Proof–Search in Modal and Temporal Logics

by

Toshimasa MATSUMOTO

submitted to
Japan Advanced Institute of Science and Technology
in partial fulfillment of the requirements
for the degree of
Doctor of Philosophy

Supervisor: Professor Hiroakira Ono

School of Information Science
Japan Advanced Institute of Science and Technology

June 2004

Abstract

This thesis investigates proof–search procedures for modal and temporal logics. A proof–search procedure is a decision procedure which gives us a proof of a give formula if it is provable. Main target of the present thesis is to give a proof–search procedure for the temporal logic $\mathbf{K_t}$. Since the study of temporal logics is now applied to various branches of computer science including software engineering and artificial intelligence, to find an efficient proof–search procedure for $\mathbf{K_t}$ will be an important problem as $\mathbf{K_t}$ is the most basic one among them. Our study will be a prototype of further studies for logicians who utilize temporal logics for their researches.

In proof–search, we usually need to check whether there are repetitions of the same sequents (or formula sets) or not in proofs. This is called loop–checkings. Naturally, loop–checking causes inefficiency. The most desirable way of avoiding loop–checkings is to introduce such a proof system that loops never occur in its proofs. In standard systems for $\mathbf{K_t}$, several kinds of loops will occur. To get a proof–search procedure for $\mathbf{K_t}$, we begin with finding one for each of modal logics **S4**, **KB** and **K4B**. For, the modal operator $\Box$ of each of these logics behaves like tense operators $[F]$ and $[P]$ in $\mathbf{K_t}$. Our study goes along as follows:

1. proof system for **S4**,
2. proof system for **KB**,
3. proof system for **K4B**,
4. proof system for $\mathbf{K_t}$.

Techniques of getting loop–free proof systems for first three modal logics will be applied to $\mathbf{K_t}$, and thus we can get a loop–free proof for it. To avoid several kinds of loop, we introduce an auxiliary modal operator $\blacksquare$ and *histories*, which are pairs of sets of $\Box$–formulas, to standard proof systems for **S4**, **KB**, **K4B** and $\mathbf{K_t}$. We will see that $\blacksquare$ and histories enable us to avoid loop–checkings. Then, we show that our proof–search procedure using each of these proof systems terminates eventually, and gives a loop–free proof if our proof–search of a given sequent is successful.

On the other hand, when our proof–search procedure fails to find a proof of a given formula, we will give a way of constructing a (finite) counter–model for it. In order to construct counter–models, we introduce finite Kripke frames called *model graphs* which facilitate construction of counter–models. We can easily get a counter–model from a model graph when we construct it. From this, both completeness and the finite model property of each of these four systems follow.

Acknowledgments

I would like to express to my supervisor, Professor Hiroakira Ono of Japan Advanced Institute of Science and Technology, my deepest gratitude for his kind guidance and constant encouragement during this work.

I wish to express to my advisor in master's course, Professor Yoshihito Toyama of Tohoku University, my heartfelt gratitude for his many invaluable comments and suggestions throughout this dissertation. It is no exaggeration to say that I owe what I am much to Professor Toyama as well as Professor Ono.

I am really grateful to Professor Yuichi Komori of Chiba University, and Professor Atsushi Ohori, Research Professor Mizuhito Ogawa and Associate Professor Hajime Ishihara of Japan Advanced Institute of Science and Technology for their useful comments and informative suggestions.

I am sincerely thankful for all members of Computational Logic Laboratory in Japan Advanced Institute of Science and Technology.

Finally, I express my sincerest thanks and appreciation to Professor Michio Takano of Niigata University, who read the first draft of this dissertation carefully and pointed out some flaws in original proofs of completeness of our sequent systems. I could not have finished this dissertation without his invaluable remarks and suggestions.

Contents

Chapter 1

Introduction

1.1 Backgrounds

This thesis investigates loop–free proof–search procedures for modal and temporal logics. We approach modal and temporal logics proof–theoretically. Proof–theory is a study which clarifies syntactical properties of logics, in particular, properties of proofs, and is discussed on systems formalizing logics, for example tableau systems, sequent systems and so on. By analyzing them, it is possible to show important properties such as the subformula property, cut elimination theorem, Craig's interpolation theorem and so on.

In such a syntactical approach to logics, as is often the case, we need to show whether a given formula is provable or not. However, most theorem provers tell only whether a given formula is provable or not. It does not suffice for our approach to logics. When a given formula is provable, we need a proof which shows us why it is provable. That is why we desire a procedure which gives us a proof of a provable formula. We call such a procedure a *proof–search procedure*, that is a decision procedure which decides whether a given formula is provable or not, and gives us a proof of it when it is provable. In addition, when a proof–search procedure always returns proofs in which there is no repetition, it is called *loop–free*. On the other hand, when a given formula is not provable, we need a counter–model which shows us why it is not provable. Such a construction of counter–models is also desirable.

Our first goal is to give loop–free proof–search procedures for some of modal and temporal logic. In proof–search, from computational point of view, we definitely need techniques to reduce search space. For example, we have to avoid repetitions in proofs, we have to check whether there are loops in proof–search or not. When cut rule is applied in proof–search, since there are infinite possibility of cut formulas, we need to restrict them strictly. That is why we are required to reduce the number of candidates for cut formula, otherwise applications of cut rule would cause a big inefficiency.

Main target of this thesis is to give a loop–free proof–search procedure for a temporal logic $\mathbf{K_t}$. For, it would be quite convenient to have such a procedure, since $\mathbf{K_t}$ is very

useful for formalizing various notions which appear in computer science, and is applied by combining with other modal logics. The temporal logic $\mathbf{K_t}$ is a bimodal logic in which there are the future operator $[F]$ and the past operator $[P]$ as modal operators. For our purpose, we first discuss proof–search procedures for monomodal logics $\mathbf{S4}, \mathbf{KB}$ and $\mathbf{K4B}$, since the modal operator $\Box$ of these logics behaves like $[F]$ and $[P]$. We note here that cut elimination theorem does not hold for the standard sequent systems for $\mathbf{KB}$ and $\mathbf{K4B}$.

Our second goal is to give a way of constructing a counter–model for a given formula when it is not provable. To do that, we use *model graphs* which are introduced in tableau system. In tableau systems, completeness is shown via model graphs, that facilitate constructing counter–models. We will apply model graphs to show completeness of sequent systems.

1.2 Outline of this thesis

This thesis discusses proof–search procedures for tableau systems and Gentzen's sequent systems of modal and temporal logics. To facilitate introducing a proof–search procedure for $\mathbf{K_t}$, we first discuss proof–search procedure for $\mathbf{S4}$, $\mathbf{KB}$ and $\mathbf{K4B}$. The techniques used in our proof systems for $\mathbf{S4}$, $\mathbf{KB}$ and $\mathbf{K4B}$ will be applied to a proof system for $\mathbf{K_t}$.

Chapter 2 reviews propositional modal logics, in particular, $\mathbf{S4}, \mathbf{KB}$, and $\mathbf{K4B}$. Kripke semantics, tableau systems and sequent systems are introduced for them in Section 2.2, Section 2.3 and Section 2.4, respectively. In Section 2.5, we show what soundness and completeness mean from point of view of proof–search procedures . In Section 2.6, it will be clear that our proving completeness construct the finite model property.

Chapter 3 is devoted to a study of proof–search procedure for the modal logic $\mathbf{S4}$ based on tableau system. In Section 3.1, a tableau system for $\mathbf{S4}$ is introduced. Then, some deficiency of the system is pointed out, if we use it for proof–search. In Section 3.2, the notion of *histories* is introduced in order to avoid loops In Section 3.3, we will introduce a tableau system, called $\mathcal{C}\mathbf{S4}+$, for $\mathbf{S4}$ with history. In Section 3.4, model graphs will be defined for $\mathcal{C}\mathbf{S4}+$. Model graphs will be used as a tool to show completeness of $\mathcal{C}\mathbf{S4}+$.

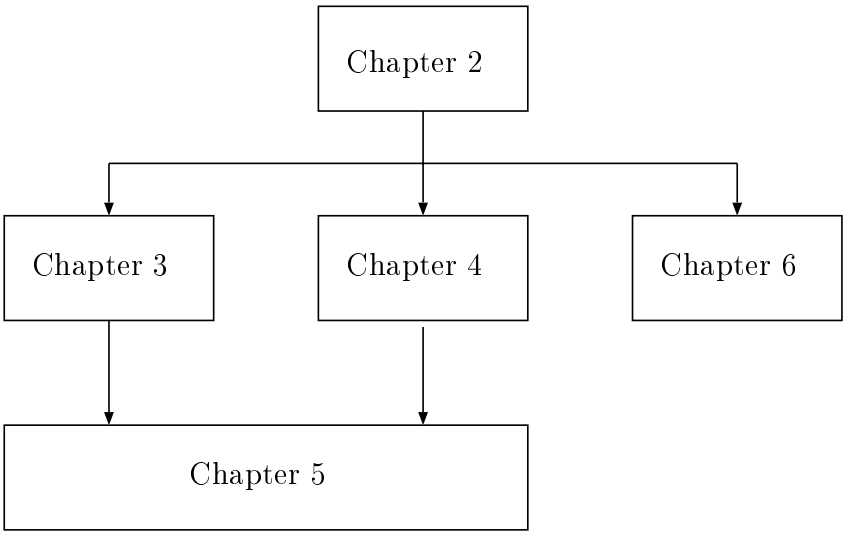
In Chapter 4, we discuss monomodal logics $\mathbf{KB}$ and $\mathbf{K4B}$. In Section 4.1, standard sequent systems for $\mathbf{KB}$ and $\mathbf{K4B}$ will be introduced. It is known that though cut elimination theorem does not hold for them, they enjoy the subformula property. In Section 4.2, a sequent system, called $\mathcal{S}\mathbf{KB}$, for $\mathbf{KB}$ will be introduced, and completeness of $\mathcal{S}\mathbf{KB}$ will be shown via model graphs. In addition, a sequent system, called $\mathcal{S}\mathbf{K4B}$, for $\mathbf{K4B}$ will be introduced, and completeness of $\mathcal{S}\mathbf{K4B}$ will be shown via model graphs in Section 4.3. We will see that each cut formula in each proof of $\mathcal{S}\mathbf{KB}$ and $\mathcal{S}\mathbf{K4B}$ can

be restricted to one among finitely many formulas of special form.

Chapter 5 discusses termination and upper bound of proof–search procedure. Termination is one of important property for proof–search procedures. Here, the upper bound of proof–search procedure means the total number of applications of rules in proof–search in the worst case. In Section 5.2, we will discuss termination and the upper bound of the proof–search procedure determined by Mouri's sequent system for **K4**. In Section 5.3 and Section 5.4, termination and upper bounds of proof–search procedures for $\mathcal{S}$**KB** and $\mathcal{S}$**K4B** will be discussed. In Section 5.5, we will show that the proof–search procedure determined by $\mathcal{C}$**S4**+ terminate always, and the upper bound of the proof–search procedure will be discussed.

Chapter 6 puts all techniques obtained in previous chapters to trial for introducing a loop–free sequent system for $\mathbf{K_t}$. This is a new approach to $\mathbf{K_t}$. In Section 6.1, temporal logic $\mathbf{K_t}$ and its Kripke semantics are introduced. In Section 6.2, we introduce a sequent systems for $\mathbf{K_t}$, which enjoy the subformula property. However, there are infinite possibilities of cut formula in proofs of the sequent system. In Section 6.3, a sequent system, called $\mathcal{S}\mathbf{K_t}$, for $\mathbf{K_t}$ will be introduced. Each cut formula is a proof of $\mathcal{S}\mathbf{K_t}$ can be restricted to one among finitely many formulas of special form. In order to show the completeness, we introduce modal graphs for $\mathcal{S}\mathbf{K_t}$ in Section 6.4. In Section 6.5, we show completeness of $\mathcal{S}\mathbf{K_t}$ by giving a way of construction of counter–models via modal graph. In Section 6.6, termination of the proof–search procedure determined by $\mathcal{S}\mathbf{K_t}$ is shown explicitly. In Section 6.7, some concluding remarks will be given.

The dependencies among chapters are given by the following diagram :



Our study goes along the following arrows in Figure 1.2.

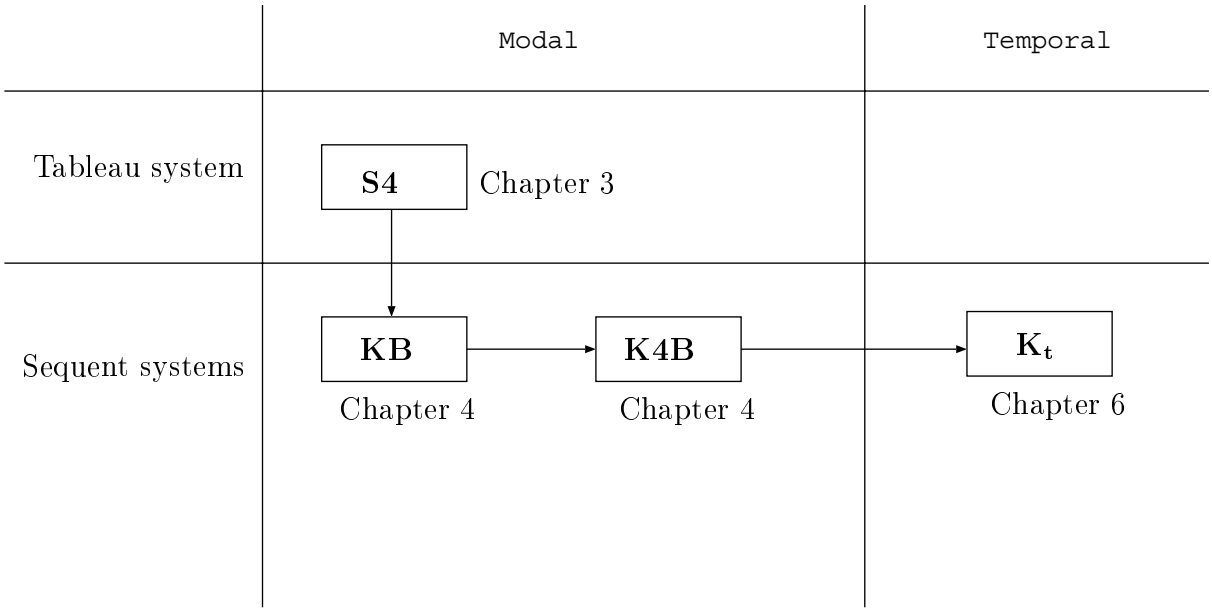


Figure 1.1: Outline of this thesis

Chapter 2

Preliminaries

In this chapter, we introduce notations and basic notions of modal logics, and give a brief survey. In Section 2.1, we will give a brief survey for modal logics, in particular ones which we discuss in our thesis. In Section 2.2, Kripke semantics will be given. In Section 2.3 and Section 2.4, we will introduce tableau systems and sequent systems. In Section 2.5, we will explain how tableau systems and sequent systems gives us proof–search procedures. In Section 2.6, it will be clarified that our proof of completeness gives us the finite model property and also decidability.

2.1 Modal logics

The language $\mathcal{L}$ of propositional modal logic consists of propositional variables, denoted by p, q etc. and logical connectives $\wedge$, $\vee$, $\supset$, $\neg$ and $\Box$. Formulas are defined in the usual way and denoted by A, B etc., Sometimes, we use propositional variables and formulas with subscripts. A *literal* is either an atomic formula (a positive literal) or negation of an atomic formula (a negative literal). A pair $(p, \neg p)$ of literals of this form is said to be complementary. Greek capital letters Γ, Δ, Π etc. denote finite (possibly empty) sets of formulas. The notation $\Box\Gamma$ denotes the set $\{\Box A | A \in \Gamma\}$, and $Sub(\Gamma)$ denotes the set of all subformulas of all formula in Γ. For a set Γ of formulas, $|\Gamma|$ denotes the cardinality of Γ.

A set $\mathbf{L}$ of formulas is a *modal logic*, if the following conditions are satisfied:

- all tautologies belong to $\mathbf{L}$,
- if A and $A \supset B \in \mathbf{L}$, then $B \in \mathbf{L}$ (modus ponens), and
- if $A \in \mathbf{L}$, then $\Box A \in \mathbf{L}$ (rule of necessitation).

Let $\mathbf{L}$ a modal logic and $\mathbf{Q}$ be a set of formulas. The symbol $\mathbf{K}$ denotes the least model logic containing the following axiom

$$\mathbf{K} \quad : \quad \Box(A \supset B) \supset (\Box A \supset \Box B).$$

Any modal logic with the axiom **K** is called a *normal* modal logic. Historical names for some well–known axiom schemes are

$$\begin{aligned} \mathbf{T} &: \Box A \supset A, \\ \mathbf{4} &: \Box A \supset \Box\Box A, \\ \mathbf{B} &: A \supset \Box\Diamond A, \end{aligned}$$

where the symbol $\Diamond$ is the abbreviation of $\neg\Box\neg$. In our thesis, the following modal logics can be discussed. Here, $\mathbf{K} \cup \{X, Y\}$ for axiom schemes X, Y denotes the smallest modal logic containing **K** and axiom schemes X, Y.

$$\begin{aligned} \mathbf{KT} &= \mathbf{K} \oplus \{\mathbf{T}\}, \\ \mathbf{K4} &= \mathbf{K} \oplus \{\mathbf{4}\}, \\ \mathbf{S4} &= \mathbf{K} \oplus \{\mathbf{T}, \mathbf{4}\}, \\ \mathbf{KB} &= \mathbf{K} \oplus \{\mathbf{B}\}, \\ \mathbf{K4B} &= \mathbf{K} \oplus \{\mathbf{4}, \mathbf{B}\}. \end{aligned}$$

2.2 Kripke semantics

Kripke semantics is an important semantics for modal logics. It is defined by using Kripke frames, where a *Kripke frame* is a pair (W, R) such that W is a non–empty set and R is a binary relation on W. Sometimes, each member of W is called a possible world and R is called an accessibility relation. We say that a possible world w' is accessible from a possible world w if and only if wRw'. Let (W, R) be a Kripke frame and V be a mapping from propositional variables to 2^W, called a valuation on (W, R). A *Kripke model* is a triple (W, R, V). For a given Kripke model (W, R, V), a relation $\models$ between an element of W and a formula is defined inductively as follows:

$$\begin{aligned} w \models p \quad & \textit{iff} \quad w \in V(p), \\ w \models A \wedge B \quad & \textit{iff} \quad w \models A \text{ and } w \models B, \\ w \models A \vee B \quad & \textit{iff} \quad w \models A \text{ or } w \models B, \\ w \models A \supset B \quad & \textit{iff} \quad w \models A \text{ implies } w \models B, \\ w \models \neg A \quad & \textit{iff} \quad w \not\models A, \\ w \models \Box A \quad & \textit{iff} \quad \text{for all } w', wRw' \text{ implies } w' \models A. \end{aligned}$$

The relation $\models$ is defined uniquely by valuation V. Therefore, $\models$ and $(W, R, \models)$ are also called a valuation and a Kripke model, respectively, as far as there is no confusion. If $w \models A$ we say that A is *true* at w or w *makes* A *true*. A formula A is *satisfiable in a model* $(W, R, \models)$ if there exists some $w \in W$ such that $w \models A$. A formula A is *true in a model* $(W, R, \models)$, written as $\models A$, if $w \models A$ for any $w \in W$. A formula A is *satisfiable in a frame* (W, R) if there exists some valuation $\models$ and some world $w \in W$ such that $w \models A$. A formula A is *valid in a frame* (W, R) if, for any valuation $\models$ and any $w \in W$, $w \models A$.

Suppose that (W, R) is a frame. Then, the binary relation R satisfies one of the following first–order conditions if and only if the axiom schemes corresponding the condition is valid in (W, R). The study of correspondence of this kind between conditions on accessibility relations and axiom schemes is called a *corresponding theory*.

T	:	$\forall u(uRu)$	(reflexive)
4	:	$\forall u \forall v \forall w(uRv$ and vRw imply $uRw)$	(transitive)
B	:	$\forall u \forall v(uRv$ implies $vRu)$	(symmetric)

For a logic **L**, every frame with the conditions corresponding to the axioms is called **L**–frame. For example, a frame (W, R) is called **S4**–frame if R is reflexive and transitive. For a modal $(W, R, \models)$ we say it is an **L**–*model* when (W, R) is **L**–frame. An **L**–model $(W, R, \models)$ is an **L**–*model for* Γ if there exists some $w \in W$ such that $w \models A$ for all $A \in \Gamma$, which is denoted $w \models \Gamma$.

For a given finite set Γ and a formula A, we say that A is a *logical consequence* of Γ in **L** if, for every **L**–model $(W, R, \models)$ and for every $w \in W$, $w \models \Gamma$ then $w \models A$. We write $\Gamma \models_{\mathbf{L}} A$ if A is a logical consequence of Γ in **L**. It is clear that the following proposition holds from the definition of logical consequence.

Proposition 2.2.1 $\Gamma \models_{\mathbf{L}} A$ *iff* $\Gamma \cup \{\neg A\}$ *is* **L**–*unsatisfiable.*

2.3 Tableau system

A tableau system consists of a set of tableau rules. Each tableau rule consists of a *numerator* above the line and a (finite) list of *denominator* below the line. It is of the following form, in general:

$$\frac{N}{D_1 | D_2 | \cdots | D_k}$$

The denominators $D_1, D_2, \cdots, D_k$ are separated by vertical bars. The numerator N is a finite set of formulas and so is each denominator D_i. We will introduce the tableau system $\mathcal{C}$**PC** for classical logic. Rules of $\mathcal{C}$**PC** are given in Figure 2.1.

The punctuation "," of numerator and denominator and the vertical bar "|" intuitively mean"and" and "or", respectively. The formula shown explicitly in the numerator of each static rule is called the *principal formula*, and formulas shown explicitly in the denominators of each static rule is called the *side formulas*. We note that each rule decomposes its main formula except rule (f) into simpler formulas.

A *tableau* for Γ is a finite upside–down tree with the root Γ each of whose nodes carries a finite set of formulas. A tableau is constructed by repeated applications of tableau rules. A tableau rule with numerator N is applicable to a node carrying a set Δ if Δ is an instance of N. When we want to test whether $\Gamma \models A$ or not, we construct a

$$\frac{\Gamma, A, \neg A}{f}\ (f) \qquad \frac{\Gamma, A \wedge B}{\Gamma, A, B}\ (\wedge)$$

$$\frac{\Gamma, A \vee B}{\Gamma, A \mid \Gamma, B}\ (\vee) \qquad \frac{\Gamma, A \supset B}{\Gamma, \neg A \mid \Gamma, B}\ (\supset)$$

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

$$\frac{\Gamma, \neg(A \supset B)}{\Gamma, A, \neg B}\ (\neg\supset) \qquad \frac{\Gamma, \neg\neg A}{\Gamma, A}\ (\neg\neg)$$

Figure 2.1: The tableau system $\mathcal{C}\mathbf{PC}$

tree whose root is $\Gamma \cup \{\neg A\}$ by repeated applications of tableau rules. The rule (f) is meant to have priority over any other rule.

We say that the tableau system has *the analytical superformula property* if for any finite set of formulas Γ we can assign a finite set of formulas, which contains all formulas that may appear in any tableau for Γ.

