision procedure has been discussed in [11,1,6]. Since our attitude toward time complexity is slightly different from theirs, it could not be easy to compare our result with the results of [11,1,6]. However, it should be worth clarifying the relation between our result and them.

It will be an interesting problem to estimate the time complexity of other proof–search procedures. In fact, the time complexity of a proof–search procedure determined by a sequent system for $\mathbf{S4}$ introduced in [8] can be also bounded by $2^{il^2}$ in the case of simple estimation, that is, when we ignore ‘and’–branches of the rules, where $l$ is the length of a given formula. On the other hand, the present author has already estimated the time complexity of a proof–search procedure determined by a tableau system for $\mathbf{S4}$ in [7].

References