Proposition 2.3.1 *The tableau system $\mathcal{C}\mathbf{PC}$ has the analytical superformula property. That is, for any finite set of formulas Γ, we can assign a finite set $\Gamma^{\mathcal{C}\mathbf{PC}*} = Sub(\Gamma) \cup \neg Sub(\Gamma) \cup \{f\}$, which contains all formulas that may appear in any tableau for Γ.*

A branch of a tableau is *closed* if its leaf node carries $\{f\}$, otherwise it is *open*. A tableau is *closed* if all its branches are closed, otherwise it is *open*. A finite set Γ is *consistent* if no tableau for Γ is closed, otherwise Γ is *inconsistent*. We write down $\Gamma \vdash A$ if $\Gamma \cup \{\neg A\}$ is inconsistent.

We say that a tableau rule (r) is *invertible* in $\mathcal{C}\mathbf{PC}$ if there is a closed tableau for an instance of the numerator N of (r) then there is a closed tableau for an appropriate instance of each of the denominators D_i of (r). For instance, $(\neg\wedge)$ is invertible if $\Gamma, \neg(A \neg B)$ has a closed tableau then both $\Gamma, \neg A$ and $\Gamma, \neg B$ have closed tableaux. Note that the converse holds always because of then rule $(\neg\wedge)$.

Proposition 2.3.2 *Each static rule of $\mathcal{C}\mathbf{PC}$ is invertible in $\mathcal{C}\mathbf{PC}$.*

Proof) Similarly to the proof in [10]. □

As for further information on tableau systems, see [10].

2.4 Sequent system

In this section, we introduce a Gentzen's style sequent system **LK** for classical logic. Let Greek capital letters Γ, Δ, Π, Σ, Θ etc. be sequences (may be sets, possibly empty set) of formulas. Any expression of the form $\Gamma \to \Delta$ is called a *sequent*, where the left hand side Γ *the antecedent* and the right hand side Δ *the succedent*. Let S_1, S_2 and S be sequents. An *inference rule* is of the form either

$$\frac{S}{S_1} \quad \text{or} \quad \frac{S_1 \quad S_2}{S}$$

In the inference rule, S_1 and S_2 are called the *upper sequents*, and S the *lower sequent*. In particular, S_1 and S_2 are called the *left* and *right upper sequent* of the inference rule, respectively. The sequent system **LK** for classical logic consists of initial sequents and inference rules in Figure 2.2.

Inference rules $(w \to)$ and $(\to w)$ are called *weakening rules*, $(c \to)$ and $(\to c)$, *contraction rules*, and $(e \to)$ and $(\to e)$, *exchange rules*. The formula A in cut rule (cut) is called the *cut formula* of the rule. The formula shown explicitly in the lower sequent of each rule is called the *principal formula*, and formulas shown explicitly in the upper sequents of each rule is called the *side formulas*.

In a sequent system $\mathcal{S}$, proofs of $\mathcal{S}$ and *end sequent* of the proof are defined as follows:

1. Each initial sequent is a proof of $\mathcal{S}$, and the end sequent of the proof is itself.

2. Let P_1 and P_2 be proofs of $\mathcal{S}$ with the end sequents S_1 and S_2, respectively. If

$$\frac{S_1}{S} \quad \text{or} \quad \frac{S_1 \quad S_2}{S}$$

   is one of the inference rules in $\mathcal{S}$, then

$$\frac{P_1}{S} \quad \text{or} \quad \frac{P_1 \quad P_2}{S}$$

   is a proof of $\mathcal{S}$, and the end sequent is S. A sequent S is provable in $\mathcal{S}$ if there exists a proof of $\mathcal{S}$ whose end sequent is S.

If a sequent S is provable in a system $\mathcal{S}$, then it is often denoted by $\mathcal{S} \vdash S$. For a formula A, A is said to be provable in a sequent system if the sequent $\to A$ is provable in the sequent system.

Cut elimination theorem for a given sequent system $\mathcal{S}$ says that any sequent S which is provable in $\mathcal{S}$ has a proof of S containing no applications of cut rule. Such a proof is a called a cut–free proof. When cut elimination theorem holds for $\mathcal{S}$, we say that $\mathcal{S}$ has the cut elimination property.

initial sequent

$$A \to A$$

inference rules

(1) structural rules

$$\frac{\Gamma \to \Delta}{A, \Gamma \to \Delta}\ (w \to) \qquad \frac{\Gamma \to \Delta}{\Gamma \to \Delta, A}\ (\to w)$$

$$\frac{A, \Gamma \to \Delta}{A, A, \Gamma \to \Delta}\ (c \to) \qquad \frac{\Gamma \to \Delta, A}{\Gamma \to \Delta, A, A}\ (\to c)$$

$$\frac{\Gamma, A, B, \Pi \to \Delta}{\Gamma, B, A, \Pi \to \Delta}\ (e \to) \qquad \frac{\Gamma \to \Delta, A, B, \Sigma}{\Gamma \to \Delta, B, A, \Sigma}\ (\to e)$$

$$\frac{\Gamma \to \Delta, A \quad A, \Pi \to \Sigma}{\Gamma, \Pi \to \Delta, \Sigma}\ (cut)$$

(2) logical rules

$$\frac{A, \Gamma \to \Delta}{A \wedge B, \Gamma \to \Delta}\ (\wedge \to)_1 \qquad \frac{B, \Gamma \to \Delta}{A \wedge B, \Gamma \to \Delta}\ (\wedge \to)_2$$

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

$$\frac{\Gamma \to \Delta, A}{\Gamma \to \Delta, A \vee B}\ (\to \vee)_1 \qquad \frac{\Gamma \to \Delta, B}{\Gamma \to \Delta, A \vee B}\ (\to \vee)_2$$

$$\frac{\Gamma \to \Delta, A \quad B, \Pi \to \Sigma}{A \supset B, \Gamma, \Pi \to \Delta, \Sigma}\ (\supset \to) \qquad \frac{\Gamma, A \to \Delta, B}{\Gamma \to \Delta, A \supset B}\ (\to \supset)$$

$$\frac{\Gamma \to \Delta, A}{\neg A, \Gamma \to \Delta}\ (\neg \to) \qquad \frac{A, \Gamma \to \Delta}{\Gamma \to \Delta, \neg A}\ (\to \neg)$$

Figure 2.2: The sequent system **LK**

Theorem 2.4.1 (cut elimination theorem for LK) *The system* **LK** *has the cut elimination property. In fact, every proof in* **LK** *can be translated, without changing the end sequent, into a cut–free one.*

As a corollary of Theorem 2.4.1, the following can be shown by checking all inference rules except cut rule (*cut*).

Corollary 2.4.2 (subformula property) *In any cut–free proof of a sequent S in* **LK**, *only subformulas of formulas in S appear.*

2.4.1 Wang's system

In 1963, Wang introduced alternative system for classical logic. Wang's system is given in Figure 2.3. Here, each side of sequent consists of a set of formulas. By this, we do not need exchange rules. Weakening rules are incorporated into initial sequents. Also, contraction rules are incorporated implicitly into logical rules. Thus, Wang's system has no structural rules. As we can see, each rule decomposes its principal formula into simper formulas. That is why we can say that Wang's system is suitable for implementing proof–search procedure.

initial sequent

$$\Gamma, p \to p, \Delta$$

inference rules

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

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

$$\frac{\Gamma \to \Delta, A \quad \Gamma, B \to \Delta}{\Gamma, A \supset B \to \Delta}\ (\supset \to) \qquad \frac{\Gamma, A \to \Delta, B}{\Gamma \to \Delta, A \supset B}\ (\to \supset)$$

$$\frac{\Gamma \to \Delta, A}{\Gamma, \neg A \to \Delta}\ (\neg \to) \qquad \frac{\Gamma, A \to \Delta}{\Gamma \to \Delta, \neg A}\ (\to \neg)$$

Figure 2.3: Wang's system

Proposition 2.4.3 *Each rule of Wang's system is invertible.*

Proof) Suppose that $\Gamma \to \Delta$ which is (an instance of) the lower sequent of (r) in Wang's system, where (r) is any of the static rules. We have to prove that there are proofs of corresponding (instance of) the upper sequents of (r) in Wang's system. By induction on the height of the given proof for $\Gamma \to \Delta$.

Base Case The base case for the induction step is when the height of the given proof for $\Gamma \to \Delta$ is 1, that is there is some propositional variable p such that $p \in \Gamma$ and $p \in \Delta$. The corresponding upper sequents of (r) must also contain p in both sides of them since p can not be the principal formula of (r). Therefore, there are proofs of instances of upper sequents.

Induction Step The induction hypothesis is that lemma holds for all proofs of height less that n. Suppose that the given proof of $\Gamma \to \Delta$ is of the height n. We show only the case $(\wedge \to)$, as other cases can be proved similarly.

When $\Gamma \to \Delta$ is (an instance of) the lower sequent of $(\wedge \to)$, $\Gamma \to \Delta$ is of the form $\Gamma', A \wedge B \to \Delta$. We will provide a proof of the upper sequent $\Gamma', A, B \to \Delta$. Consider the application of (r') which is the lowest application in the given proof of $\Gamma', A \wedge B \to \Delta$.

1. If $A \wedge B$ is not the principal formula C of (r'), the each of upper sequent(s) of (r') is of the form $\Pi, A \wedge B \to \Sigma$, since C must be some formula from Γ' or Δ. Because there is a proof of $\Gamma', A \wedge B \to \Delta$, there must exist a proof of each $\Pi, A \wedge B \to \Sigma$ of the height less than n. Then, by the induction hypothesis, there is a proof if the height less than n for each $\Pi, A, B \to \Sigma \ \cdots (1)$.

   If we now start a separate proof of $\Gamma', A, B \to \Delta$ and use (r') with the same principal formula $C \in \Gamma' \cup \Delta$, we obtain $\Pi, A \wedge B \to \Sigma$ as the upper sequent(s) of (r'). By (1), there is a proof of $\Gamma', A, B \to \Delta$ as desired.

2. If $A \wedge B$ is the principal formula C of (r') then (r') is $(\wedge \to)$. Only one upper sequent of the form $\Gamma', A \wedge B \to \Delta$ and there is clearly a proof of it. This is the proof we have to show. □

In this thesis, tableau or sequent systems are used to give proof–search procedures. As long as we use each of them as such a tool, there seem to be no substantial difference between them. Actually, each of rules of the tableau systems can be obtained from a rule of the sequent systems by moving all formulas in the right hand side into the left hand side with $\neg$. For instance, $(\neg\wedge)$ of $\mathcal{C}\mathbf{PC}$ can be obtained from $(\to \wedge)$ of Wang's system as follows:

$$\frac{\Gamma \to \Delta, A \quad \Gamma \to \Delta, B}{\Gamma \to \Delta, A \wedge B}\,(\to \wedge) \quad \Rightarrow \quad \frac{\Gamma, \neg\Delta, \neg A \to \quad \Gamma, \neg\Delta, \neg B \to}{\Gamma, \neg\Delta, \neg(A \wedge B) \to} \quad \Rightarrow \quad \frac{\Gamma, \neg(A \wedge B)}{\Gamma, \neg A \mid \Gamma, \neg B}\,(\neg\wedge)$$

where $\neg\Delta$ denotes the set $\{\neg A \mid A \in \Delta\}$. Recall that the direction of proof–search by tableau system proceeds top down, on the other hand, that by sequent system does bottom up.

2.5 Proof–search procedure based on tableau systems and sequent systems

We give proof–search procedures based on tableau system and sequent system. To do that, we are required to show soundness, completeness of them and also termination of proof–search procedures determined by them.

- Soundness means that our proof–search procedures gives us only correct proofs.
- Completeness means that our proof–search procedures gives us all correct proofs.
- Termination means that our proof–search procedures terminates for all formulas.

2.6 Construction of the finite model property

It is quite useful to obtain a finite model in which a given unprovable formula is false. A logic $\mathbf{L}$ has *the finite model property* if the following condition is satisfied,

$$\text{if } A \notin \mathbf{L}, \text{ then there is a finite } \mathbf{L}\text{–model such that } \not\models A.$$

A logic $\mathbf{L}$ is *finitely axiomatizable* if $\mathbf{L}$ is defined by adding finite axiom schemes to the logic $\mathbf{K}$. If there exists a finite procedure which decides whether a given formula is provable in $\mathbf{L}$ or not, $\mathbf{L}$ is said to be *decidable*, and such a procedure is called *decision procedure*. By the Harrop's theorem (see [8] for the details),

if a finitely axiomatizable logic has the finite model property, then it is decidable.

Therefore, we can say that the finite model property of finite axiomatizable logic $\mathbf{L}$ leads to the decidability of a logic $\mathbf{L}$. In Chapter 3, Chapter 4 and Chapter 6, we will show a way of giving counter–models. That brings us the finite model property, completeness and decidability for $\mathbf{S4}, \mathbf{KB}, \mathbf{K4B}$ and $\mathbf{K_t}$ simultaneously.

Chapter 3

Proof–search procedure for S4 based on tableau system

We will give an proof–search procedure for **S4** based on the tableau system introduced in this chapter.

3.1 Tableau system for S4

In this section, we will discuss modal tableau systems for **S4**. Usually the modal tableau systems for **S4** has the following rules called (T) and $(S4)$:

$$\frac{\Gamma, \Box A}{\Gamma, \Box A, A}\ (T) \qquad \frac{\Gamma, \Box\Delta, \neg\Box A}{\Box\Delta, \neg A}\ (S4)$$

In this application of (T), the denominator $\Gamma, \Box A, A$ is not always simpler than the numerator $\Gamma, \Box A$. Therefore, (T) may be applicable once again for this $\Box A$. Moreover, alternating applications of (T) and $(S4)$ may cause redundant repeated applications. It arises from that $\Box\Delta$ in numerator of $(S4)$ remains in its denominator. Any loop–checking for avoiding such redundant repeated applications of (T) and $(S4)$ would obviously cause a big inefficiency.

The decision procedure for **S4** have been proposed [13, 14, 11, 23]. Strictly speaking, they give proof–search procedures. However, the procedure of [13, 14] holds only for formulas in a certain normal form. That is why we are interested in Heuerding, Seyfried and Zimmermann's system [11] and Mouri's one [23]. In order to store the necessary information, the notion called histories was introduced in [11]. In [23], an auxiliary modal operator [22] and histories have been applied to sequent systems. The procedure in [23] is not only a proof–search procedure but also a procedure of construction of counter–models.

In our work, by introducing an auxiliary modal operator and histories to the ordinary modal tableau system for **S4**, we give a proof–search procedure for **S4**, which returns not only a closed tableau as its output when a given formula is a theorem of **S4**, but also gives

us a counter–model for it when it is not the case. Our work is the continuation of Mouri's work in [23]. We show completeness by giving a way to construct of counter–models via model graphs introduced later. It is easily seen that our proof of completeness is simpler than Mouri's proof of completeness.

3.2 Loop–checking

There are two kinds of loops that will occur in proofs of the tableau systems for **S4**. Examples are given as follows:

$$\dfrac{\dfrac{\dfrac{\Gamma, \Box A}{\Gamma, \Box A, A}\,(T)}{\Gamma, \Box A, A}\,(T)}{\vdots}\,(T) \qquad \dfrac{\dfrac{\dfrac{\Box\neg\Box A, \neg\Box A, \neg A}{\Box\neg\Box A, \neg A}\,(S4)}{\Box\neg\Box A, \neg\Box A, \neg A}\,(T)}{\vdots}\,(S4)$$

Usually, loop–checking cause a big inefficiency in the proof–search, for, in each time when a new formula set is generated, we have to check whether this is generated already in a tableau or not. To do so, we need to compare it with every formula sets generated so far. To avoid this kind of loop checking, we will introduce the following two techniques:

1. In the left example above, a loop is caused by redundant repeated applications of (T). In fact, a single application of (T) suffices for each formula of the form $\Box A$. In order to avoid this kind of loop, we introduce an auxiliary modal operator $\blacksquare$, which has the same meaning as $\Box$ but plays a different syntactical role. That is, $\blacksquare A$ means that the rule (T) is already applied to this $\Box A$.

2. While, in the right example, a loop is caused by an alternate application of (T) and $(S4)$. We can see that a $\Box$–formula occurs $\Box\neg\Box A$ in both denominator and numerator of both $(S4)$ and (T). To avoid loops of this kind, we introduce tableaux with histories, which will be defined at the beginning of next section. Once $\Box$–formula appear, they never disappear by any application of $(S4)$ and (T). In addition, a formula of the form $\neg\Box A$ is fixed in every application of $(S4)$. That is, as long as new $\Box$–formulas never appear, we do not need to apply $(S4)$ by fixing same formula of the form $\neg\Box A$. This is an intuitive meaning of histories.

3.3 A modal tableau system for S4

Our tableau rules are different from the standard tableau rules only in the following point. Each of numerators and denominators also carries a pair of sets of formulas $\langle\Box\Pi|\neg\Box\Sigma\rangle$. Any pair $\langle\Box\Pi|\neg\Box\Sigma\rangle$ is called a *history*. A $\mathcal{C}\mathbf{S4}$+–tableau for a finite set with history $\Gamma\langle\Box\Pi|\neg\Box\Sigma\rangle$ is a finite tree with the root $\Gamma\langle\Box\Pi|\neg\Box\Sigma\rangle$ each of whose nodes carries a finite

set of formulas and a history. A tableau is constructed by repeated applications of tableau rules.

The tableau rules of $\mathcal{CS}\mathbf{4}+$ are given in Figure 3.1. In $(S4+)_s$ and $(S4+)_t$ we suppose that $\{p_1, \cdots, p_n\}$ and $\{\neg q_1, \cdots, \neg q_m\}$ do not contain any complementary pair. We call each of $(\wedge)$, $(\vee)$, $(\supset)$, $(\neg\wedge)$, $(\neg\vee)$, $(\neg\supset)$, $(\neg)$ and $(T+)$ a *static rule*, and each of $(S4+)_s$ and $(S4+)_t$ a *transitional rule*. The upper sequent of $(S4+)_s$ and $(S4+)_t$ is taken 'or'–branch, though it should be understand as 'and'–branch in other rules. It means that if one of the upper sequent of $(S4+)_s$ and $(S4+)_t$ is provable then the lower sequent of it is provable. In order to emphasize 'or'–branch, double lines are used in $(S4+)_s$ and $(S4+)_t$. The idea of using 'or'–branch was introduced in Pinto and Dyckhoff [29].

Note that histories change only by the application of transitional rules. In proof–search of $\mathcal{CS}\mathbf{4}+$, we have only to check history only in each application of $(S4+)_t$ and $(S4+)_s$. In other words, $\mathcal{CS}\mathbf{4}+$ cuts down the number of loop–checking.

Theorem 3.3.1 *The modal tableau system* $\mathcal{CS}\mathbf{4}+$ *has the analytical superformula property. For a given formula set* Γ*, we can put* $\Gamma^{\mathcal{CS}\mathbf{4}+*} = Sub(\Gamma) \cup \{\neg A | A \in Sub(\Gamma)\} \cup \{\blacksquare A | \Box A \in Sub(\Gamma)\} \cup \{f\}$.

Lemma 3.3.2 *There are only a finite number of* $\mathcal{CS}\mathbf{4}+$*–tableaux for a finite set* Γ.

Proof) Since $\mathcal{CS}\mathbf{4}+$ has the analytical superformula property, it is clear. □

Now, to see whether $\Gamma \models_{\mathbf{S4}} A$ or not, we apply tableau rules to $(\Gamma \cup \{\neg A\})\langle\epsilon|\epsilon\rangle$. As we see, the tableau procedure is a refutation procedure. From the form of $(S4+)_t$ or $(S4+)_s$, it is easily seen that we can see that we can apply $(S4+)_t$ or $(S4+)_s$ only after static rules are applied as many time as possible. In fact, as is discussed in Section 5.5, to show the termination of proof–search of $\mathcal{CS}\mathbf{4}+$, it is essential in which order rules of $\mathcal{CS}\mathbf{4}+$ are applied. By checking histories in every application of $(S4+)_s$ and $(S4+)_t$, we can see whether same formula set appear so far or not in a tableau. In other words, histories tells us whether a loop is caused or not, hence we can avoid any loop. We note that same formula set with history never appear on each branch from the root to any node in every $\mathcal{CS}\mathbf{4}+$–tableau for any given finite set Γ.

Example 1 The left tableau is a closed $\mathcal{CS}\mathbf{4}+$–tableau for $\neg(\Box p \supset \Box\Box p)$, while the right one is an open $\mathcal{CS}\mathbf{4}+$–tableau for $\neg\neg\Box A$, where A denotes $\neg(\Box p \vee \Box\neg p)$.

In the right tableau below, $\Box A, \neg\Box p$ and $\neg\Box\neg p$ are included in the lowest history. That is, since neither $(S4+)_t$ nor $(S4+)_s$ is applicable, proof–search stops. Even if we apply $(S4+)_t$ or $(S4+)_s$ by fixing any of $\neg\Box p$ and $\neg\Box\neg p$, formula sets generated so far appear again.

$$\frac{\Gamma, p, \neg p\langle\Box\Pi|\neg\Box\Sigma\rangle}{f}\ (f) \qquad \frac{\Gamma, A \wedge B\langle\Box\Pi|\neg\Box\Sigma\rangle}{\Gamma, A, B\langle\Box\Pi|\neg\Box\Sigma\rangle}\ (\wedge)$$

$$\frac{\Gamma, A \vee B\langle\Box\Pi|\neg\Box\Sigma\rangle}{\Gamma, A\langle\Box\Pi|\neg\Box\Sigma\rangle \mid \Gamma, B\langle\Box\Pi|\neg\Box\Sigma\rangle}\ (\vee) \qquad \frac{\Gamma, A \supset B\langle\Box\Pi|\neg\Box\Sigma\rangle}{\Gamma, \neg A\langle\Box\Pi|\neg\Box\Sigma\rangle \mid \Gamma, B\langle\Box\Pi|\neg\Box\Sigma\rangle}\ (\supset)$$

$$\frac{\Gamma, \neg(A \wedge B)\langle\Box\Pi|\neg\Box\Sigma\rangle}{\Gamma, \neg A\langle\Box\Pi|\neg\Box\Sigma\rangle \mid \Gamma, \neg B\langle\Box\Pi|\neg\Box\Sigma\rangle}\ (\neg\wedge) \qquad \frac{\Gamma, \neg(A \vee B)\langle\Box\Pi|\neg\Box\Sigma\rangle}{\Gamma, \neg A, \neg B\langle\Box\Pi|\neg\Box\Sigma\rangle}\ (\neg\vee)$$

$$\frac{\Gamma, \neg(A \supset B)\langle\Box\Pi|\neg\Box\Sigma\rangle}{\Gamma, A, \neg B\langle\Box\Pi|\neg\Box\Sigma\rangle}\ (\neg\supset) \qquad \frac{\Gamma, \neg\neg A\langle\Box\Pi|\neg\Box\Sigma\rangle}{\Gamma, A\langle\Box\Pi|\neg\Box\Sigma\rangle}\ (\neg\neg)$$

$$\frac{\Gamma, \Box A\langle\Box\Pi|\neg\Box\Sigma\rangle}{\Gamma, \blacksquare A, A\langle\Box\Pi|\neg\Box\Sigma\rangle}\ (T+)$$

$$\frac{\blacksquare\Gamma, \neg\Box A_1, \cdots, \neg\Box A_l, \neg\Box\Delta, p_1, \cdots, p_m, \neg q_1, \cdots, \neg q_n\langle\Box\Gamma|\neg\Box\Sigma\rangle}{\Box\Gamma, \neg A_1\langle\Box\Gamma|\neg\Box\Sigma, \neg\Box\Theta\rangle \mid \cdots \mid \Box\Gamma, \neg A_l\langle\Box\Gamma|\neg\Box\Sigma, \neg\Box\Theta\rangle}\ (S4+)_s$$

$$(\neg\Box\Theta \equiv \neg\Box A_1, \cdots, \neg\Box A_l,\ \neg\Box\Theta \cap \neg\Box\Sigma = \emptyset,\ \neg\Box\Delta \subseteq \neg\Box\Sigma)$$

$$\frac{\blacksquare\Gamma, \neg\Box A_1, \cdots, \neg\Box A_l, p_1, \cdots, p_m, \neg q_1, \cdots, \neg q_n\langle\Box\Pi|\neg\Box\Sigma\rangle}{\Box\Gamma, \neg A_1\langle\Box\Gamma|\neg\Box\Theta\rangle \mid \cdots \mid \Box\Gamma, \neg A_l\langle\Box\Gamma|\neg\Box\Theta\rangle}\ (S4+)_t$$

$$(\neg\Box\Theta \equiv \neg\Box A_1, \cdots, \neg\Box A_l,\ \Box\Pi \subsetneq \Box\Gamma)$$

Figure 3.1: Tableau system $\mathcal{C}\mathbf{S4}+$

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\neg(\Box p \supset \Box\Box p)\ \langle\epsilon|\epsilon\rangle}{\Box p, \neg\Box\Box p\ \langle\epsilon|\epsilon\rangle}}{\blacksquare p, p, \neg\Box\Box p\ \langle\epsilon|\epsilon\rangle}}{\Box p, \neg\Box p\ \langle\Box p|\neg\Box\Box p\rangle}}{\blacksquare p, p, \neg\Box p\ \langle\Box p|\neg\Box\Box p\rangle}}{\Box p, \neg p\ \langle\Box p|\neg\Box p, \neg\Box\Box p\rangle}}{\blacksquare p, p, \neg p\ \langle\Box p|\neg\Box p, \neg\Box\Box p\rangle}}{f}$$

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\neg\neg\Box A\ \langle\epsilon|\epsilon\rangle}{\Box A\ \langle\epsilon|\epsilon\rangle}}{\blacksquare A, A\ \langle\epsilon|\epsilon\rangle}}{\blacksquare A, \neg\Box p, \neg\Box\neg p\ \langle\epsilon|\epsilon\rangle}}{\Box A, \neg p\ \langle\Box A|\neg\Box p\rangle}}{\blacksquare A, A, \neg p\ \langle\Box A|\neg\Box p\rangle}}{\blacksquare A, \neg\Box p, \neg\Box\neg p, \neg p\ \langle\Box A|\neg\Box p\rangle}}{\Box A, \neg\neg p\ \langle\Box A|\neg\Box\neg p, \neg\Box p\rangle}}{\Box A, p\ \langle\Box A|\neg\Box\neg p, \neg\Box p\rangle}}{\blacksquare A, A, p\ \langle\Box A|\neg\Box\neg p, \neg\Box p\rangle}}{\blacksquare A, \neg\Box p, \neg\Box\neg p, p\ \langle\Box A|\neg\Box\neg p, \neg\Box p\rangle}$$

3.4 Completeness theorem of $\mathcal{C}$S4+

In this section, we show soundness and completeness of $\mathcal{C}$**S4**+. We say that $\mathcal{C}$**S4**+ is *sound with respect to* **S4***–frames* if, for any Γ and A, there is a closed $\mathcal{C}$**S4**+–tableau for $\Gamma \cup \{\neg A\}$ then any **S4**–model that makes Γ true at any possible world w must make A true at w. We say that $\mathcal{C}$**S4**+ is *complete with respect to* **S4***–frames* if, for any Γ and A, every **S4**–model that makes Γ true at any possible world w also makes A true at w, then some $\mathcal{C}$**S4**+–tableau for $\Gamma \cup \{\neg A\}$ is closed. We have to define the semantics of $\blacksquare A$ in Kripke frames. For a given Kripke model $(W, R, \models)$,

$$w \models \blacksquare A \quad \textit{iff} \quad w \models \Box A.$$

This means that $\blacksquare$ has the same semantics as $\Box$.

Theorem 3.4.1 (soundness) *The modal tableau system* $\mathcal{C}$**S4**+ *is sound with respect to* **S4***–frames.*

Proof) The soundness of $\mathcal{C}$**S4**+ is equivalent to the condition that if there is a closed $\mathcal{C}$**S4**+–tableau for $\Gamma \cup \{\neg A\}$ then $\Gamma \cup \{\neg A\}$ is **S4**–unsatisfiable. But, the latter is shown by using induction on the length of $\mathcal{C}$**S4**+–tableau. □

Next, we show the completeness of $\mathcal{C}$**S4**+. Taking the contraposition, it suffices to that if there is no closed $\mathcal{C}$**S4**+–tableau for $\Gamma \cup \{\neg A\}$ then there is an **S4**–model for $\Gamma \cup \{\neg A\}$. To show completeness, we need some machinery.

Lemma 3.4.2 *Each static rule of* $\mathcal{C}$**S4**+ *is invertible in* $\mathcal{C}$**S4**+.

Proof) Similarly to the proof in [10]. □

Definition 3.4.3 *A finite set* Γ *is* **closed with respect to a tableau rule** *if whenever* (*an instance of*) *the numerator of the rule is in* Γ*, so is* (*a corresponding instance of*) *at least one of the denominators of the rule.*

Definition 3.4.4 *A finite set* Γ *is* $\mathcal{CS}4+$**–saturated** *if it is* $\mathcal{CS}4+$*–consistent and closed with respect to every static rule of* $\mathcal{CS}4+$*.*

Lemma 3.4.5 *For a each finite* $\mathcal{CS}4+$*–consistent* Γ *there is an effective procedure for constructing a finite* $\mathcal{CS}4+$*–saturated* Γ^s *with* $\Gamma \subseteq \Gamma^s \subseteq \Gamma^{\mathcal{CS}4+*}$*.*

Proof) Similarly to the proof in [10]. □

As far as there is no confusion, we call this process just *saturation*. Such sets generated by saturation are called downward saturated sets in [12].

Definition 3.4.6 (model graph) *Let* W *be a nonempty set and* R *be a binary relation on* W*, that is* $R \subseteq WtimesW$*. Then, an* **S4–model graph** *for a finite set of formulas* Γ *is a finite* **S4***–frame* (W, R) *such that all* $w\langle\Box\Pi|\neg\Box\Sigma\rangle \in W$ *and every* w *is* $\mathcal{CS}4+$*–saturated set with* $w \subseteq \Gamma^{\mathcal{CS}4+*}$ *and*

1. $\Gamma \subseteq w_0$ *for some* $w_0\langle\Box\Pi_0|\neg\Box\Sigma_0\rangle \in W$*,*
2. *if* $\neg\Box A \in w_i$ *then there exists some* $w_j\langle\Box\Pi_j|\neg\Box\Sigma_j\rangle \in W$ *with* $(w_i\langle\Box\Pi_i|\neg\Box\Sigma_i\rangle)R(w_j\langle\Box\Pi_j|\neg\Box\Sigma_j\rangle)$ *and* $\neg A \in w_j$*,*
3. *if* $(w_i\langle\Box\Pi_i|\neg\Box\Sigma_i\rangle)R(w_j\langle\Box\Pi_j|\neg\Box\Sigma_j\rangle)$ *and* $\blacksquare A \in w_i$ *then* $A \in w_j$*.*

As long as there is no confusion, we write down **S4**–model graph just as model graph. It remains to construct a model for Γ from model graph for Γ. By the following lemma, we can obtain a model immediately from a model graph.

Lemma 3.4.7 (satisfiability lemma) *If* (W, R) *is a model graph for a finite set* Γ *then there exists an model for* Γ*.*

Proof) Similarly to the proof in [10]. Let Γ be $\mathcal{CS}4+$–consistent set and $w\langle\Box\Pi|\neg\Box\Sigma\rangle \in W$. For any propositional variable p, we define a valuation as follows: $w\langle\Box\Pi|\neg\Box\Sigma\rangle \models p$ *iff* $p \in w$. Then, we show

1. $A \in w$ implies $w\langle\Box\Pi|\neg\Box\Sigma\rangle \models A$,
2. $\neg A \in w$ implies $w\langle\Box\Pi|\neg\Box\Sigma\rangle \not\models A$

by using simultaneous induction. □

Our completeness proof is based on Goré's in [10]. The proof is based on Goré [10]. Recall the definition of the completeness of $\mathcal{CS}4+$. We prove the contraposition. That is, we assume that there is no $\mathcal{CS}4+$–tableau for $\Gamma \cup \{\neg A\}$. Then, there are open tableaux for $\Gamma \cup \{\neg A\}$. We pick and choose formula sets with history from each of them. The formula set with history is used as possible worlds to construct an **S4**–model M for $\Gamma \cup \{\neg A\}$.

The model M is deliberately constructed so as to contain a possible world w_0 such that $w_0 \models \Gamma \cup \{\neg A\}$. Hence we demonstrate by construction that $\Gamma \not\models_{\mathbf{S4}} A$. That is, proving completeness boils down to the following: if there is no closed $\mathcal{C}\mathbf{S4}+$–tableau for $\Gamma \cup \{\neg A\}$ then there is an $\mathbf{S4}$–model for $\Gamma \cup \{\neg A\}$ on an $\mathbf{S4}$–frame (W, R).

As we have stated above, we shall associate sets of formulas with possible worlds. In our completeness proof of $\mathcal{C}\mathbf{S4}+$, the difference with Goré's method is that we adopt a set of formulas with history $\Pi\langle\Box\Sigma|\neg\Box\Omega\rangle$ as a possible world. As we have stated previously, proving completeness boils down to the following: if there is no closed $\mathcal{C}\mathbf{S4}+$–tableau for $\Gamma \cup \{\neg A\}$ then there is an $\mathbf{S4}$–model for $\Gamma \cup \{\neg A\}$ on an $\mathbf{S4}$–frame. Now, we give the sketch of completeness proof of $\mathcal{C}\mathbf{S4}+$. It goes as follows:

1. Suppose $\Gamma \not\vdash_{\mathcal{C}\mathbf{S4}+} A$.
2. Create a $\mathcal{C}\mathbf{S4}+$–saturated $(\Gamma \cup \{\neg A\})^s$ from $\Gamma \cup \{\neg A\}$ (By Lemma 3.4.5) and put $w_0\langle\Box\Pi_0|\neg\Box\Sigma_0\rangle = (\Gamma \cup \{\neg A\})^s\langle\epsilon|\epsilon\rangle$.
3. Construct a finite model graph for $\Gamma \cup \{\neg A\}$ (By Theorem 3.4.8).
4. Construct an $\mathbf{S4}$–model for $\Gamma \cup \{\neg A\}$ (By Lemma 3.4.7).
5. Thus, $\Gamma \cup \{\neg A\}$ is $\mathbf{S4}$–satisfiable, hence $\Gamma \not\models_{\mathbf{S4}} A$.

The following property holds for histories.

Remark 1 Let Γ be of the form $\{p_1, \cdots, p_n, \neg q_1, \cdots, \neg q_m, \neg\Box A_1, \cdots, \neg\Box A_l\}$ and $(\Gamma \cup \blacksquare\Delta)\langle\Box\Pi|\neg\Box\Sigma\rangle$ be any node of a $\mathcal{C}\mathbf{S4}+$–tableaux. Then, $\Box\Pi \subseteq \Box\Delta$ holds.

Proof) Clear from how to update histories by applications of rules of $\mathcal{C}\mathbf{S4}+$. □

Theorem 3.4.8 (completeness) *If Γ is a finite set of formulas and is $\mathcal{C}\mathbf{S4}+$–consistent then there is an $\mathbf{S4}$–model for Γ on a finite $\mathbf{S4}$–frame.*

Proof) We give a way of constructing a finite model graph (W, R) for Γ. The first step is to create a $\mathcal{C}\mathbf{S4}+$–saturated Γ^s from Γ with $\Gamma \subseteq \Gamma^s \subseteq \Gamma^{\mathcal{C}\mathbf{S4}+*}$. We can create Γ^s by Lemma 3.4.5. Put $w_0\langle\Box\Pi_0|\neg\Box\Sigma_0\rangle = \Gamma^s\langle\epsilon|\epsilon\rangle$, which is a possible world of W, and add $w_0\langle\Box\Pi_0|\neg\Box\Sigma_0\rangle$ to W. We construct a model graph for Γ based on the node $w_0\langle\Box\Pi_0|\neg\Box\Sigma_0\rangle$ as follows:

1. If no formula of the form $\neg\Box A$ occurs in w_0 then $(\{w_0\langle\epsilon|\epsilon\rangle\}, \{(w_0\langle\epsilon|\epsilon\rangle, w_0\langle\epsilon|\epsilon\rangle)\})$ is the desired model graph for Γ since this is a $\mathbf{S4}$–frame and satisfies conditions of model graphs.

2. Otherwise, we give show how to define the immediate successors of any $w_k\langle\Box\Pi_k|\neg\Box\Sigma_k\rangle \in W$. Do the following as many times as possible:

   If no formula of the form $\neg\Box A$ occurs in w_k then then we do not create any successor of w_k. Otherwise, let $\Theta_k = \{A_1, A_2, \cdots, A_l\}$ be the set of all formulas such that $\neg\Box B \in w_k$ and $\neg B \notin w_k$. Since w_k is $\mathcal{CS4}+$–consistent, $\blacksquare\Delta_k \cup \neg\Box\Theta_k \subseteq w_k$ also must be $\mathcal{CS4}+$–consistent. We have two cases by Remark 1.

   (a) If $\Box\Pi_k \subsetneq \Box\Delta_k$, apply $(S4+)_t$ to $(\blacksquare\Delta_k \cup \neg\Box\Theta_k)\langle\Box\Pi_k|\neg\Box\Sigma_k\rangle$ with each $\neg\Box A_i$ chosen, and we can obtain $(\Box\Delta_k \cup \{\neg A_i\})\langle\Box\Delta_k|\neg\Box\Theta_k\rangle$. Put

   $$w_{ki}\langle\Box\Pi_{ki}|\neg\Box\Sigma_{ki}\rangle = (\Box\Delta_k \cup \{\neg A_i\})^s\langle\Box\Delta_k|\neg\Box\Theta_k\rangle.$$

   Then, add $w_{ki}\langle\Box\Pi_{ki}|\neg\Box\Sigma_{ki}\rangle$ to W and put

   $$(w_k\langle\Box\Pi_k|\neg\Box\Sigma_k\rangle)R(w_{ki}\langle\Box\Pi_{ki}|\neg\Box\Sigma_{ki}\rangle).$$

   Here, let $\Box\Lambda_{ki} = \{\Box B|\blacksquare B \in w_{ki}\}$. In addition, if $\Box\Delta_k = \Box\Lambda_{ki}$, put

   $$(w_{ki}\langle\Box\Pi_{ki}|\neg\Box\Sigma_{ki}\rangle)R(w_k\langle\Box\Pi_k|\neg\Box\Sigma_k\rangle).$$

   (b) If $\Box\Pi_k = \Box\Delta_k$, we can write $\blacksquare\Delta_k \cup \neg\Box\Theta_k$ as $\blacksquare\Delta_k \cup \neg\Box\Gamma_k \cup \neg\Box\Omega_k$, where $\neg\Box\Theta_k = \neg\Box\Gamma_k \cup \neg\Box\Omega_k$, $\neg\Box\Gamma_k \cap \neg\Box\Sigma_k = \emptyset$ and $\neg\Box\Omega_k \subseteq \neg\Box\Sigma_k$. We have two cases.

      i. If $\neg\Box\Gamma_k \neq \emptyset$, apply $(S4+)_s$ to $(\blacksquare\Delta_k \cup \neg\Box\Gamma_k \cup \neg\Box\Omega_k)\langle\Box\Pi_k|\neg\Box\Sigma_k\rangle$, and we can obtain $(\Box\Delta_k \cup \{\neg A_i\})\langle\Box\Pi_k|\neg\Box\Sigma_k, \neg\Box\Gamma_k\rangle$. Similarly to the previous case, below.

      ii. Otherwise, do nothing. In this case, $w_k\langle\Box\Pi_k|\neg\Box\Sigma_k\rangle$ is an end node of the model graph.

Next, we show that this construction terminates. Since $\mathcal{CS4}+$ has the analytical superformula property, we can express the set of all formulas in any $\mathcal{CS4}+$–tableau for Γ as $\Gamma^{\mathcal{CS4}+*}$. For each node $w\langle\Box\Pi|\neg\Box\Sigma\rangle$, we define the degree $ndeg(w\langle\Box\Pi|\neg\Box\Sigma\rangle) \in \mathbf{N}^2$ as follows:

$$ndeg(w\langle\Box\Pi|\neg\Box\Sigma\rangle) = (|\Gamma^{\mathcal{CS4}+*}| - |\Box\Pi|,\ |\Gamma^{\mathcal{CS4}+*}| - |\neg\Box\Sigma|).$$

We note that $|\Gamma^{\mathcal{CS4}+*}| - |\Box\Pi|, |\Gamma^{\mathcal{CS4}+*}| - |\neg\Box\Sigma| \geq 1$. We define a lexicographic order $\ll$ over pairs of positive integers as follows:

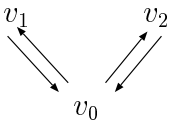
$$(x_1, x_2) \ll (y_1, y_2) \quad \textit{iff} \quad x_1 < y_1 \text{ or } (x_1 = y_1 \text{ and } x_2 < y_2).$$

Every successor of a node $w\langle\Box\Pi|\neg\Box\Sigma\rangle$ is generated by an application of $(S4+)_t$ or $(S4+)_s$ after $\mathcal{CS4}+$–saturation. We can see $(S4+)_t$ strictly decreases $|\Gamma^{\mathcal{CS4}+*}|-|\Box\Pi|$. On the other hand, $(S4+)_s$ leaves $|\Gamma^{\mathcal{CS4}+*}|-|\Box\Pi|$ unchanged but strictly decreases $|\Gamma^{\mathcal{CS4}+*}|-|\neg\Box\Sigma|$. We note that the $\mathcal{CS4}+$–saturation does not increase $ndeg(w\langle\Box\Pi|\neg\Box\Sigma\rangle)$. Hence, immediate successors are finite, that is this construction terminates.

Finally, the set W consisting of all the nodes created in this process and take R^* as the reflexive and transitive closure of R. The construction shows that (W, R^*) is a model graph for Γ. Thus, by Lemma 3.4.7, we can obtain an **S4**–model for Γ with the root $w_0\langle\epsilon|\epsilon\rangle$. □

Our completeness proof of $\mathcal{CS4}+$ actually gives a way of constructing a counter–models for **S4** when the proof–search fails. Therefore, we can say that the procedure determined by $\mathcal{CS4}+$ is not only a proof–search procedure for **S4** but also a procedure of construction of counter–models for **S4** when the proof–search fails. In addition, **S4**–models constructed by our completeness proof are always finite. It means that our completeness proof constructively shows that **S4** has the finite model property.

Example 2 Now we give a counter–model for $\neg\Box\neg(\Box p \vee \Box\neg p)$. The following figure is a counter–model of **S4** for $\neg\neg\Box\neg(\Box p \vee \Box\neg p)$, where A denotes $\neg(\Box p \vee \Box\neg p)$.

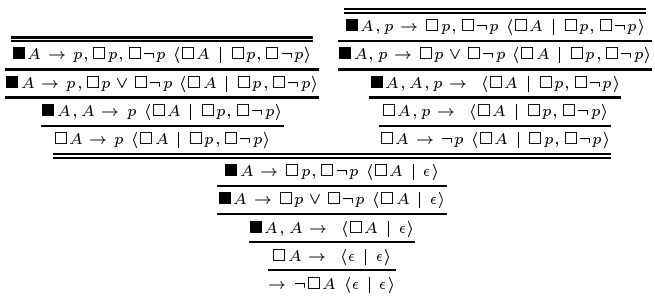


$v_0 = \{\neg\neg\Box A, \blacksquare A, \neg\Box p, \neg\Box\neg p\}\langle\epsilon|\epsilon\rangle$
$v_1 = \{\Box A, \blacksquare A, \neg\Box p, \neg\Box\neg p, \neg p\}\langle\Box A|\neg\Box p, \neg\Box\neg p\rangle$
$v_2 = \{\Box A, \blacksquare A, \neg\Box p, \neg\Box\neg p, \neg\neg p, p\}\langle\Box A|\neg\Box p, \neg\Box\neg p\rangle$
$v_2 \models p$

Figure 3.2: A counter–model for $\neg\Box\neg(\Box p \vee \Box\neg p)$

In the completeness proof, it is obvious that clusters are generated by constructing model graph and the clusters form a tree.

As a matter of fact, our proof of completeness tells us that it is possible to construct a counter–model directly from open tableaux. Mouri has given a construction of counter–model of **S4**, which is based on sequent system, directly from failed proof figure in [23]. The following is a failed proof figure for $\neg\Box A$ in Mouri's system:



Mouri's method goes as follows: each part between double lines denotes a possible world. If the number of formulas of the form $\Box B$ of the left hand side of the top sequent in the upper part u is more than that of the left hand side of the top sequent in the lower part w, put wRu. Otherwise, put wRu and uRw. Repeat this process. Finally, take the reflexive and transitive closure of R. In this case, we can obtain the same counter–model as one in Figure 3.2.

3.5 Conclusion

In this chapter, we gave a proof–search procedure determined by the tableau system $\mathcal{C}\mathbf{S4}+$, and termination of it was also shown. Since the proof–search procedure needs loop–checking only in each of applications of $(S4+)_s$ and $(S4+)_t$, we can say that $\mathcal{C}\mathbf{S4}+$ cuts down the number of loop–checking. This would contribute to technique for implementing of proof–search procedure for **S4**.

3.6 Note

Tableau system is used to formulate some logics like classical logic, intuitionistic logic, modal logics, substructural logics [28] and so on [9, 10]. Tableau became widely known by the book [32] of Smullyan. Smullyan used tableaux as the basis of a general treatment of classical logic. Then, the method is applied to several modal logics by Fitting [4] and temporal logics by Rescher and Urquhart [30]. In [5], tableau systems for both normal and non–normal modal logics extensively discussed.

The author gave a tableau system for **KT** in [20]. It avoids loops generated by redundant applications of (T) by using an auxiliary modal operator. The technique of using an extra modal operator as a blocked version of $\Box$ has previously been used in the work by Hudelmaier [14].

The proof–search procedures for **S4** have been proposed in [13, 14, 11, 23]. In [11], Heuerding, Seyfried and Zimmermann have introduced histories. The procedure by Hudelmaier [13, 14] holds only for formulas in a certain normal form. In [23], Mouri gave not only a proof–search procedure for **S4** but also a way of construction of counter–model of **S4**, though it is hard to follow his proof of completeness.

Chapter 4

Proof–search procedures for KB and K4B based on sequent system

In this chapter, we will give sequent systems for **KB** and **K4B**. Then, we will show their completeness by giving a way of construction of counter–model as we gave in the previous chapter. This chapter is organized as follows. In Section 4.2, we will introduce a sequent system for **KB**, and will show its completeness via modal graph. In Section 4.3, similarly, we will introduce a sequent system for **K4B**, and will show its completeness via modal graph.

4.1 Sequent systems for KB and K4B

Throughout this section, Γ, Σ etc. denote (finite, possibly empty) sequence of formulas. Standard sequent systems for **KB** and **K4B** are obtained from **LK** by including the following rules in **LK**, $(\to \Box_B)$ in **LK** and $(\to \Box_{4BE})$ in **LK**, respectively:

$$\frac{\Gamma \to \Box\Theta, A}{\Box\Gamma \to \Theta, \Box A}\ (\to \Box)_B \qquad \frac{\Gamma, \Box\Gamma \to \Box\Theta, \Box\Omega, A}{\Box\Gamma \to \Box\Theta, \Omega, \Box A}\ (\to \Box)_{4BE}$$

The sequent system obtained from **LK** adding $(\to \Box_B)$ is called $\mathcal{G}$**KB**, and the sequent system obtained from **LK** adding $(\to \Box_{4BE})$ is called $\mathcal{G}$**K4B**. We note that in each of $\mathcal{G}$**KB** and $\mathcal{G}$**K4B** the cut elimination theorem does not hold. This means that we can not obtain the subformula property for these sequent systems from the cut elimination theorem. For example, a sequent $p \to \Box\neg\Box\neg p$ has a proof in the following:

$$\dfrac{\dfrac{\dfrac{\Box\neg p \to \Box\neg p}{\to \Box\neg p, \neg\Box\neg p}}{\to \neg p, \Box\neg\Box\neg p}\,(cut) \quad \dfrac{p \to p}{\neg p, p \to}}{p \to \Box\neg\Box\neg p}$$

But it is not provable without any application of cut rule, Therefore, proof–search procedures determined by $\mathcal{G}$**KB** and $\mathcal{G}$**K4B** would cause an inefficiency because there

are extremely many choices of cut formulas in proof–search. In [34], Takano succeeded to introduce sequent systems for **KB** and **K4B**, which have the subformula property. Takano's sequent systems for **KB** and **K4B**, called $\mathbf{KB}^*$ and $\mathbf{K4B}^*$ respectively, are obtained by adding the following restrictions to each of the rules $(cut), (\to \Box)_B$ and $(\to \Box)_{4BE}$:

$$\frac{\Gamma \to \Delta, A \quad A, \Pi \to \Sigma}{\Gamma, \Pi \to \Delta, \Sigma}\ (ac)$$
$$A \in Sub(\Gamma \cup \Pi \cup \Delta \cup \Sigma)$$

$$\frac{\Gamma \to \Box\Theta, A}{\Box\Gamma \to \Theta, \Box A}\ (\to \Box)_B$$
$$\Box\Theta \subseteq Sub(\Gamma \cup \{A\})$$

$$\frac{\Gamma, \Box\Gamma \to \Box\Theta, \Box\Omega, A}{\Box\Gamma \to \Box\Theta, \Omega, \Box A}\ (\to \Box)_{4BE}$$
$$\Box\Omega \subseteq Sub(\Box\Gamma \cup \Omega \cup \{A\})$$

We say that a cut rule is *acceptable* if the cut formula A is a subformula of a formula occurring in the lower sequent, that is $A \in Sub(\Gamma \cup \Pi \cup \Delta \cup \Sigma)$. Also, we say that $(\to \Box)_B$ and $(\to \Box)_{4BE}$ are *acceptable*, if $\Box\Theta \subseteq Sub(\Gamma \cup \{A\})$ and $\Box\Omega \subseteq Sub(\Box\Gamma \cup \Omega \cup \{A\})$, respectively.

Proposition 4.1.1 *Every proof of* $\mathcal{G}$**KB** *or* $\mathcal{G}$**K4B** *can be transformed, without changing the end sequent, into another proof such that every cut,* $(\to \Box)_B$ *and* $(\to \Box)_{4BE}$ *applied in it, is acceptable. Thus, every proof of* $\mathcal{G}$**KB** *or* $\mathcal{G}$**K4B** *can be transformed into Takano's sequent system* $\mathbf{KB}^*$ *or* $\mathbf{K4B}^*$.

It is easy to see that $\mathbf{KB}^*$ and $\mathbf{K4B}^*$ have the subformula property. Moreover, another sequent system for **K4B** is also given by adding the following rules to **LK** :

$$\frac{\Gamma, \Box\Gamma \to \Box\Delta, A}{\Box\Gamma \to \Box\Delta, \Box A}\ (\to \Box)_{K4B^*} \qquad \frac{A, \Gamma \to \Delta, \Box B}{\Box A, \Gamma \to \Delta, \Box B}\ (\Box \to)_{K4B^*}$$

This system is also due to Takano. In [15], Kobayashi restricted cut formula to a formula A such that $\Box A$ occurring in both sides of sequent of this sequent system for **K4B**, by introducing an alternative cut rule, which restricts cut formula more strictly. That contributes to techniques for reducing choices of cut formulas in proof–search. However, Takano's acceptable cut rule and Kobayashi's cut rule have a certain deficiency in applying them. In fact, cut rules can be applied repeatedly and redundantly, which will cause inefficient proof–searches. To avoid them and to make proof–search procedures more efficient, we are required to modify Takano's sequent system for **KB** and **K4B**. To construct counter–models, we use model graphs in proofs of completeness of our sequent systems for **KB** and **K4B**.

4.2 A sequent system for KB

Now, we will introduce our a sequent system $\mathcal{S}\mathbf{KB}$ for $\mathbf{KB}$, which is based on Takano's sequent system for $\mathbf{KB}$. (See [34].) In our sequent system $\mathcal{S}\mathbf{KB}$, the form of cut formulas is restricted more strictly than that of Takano's sequent system $\mathbf{KB}^*$. Actually, our sequent system $\mathcal{S}\mathbf{KB}$ gives us a proof–search procedure for $\mathbf{KB}$ which avoids redundant applications of the cut rule.

4.2.1 $\mathcal{S}$KB

In the following, we will introduce our sequent system for $\mathbf{KB}$, called $\mathcal{S}\mathbf{KB}$, which has auxiliary modal operators $\blacksquare$ and $\sharp$. Initial sequents of $\mathcal{S}\mathbf{KB}$ are sequents of the form $\Gamma, A \to A, \Delta$. Rules of inference of $\mathcal{S}\mathbf{KB}$ are given in Figure 4.1. Both sides of sequent are sets of formulas.

Here, $\blacksquare\Pi$ and $\sharp\Omega$ etc. denote the sets $\{\blacksquare A | A \in \Pi\}$ and $\{\sharp A | A \in \Omega\}$, respectively. In $(cut)_L$ and $(cut)_R$, we assume that the sets Λ and Ξ are disjoint, respectively. We call $(\Box)_B$ *transitional rule* and call others *static rules*. We note that $(\to \wedge)$, $(\vee \to)$ and $(\supset\to)$ have three upper sequents. The upper sequents of $(\Box)_B$ must be regarded as 'or'–branch, while in other rules those should be understood as 'and'–branch. Here, by 'or'–branch we mean that if one of upper sequents of $(\Box)_B$ is provable then the lower sequent of it is provable. In order to emphasize 'or'–branch, double lines are used in $(\Box)_B$ (similarly to $\mathcal{C}\mathbf{S4}+$ in Section 3.3). We note that cut formula is taken always from a $\Box$–formula in the lower sequent of $(cut)_L$ or $(cut)_R$. The subformula property in the strict sense does not hold in $\mathcal{S}\mathbf{KB}$, but we can see that for a given sequent $\Gamma \to \Delta$, any formula occurring in all proofs of $\Gamma \to \Delta$ is in $(\Gamma \cup \Delta)^{\mathcal{S}\mathbf{KB}*}$, where

$$(\Gamma \cup \Delta)^{\mathcal{S}\mathbf{KB}*} = Sub(\Gamma \cup \Delta) \ \cup \ \{\blacksquare A \mid \Box A \in Sub(\Gamma \cup \Delta)\} \ \cup \ \{\sharp A \mid \Box A \in Sub(\Gamma \cup \Delta)\}.$$

The proof–search procedure determined by $\mathcal{S}\mathbf{KB}$ proceeds 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 an application of a rule is meant to get upper sequents from a given lower sequent of this rule.

4.2.2 Model graphs for $\mathcal{S}$KB

In order to show completeness of $\mathcal{S}\mathbf{KB}$, we need to introduce some technical machinery as we defined model graphs for $\mathcal{C}\mathbf{S4}+$ in Section 3.4. We will define model graphs for sequent system $\mathcal{S}\mathbf{KB}$.

A rule (r) of $\mathcal{S}\mathbf{KB}$ is *invertible* in $\mathcal{S}\mathbf{KB}$ if there is a proof of (an instance of) the lower sequent of (r) in $\mathcal{S}\mathbf{KB}$ then there are proofs of (appropriate instances of) the upper sequents of (r) in $\mathcal{S}\mathbf{KB}$, respectively.

initial sequent

$$\Gamma, A \to A, \Delta$$

static rule

$$\frac{\Gamma, A, B \to \Delta}{\Gamma, A \wedge B \to \Delta}\ (\wedge \to) \qquad \frac{\Gamma \to \Delta, A, B}{\Gamma \to \Delta, A \vee B}\ (\to \vee) \qquad \frac{\Gamma, A \to \Delta, B}{\Gamma \to \Delta, A \supset B}\ (\to \supset)$$

$$\frac{\Gamma \to \Delta, A}{\Gamma, \neg A \to \Delta}\ (\neg \to) \qquad \frac{\Gamma, A \to \Delta}{\Gamma \to \Delta, \neg A}\ (\to \neg)$$

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

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

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

$$\frac{\Box\Gamma, \blacksquare A, \blacksquare\Pi, \Lambda \to \Box\Delta, \blacksquare\Sigma, \sharp\Omega, \sharp A, \Xi, A \quad A, \Box\Gamma, \blacksquare A, \blacksquare\Pi, \Lambda \to \Box\Delta, \blacksquare\Sigma, \sharp\Omega, \Xi}{\Box\Gamma, \Box A, \blacksquare\Pi, \Lambda \to \Box\Delta, \blacksquare\Sigma, \sharp\Omega, \Xi}\ (cut)_L$$

$$\frac{\Box\Gamma, \blacksquare\Pi, \Lambda \to \Box\Delta, \blacksquare A, \blacksquare\Sigma, \sharp\Omega, \sharp A, \Xi, A \quad A, \Box\Gamma, \blacksquare\Pi, \Lambda \to \Box\Delta, \blacksquare A, \blacksquare\Sigma, \sharp\Omega, \Xi}{\Box\Gamma, \blacksquare\Pi, \Lambda \to \Box\Delta, \Box A, \blacksquare\Sigma, \sharp\Omega, \Xi}\ (cut)_R$$

$$\Lambda \equiv p_1, \cdots, p_n,\ \Xi \equiv q_1, \cdots, q_m$$

transitional rule

$$\frac{\Gamma \to A_1, \Box\Omega_1 \qquad \cdots \qquad \Gamma \to A_m, \Box\Omega_m}{\blacksquare\Gamma, p_1, \cdots, p_n \to \blacksquare A_1, \cdots, \blacksquare A_m, \sharp\Omega, q_1, \cdots, q_l}\ (\Box)_B$$

$$\Omega_i \subseteq \Omega,\ \Box\Omega_i \subseteq Sub(\Gamma \cup \{A_i\}),\ \Box(\Omega - \Omega_i) \cap Sub(\Gamma \cup \{A_i\}) = \emptyset\ (1 \leq i \leq m)$$

Figure 4.1: Sequent system for $\mathbf{KB} : \mathcal{S}\mathbf{KB}$

Lemma 4.2.1 *Every static rule of* $\mathcal{S}$**KB** *is invertible in* $\mathcal{S}$**KB**.

Proof) Suppose that $\Gamma \to \Delta$ which is (an instance of) the lower sequent of (r) in $\mathcal{S}$**KB** where (r) is any of static rules. We will show that there are proofs of corresponding (instance of) the upper sequents of (r) in $\mathcal{S}$**KB**, by using induction on the height of the given proof for $\Gamma \to \Delta$.
Base Case Trivial.

Induction Step The induction hypothesis is that lemma holds for all proofs of the height less that n. Suppose that the height of a given proof of $\Gamma \to \Delta$ is n. For simplicity's sake, throughout this proof, we assume that $(cut)_L$ and $(cut)_R$ are applicable even if each lower sequent of them contain formulas which can be principal formulas of static rules other than $(cut)_L$ and $(cut)_R$.

1. When $\Gamma \to \Delta$ is (an instance of) the lower sequent of any of static rules except $(cut)_L$ and $(cut)_R$. Consider the application of (r') which is at the lowest application in the given proof of $\Gamma \to \Delta$. Here, we note that (r') is none of $(\Box)_B, (cut)_L$ and (cut_R) because the lower sequents of any of $(\Box)_B, (cut)_L$ and (cut_R) do not contain any formula which can be a principal formula of a static rule of $\mathcal{S}$**KB** other than $(cut)_L$ and $(cut)_R$. In this case, we can show like Proposition 2.4.3.

2. When $\Gamma \to \Delta$ is the lower sequent of $(cut)_L$, $\Gamma \to \Delta$ is of the form

$$\Box\Gamma', \Box A, \blacksquare\Sigma, \Lambda \to \Box\Delta', \blacksquare\Pi, \sharp\Omega, \Xi \quad \cdots \quad (1),$$

where Λ and Ξ denote sets of propositional variables. We will provide proofs of the following upper sequents:

$$\Box\Gamma', \blacksquare A, \blacksquare\Sigma, \Lambda \to \Box\Delta', \blacksquare\Pi, \sharp\Omega, \sharp A, \Xi, A \quad \cdots \quad (2)$$
$$A, \Box\Gamma', \blacksquare\Sigma, \blacksquare A, \Lambda \to \Box\Delta', \blacksquare\Pi, \sharp\Omega, \Xi \quad \cdots \quad (3)$$

Consider the application of (r') which is the lowest application in a given proof of (1). Here, we note that (r') must be either $(cut)_L$ or $(cut)_R$ because of the form (1).

   (a) If (r') is $(cut)_L$ with the cut formula A then the upper sequents are of the form (2) and (3). Since there is a proof of (1), there are clearly proofs of (2) and (3). These are proofs that we want.

   (b) If (r') is $(cut)_R$ with a cut formula B such that $\Box B$ occurs in the right hand side of (1) then (1) is of the form

$$\Box\Gamma', \Box A, \blacksquare\Sigma, \Lambda \to \Box\Delta'', \Box B, \blacksquare\Pi, \sharp\Omega, \Xi,$$

where $\Box\Delta' = \Box\Delta'' \cup \{\Box B\}$. The upper sequents are of the forms

$$\Box\Gamma', \Box A, \blacksquare\Sigma, \Lambda \to \Box\Delta'', \blacksquare B, \blacksquare\Pi, \sharp\Omega, \sharp B, \Xi, B \quad \cdots \quad (4)$$
$$B, \Box\Gamma', \Box A, \blacksquare\Sigma, \Lambda \to \Box\Delta'', \blacksquare B, \blacksquare\Pi, \sharp\Omega, \Xi \quad \cdots \quad (5)$$

Because there is a proof (1), there must exist proofs of (4) and (5) of the height less than n. Then, by the induction hypothesis, there are proofs of the height less than n for

$$\Box\Gamma', \blacksquare A, \blacksquare\Sigma, \Lambda \to \Box\Delta'', \blacksquare B, \blacksquare\Pi, \sharp\Omega, \sharp B, \Xi, B, \sharp A, A \quad \cdots \quad (6)$$
$$A, \Box\Gamma', \blacksquare A, \blacksquare\Sigma, \Lambda \to \Box\Delta'', \blacksquare B, \blacksquare\Pi, \sharp\Omega, \sharp B, \Xi, B \quad \cdots \quad (7)$$
$$B, \Box\Gamma', \blacksquare A, \blacksquare\Sigma, \Lambda \to \Box\Delta'', \blacksquare B, \blacksquare\Pi, \sharp\Omega, \Xi, \sharp A, A \quad \cdots \quad (8)$$
$$A, B, \Box\Gamma', \blacksquare A, \blacksquare\Sigma, \Lambda \to \Box\Delta'', \blacksquare B, \blacksquare\Pi, \sharp\Omega, \Xi \quad \cdots \quad (9)$$

They are of the form upper sequents obtained from (4) and (5) by applying $(cut)_L$ with the cut formula A. In other words,

$$\frac{(6) \quad (7)}{(4)}\,(cut)_L \qquad\qquad \frac{(8) \quad (9)}{(5)}\,(cut)_L$$

If we now start separate proofs of (4) and (5), and apply $(cut)_R$ with the same principal formula $\Box B$, we can obtain the following:

$$\frac{(6) \quad (8)}{(2)}\,(cut)_R \qquad\qquad \frac{(7) \quad (9)}{(3)}\,(cut)_R$$

Since there are proofs of (6), (7), (8) and (9), there are proofs of (2) and (3) as desired.

(c) If (r') is $(cut)_L$ with cut formula C such that $\Box C \in \Box\Gamma'$, that is $\Box C \neq \Box A$ then (1) is of the form

$$\Box\Gamma'', \Box C, \Box A, \blacksquare\Sigma, \Lambda \to \Box\Delta', \blacksquare\Pi, \sharp\Omega, \Xi,$$

where $\Box\Gamma' = \Box\Gamma'' \cup \{\Box C\}$. Similarly to the above case, below.

3. We can show the case $(cut)_R$ similarly to the case $(cut)_L$. □

Suppose that $\Gamma \to \Delta$ is given. Then, let $a(\Gamma \to \Delta)$ and $s(\Gamma \to \Delta)$ denote Γ and Δ, respectively. For sequents $\Gamma_1 \to \Delta_1$ and $\Gamma_2 \to \Delta_2$, when $\Gamma_1 \subseteq \Gamma_2$ and $\Delta_1 \subseteq \Delta_2$, we say that $\Gamma_1 \to \Delta_1$ *is in* $\Gamma_2 \to \Delta_2$. and we write also as $\Gamma_1 \to \Delta_1 \ \subseteq \ \Gamma_2 \to \Delta_2$. A sequent $\Gamma \to \Delta$ is *closed with respect to a rule* (r) if whenever (an instance of) the lower sequent of (r) is in $\Gamma \to \Delta$, so is (a corresponding instance of) at least one of the upper sequents of (r). A sequent $\Gamma \to \Delta$ is $\mathcal{S}\mathbf{KB}$*–saturated* if it is not provable in $\mathcal{S}\mathbf{KB}$ and closed with respect to all rules of $\mathcal{S}\mathbf{KB}$ except $(\Box_B)$.

Lemma 4.2.2 (saturation lemma) *Suppose that a sequent* $\Gamma \to \Delta$ *is not provable in* $\mathcal{S}\mathbf{KB}$. *Then, there is an effective procedure of constructing a* $\mathcal{S}\mathbf{KB}$*–saturated* $(\Gamma \to \Delta)^s$ *such that* $\Gamma \to \Delta \subseteq (\Gamma \to \Delta)^s$, *and that* $a((\Gamma \to \Delta)^s),\ s((\Gamma \to \Delta)^s) \subseteq (\Gamma \cup \Delta)^{\mathcal{S}\mathbf{KB}*}$.

Proof) We will give the procedure for getting such a $\mathcal{S}$**KB**–saturated sequent. Let $s_0 = \Gamma \to \Delta$ and (r) be a static rule of $\mathcal{S}$**KB** with respect to which s_0 is not closed. If there is no such static rule, let $(\Gamma \to \Delta)^s = s_0$.

Suppose that a sequent s_i which is not closed with respect to the static rule (r) is given. Apply (r) to s_i and we can obtain the corresponding upper sequent(s). Since s_i is not provable in $\mathcal{S}$**KB**, at least one of these upper sequents do not have any proof in $\mathcal{S}$**KB**. Let t_i be one of such an upper sequent, i.e. an upper sequent which has no proof in $\mathcal{S}$**KB**. Suppose that the principal formula of the application of (r) is A and $A \in a(s_i)$. Let $\Gamma_i = a(s_i)$ and $\Delta_i = s(s_i)$. Below, the sets of formulas $\Pi, \Sigma, \Lambda, \Theta, \Psi$ and Ω may be empty. Then,

$$\frac{\Pi,\ (\Gamma_i - \{A\}) \to \Delta_i, \Sigma \quad \Lambda,\ (\Gamma_i - \{A\}) \to \Delta_i, \Theta \quad \Psi,\ (\Gamma_i - \{A\}) \to \Delta_i, \Omega}{A,\ (\Gamma_i - \{A\}) \to \Delta_i}\ (r)_A$$

or

$$\frac{\Pi,\ (\Gamma_i - \{A\}) \to \Delta_i, \Sigma}{A,\ (\Gamma_i - \{A\}) \to \Delta_i}\ (r)_A$$

Here, $(r)_A$ means that A is the principal formula in the application of (r). For t_i defined above, put $s_{i+1} = A, a(t_i) \to s(t_i)$. For example, $s_{i+1} = A, \Pi,\ (\Gamma_i - \{A\}) \to \Delta_i, \Sigma$. (When $A \in s(s_i)$, put $s_{i+1} = a(t_i) \to s(t_i), A$, instead.) Therefore, s_{i+1} is closed with respect to $(r)_A$ and

$$s_i = [\Gamma_i \to \Delta_i] \ \subseteq\ [A, \Pi,\ (\Gamma_i - \{A\}) \to \Delta_i, \Sigma] \ =\ [A, a(t_i) \to s(t_i)] \ =\ s_{i+1}.$$

Next, we show that s_{i+1} is not provable in $\mathcal{S}$**KB**. To the contrary, we assume that s_{i+1} is provable in $\mathcal{S}$**KB**, that is, assume that $s_{i+1} = A, \Pi,\ (\Gamma_i - \{A\}) \to \Delta_i, \Sigma$ is provable in $\mathcal{S}$**KB**. Since $(r)_A$ is applicable also to s_{i+1}, we can obtain the following :

$$\frac{\Pi,\ (\Gamma_i - \{A\}) \to \Delta_i, \Sigma \quad \Lambda, \Pi,\ (\Gamma_i - \{A\}) \to \Delta_i, \Sigma, \Theta \quad \Psi, \Pi,\ (\Gamma_i - \{A\}) \to \Delta_i, \Sigma, \Omega}{A, \Pi,\ (\Gamma_i - \{A\}) \to \Delta_i, \Sigma}\ (r)_A$$

or

$$\frac{\Pi,\ (\Gamma_i - \{A\}) \to \Delta_i, \Sigma}{A, \Pi,\ (\Gamma_i - \{A\}) \to \Delta_i, \Sigma}\ (r)_A$$

Since (r) is invertible from Lemma 4.2.1 and s_{i+1} is provable in $\mathcal{S}$**KB**, the sequent $\Pi,\ (\Gamma_i - \{A\}) \to \Delta_i, \Sigma$ is provable in $\mathcal{S}$**KB**. But this sequent none other than t_i, which is not provable. This is a contradiction. Hence, s_{i+1} is not provable in $\mathcal{S}$**KB**.

Now, repeat this procedure on s_{i+1}. Since s_{i+1} is closed with respect to at least one more static rules than those of s_i, the number of static rules under with respect to which

s_{i+1} is not closed becomes smaller. Furthermore, the resulting sequents s_{i+2} are guaranteed to be unprovable in $\mathcal{S}\mathbf{KB}$. By iterating this, we obtain a sequence of unprovable sequents

$$s_0 \subseteq s_1 \subseteq s_2 \subseteq \cdots$$

This sequence terminates at some s_n when s_n is closed with respect to every static rule of $\mathcal{S}\mathbf{KB}$. Of course s_n is not provable in $\mathcal{S}\mathbf{KB}$. Let $(\Gamma \to \Delta)^s = s_n$. It is clear that $\Gamma \to \Delta \subseteq (\Gamma \to \Delta)^s$. Since every formula occurring in this procedure is in $(\Gamma \cup \Delta)^{\mathcal{S}\mathbf{KB}*}$, we can see that $a((\Gamma \to \Delta)^s), s((\Gamma \to \Delta)^s) \subseteq (\Gamma \cup \Delta)^{\mathcal{S}\mathbf{KB}*}$. □

A sequent $\Gamma \to \Delta$ is *subformula–complete* if $A \in Sub(\Gamma \cup \Delta)$ implies either $A \in \Gamma$ or $A \in \Delta$.

Lemma 4.2.3 *If a sequent $\Gamma \to \Delta$ is closed with respect to all static rules of $\mathcal{S}\mathbf{KB}$ then $\Gamma \to \Delta$ is subformula–complete.*

Proof) Clear. □

Definition 4.2.4 *Let W be a nonempty set and R be a binary relation on W, that is $R \subseteq W \times W$. Then a* **KB–model graph** *for a sequent $\Gamma \to \Delta$ is a finite* **KB***–frame (W, R) such that W consists of $\mathcal{S}\mathbf{KB}$–saturated sequents w such that $a(w),\ s(w) \subseteq (\Gamma \cup \Delta)^{\mathcal{S}\mathbf{KB}*}$ and*

1. *$\Gamma \to \Delta \subseteq w_0$ for some $w_0 \in W$,*
2. *if $\blacksquare A \in s(w)$ then there exists some $w' \in W$ with wRw' and $A \in s(w')$,*
3. *if wRw' and $\blacksquare A \in a(w)$ then $A \in a(w')$.*

As long as no confusion occurs , we write a **KB**–model graph simply as a model graph in the following. For sets of formulas $\Gamma = \{A_1, \cdots, A_n\}$ and $\Delta = \{B_1, \cdots, B_m\}$, let Γ_* denote $A_1 \vee \cdots \vee A_n$ and Δ^* denote $B_1 \vee \cdots \vee B_m$ $(n, m \geq 0)$.

Lemma 4.2.5 (satisfiability lemma) *If (W, R) is a model graph for $\Gamma \to \Delta$ then there exists a* **KB***–model $(W, R, \models)$ such that $w \not\models \Gamma_* \supset \Delta^*$ for some $w \in W$.*

Proof) In order to show this lemma, we need to define a valuation $\models$. For any $w \in W$ and any propositional variable p, define $w \models p$ *iff* $p \in a(w)$. Then, we can show the following:

1. if $A \in a(w)$ then $w \models A$,
2. if $A \in s(w)$ then $w \not\models A$

by simultaneous induction for any formula A in $a(w) \cup s(w)$. From this our lemma follows by taking w_0 for w, where w_0 satisfies that $\Gamma \to \Delta \subseteq w_0$. □

4.2.3 Completeness of $\mathcal{S}$KB

In this section, we will show soundness and completeness of $\mathcal{S}$**KB**. We say $\mathcal{S}$**KB** is *sound* with respect to **KB**–frames if $\Gamma \to \Delta$ is provable then $\Gamma_* \supset \Delta^*$ is valid for any **KB**–frame. We say $\mathcal{S}$**KB** is *complete* with respect to **KB**–frames if $\Gamma_* \supset \Delta^*$ is valid for any **KB**–frame then $\Gamma \to \Delta$ is provable.

We first need to define the semantics of $\blacksquare A$ and $\sharp A$. For a given Kripke model $(W, R, \models)$ and $w \in W$, we define $w \models \blacksquare A$ and $w \models \sharp A$ as follows:

$$\begin{array}{lll} w \models \blacksquare A & \textit{iff} & w \models \Box A, \\ w \models \sharp A & \textit{iff} & w \models A. \end{array}$$

This means that $\blacksquare$ has the same meaning as $\Box$ and $\sharp$ has no semantical role, though each of them has an intrinsic syntactical meaning. The idea of introducing $\sharp$ is due to Nguyen [26]. We can show soundness of $\mathcal{S}$**KB** straightforwardly.

Theorem 4.2.6 *The sequent system* $\mathcal{S}$**KB** *is sound with respect to* **KB***–frames.*

Proof) We can show soundness of $\mathcal{S}$**KB** by using the length of proof–figure. □

Now, we will show completeness of $\mathcal{S}$**KB** via model graphs. We show the contraposition. A sketch of completeness of **L** goes as follows:

1. Suppose that $\Gamma \to \Delta$ is not provable in **L**,
2. Create a $\mathcal{S}$**KB**–saturated $(\Gamma \to \Delta)^s$ by Saturation Lemma,
3. Put $w_0 = (\Gamma \to \Delta)^s$,
4. Construct a model graph for $\Gamma \to \Delta$,
5. Thus, we can obtain a **L**–model $(W, R, \models)$ such that $w_0 \not\models \Gamma_* \supset \Delta^*$ by Satisfiability Lemma.

The main point in this proof of completeness is to construct a modal graph. By giving finite counter–model, we can prove not only completeness of $\mathcal{S}$**KB** but also the finite model property of **KB**. This implies that $\mathcal{S}$**KB** gives us a proof–search procedure for **KB** and that **KB** is decidable.

Theorem 4.2.7 *The sequent system* $\mathcal{S}$**KB** *is complete with respect to* **KB***–frames.*

Proof) We give a way to construct a finite model graph (W, R) for $\Gamma \to \Delta$ when it is not provable. The first step is to create a $\mathcal{S}$**KB**–saturated w_0 from $\Gamma \to \Delta$ with $\Gamma \to \Delta \subseteq w_0$, where $a(w_0), s(w_0) \subseteq (\Gamma \cup \Delta)^{\mathcal{S}\mathbf{KB}*}$. This is possible by Lemma 4.2.2. We construct a model graph for $\Gamma \to \Delta$ from w_0 as follows:

1. If no formula of the form $\blacksquare A$ in $s(w_0)$, then $(W, R) = (\{w_0\}, \emptyset)$ is the desired model graph since this is a **KB**–frame and all the properties of model graphs in Definition 4.2.4 are satisfied.

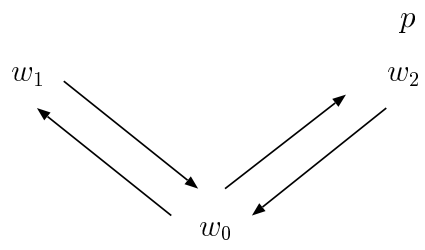
2. Otherwise, let $A_1, \cdots, A_m$ be all formulas such that $\blacksquare A_i \in s(w_0)$ $(1 \leq i \leq m)$. Since w_0 is not provable in $\mathcal{S}$**KB**, $\blacksquare\Gamma_0 \to \blacksquare A_1, \cdots, \blacksquare A_m, \sharp\Omega$is neither provable in $\mathcal{S}$**KB**, where $\blacksquare\Gamma_0$ consists of all $\blacksquare$–formulas in $a(w_0)$ and $\sharp\Omega$ contains of all $\sharp$–formulas in $s(w_0)$. Then, apply $(\Box)_B$ to $\blacksquare\Gamma_0 \to \blacksquare A_1, \cdots, \blacksquare A_m, \sharp\Omega$ and we can obtain $\Gamma_0 \to A_i, \Box\Omega_i$, where $\Omega_i \subseteq \Omega$, $\Box\Omega_i \subseteq Sub(\Gamma_0 \cup \{A_i\})$ and $\Box(\Omega - \Omega_i) \cap Sub(\Gamma \cup \{A_i\}) = \emptyset$ $(1 \leq i \leq m)$. Since $\blacksquare\Gamma_0 \to \blacksquare A_1, \cdots, \blacksquare A_m, \sharp\Omega$ is not provable in $\mathcal{S}$**KB**, none of $\Gamma_0 \to A_i, \Box\Omega_i$ is provable in $\mathcal{S}$**KB**. Thus we can create a $\mathcal{S}$**KB**–saturated $w_i = (\Gamma_0 \to A_i, \Box\Omega_i)^s$ by Lemma 4.2.2, and put $w_0 R w_i$ for each i.

3. We repeat the above by taking each w_i, instead of w_0.

In order to show that this construction terminates, we need a modal degree introduced in next section. In next section, we will show that each the upper sequents of $(\Box)_B$ has the modal degree which strictly less than that of its the lower sequent, and the $\mathcal{S}$**KB**–saturation process does not increase the modal degree. Hence, each successor created by $(\Box)_B$ has the modal degree which is strictly less than that of the parent node, and hence this construction terminates.

Let W be the set of all the nodes created in this process. Then W is finite. Take R^* as the symmetric closure of R. It remains to show that (W, R^*) satisfies the properties $1, 2$ and 3 of model graphs in Definition 4.2.4. It is clear by the way of construction of (W, R^*) that both properties 1 and 2 are satisfied. We show (W, R^*) satisfies property 3. Let w and w' be in W. When w' is a successor of w, we can see by the form of $(\Box)_B$ that wR^*w' and $\blacksquare A \in a(w)$ imply $A \in a(w')$. Since R^* is symmetric, we also have to show that $w'R^*w$ and $\blacksquare A \in a(w')$ imply $A \in a(w)$. Suppose that $w'R^*w$ and $\blacksquare A \in a(w')$. Because $\blacksquare A \in a(w')$, $\Box A$ must be in $Sub(a(w) \cup s(w))$. Since w is subformula–complete, $\Box A \in (a(w) \cup s(w))$. Since w is closed with respect to $(cut)_L$ and $(cut)_R$, $A \in a(w)$ or $A \in s(w)$. Suppose that $A \in s(w)$. That implies $\sharp A \in s(w)$. In this case, when w' is generated from w by using $(\Box)_B$, $\Box A \in s(w')$, which is from $\sharp A \in s(w)$. Since w' must be unprovable, this is a contradiction. Therefore, $A \in a(w)$. Thus, (W, R^*) is a model graph for $\Gamma \to \Delta$. Finally, we can get a **KB**–model $(W, R^*, \models)$ such that $w_0 \not\models \Gamma_* \supset \Delta^*$ from Lemma 4.2.5. $\Box$

Example 1 The following figure is a **KB**–model graph for $\Box\Box p \to \Box p \vee \Box\neg p$.
It is easily seen that $w_0 \not\models \Box\Box p \supset \Box p \vee \Box\neg p$.



$w_0 = \Box\Box p, \blacksquare\Box p, p \to \Box p \vee \Box\neg p, \Box p, \Box\neg p, \blacksquare p, \blacksquare\neg p, \sharp\Box p, \sharp\neg p, \neg p$

$w_1 = \Box p, \blacksquare p \to \sharp p, p$

$w_2 = \Box p, \blacksquare p, p \to \neg p$

Figure 4.2: A counter–model for $\Box\Box p \supset \Box p \vee \Box\neg p$

4.2.4 Conversion of proofs of $\mathcal{S}$KB

Since auxiliary modal operators $\blacksquare$ and $\sharp$ occur in proofs of $\mathcal{S}$**KB**, it is hard to read proofs of $\mathcal{S}$**KB** than those of **KB***. Therefore, we require a way to convert proofs of $\mathcal{S}$**KB** to proofs which are easier for us to read, i.e. proofs of **KB***. In this section, we discuss a conversion of proofs of $\mathcal{S}$**KB** to those of **KB***. First, we will clarify the relation between **KB**, **KB*** and $\mathcal{S}$**KB**. The relation among them is shown in Figure 4.3.

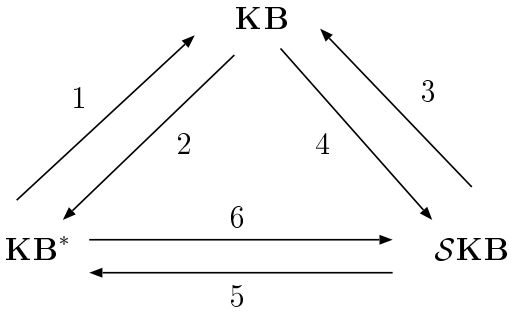


Figure 4.3: Relation between **KB**, **KB*** and $\mathcal{S}$**KB**

For any given sequent $\Gamma \to \Delta$, each of Arrows $1, 2, 3, 4, 5$ and 6 means the following:

1. (Soundness) : If $\Gamma \to \Delta$ is provable in **KB*** then it is valid in **KB**.
2. (Completeness) : If $\Gamma_* \supset \Delta^*$ is valid in **KB** then $\Gamma \to \Delta$ is provable in **KB***.
3. (Soundness) : If $\Gamma \to \Delta$ is provable in $\mathcal{S}$**KB** then $\Gamma_* \supset \Delta^*$ is valid in **KB**.
4. (Completeness) : If $\Gamma_* \supset \Delta^*$ is valid in **KB** then $\Gamma \to \Delta$ is provable in $\mathcal{S}$**KB**.
5. (Soundness) : If $\Gamma \to \Delta$ is provable in $\mathcal{S}$**KB** then it is provable in **KB***.
6. (Completeness) : If $\Gamma \to \Delta$ is provable in **KB*** then it is provable in $\mathcal{S}$**KB**.

What we would like to highlight is Arrow 6, that is, it is desirable a conversion proofs of $\mathcal{S}$**KB** to those of **KB***. An *internal subformula* of a formula A is a subformula of

some formula C such that $\Box C$ is a subformula is of A. For example, $\Box p$ is an internal subformula of $\Box\Box p$. This definition is due to Takano in [35]. In addition, when a formula $\Box C$ is an internal subformula of a formula A, we call C a *proper internal subformula* of A. For example, proper internal subformulas of $\Box\Box\Box p$ are $\Box p$ and p, among which $\Box p$ is called a *maximal proper internal subformula* of $\Box\Box\Box p$.

Our interest is how to convert $(cut)_L$, $(cut)_R$ and $(\Box)_B$. Let $\Lambda = \{p_1, \cdots, p_n\}$ and $\Xi = \{p_1, \cdots, p_n\}$. A conversion of $(cut)_L$ goes as follows:

1. Replace $\blacksquare$ with $\Box$ and eliminate $\sharp$:

$$\frac{\Box\Gamma, \blacksquare A, \blacksquare\Pi, \Lambda \to \Box\Delta, \blacksquare\Sigma, \sharp\Omega, \sharp A, \Xi, A \quad A, \Box\Gamma, \blacksquare A, \blacksquare\Pi, \Lambda \to \Box\Delta, \blacksquare\Sigma, \sharp\Omega, \Xi}{\Box\Gamma, \Box A, \blacksquare\Pi, \Lambda \to \Box\Delta, \blacksquare\Sigma, \sharp\Omega, \Xi}\ (cut)_L$$

2. Add contraction and exchange rules :

$$\frac{\Box\Gamma, \Box A, \Box\Pi, \Lambda \to \Box\Delta, \Box\Sigma, \Omega, A, \Xi, A \quad A, \Box\Gamma, \Box A, \Box\Pi, \Lambda \to \Box\Delta, \Box\Sigma, \Omega, \Xi}{\Box\Gamma, \Box A, \Box\Pi, \Lambda \to \Box\Delta, \Box\Sigma, \Omega, \Xi}$$

3. We can obtain as follows :

$$\dfrac{\dfrac{\dfrac{\Box\Gamma, \Box A, \Box\Pi, \Lambda \to \Box\Delta, \Box\Sigma, \Omega, \Xi, A, A}{\Box\Gamma, \Box A, \Box\Pi, \Lambda \to \Box\Delta, \Box\Sigma, \Omega, \Xi, A}\ (\to c), (\to e) \qquad A, \Box\Gamma, \Box A, \Box\Pi, \Lambda \to \Box\Delta, \Box\Sigma, \Omega, \Xi}{\Box\Gamma, \Box\Gamma, \Box A, \Box A, \Box\Pi, \Box\Pi, \Lambda, \Lambda \to \Box\Delta, \Box\Delta, \Box\Sigma, \Box\Sigma, \Omega, \Omega, \Xi, \Xi}\ (cut)}{\vdots\ (c \to), (\to c), (e \to), (\to e)}}{\Box\Gamma, \Box A, \Box\Pi, \Lambda \to \Box\Delta, \Box\Sigma, \Omega, \Xi}$$

We can convert of $(cut)_R$ similarly. A conversion from $(\Box)_B$ goes as follows:

1. Replace $\blacksquare$ with $\Box$ and eliminate $\sharp$:

$$\frac{\Gamma \to A_i, \Box\Omega_i}{\blacksquare\Gamma, \Lambda \to \blacksquare A_1, \cdots, \blacksquare A_n, \sharp\Omega, \Xi}\ (\Box)_B$$

$$\Omega_i \subseteq \Omega,\ \Box\Omega_i \subseteq Sub(\Gamma \cup \{A_i\}),\ \Box(\Omega - \Omega_i) \cap Sub(\Gamma_i \cup \{A_i\}) = \emptyset\ (1 \leq \imath \leq m)$$

2. Add weakening and exchange rules :

$$\frac{\Box\Gamma \to A_i, \Box\Omega_i}{\Box\Gamma, \Lambda \to \Box A_1, \cdots, \Box A_n, \Omega, \Xi}$$

3. We can obtain as follows :

$$\dfrac{\dfrac{\Gamma \to A_i, \Box\Omega_i}{\Box\Gamma \to \Box A_i, \Omega_i}\ (\to \Box)_B}{\vdots\ (w \to), (\to w), (e \to), (\to e)}$$
$$\Box\Gamma, \Lambda \to \Box A_1, \cdots, \Box A_m, \Omega, \Xi$$

Example 2 In the following, we will convert a proof of $\mathcal{S}\mathbf{KB}$ of $p \to \Box\neg\Box\neg p$ to that of $\mathbf{KB}^*$ of it.

1. A given proof of $\mathcal{S}\mathbf{KB}$ of $p \to \Box\neg\Box\neg p$:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\Box\neg p \to \Box\neg p, p}{\to \neg\Box\neg p, \Box\neg p, p}(\to\neg)}{\neg p \to \neg\Box\neg p, \Box\neg p}(\neg\to)}{\blacksquare\neg p, p \to \blacksquare\neg\Box\neg p, \sharp\neg\Box\neg p, \sharp\neg p}(\Box)_B}{\blacksquare\neg p, p \to \blacksquare\neg\Box\neg p, \sharp\neg\Box\neg p, \sharp\neg p, \neg p}(\to\neg) \quad \dfrac{\blacksquare\neg p, p \to \blacksquare\neg\Box\neg p, \sharp\neg\Box\neg p, p}{\neg p, \blacksquare\neg p, p \to \blacksquare\neg\Box\neg p, \sharp\neg\Box\neg p}(\neg\to)}{\Box\neg p, p \to \blacksquare\neg\Box\neg p, \sharp\neg\Box\neg p}(cut)_L \quad \dfrac{\dfrac{\dfrac{\dfrac{\Box\neg p \to \Box\neg p}{\to \neg\Box\neg p, \Box\neg p}(\to\neg)}{p \to \blacksquare\neg\Box\neg p, \blacksquare\neg p, \sharp\neg p}(\Box)_B}{p \to \blacksquare\neg\Box\neg p, \blacksquare\neg p, \sharp\neg p, \neg p}(\to\neg) \quad \dfrac{p \to \blacksquare\neg\Box\neg p, \blacksquare\neg p, p}{\neg p, p \to \blacksquare\neg\Box\neg p, \blacksquare\neg p}(\neg\to)}{p \to \blacksquare\neg\Box\neg p, \Box\neg p}(cut)_R}{\dfrac{\Box\neg p, p \to \blacksquare\neg\Box\neg p, \sharp\neg\Box\neg p}{p \to \blacksquare\neg\Box\neg p, \sharp\neg\Box\neg p, \neg\Box\neg p}(\to\neg) \quad \dfrac{p \to \blacksquare\neg\Box\neg p, \Box\neg p}{\neg\Box\neg p, p \to \blacksquare\neg\Box\neg p}(\neg\to)}(cut)_R}{p \to \Box\neg\Box\neg p}$$

2. Convert each of rules of $\mathcal{S}\mathbf{KB}$ to those of $\mathbf{KB}^*$:

$$\dfrac{\begin{array}{c}\vdots\\ p, p \to \Box\neg\Box\neg p, \Box\neg\Box\neg p\end{array}}{p \to \Box\neg\Box\neg p}(c\to), (\to c), (e\to), (\to e)$$

The remainder is as follows:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\Box\neg p \to \Box\neg p, p}{\to \neg\Box\neg p, \Box\neg p, p}(\to\neg)}{\neg p \to \neg\Box\neg p, \Box\neg p}(\neg\to)}{\Box\neg p, p \to \Box\neg\Box\neg p, \neg\Box\neg p, \neg p}(\to\Box)_B}{\Box\neg p \to \Box\neg\Box\neg p, \neg p}(R2)}{\Box\neg p, p \to \Box\neg\Box\neg p, \neg\Box\neg, \neg p, \neg p}(\to\neg) \quad \dfrac{\neg p, p \to \Box\neg\Box\neg p, \neg\Box\neg p, p}{\neg p, \Box\neg p, p \to \Box\neg\Box\neg p, \neg\Box\neg p}(\neg\to)}{\Box\neg p, \Box\neg p, p, p \to \Box\neg\Box\neg p, \Box\neg\Box\neg p, \neg\Box\neg p, \neg\Box\neg p}(cut)}{\dfrac{\Box\neg p, p \to \Box\neg\Box\neg p, \neg\Box\neg p}{p \to \Box\neg\Box\neg p, \neg\Box\neg p, \neg\Box\neg p}(\to\neg)}(R1) \quad \dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\Box\neg p \to \Box\neg p}{\to \neg\Box\neg p, \Box\neg p}(\to\neg)}{\to \Box\neg\Box\neg p, \neg p}(\to\Box)_B}{p \to \Box\neg\Box\neg p, \Box\neg p, \neg p}(R2)}{p \to \Box\neg\Box\neg p, \Box\neg p, \neg p, \neg p}(\to\neg) \quad \dfrac{p \to \Box\neg\Box\neg p, \Box\neg p, p}{\neg p, p \to \Box\neg\Box\neg p, \Box\neg p}(\neg\to)}{p, p \to \Box\neg\Box\neg p, \Box\neg\Box\neg p, \Box\neg p, \Box\neg p}(cut)}{\dfrac{p \to \Box\neg\Box\neg p, \Box\neg p}{\neg\Box\neg p, p \to \Box\neg\Box p}(\neg\to)}(R1)}{p, p \to \Box\neg\Box\neg p, \Box\neg\Box\neg p}(cut)$$

where $(R1)$ denotes $(c\to), (\to c), (e\to), (\to e)$ and $(R2)$ denotes $(w\to), (\to w), (e\to), (\to e)$.

3. In addition, for readability's sake, to simplify the above proof is required and we can obtain the following proof of $\mathbf{KB}^*$:

$$\dfrac{\dfrac{\dfrac{\dfrac{\Box\neg p \to \Box\neg p}{\to \neg\Box\neg p, \Box\neg p}(\to\neg)}{\to \Box\neg\Box\neg p, \neg p}(\to\Box)_B \quad \dfrac{p \to p}{\neg p, p \to}(\neg\to)}{p \to \Box\neg\Box\neg p}(cut)$$

Though it is possible to convert each of rules of $\mathcal{S}\mathbf{KB}$ to that of $\mathbf{KB}$, converted proofs will be larger without simplification. In $\mathcal{S}\mathbf{KB}$, cut formulas are always taken from internal subformulas. Actually, in every application of $(cut)_L$ or $(cut)_R$, a maximal proper internal subformula suffices for the cut formula. Taking into account that, we can eliminate unnecessary applications of $(cut)_L$ and $(cut)_R$ from proofs of $\mathcal{S}\mathbf{KB}$. Elimination of unnecessary applications of cut rule will be one way of simplification. Since applications of rules depends much on the context, we need more to discuss simplification.

4.3 A sequent system for K4B

In this section, we introduce a sequent system for **K4B** based on Takano's sequent system for **K4B**. As we know, the modal logic **K4B** can be defined by adding the axiom **4** to

the modal logic **KB**. The axiom **4** makes the accessibility relation to Kripke frames, transitive. Therefore in constructing model graphs, we have to keep in mind that the accessibility relation is not only symmetric but also transitive in this case.

4.3.1 $\mathcal{S}$K4B

In the following, we will introduce a sequent system for **K4B**, called $\mathcal{S}$**K4B**, which has the auxiliary modal operator ■ (but without ♯). As we discussed in Chapter 3, transitivity would cause loops as follows:

$$\dfrac{\dfrac{\dfrac{\vdots}{\Box\neg\Box A \to \Box A, A}}{\Box\neg\Box A, \neg\Box A \to A, \Box A}\,(\neg\to)}{\Box\neg\Box A \to \Box A, A}\,(\to\Box)_{4BE}$$

We can see that once $\Box$–formulas in the left hand side of a sequent, they disappear by any application of $(\to \Box)_{4BE}$. To avoid such loops caused by transitivity, we use history of the form $\langle \Box\Pi | \Box\Sigma \rangle$ like that of Chapter 3. Each of sequents also carries history. Let $\mathcal{H}$ denote a history $\langle \Box\Pi | \Box\Sigma \rangle$. Initial sequents of $\mathcal{S}$**K4B** are sequents of the form $\Gamma, A \to A, \Delta\ \mathcal{H}$. Rules of inference of $\mathcal{S}$**K4B** are given in Figure 4.4.

Here, ■Π denotes the set $\{\blacksquare A | A \in \Pi\}$. Similarly to the case of $\mathcal{S}$**KB**, we call each of $(\Box)_s$ and $(\Box)_t$ *transitional rule* and call others *static rules*. The upper sequent of each of $(\Box)_s$ and $(\Box)_t$ is also taken 'or'–branch. In $(cut)_L$ and $(cut)_R$, the sets Λ and Ξ are disjoint, respectively. Also in the previous case, cut formula is taken from $\Box$–formula in the lower sequent of $(cut)_L$ or $(cut)_R$. Similarly to the case of $\mathcal{S}$**KB**, the subformula property in the strict sense does not hold in $\mathcal{S}$**K4B**, but we can see that for a sequent $\Gamma \to \Delta\ \langle\epsilon|\epsilon\rangle$, any formula occurring in all proofs of $\Gamma \to \Delta\ \langle\epsilon|\epsilon\rangle$, including any formula occurring in all histories, is in $(\Gamma \cup \Delta)^{\mathcal{S}\mathbf{K4B}*}$, where

$$(\Gamma \cup \Delta)^{\mathcal{S}\mathbf{K4B}*} = Sub(\Gamma \cup \Delta)\ \cup\ \{\blacksquare A \mid \Box A \in Sub(\Gamma \cup \Delta)\}.$$

The proof–search procedure determined by $\mathcal{S}$**K4B** goes similarly to that of $\mathcal{S}$**KB**. Here, we note that as for applications of rules of $\mathcal{S}$**K4B**, every application of (T) is meant to have priority over any application of $(cut)_L$ in order to avoid redundant applications of $(cut)_L$. For example, assuming that we ignore ■, when we apply (T) after an application of $(cut)_L$, we can have the following :

$$\dfrac{\dfrac{\Box A, A \to \Box B, \sharp A, A\ \mathcal{H}}{\Box A \to \Box B, \sharp A, A\ \mathcal{H}}\,(T) \qquad \dfrac{\dfrac{\Box A, A \to \Box B, \sharp B, B\ \mathcal{H} \quad B, A, \Box A \to \Box B\ \mathcal{H}}{\Box A, A \to \Box B\ \mathcal{H}}\,(cut)_R}{\Box A, A \to \Box B\ \mathcal{H}}\,(T)}{\Box A \to \Box B\ \mathcal{H}}\,(cut)_L$$

initial sequent

$$\Gamma, A \to A, \Delta \ \mathcal{H}$$

static rule

$$\frac{\Gamma, A, B \to \Delta \ \mathcal{H}}{\Gamma, A \wedge B \to \Delta \ \mathcal{H}} \ (\wedge \to) \qquad \frac{\Gamma \to \Delta, A, B \ \mathcal{H}}{\Gamma \to \Delta, A \vee B \ \mathcal{H}} \ (\to \vee) \qquad \frac{\Gamma, A \to \Delta, B \ \mathcal{H}}{\Gamma \to \Delta, A \supset B \ \mathcal{H}} \ (\to \supset)$$

$$\frac{\Gamma \to \Delta, A \ \mathcal{H}}{\Gamma, \neg A \to \Delta \ \mathcal{H}} \ (\neg \to) \qquad \frac{\Gamma, A \to \Delta \ \mathcal{H}}{\Gamma \to \Delta, \neg A \ \mathcal{H}} \ (\to \neg)$$

$$\frac{\Gamma \to \Delta, A, B \ \mathcal{H} \quad \Gamma, B \to \Delta, A \ \mathcal{H} \quad \Gamma, A \to \Delta, B \ \mathcal{H}}{\Gamma \to \Delta, A \wedge B \ \mathcal{H}} \ (\to \wedge)$$

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

$$\frac{\Gamma, A, B \to \Delta \ \mathcal{H} \quad \Gamma, B \to \Delta, A \ \mathcal{H} \quad \Gamma \to \Delta, A, B \ \mathcal{H}}{\Gamma, A \supset B \to \Delta \ \mathcal{H}} \ (\supset \to)$$

$$\frac{\Box\Gamma, \blacksquare A, \blacksquare\Phi, \Lambda \to \Box\Delta, \blacksquare\Psi, \Xi, A \ \mathcal{H} \quad A, \Box\Gamma, \blacksquare A, \blacksquare\Phi, \Lambda \to \Box\Delta, \blacksquare\Psi, \Xi \ \mathcal{H}}{\Box\Gamma, \Box A, \blacksquare\Phi, \Lambda \to \Box\Delta, \blacksquare\Psi, \Xi \ \mathcal{H}} \ (cut)_L$$

$$\frac{\Box\Gamma, \blacksquare\Phi, \Lambda \to \Box\Delta, \blacksquare A, \blacksquare\Psi, \Xi, A \ \mathcal{H} \quad A, \Box\Gamma, \blacksquare\Phi, \Lambda \to \Box\Delta, \blacksquare A, \blacksquare\Psi, \Xi \ \mathcal{H}}{\Box\Gamma, \blacksquare\Phi, \Lambda \to \Box\Delta, \Box A, \blacksquare\Psi, \Xi \ \mathcal{H}} \ (cut)_R$$

$$\Lambda \equiv p_1, \cdots, p_n, \ \Xi \equiv q_1, \cdots, q_m$$

$$\frac{\blacksquare A, A, \Gamma \to \Delta, \blacksquare B \ \mathcal{H}}{\Box A, \Gamma \to \Delta, \blacksquare B \ \mathcal{H}} \ (T)$$

transitional rule

$$\frac{\Box\Gamma, \Gamma \to \Box\Theta, \Box\Delta, A_1 \ \langle \Box\Gamma | \Box\Sigma, \Box\Theta \rangle \qquad \cdots \qquad \Box\Gamma, \Gamma \to \Box\Theta, \Box\Delta, A_m \ \langle \Box\Gamma | \Box\Sigma, \Box\Theta \rangle}{\blacksquare\Gamma, p_1, \cdots, p_n \to \blacksquare A_1, \cdots, \blacksquare A_m, \blacksquare\Delta, q_1, \cdots, q_l \ \langle \Box\Gamma | \Box\Sigma \rangle} \ (\Box)_s$$

$$\Box\Theta \equiv \Box A_1, \cdots, \Box A_i, \cdots, \Box A_m, \ \Box\Theta \cap \Box\Sigma = \emptyset, \ \Box\Delta \subseteq \Box\Sigma$$

$$\frac{\Box\Gamma, \Gamma \to \Box\Theta, A_1 \ \langle \Box\Gamma | \Box\Theta \rangle \qquad \cdots \qquad \Box\Gamma, \Gamma \to \Box\Theta, A_m \ \langle \Box\Gamma | \Box\Theta \rangle}{\blacksquare\Gamma, p_1, \cdots, p_n \to \blacksquare A_1, \cdots, \blacksquare A_m, q_1, \cdots, q_l \ \langle \Box\Pi | \Box\Sigma \rangle} \ (\Box)_t$$

$$\Box\Theta \equiv \Box A_1, \cdots, \Box A_i, \cdots, \Box A_m, \ \Box\Pi \subsetneq \Box\Gamma$$

Figure 4.4: Sequent system for **K4B** : $\mathcal{S}$**K4B**

We can see that the left uppermost sequent is a initial sequent. If we apply (T) before an application of $(cut)_L$ using $\blacksquare$, we can have the following:

$$\dfrac{\dfrac{\blacksquare A, A \to \blacksquare B, \sharp B, B\ \mathcal{H} \quad B, \blacksquare A, A \to \blacksquare B\ \mathcal{H}}{\blacksquare A, A \to \Box B\ \mathcal{H}}\ (cut)_R}{\Box A \to \Box B\ \mathcal{H}}\ (T)$$

By using $\blacksquare$ also in (T), $(cut)_L$ is make to be unapplicable and hence we can avoid redundant applications of $(cut)_L$. That is why (T) is meant to have priority over $(cut)_L$.

4.3.2 Model graphs for $\mathcal{S}$K4B

We need to modify the technical machinery given in Subsection 4.2.2 so as to deal with sequents with histories. We will define model graphs for the sequent system $\mathcal{S}$**K4B**.

Lemma 4.3.1 *Each static rule of* $\mathcal{S}$**K4B** *is invertible in* $\mathcal{S}$**K4B**.

Proof) Similarly to Theorem 4.2.1. □

For sequents $\Gamma_1 \to \Delta_1\ \mathcal{H}$ and $\Gamma_2 \to \Delta_2\ \mathcal{H}$ with the same history, we say that $\Gamma_1 \to \Delta_1\ \mathcal{H}$ *is in* $\Gamma_2 \to \Delta_2\ \mathcal{H}$, when $\Gamma_1 \subseteq \Gamma_2$ and $\Delta_1 \subseteq \Delta_2$. In this case, we write $\Gamma_1 \to \Delta_1\ \mathcal{H} \subseteq \Gamma_2 \to \Delta_2\ \mathcal{H}$. A sequent $\Gamma \to \Delta\ \mathcal{H}$ is *closed with respect to a rule* (r) if whenever (an instance of) the lower sequent of (r) is in $\Gamma \to \Delta\ \mathcal{H}$, so is (a corresponding instance of) at least one of the upper sequents of (r). $\mathcal{S}$**K4B**–saturated sequents are defined like $\mathcal{S}$**KB**–saturated sequents.

Lemma 4.3.2 (saturation lemma) *Suppose that a sequent* $\Gamma \to \Delta\ \mathcal{H}$ *is not provable in* $\mathcal{S}$**K4B**. *Then, there is an effective procedure to construct some* $\mathcal{S}$**K4B**–*saturated* $(\Gamma \to \Delta)^s\ \mathcal{H}$ *with* $\Gamma \to \Delta \subseteq (\Gamma \to \Delta)^s$, *where* $a((\Gamma \to \Delta)^s),\ s((\Gamma \to \Delta)^s) \subseteq (\Gamma \cup \Delta)^{\mathcal{S}\mathbf{K4B}*}$.

Proof) Similarly to Theorem 4.2.2. Note that any application of the static rules does not change history. □

Lemma 4.3.3 *If a sequent* $\Gamma \to \Delta\ \mathcal{H}$ *is closed with respect to all static rules of* $\mathcal{S}$**K4B** *then* $\Gamma \to \Delta$ *is subformula–complete.*

Proof) Clear. □

Definition 4.3.4 *Let* W *be a nonempty set and* R *be a binary relation on* W, *that is* $R \subseteq W \times W$. *Then a* **K4B–model graph** *for a sequent* $\Gamma \to \Delta\ \langle\epsilon|\epsilon\rangle$ *is a finite* **K4B**–*frame* (W, R) *such that all* $w_i\mathcal{H}_i \in W$ *are* $\mathcal{S}$**K4B**–*saturated sequents, where* $a(w_i),\ s(w_i) \subseteq (\Gamma \cup \Delta)^{\mathcal{S}\mathbf{K4B}*}$ *and*

1. $\Gamma \to \Delta \subseteq w_0$ *for some* $w_0\mathcal{H}_0 \in W$,

2. *if* $\blacksquare A \in s(w_i)$ *then there exists some* $w_j\mathcal{H}_j \in W$ *with* $(w_i\mathcal{H}_i)R(w_j\mathcal{H}_j)$ *and* $A \in s(w_j)$,

3. *if* $(w_i\mathcal{H}_i)R(w_j\mathcal{H}_j)$ *and* $\blacksquare A \in a(w_i)$ *then* $A \in a(w_j)$.

When no confusion occurs, we call each **K4B**–model graph simply, a model graph.

Lemma 4.3.5 (satisfiability lemma) *If* (W, R) *is a model graph for* $\Gamma \to \Delta$ $\langle\epsilon|\epsilon\rangle$ *then there exists a* **K4B***–model* $(W, R, \models)$ *such that* $w \not\models \Gamma_* \supset \Delta^*$ *for some* $w\mathcal{H} \in W$.

Proof) Similarly to Theorem 4.2.5. □

4.3.3 Completeness of $\mathcal{S}$K4B

In this section, we will show soundness and completeness of $\mathcal{S}$**K4B**. As before, $\blacksquare A$ is interpreted as $\Box A$.

Theorem 4.3.6 *The sequent system* $\mathcal{S}$**K4B** *is sound with respect to* **K4B***–frames.*

Proof) By induction on the length of a given proof. In the following, histories are omitted for brevity's sake. We need to show

- every initial sequent is valid,
- in every rule, if each upper sequent of the rule is valid then the lower sequent is valid.

Base Case Trivial.

Induction Step Suppose that each upper sequent of each rule is valid. We show the above for the following two cases.

1. **Case** (T). Suppose that $(\blacksquare A \wedge A \wedge \Gamma_*) \supset (\Delta^* \vee \Box B)$ is valid in a given **K4B**–frame (W, R). In addition, suppose that for $w_0 \in W$, $w_0 \models \Box A \wedge \Gamma_*$. Then, for any w_1 such that w_0Rw_1, w_1Rw_0 holds since R is symmetric. Since R is transitive, w_0Rw_1 and w_1Rw_0 imply w_0Rw_0. Since $w_0 \models \Box A$ and w_0Rw_0, $w_0 \models A$. Therefore, $w_0 \models \Box A \wedge A \wedge \Gamma_*$ and hence $w_0 \models \blacksquare A \wedge A \wedge \Gamma_*$. Since $w_0 \models (\blacksquare A \wedge A \wedge \Gamma_*) \supset (\Delta^* \vee \Box B)$ from the assumption, we have $w_0 \models \Delta^* \vee \Box B$. Thus, $(\Box A \wedge \Gamma^*) \supset (\Delta^* \vee \Box B)$ is valid.

2. **Case** $(\Box)_t$. Suppose that for some i, $((\Box\Gamma)_* \wedge \Gamma_*) \supset (\Box\Theta)^* \vee A_i$ is valid in a given **K4B**–frame (W, R). We assume that the lower sequent of $(\Box)_t$ is of the form $\blacksquare\Gamma, \Lambda \to \blacksquare\Theta, \Xi$, where Λ and Ξ are sets of propositional variables. Suppose that for $w_0 \in W$, $w_0 \models (\blacksquare\Gamma)_* \wedge \Lambda_*$. Then, for any w_1 such that $w_0 R w_1$, $w_1 \models \Gamma_*$. Here, for any w_2 such that $w_1 R w_2$, $w_0 R w_2$ since R is transitive. From $w_0 \models (\blacksquare\Gamma)_*$, $w_2 \models \Gamma_*$. Therefore, for any w_2 such that $w_1 R w_2$, $w_2 \models \Gamma_*$, that is $w_1 \models (\Box\Gamma)_*$. From the assumption, $w_1 \models ((\Box\Gamma)_* \wedge \Gamma_*) \supset (\Box\Theta)^* \vee A_i$. Hence, $w_1 \models (\Box\Theta)^* \vee A_i$ for all w_1 such that $w_0 R w_1$.

   (a) The case where for some possible world w_1 which is one of possible worlds w such that $w_0 R w$, $w_1 \models (\Box\Theta)^*$. Since R is symmetric, $w_1 R w_0$. From $w_1 \models (\Box\Theta)^*$, $w_0 \models \Theta^*$. In addition, for any w_3 such that $w_0 R w_3$, $w_1 R w_3$ since R is transitive. From $w_1 \models (\Box\Theta)^*$, $w_3 \models \Theta^*$. Since $w_3 \models \Theta^*$ for any w_3 such that $w_0 R w_3$, $w_0 \models (\Box\Theta)^*$. in other words, $w_0 \models (\blacksquare\Theta)^*$.

   (b) Otherwise. The case where for all w_1 such that $w_0 R w_1$, $w_1 \models A_i$ always holds. In this case, $w_0 \models \Box A_i$. Hence, $w_0 \models \Box A_1 \vee \cdots \vee sqA_i \vee \cdots \vee \Box A_m$, that is $w_0 \models (\Box\Theta)^*$. Therefore, $w_0 \models (\blacksquare\Theta)^*$.

   Thus, in either case $w_0 \models (\blacksquare\Theta)^*$ and hence $w_0 \models (\blacksquare\Theta)^* \vee \Xi$. Thus, $(\Lambda_* \wedge (\blacksquare\Gamma)_*) \supset ((\blacksquare\Theta)^* \vee \Xi)$ is valid.

We can show other cases similarly to the above. □

Lemma 4.3.7 *If a sequent* $\Gamma \to \Delta\ \langle\epsilon|\epsilon\rangle$ *is not provable in* $\mathcal{S}$**K4B**, *then there exists a finite* **K4B**–*model* $(W, R, \models)$ *such that* $w \not\models \Gamma_* \supset \Delta^*$ *for some* $w\mathcal{H} \in W$.

Proof) Suppose that $\Gamma \to \Delta\ \mathcal{H}_0$ is not provable in $\mathcal{S}$**K4B**, where $\mathcal{H}_0$ is $\langle\epsilon|\epsilon\rangle$. We give a way to construction of finite a **K4B**–model graph (W, R) for $\Gamma \to \Delta\ \mathcal{H}_0$ with a symmetric and transitive relation R. The first step is to create a $\mathcal{S}$**K4B**–saturated $w_0\ \mathcal{H}_0$ from $\Gamma \to \Delta\ \mathcal{H}_0$ with $\Gamma \to \Delta \subseteq w_0$, where $a(w_0), s(w_0) \subseteq (\Gamma \cup \Delta)^{\mathcal{S}\mathbf{K4B}*}$. This is possible by Lemma 4.3.2. Then we construct a **K4B**–model graph for $\Gamma \to \Delta$ from $w_0\ \mathcal{H}_0$ as follows:

1. If no formula of the form $\blacksquare A$ in $s(w_0)$, then $(W, R) = (\{w_0\mathcal{H}_0\}, \emptyset)$ is the desired model graph since this is a **K4B**–frame and all the properties of model graphs in Definition 4.3.4 are satisfied.

2. Otherwise. Let $A_1, \cdots, A_n$ be all formulas such that $\blacksquare A_i \in s(w_0)$ $(1 \leq i \leq n)$. Since $w_0\mathcal{H}_0$ is not provable in $\mathcal{S}$**K4B**,

$$\blacksquare\Gamma_0 \to \blacksquare A_1, \cdots, \blacksquare A_n\ \mathcal{H}_0,$$

is neither provable in $\mathcal{S}\mathbf{K4B}$, where $\blacksquare\Gamma_0$ consists of all $\blacksquare$–formulas in $a(w_0)$. Apply $(\Box)_t$ to it (if $\Gamma_0 = \emptyset$, we apply $(\Box)_s$, instead), and we can obtain

$$\Box\Gamma_0, \Gamma_0 \to \Box\Theta, A_i \ \langle \Box\Gamma_0 | \Box\Theta \rangle,$$

where $\Box\Theta = \{\Box A_1, \cdots, \Box A_i, \cdots, \Box A_n\}$. Since $\blacksquare\Gamma_0 \to \blacksquare A_1, \cdots, \blacksquare A_n \ \mathcal{H}_0$ is not provable in $\mathcal{S}\mathbf{K4B}$, none of $\Box\Gamma_0, \Gamma_0 \to \Box\Theta, A_i \ \langle \Box\Gamma_0 | \Box\Theta \rangle$ can be provable in $\mathcal{S}\mathbf{K4B}$. Put

$$w_i\mathcal{H}_i = (\Box\Gamma_0, \Gamma_0 \to \Box\Theta, A_i)^s \ \langle \Box\Gamma_0 | \Box\Theta \rangle \text{ and } (w_0\mathcal{H}_0)R(w_i\mathcal{H}_i),$$

and stop this process.

We do not need to repeat the above anymore unlike the construction of **KB**–model graphs. The process suffices to construct model graphs. Actually, neither $(\Box)_s$ nor $(\Box)_t$ are applicable to all w_i where $i \neq 0$, Note that all $\Box$–formulas occurring in w_0 are stored in $\langle \Box\Gamma_0 | \Box\Theta \rangle$ because w_0 is subformula–complete.

Next, we will show that modal graphs are constructed by this process. Let W be the set of all nodes created in this process, and R is the symmetric and transitive closure of R. It remains to show that (W, R) is a **K4B**–model graph. We need to show properties $1, 2$ and 3 of mode graphs. In order to show them, first we show the following claims : For any $w_i\mathcal{H}_i \in W$,

(a) if $\blacksquare A \in s(w_i)$ then $\blacksquare A \in s(w_0)$

(b) if $\blacksquare A \in a(w_i)$ then $\blacksquare A \in a(w_0)$.

(a) Suppose that $\blacksquare A \in s(w_i)$. In this case, $\Box A \in s(w_i)$ also holds. By the definition, $\Box A \in Sub(a(w_0) \cup s(w_0))$ must hold. (Recall forms of $(\Box)_t$ and $(\Box)_s$). Since w_0 is subformula–complete, $\Box A \in a(w_0)$ or $\Box A \in s(w_0)$. If $\Box A \in a(w_0)$ then $\Box A \in a(w_i)$. Since w_i is unprovable, this is a contradiction. Therefore, $\Box A \in s(w_0)$ and $\blacksquare A \in s(w_0)$ since w_0 must be closed with respect to $(cut)_R$.

(b) Suppose that $\blacksquare A \in a(w_i)$. As above $\Box A \in Sub(a(w_0) \cup s(w_0))$ and also $\Box A \in a(w_0)$ or $\Box A \in s(w_0)$. If $\Box A \in s(w_0)$ then $\Box A \in s(w_i)$. Since w_i is unprovable, this is a contradiction. Therefore, $\Box A \in a(w_0)$ and hence $\blacksquare A \in a(w_0)$ since w_0 must be closed with respect to $(cut)_L$.

Now, we show (W, R) satisfies properties $1, 2$ and 3 of model graphs in Definition 4.3.4.

1. Clear.

2. Suppose that $\blacksquare A \in s(w)$.

    1) When $w = w_0$, $A \in s(w_i)$ holds. Therefore, there exists w_i with $w_0 R w_i$ and $s(w_i)$.

2) When $w = w_i$, $\blacksquare A \in s(w_i)$ holds, $\blacksquare A \in s(w_0)$ from (a). Hence, for some $1 \leq i \leq n$, $A \in s(w_j)$ holds from the construction. Since R is transitive, $w_i R w_0$ and $w_0 R w_j$ implies $w_i R w_j$. Therefore, there exists w_j with $w_i R w_j$ and $A \in s(w_j)$.

3. Suppose that wRw' and $\blacksquare A \in a(w)$.

   1) **Case** $w = w_0$:

      i. **Case** $w' = w_0$: When $w_0 R w_0$, $A \in a(w_0)$, since w_0 is closed with respect to $(T)_{\mathbf{K4B}}$.

      ii. **Case** $w' = w_i$ $(i \neq 0)$: When $w_0 R w_i$, clear from the construction.

   2) **Case** $w = w_i$ $(i \neq 0)$:

      i. **Case** $w' = 0$: When $w_i R w_0$, from (b), $\blacksquare A \in a(w_i)$ implies $\blacksquare A \in a(w_0)$. By using the same argument as the case $w_0 R w_0$, we have $A \in a(w_0)$.

      ii. **Case** $w' = w_i$: When $w_i R w_i$, from (b), $\blacksquare A \in a(w_i)$ implies $\blacksquare A \in a(w_0)$. From the construction, $A \in a(w_i)$.

      iii. **Case** $w' = w_j$ $(i \neq j)$: When $w_i R w_j$, from (b), $\Box A \in a(w_i)$ implies $\Box \in a(w_0)$. From the construction, there exists w_j such that $w_0 R w_j$ and $A \in a(w_j)$.

Finally, we can obtain **K4B**–model $(W, R, \models)$ such that $w_0 \not\models \Gamma_* \supset \Delta^*$ from Lemma 4.3.5. □

Thus, we can obtain completeness of $\mathcal{S}$**K4B**.

Theorem 4.3.8 *The sequent system* $\mathcal{S}$**K4B** *is complete with respect to* **K4B***–frames.*

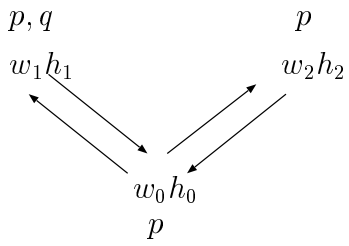
Proof) This is shown in Lemma 4.3.7. □

Example 1 As an example, the following figure is a **K4B**–model for $\Box\Box p \to \Box\neg p \vee \Box q$. It is easily seen that $w_0\langle\epsilon|\epsilon\rangle \not\models \Box\Box p \supset \Box\neg p \vee \Box q$.

4.3.4 Conversion of proofs of $\mathcal{S}$K4B

Similarly to the conversion of proofs of $\mathcal{S}$**KB** to those of **KB***, it is possible to convert proofs of $\mathcal{S}$**K4B** to those of **K4B***. In this section, we only show a conversion of $(\Box)_t$. Other rules are also convertible like the conversion of $(\Box)_t$. Let $\Lambda = \{p_1, \cdots, p_n\}$ and $\Xi = \{q_1, \cdots, q_l\}$.

A conversion of $(\Box)_t$ goes as follows:



$w_0 = p, \Box p, \Box\Box p, \blacksquare p, \blacksquare\Box p \to \Box\neg p \vee \Box q, \Box\neg p, \Box q, \blacksquare\neg p, \blacksquare q, \neg p, q$

$h_0 = \langle \epsilon | \epsilon \rangle$

$w_1 = p, q, \Box p, \Box\Box p, \blacksquare p, \blacksquare\Box p \to \Box\neg p, \Box q, \blacksquare\neg p, \blacksquare q, \neg p$

$h_1 = \langle \Box p, \Box\Box p | \Box\neg p, \Box q \rangle$

$w_2 = p, \Box p, \Box\Box p, \blacksquare p, \blacksquare\Box p \to \Box\neg p, \Box q, \blacksquare\neg p, \blacksquare q, \neg p, q$

$h_2 = \langle \Box p, \Box\Box p | \Box\neg p, \Box q \rangle$

Figure 4.5: A counter–model for $\Box\Box p \supset \Box\neg p \vee \Box q$

1. Replace $\blacksquare$ with $\Box$ and eliminate histories:

$$\frac{\Box\Gamma, \Gamma \to \Box\Theta, A_i \ \langle\Gamma|\Box\Theta\rangle}{\blacksquare\Gamma, \Lambda \to \blacksquare A_1, \cdots, \blacksquare A_m, \Xi \ \langle\Box\Pi|\Box\Sigma\rangle} \ (\Box)_t$$

$$\Theta = \{\Box A_1, \cdots, \Box A_i, \cdots, \Box A_m\}, \ \Box\Pi \subsetneq \Box\Gamma$$

2. Add weakening and exchange rules:

$$\frac{\Box\Gamma, \Gamma \to \Box\Theta, A_i}{\Box\Gamma, \Lambda \to \Box A_1, \cdots, \Box A_m, \Xi}$$

3. We can obtain as follows:

$$\begin{array}{r} \dfrac{\Box\Gamma, \Gamma \to \Box\Theta, A_i}{\Box\Gamma \to \Box\Theta, \Box A_i} \ (\to\Box)_{\mathbf{K4B}^*} \\ \vdots \ (\to w), (\to e) \\ \Box\Gamma \to \Box A_1, \cdots, \Box A_m \\ \vdots \ (w\to), (\to w), (c\to), (\to c) \\ \Box\Gamma, \Lambda \to \Box A_1, \cdots, \Box A_m, \Xi \end{array}$$

As we can see, this conversion makes proofs of $\mathcal{S}\mathbf{K4B}$ larger because of adding structural rules. Therefore, we need to simplify proofs like the conversion of proofs of $\mathcal{S}\mathbf{KB}$. This will also be a further issue.

4.4 Conclusion

In this chapter, we gave proof–search procedures for **KB** and **K4B**. In both $\mathcal{S}$**KB** and $\mathcal{S}$**K4B**, each cut formula is restricted to a formula A taken from $\Box A$ occurring either side of a lower sequent of cut rules. This will contribute to development of techniques for implementing of proof–search procedures for **KB** and **K4B**. Since completeness of $\mathcal{S}$**KB** and $\mathcal{S}$**K4B** are shown by constructing finite counter–models, the finite model property for them is also obtained.

4.5 Note

In this chapter, we took up not tableau systems but sequent systems for **KB** and **K4B**. As concerns our work in this chapter, it will be possible to obtain the same result in tableau system.

As for tableau systems, Fitting gave a tableau system for **KB** in [5], but it does not have the analytic superformula property. On the other hand, Nguyen proposed a tableau system for **KB** in [26], in which auxiliary modal operator was introduced in order to obtain the analytic superformula property. Our idea of the auxiliary modal operator $\sharp$ is due to Nguyen. In the tableau system for **KB**, cut rules called analytic cut rules are applicable to formulas of the form $\Box A$ or $\neg\Box A$ occurring in a formula set, repeatedly. It is easily seen that there are possibilities loops cause. To avoid such loops, the auxiliary modal operator $\blacksquare$ is introduced in our $\mathcal{S}$**KB**. Therefore, $\mathcal{S}$**KB** gives us proofs in there is no loop caused by applications of cut rule.

In [10], Goré proposed a tableau system for **K4B**, which has analytic superformula property. In the tableau system for **K4B**, the following rules are included :

$$\frac{\Gamma, \neg\Box A}{\Gamma, \neg\Box A, \Box\neg\Box A}\ (5)$$

$$\frac{\Gamma, \Box A}{\Gamma, \Box A, A \ \mid\ \Gamma, \Box A, \neg A}\ (sfc\Box)$$

$$\frac{\Gamma, \neg\Box A}{\Gamma, \neg\Box A, A \ \mid\ \Gamma, \neg\Box A, \neg A}\ (sfc\Diamond)$$

It is easily seen that applications of $(sfc\Box)$ and $(sfc\Diamond)$ cause loops. Also in our $\mathcal{S}$**K4B**, to avoid such loops, $\blacksquare$ is introduced. Moreover, we can see that redundant applications of (5) cause loops. In $\mathcal{S}$**K4B**, to avoid such loops caused by applications of (5), $(\Box)_t$ and $(\Box)_s$ are designed so that all $\Box$–formulas occurring in the right hand side of lower sequents remain in the right hand side of upper sequent also after applications of $(\Box)_t$ or $(\Box)_s$. We note that the role of (5) is included in $(\Box)_t$ or $(\Box)_s$. In addition, As we see, it

is simpler to construct $\mathcal{S}\mathbf{K4B}$–model graphs than $\mathcal{S}\mathbf{KB}$–model graphs. Histories tells us the reason why we can construct $\mathcal{S}\mathbf{K4B}$–model graphs simpler than $\mathcal{S}\mathbf{KB}$–model graphs.

Chapter 5

Termination and upper bounds of proof–search procedures for K4, KB, K4B and S4

In this chapter, we will show termination of proof–search procedures determined by sequent systems for **K4**, **KB**, **K4B** and the tableau system $\mathcal{C}$**S4**+, and give upper bounds of them. Termination is one of the most important properties of proof–search procedures, and it guarantees that it is possible to estimate an upper bound of the number of applications of rules in proof–search.

Upper bound of a given proof–search procedure is a function f on the set of natural numbers such that for any l, $f(l)$ is the total number of applications of rules in proof–search in the worst case for any formula of length l. In other words, $f(l)$ gives us an upper bound for the number of required steps in proof–search for a given formula. In Section 5.2, we first introduce Mouri's sequent system for **K4**. Next, we will estimate an upper bound of the proof–search procedure determined by Mouri's sequent system for **K4** with an order which guarantees termination of the proof–search procedure. Three sections from Section 5.3 are devoted to termination and upper bounds of proof–search procedures determined by **K4**, **KB** and **K4B**, respectively.

5.1 Preliminaries

Let $*$ be any of $\wedge$, $\vee$ and $\supset$. For a formula A, we define the length $\ell(A)$ of A inductively as follows:

$\ell(p) = 1$ for any propositional variable p,
$\ell(A * B) = 1 + \ell(A) + \ell(B)$,
$\ell(\neg A) = 1 + \ell(A)$,
$\ell(\Box A) = 1 + \ell(A)$,
$\ell(\blacksquare A) = 1 + \ell(A)$,
$\ell(\sharp A) = 0$.

Note that the length of $\sharp A$ is 0. For a set (or multiset) Γ with distinct formulas $A_1, \cdots, A_n$. $\ell(\Gamma) = \ell(A_1) + \cdots + \ell(A_n)$.

Next, we introduce the notion of the *modal degree* $mdeg(A)$ and the *modal depth* $mdep(A)$ of a formula A. The modal degree and the modal depth are defined inductively as follows:

$mdeg(p) = 0$	$mdep(p) = 0$
$mdeg(A * B) = max\{mdeg(A), mdeg(B)\}$,	$mdep(A * B) = mdep(A) + mdep(B)$,
$mdeg(\neg A) = mdeg(A)$,	$mdep(\neg A) = mdep(A)$,
$mdeg(\Box A) = 1 + mdeg(A)$,	$mdep(\Box A) = 1 + mdep(A)$,
$mdeg(\blacksquare A) = 1 + mdeg(A)$,	$mdep(\blacksquare A) = 0$,
$mdeg(\sharp A) = 0$.	$mdep(\sharp A) = 0$.

For a set $\Gamma = \{A_1, \cdots, A_n\}$ with distinct formulas $A_1, \cdots, A_n$.

$$mdeg(\Gamma) = max\{mdeg(A_1), \cdots, mdeg(A_n)\},$$
$$mdep(\Gamma) = mdep(A_1) + \cdots + mdep(A_n).$$

For all positive integers $x_1, \cdots, x_n$ and $y_1, \cdots, y_n$, where $n \geq 1$, we define the lexicographic orders $\ll$ over n–tuples of positive integers as follows:

$$(x_1, \cdots, x_n) \ll (y_1, \cdots, y_n) \quad \textit{iff} \quad \text{for some } k \leq n \ ((\text{for all } i < k \ (x_i = y_i)) \text{ and } x_k < y_k).$$

By abuse of language, we use the same symbol $\ll$ for these orders. But we can distinguish them from each other by its context. Each of the lexicographic orders $\ll$ over over n–tuples is well–order.

Next, we define multisets. A *multiset* on over a set X is a function M from X to natural numbers. Intuitively, $M(a)$ is the number of copies of $a \in X$ in M. A multiset M is *finite* if there are only finitely many $a \in X$ such that $M(a) > 0$. We use standard set notation like $\{a, a, b\}$ as an abbreviation of the multiset $\{a \mapsto 2, b \mapsto 1, c \mapsto 0\}$ over the base set $X = \{a, b, c\}$. It will be obvious from the context if we refer to a set or multiset. As for further information on multisets, see [1].

For a formula A, let $Sub_{mul}(A)$ denote the multiset of consisting of all occurrences of subformulas of A.

Lemma 5.1.1 *For any formula A, $|Sub(A)| \leq \ell(A)$ and $|Sub_{mul}(A)| = \ell(A)$.*

Proof) Trivial. □

Lemma 5.1.2 *Let $l = \ell(A)$ for a given formula A. The upper bound of proof–search procedure determined by Wang's system is bounded by $2^l - 1$. In addition, the upper bound of the total number of applications of only static rules of $\mathcal{S}$**KB** or $\mathcal{S}$**K4B** is bounded by $\frac{3^l - 1}{2}$.*

Proof) See the tree in Figure 5.1. It expresses a tree generated by applications of the rules of Wang's system. Each node denotes a sequent. Each branch is generated by an application of a rule. We note that each node has two branches in the worst case. The order of applications of the rules of Wang's system is inessential.

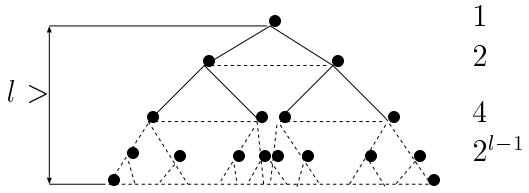


Figure 5.1: The total number of applications of the rules of Wang's system

Next, count the total number of branches generated by applications of the rules of Wang's system : It is at most

$$1 + 2 + 4 + \cdots + 2^{l-1} = 2^l - 1.$$

On the other hand, we note that the total number of leaves is 2^l.

Next, recall that in $\mathcal{S}$**KB** and $\mathcal{S}$**K4B**, each of $(\rightarrow \wedge)$, $(\vee \rightarrow)$ and $(\supset\rightarrow)$ has three upper sequent. By using the same argument as the case of Wang's system, we can obtain the following:

$$1 + 3 + 9 + \cdots + 3^{l-1} = \frac{3^l - 1}{2}.$$

We note that the total number of leaves is 3^l. □

We say that a proof (or a tableau) is *loop–free* if there is no repetition of the same sequent (or the same formula set) in any path starting from the end sequent (or the root) and ending at an initial sequent (or a leaf). Also, we call a sequent system (or a tableau system) which gives us only loop–free proofs (or tableaus), loop–free.

5.2 Termination and the upper bound of Mouri's sequent system for K4

We will discuss termination and the upper bound of the proof–search procedure determined by Mouri's sequent system for **K4** introduced in [24].

5.2.1 Mouri's sequent system for K4

In this section, we will first explain Mouri's sequent system for **K4** and then show that the proof–search procedure determined by it always terminates for any given formula. The sequent system for **K4** gives a proof–search procedure for **K4**, in the sense that it checks whether a given formula A is provable in **K4** or not and moreover gives us a proof of A if it is provable in **K4**. In addition, a counter–model for A is constructed by using information on failed proofs when the proof–search procedure fails to find a proof of A. (See [24] for the details.)

In this subsection, we follow the notation Mouri used in his paper [24]. We take $\wedge$, $\neg$ and $\Box$ for logical connectives. Formulas of the form $A \vee B$ and $A \supset B$ are regarded as abbreviations of $\neg(\neg A \wedge \neg B)$ and $\neg(A \wedge \neg B)$, respectively. Greek capital letters Γ, Σ, Π etc. denote (finite, possibly empty) multisets of formulas. The notation $\Box\Gamma$ denotes the multiset $\{\Box A | A \in \Gamma\}$.

Now, rules of Mouri's sequent system for **K4** are shown in Figure 5.2. It is slightly different but not in an essential way from the standard system for **K4**. Sequents of the sequent system are expressions of the form $\Gamma \to \Delta$ with $\langle \Box\Pi | \Box\Sigma \rangle$. We use histories like those of our tableau system $\mathcal{CS4}+$. (Note that both Γ and Δ are multisets of formulas.) The rules $(init)_s$ and $(init)_t$ substantially denote initial sequents. In oder to emphasize 'or'–branch, double lines are used in $(\Box)_s$ and $(\Box)_t$.

$$\frac{}{\Box\Gamma, p_1, \cdots, p_n \to q_1, \cdots, q_m, \Box\Delta \ \langle \Box\Gamma | \Box\Sigma \rangle}\ (init)_s (\Box\Delta \subseteq \Box\Sigma, \Box\Delta \neq \emptyset)$$

$$\frac{}{\Box\Gamma, p_1, \cdots, p_n \to q_1, \cdots, q_m \ \langle \Box\Pi | \Box\Sigma \rangle}\ (init)_t$$

$$\frac{\Gamma, A, B \to \Delta \ \langle \Box\Pi | \Box\Sigma \rangle}{\Gamma, A \wedge B \to \Delta \ \langle \Box\Pi | \Box\Sigma \rangle}\ (\wedge L) \quad \frac{\Gamma \to \Delta, A \ \langle \Box\Pi | \Box\Sigma \rangle \quad \Gamma \to \Delta, B \ \langle \Box\Pi | \Box\Sigma \rangle}{\Gamma \to \Delta, A \wedge B \ \langle \Box\Pi | \Box\Sigma \rangle}\ (\wedge R)$$

$$\frac{\Gamma \to A, \Delta \ \langle \Box\Pi | \Box\Sigma \rangle}{\Gamma, \neg A \to \Delta \ \langle \Box\Pi | \Box\Sigma \rangle}\ (\neg L) \qquad \frac{\Gamma, A \to \Delta \ \langle \Box\Pi | \Box\Sigma \rangle}{\Gamma \to \Delta, \neg A \ \langle \Box\Pi | \Box\Sigma \rangle}\ (\neg R)$$

$$\frac{\Box\Gamma, \Gamma \to A_1 \ \langle \Box\Gamma | \Box\Sigma, \Box\Theta \rangle \quad \cdots \quad \Box\Gamma, \Gamma \to A_m \ \langle \Box\Gamma | \Box\Sigma, \Box\Theta \rangle}{\Box\Gamma, p_1, \cdots, p_n \to \Box\Theta, \Box\Delta, q_1, \cdots, q_l \ \langle \Box\Gamma | \Box\Sigma \rangle}\ (\Box)_s$$
$$(\Box\Theta \equiv \Box A_1, \cdots, \Box A_m,\ \Box\Theta \cap \Box\Sigma = \emptyset,\ \Box\Delta \subseteq \Box\Sigma)$$

$$\frac{\Box\Gamma, \Gamma \to A_1 \ \langle \Box\Gamma | \Box\Theta \rangle \quad \cdots \quad \Box\Gamma, \Gamma \to A_m \ \langle \Box\Gamma | \Box\Theta \rangle}{\Box\Gamma, p_1, \cdots, p_n \to \Box\Theta, q_1, \cdots, q_l \ \langle \Box\Pi | \Box\Sigma \rangle}\ (\Box)_t$$
$$(\Box\Theta \equiv \Box A_1, \cdots, \Box A_m, \Box\Gamma \supsetneq \Box\Pi)$$

Figure 5.2: Mouri's sequent system for K4

In every application of $(init)_s$, $(init)_t$, $(\Box)_s$ and $(\Box)_t$, we need to check whether side conditions are satisfied or not, then we regard expressions like $\Box\Delta$ in side conditions not as multisets but as a set. For instance, both side of the inclusion of the condition $\Box\Delta \subseteq \Box\Sigma$ of $(\Box)_s$ is regarded as sets of formulas. That is why, in every application of $(\Box)_s$ and $(\Box)_t$, duplications of identical formulas are eliminated from both sides of upper sequents of $(\Box)_s$ and $(\Box)_t$.

As we can see, the subformula property holds in this system because it is a cut–free system. Mouri proved that his sequent system is sound and complete with respect to any **K4**–frame in [24]. 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 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. We can see from the form of each lower sequent of $(\Box)_s$ and $(\Box)_t$, that it is possible to apply each of $(\Box)_s$ and $(\Box)_t$ only when we cannot apply any of $(\wedge L)$, $(\wedge R)$, $(\neg L)$ and $(\neg R)$ any more. On the other hand, we can apply $(\wedge L)$, $(\wedge R)$, $(\neg L)$ and $(\neg R)$ in any order.

5.2.2 Termination of Mouri's sequent system for K4

We will show termination of the proof–search procedure determined by Mouri's sequent system. Although its termination is discussed already in his paper [24], we will give here an alternative proof of termination, which leads us to an estimation of its upper bound. 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.

Theorem 5.2.1 *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 proofs of A always terminates. Since the subformula property holds in the sequent system, any formula occurring in a proof of A, if exists, belongs to $Sub(A)$. Let $c = |Sub(A)|$. For any sequent $\Gamma \to \Delta\langle\Box\Pi|\Box\Sigma\rangle$ occurring in a proof of A, we define the degree $D(\Gamma \to \Delta\langle\Box\Pi|\Box\Sigma\rangle)$ of $\Gamma \to \Delta\langle\Box\Pi|\Box\Sigma\rangle$, which is a triple of natural numbers, as follows:

$$D(\Gamma \to \Delta\langle\Box\Pi|\Box\Sigma\rangle) = (c - |\Box\Pi|,\ c - |\Box\Sigma|,\ \ell(\Gamma \cup \Delta)).$$

We note that both $c - |\Box\Pi|$ and $c - |\Box\Sigma|$ are positive.

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:

	$c - \|\Box\Pi\|$	$c - \|\Box\Sigma\|$	$\ell(\Gamma \cup \Delta)$
$(\wedge L)$	$=$	$=$	$<$
$(\wedge R)$	$=$	$=$	$<$
$(\neg L)$	$=$	$=$	$<$
$(\neg R)$	$=$	$=$	$<$
$(\Box)_s$	$=$	$<$	$-$
$(\Box)_t$	$<$	$-$	$-$

The interpretation is obvious: $(\Box)_t$ and $(\Box)_s$ strictly decrease $c - |\Box\Pi|$ and $c - |\Box\Sigma|$, respectively. Others leave $c-|\Box\Pi|$ and $c-|\Box\Sigma|$ unchanged but strictly decrease $\ell(\Gamma \to \Delta)$. Thus, we can obtain our theorem.

5.2.3 Upper bound of Mouri's sequent system for K4

In estimating the upper bound 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 upper bound.

Now, we give an estimation of the upper bound of proof–search procedure determined by Mouri's sequent system for **K4** in terms of the length of a given formula A.

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 upper bound of the proof–search procedure. Then, taking the branching by $(\wedge R)$ into account, we will briefly discuss a more exact estimation of the upper bound.

Simple estimation

By analyzing the proof of termination of Mouri's sequent system for **K4**, 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. For a given formula A, let $f_{\mathbf{K4}}(A)$ denote the upper bound of the proof–search procedure determined by Mouri's sequent system for **K4**.

Theorem 5.2.2 *Let $l = \ell(A)$ for a given formula A. Then $f_{\mathbf{K4}}(A)$ is bounded by $2l^{l^2}$. In other words, the proof–search procedure determined by Mouri's sequent system for* **K4** *terminates within $2l^{l^2}$ steps.*

Proof) To show this, consider the degree of any possible sequents occurring in all proofs of A. Recall that the degree of sequent is as follows:

$$D(\Gamma \to \Delta\langle\Box\Pi|\Box\Sigma\rangle) = (c - |\Box\Pi|,\ c - |\Box\Sigma|,\ \ell(\Gamma \cup \Delta)\ ,$$

where $c = |Sub(A)|$. 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 $f_{\mathbf{K4}}(A)$ is the summation of them.

First, in order to estimate the number of applications of $(\Box)_s$ and $(\Box)_t$ in all possible proofs of A, we fix $\ell(\Gamma \cup \Delta)$. Here, see Figure 5.2.3. 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 5.2.3, 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 proofs to be searched. The tree shown in Figure 5.2.3 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 5.2.3 is $(c-1)c$. This means that the total number of applications of $(\Box)_s$ and $(\Box)_t$ on a path in a proof 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:

$$b\ +\ b^2 +\ \cdots +\ b^{(c-1)c} = \frac{b\{b^{(c-1)c} - 1\}}{b-1}\quad .$$

Next, we estimate the total number of $(\wedge L)$, $(\wedge R)$, $(\neg L)$ and $(\neg R)$. Let s be the total number of applications of the rules $(\wedge L)$, $(\wedge R)$, $(\neg L)$ 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 5.2.3 can be regarded also as possible timing of applications of $(\wedge L)$, $(\wedge R)$, $(\neg L)$ and $(\neg R)$. Since the total number of the nodes of the tree is:

$$1\ +\ b\ +\ b^2\ +\ \cdots +\ b^{(c-1)c} = 1 + \frac{b\{b^{(c-1)c} - 1\}}{b-1}\quad .$$

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

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

.

Therefore, the summation $f_{\mathbf{K4}}(A)$ of the total number of applications of the rules $(\Box)_s$ and $(\Box)_t$, and that of the rules $(\wedge L)$, $(\wedge R)$, $(\neg L)$ and $(\neg R)$ in the worst case can be

estimated at:

$$\begin{aligned} f_{\mathbf{K4}}(A) &\leq \frac{b\{b^{(c-1)c}-1\}}{b-1} + s \cdot \left(1 + \frac{b\{b^{(c-1)c}-1\}}{b-1}\right) \\ &= (s+1) \cdot \frac{b\{b^{(c-1)c}-1\}}{b-1} + s \\ &\leq (s+1) \cdot b\{b^{(c-1)c}-1\} + s \quad . \end{aligned}$$

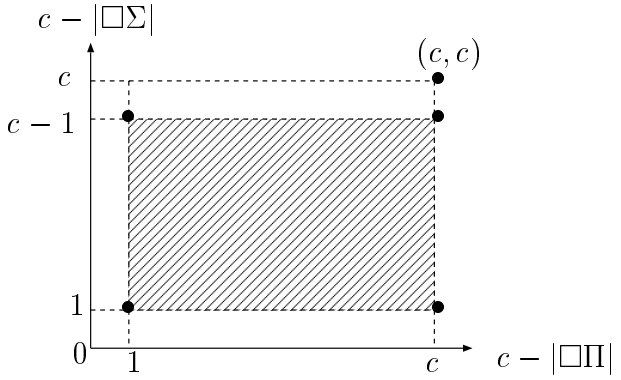


Figure 5.3: The total number of applications of $(\Box)_s$ and $(\Box)_t$

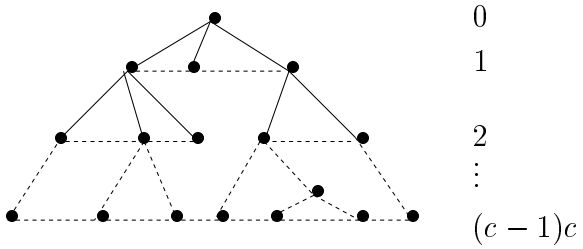


Figure 5.4: The search tree generated by 'or'–branches

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 will estimate s. All principal formulas of $(\wedge L)$, $(\wedge R)$, $(\neg L)$ 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 $(\wedge R)$ are ignored here. Taking these facts into account, s is estimated to be less than or equal to $|Sub_{mul}(A)|$. From Lemma 5.1.1, it follows that $c \leq l$, $b \leq l$ and $s \leq l$. Therefore,

$$\begin{aligned} f_{\mathbf{K4}}(A) &\leq (s+1) \cdot b\{b^{(c-1)c}-1\} + s \\ &\leq (l+1) \cdot l\{l^{(l-1)l}-1\} + l \\ &= l^{l^2-l+2} + l^{l^2-l+1} - l^2 \\ &\leq 2l^{l^2}. \end{aligned}$$

Thus, $f_{\mathbf{K4}}(A)$ can be bounded by $2l^{l^2}$. $\Box$

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

Another estimation
Next, taking branching by $(\wedge R)$ into account, we consider the upper bound again. We have to consider not only 'or–branch' by $(\Box)_s$ and $(\Box)_t$ but also branches by $(\wedge R)$. Let $\widehat{f}_{\mathbf{K4}}(A)$ denote the upper bound of this case. We can estimated $\widehat{f}_{\mathbf{K4}}(A)$ as follows:

Theorem 5.2.3 *Let $l = \ell(A)$ for a given formula A. Then $\widehat{f}_{\mathbf{K4}}(A)$ is bounded by $(2^l)^{2l^2}$.*

We will omit the proof of Theorem 5.2.3, since more complicated arguments are necessary, but we will give a strict estimation of the proof–search procedure determined by $\mathcal{S}$**KB** in the next section. We can give the proof of Theorem 5.2.3 similarly to it.

5.3 Termination and the upper bound of $\mathcal{S}$KB

In this and subsequent sections, we will show that similar results hold for both $\mathcal{S}$**KB** and $\mathcal{S}$**K4B**.

5.3.1 Termination of $\mathcal{S}$KB

Theorem 5.3.1 *The proof–search procedure determined by $\mathcal{S}$**KB** always terminates for any given sequent, no matter how rules are applied.*

Proof) To show termination of $\mathcal{S}$**KB**, we define the degree $D(\Gamma \to \Delta)$ of a sequent $\Gamma \to \Delta$, which is a triple of positive integers, as follows:

$$D(\Gamma \to \Delta) = (mdeg(\Gamma \cup \Delta), mdep(\Gamma \cup \Delta), \ell(\Gamma \cup \Delta)).$$

We can show that every application of rules strictly decreases the value of the degree D with respect to the lexicographic order $\ll$ over triples of positive integers. In other words, we can check the values of the degree of each of the upper sequent is less then that of the lower sequent. Here, we show only the cases for the $(cut)_L$ and $(\to \Box)_B$ in detail. Let $\Lambda = \{p_1, \cdots, p_n\}$ and $\Xi = \{q_1, \cdots, q_l\}$.

Case $(cut)_L$: We compare the modal degree and the modal depth of the left upper sequent and those of the lower sequent.

$$\begin{aligned} & mdeg(\Box\Gamma \cup \{\blacksquare A\} \cup \blacksquare\Pi \cup \Lambda \cup \Box\Delta \cup \blacksquare\Sigma \cup \sharp\Omega \cup \{\sharp A\} \cup \Xi \cup \{A\}) \\ = \; & mdeg(\Box\Gamma \cup \{\blacksquare A\} \cup \blacksquare\Pi \cup \Box\Delta \cup \blacksquare\Sigma) \\ = \; & mdeg(\Box\Gamma \cup \{\Box A\} \cup \blacksquare\Pi \cup \Box\Delta \cup \blacksquare\Sigma) \\ = \; & mdeg(\Box\Gamma \cup \{\Box A\} \cup \blacksquare\Pi \cup \Lambda \cup \Box\Delta \cup \blacksquare\Sigma \cup \sharp\Omega \cup \Xi). \end{aligned}$$

On the other hand,

$$\begin{aligned} & mdep(\Box\Gamma \cup \{\blacksquare A\} \cup \blacksquare\Pi \cup \Lambda \cup \Box\Delta \cup \blacksquare\Sigma \cup \sharp\Omega \cup \{\sharp A\} \cup \{A\} \cup \Xi) \\ = \; & mdep(\Box\Gamma \cup \Box\Delta \cup \{A\}). \end{aligned}$$

In addition,

$$\begin{aligned} & mdep(\Box\Gamma \cup \{\Box A\} \cup \blacksquare\Pi \cup \Lambda \cup \Box\Delta \cup \blacksquare\Sigma \cup \sharp\Omega \cup \Xi) \\ = \; & mdep(\Box\Gamma \cup \{\Box A\} \cup \Box\Delta) \\ = \; & 1 + mdep(\Box\Gamma \cup \{A\} \cup \Box\Delta). \end{aligned}$$

We can check this for the right upper sequent similarly.

Case $(\to \Box_B)$: Since $\Box\Omega_i \subseteq Sub(\Gamma \cup \{A_i\})$,

$$mdeg(\Box\Gamma \cup \{A_i\} \cup \Box\Omega_i) = mdeg(\Gamma \cup \{A_i\}).$$

On the other hand,

$$mdeg(\Box\Gamma \cup \Lambda \cup \{\Box A_1, \cdots, \Box A_m\} \cup \sharp\Omega \cup \Xi) = mdeg(\blacksquare\Gamma \cup \{\Box A_1, \cdots, \Box A_m\}).$$

Thus,

$$mdeg(\Gamma \cup \{A_i\}) < mdeg(\blacksquare\Gamma \cup \{\blacksquare A_1, \cdots, \blacksquare A_m\}).$$

Other cases can be checked similarly to the above. The following table shows that each application of the rules strictly decreases the triples with respect to the lexicographic order:

	$mdeg(\Gamma \cup \Delta)$	$mdep(\Gamma \cup \Delta)$	$\ell(\Gamma \cup \Delta)$
$(\wedge \to)$	$=$	$=$	$<$
$(\to \wedge)$	$=$	$=$	$<$
$(\vee \to)$	$=$	$=$	$<$
$(\to \vee)$	$=$	$=$	$<$
$(\supset \to)$	$=$	$=$	$<$
$(\to \supset)$	$=$	$=$	$<$
$(\neg \to)$	$=$	$=$	$<$
$(\to \neg)$	$=$	$=$	$<$
$(cut)_L$	$=$	$<$	–
$(cut)_R$	$=$	$<$	–
$(\Box)_B$	$<$	–	–

In this table, we can see the following : $(\Box)_B$ strictly decreases $mdeg(\Gamma \cup \Delta)$. On the other hand, $(cut)_L$ and $(cut)_R$ leave $mdeg(\Gamma \cup \Delta)$ unchanged but strictly decrease $mdep(\Gamma \cup \Delta)$. Others leave $mdeg(\Gamma \cup \Delta)$ and $mdep(\Gamma \cup \Delta)$ unchanged but strictly decrease $\ell(\Gamma \cup \Delta)$. Thus, we can obtain termination of $\mathcal{S}\mathbf{KB}$. $\Box$

Theorem 5.3.2 *The sequent system* $\mathcal{S}$**KB** *is a loop–free sequent system for* **KB**.

Proof) Take any proof of a sequent $\Gamma \to \Delta$ in $\mathcal{S}$**KB**, and take any path starting from $\Gamma \to \Delta$ and ending at an initial sequent. Then, from the above argument, the degree of a sequent in the path is always decreasing when we go up in the path. This implies that the same sequent never appears twice in it. Thus, $\mathcal{S}$**KB** is a loop–free system sequent for **KB**. □

5.3.2 Upper bound of $\mathcal{S}$KB

Here, we will discuss the upper bound of the proof–search procedure determined by $\mathcal{S}$**KB**. Let $f_{\mathcal{S}\mathbf{KB}}(A)$ denote the upper bound of the proof–search procedure determined by $\mathcal{S}$**KB**. We give a sketch of the estimation of $f_{\mathcal{S}\mathbf{KB}}(A)$. First, consider the order of applying of the rules of $\mathcal{S}$**KB** in proof–search. From the form of each lower sequent of $(\Box)_B$, we can see that it is impossible to apply $(\Box)_B$ until we cannot apply rules except $(\Box)_B$ any more. After an application of $(\Box)_B$, it will be possible to apply rules except $(\Box)_B$ again. During the proof–search by $\mathcal{S}$**KB**, this process is repeated as many times as possible. The total number of applications of rules except $(\Box)_B$ until next application of $(\Box)_B$ can bounded in Lemma 5.1.2. By counting the total number of branches generated by applications of the rules of $\mathcal{S}$**KB**, we can estimate $f_{\mathcal{S}\mathbf{KB}}(A)$.

Theorem 5.3.3 *Let* $l = \ell(A)$ *for a given formula* A. *Then* $f_{\mathcal{S}\mathbf{KB}}(A)$ *is bounded by* $l^{l+1} \cdot 3^{l(l+2)}$. *In other words, the proof–search procedure determined by* $\mathcal{S}$**KB** *terminates within* $l^{l+1} \cdot 3^{l(l+2)}$ *steps.*

Proof) Recall the order of applying rules of $\mathcal{S}$**KB** in proof–search. First, we apply static rules as many times as possible. Let s be the total number of applications of static rules in the worst case which are applied before the next application of $(\Box)_B$. In addition, let g be the total number of leaves generated after applying static rules s times. Next, consider an application of $(\Box)_B$. Since upper sequents of these rules must be interpreted as 'or'–branches, there will be many possibilities of the choice of proofs. Let b be the maximal number of possible 'or'–branches generated by of $(\Box)_B$. Therefore, we can obtain $b \cdot g$ nodes after the first application of $(\Box)_B$. After that, we repeat this process for each of $b \cdot g$ nodes. Let m be how many times this process repeat. The following table shows branches are generated by applications of rules of $\mathcal{S}$**KB**.

	The number of application static rules	The number of branches after application of static rules	The number of branches generated by application of $(\Box)_B$
1	s	g	$b \cdot g$
2	$s \cdot b \cdot g$	$b \cdot g^2$	$b^2 \cdot g^2$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
m	$s \cdot b^{m-1} \cdot g^{m-1}$	$s \cdot b^{m-1} \cdot g^m$	$b^m \cdot g^m$
–	$s \cdot b^m \cdot g^m$	–	–

Let f_1 be the total number of application of $(\Box)_B$ in the worst case and f_2 be the total number of application of static rules in the worst case. They can be estimated as follows:

$$\begin{aligned} f_1 &= h + h^2 + \cdots + h^m = \frac{h(h^m - 1)}{h - 1}, \\ f_2 &= s(1 + h + h^2 + \cdots + h^m) = s\left\{1 + \frac{h(h^m - 1)}{h - 1}\right\}, \end{aligned}$$

$$\begin{aligned} f_{\mathcal{S}\mathbf{KB}}(A) &\leq f_1 + f_2 \\ &= (s+1) \cdot \frac{h(h^m - 1)}{h - 1} + s \\ &\leq (s+1) \cdot h(h^m - 1) + s, \end{aligned}$$

where $h = b \cdot g$.

Now, it remains to estimate s, g, b and m in terms of the length of A. They can be estimated as follows:

$$\begin{aligned} s &\leq 3^l - 1 && \text{(from Lemma 5.1.2)}, \\ g &\leq 3^l && \text{(from Lemma 5.1.2)}, \\ b &\leq l, \\ h &= b \cdot g \leq l \cdot 3^l, \\ m &\leq mdeg(A) \leq l. \end{aligned}$$

Thus, the $f_{\mathcal{S}\mathbf{KB}}(A)$ of the total number of applications of the rules can be estimated at:

$$\begin{aligned} f_{\mathcal{S}\mathbf{KB}}(A) &\leq l \cdot t^2 \cdot (l \cdot t)^l - l \cdot t^2 + t - 1, \quad \text{where } t = 3^l \\ &\leq l \cdot t^2 \cdot (l \cdot t)^l \\ &= l^{l+1} \cdot 3^{l(l+2)}. \end{aligned}$$

□

5.4 Termination and the upper bound of $\mathcal{S}$K4B

5.4.1 Termination of $\mathcal{S}$K4B

Next, we discuss termination of $\mathcal{S}$**K4B**.

Theorem 5.4.1 *The proof–search procedure determined by* $\mathcal{S}$**K4B** *always terminates for any given sequent, no matter how rules are applied.*

Proof) For a sequent $\Gamma \to \Delta\ \langle \Box\Pi | \Box\Sigma \rangle$, we define the degree $D(\Gamma \to \Delta\ \langle \Box\Pi | \Box\Sigma \rangle)$ of $\Gamma \to \Delta\ \langle \Box\Pi | \Box\Sigma \rangle$, which is a quadruple of positive integers, as follows:

$$D(\Gamma \to \Delta\ \langle \Box\Pi | \Box\Sigma \rangle) = (c - |\Box\Pi|, c - |\Box\Sigma|, mdep(\Gamma \cup \Delta), \ell(\Gamma \cup \Delta)),$$

where $c = (\Gamma \cup \Delta)^{\mathcal{S}\mathbf{K4B}*}$. We note that $c - |\Box\Pi|$ and $c - |\Box\Sigma|$ are always non–negative.

With respect to the lexicographic order $\ll$ over quadruples of positive integers, every application of rules strictly decreases the value of the degree D, similarly to Theorem 5.3.1. The following table shows that each application of the rules strictly decreases the quadruples with respect to the lexicographic order:

	$c - \|\Box\Pi\|$	$c - \|\Box\Sigma\|$	$mdep(\Gamma \cup \Delta)$	$\ell(\Gamma \cup \Delta)$
$(\wedge \rightarrow)$	$=$	$=$	$=$	$<$
$(\rightarrow \wedge)$	$=$	$=$	$=$	$<$
$(\vee \rightarrow)$	$=$	$=$	$=$	$<$
$(\rightarrow \vee)$	$=$	$=$	$=$	$<$
$(\supset \rightarrow)$	$=$	$=$	$=$	$<$
$(\rightarrow \supset)$	$=$	$=$	$=$	$<$
$(\neg \rightarrow)$	$=$	$=$	$=$	$<$
$(\rightarrow \neg)$	$=$	$=$	$=$	$<$
$(cut)_L$	$=$	$=$	$<$	$-$
$(cut)_R$	$=$	$=$	$<$	$-$
(T)	$=$	$=$	$<$	$-$
$(\Box)_s$	$=$	$<$	$-$	$-$
$(\Box)_t$	$<$	$-$	$-$	$-$

We can see the following : $(\Box)_t$ strictly decreases $c - |\Box\Pi|$. On the other hands, $(\Box)_s$ leaves $c - |\Box\Pi|$ unchanged but strictly decreases $c - |\Box\Sigma|$. In addition, $(cut)_L$, $(cut)_R$ and (T) leave $c - |\Box\Pi|$ and $c - |\Box\Sigma|$ unchanged but strictly decrease $mdep(\Gamma \cup \Delta)$. Others leave $c - |\Box\Pi|$, $c - |\Box\Sigma|$ and $mdep(\Gamma \cup \Delta)$ unchanged but strictly decrease $\ell(\Gamma \cup \Delta)$. Thus, we can obtain termination of $\mathcal{S}\mathbf{K4B}$. □

Theorem 5.4.2 *The sequent system* $\mathcal{S}\mathbf{K4B}$ *is a loop–free sequent system for* **K4B**.

Proof) Similarly to the proof of Theorem 5.3.2. □

5.4.2 Upper bound of $\mathcal{S}$K4B

Theorem 5.4.3 *Let* $l = \ell(A)$ *for a given formula* A. *Then* $f_{\mathcal{S}\mathbf{K4B}}(A)$ *is bounded by* $3^l \cdot (1 + l \cdot 3^l)$. *In other words, the proof–search procedure determined by* $\mathcal{S}\mathbf{K4B}$ *terminates within* $3^l \cdot (1 + l \cdot 3^l)$ *steps.*

Proof) we can estimate $f_{\mathcal{S}\mathbf{K4B}}(A)$ similarly to the the upper bound of $\mathcal{S}\mathbf{KB}$. First, we apply static rules as many times as possible. Let s be the total number of applications of static rules in the worst case which are applied before the next application of $(\Box)_s$ or $(\Box)_t$. In addition, let g be the total number of leaves generated after applying static rules

s times. Next, consider an application of $(\Box)_s$ or $(\Box)_t$. Since upper sequents of these rules must be interpreted as 'or'–branches, there will be many possibilities of the choice of proofs. Let b be the maximal number of possible 'or'–branches generated by of $(\Box)_s$ or $(\Box)_t$. 1Therefore, we can obtain $b \cdot g$ nodes after the first application of $(\Box)_s$ or $(\Box)_t$. After that, we repeat this process for each of $b \cdot g$ nodes. Let m be how many times this process repeat. Here, recall the proof of completeness of $\mathcal{S}$**K4B**. We do not need to apply $(\Box)_t$ and $(\Box)_s$ more than 2 times. The following table shows how branches are generated by applications of rules of $\mathcal{S}$**K4B**.

	The number of application static rules	The number of branches after application of static rules	The number of branches generated by application of $(\Box)_s$ or $(\Box)_t$
1	s	g	$b \cdot g$
2	$s \cdot b \cdot g$	–	–

$$f_{\mathcal{S}\mathbf{K4B}}(A) \ \leq \ s + b \cdot g + s \cdot b \cdot g = s + (s+1) \cdot b \cdot g.$$

Now, it remains to estimate s, g and b in terms of the length of A. They can be estimated as follows:

$$\begin{array}{lll} s & \leq & 3^l - 1 \quad \text{(from Lemma 5.1.2)}, \\ g & \leq & 3^l \qquad \text{(from Lemma 5.1.2)}, \\ b & \leq & l, \end{array}$$

Thus, the $f_{\mathcal{S}\mathbf{K4B}}(A)$ of the total number of applications of the rules can be estimated at:

$$\begin{array}{rcl} f_{\mathcal{S}\mathbf{K4B}}(A) & \leq & s + (s+1) \cdot b \cdot g \\ & \leq & (3^l - 1) + 3^l \cdot l \cdot 3^l \\ & \leq & 3^l + 3^l \cdot l \cdot 3^l \\ & = & 3^l \cdot (1 + l \cdot 3^l). \end{array}$$

□

5.5 Termination and the upper bound of $\mathcal{C}$S4+

5.5.1 Termination of $\mathcal{C}$S4+

In this section, we will show termination of the proof–search procedure determined by $\mathcal{C}$**S4**+. We claim that the same node never appear on each branch from the root to any node in every $\mathcal{C}$**S4**+–tableau for any finite set Γ. In fact, we can show in the following that applications of tableau rules of $\mathcal{C}$**S4**+ never continue infinitely.

Theorem 5.5.1 *The proof–search procedure determined by* $\mathcal{C}$**S4**+ *always terminates for any given set of formulas, no matter how rules are applied.*

Proof) For a formula A, we first define the degree $deg(A)$ of A inductively as follows:

$$
\begin{aligned}
&deg(f) = 1,\\
&deg(p) = 1 \text{ for all } p \in \mathcal{P},\\
&deg(\neg A) = 1 + deg(A),\\
&deg(A \wedge B) = 2 + deg(A) + deg(B),\\
&deg(A \vee B) = 2 + deg(A) + deg(B),\\
&deg(A \supset B) = 2 + deg(A) + deg(B),\\
&deg(\Box A) = 3 \cdot deg(A),\\
&deg(\blacksquare A) = deg(A).
\end{aligned}
$$

For a finite set $\Gamma = \{A_1, \cdots, A_m\}$, where A_i and A_j are mutually distinct when $i \neq j$,

$$deg(\Gamma) = \sum_{i=1}^{m} deg(A_i).$$

Next, for any node $\Delta\langle\Box\Pi|\neg\Box\Sigma\rangle$ in any $\mathcal{CS4}$+–tableau for Γ, we define the degree $D(\Delta\langle\Box\Pi|\neg\Box\Sigma\rangle) \in \mathbf{N}^3$ of $\Delta\langle\Box\Pi|\neg\Box\Sigma\rangle$ as follows:

$$D(\Delta\langle\Box\Pi|\neg\Box\Sigma\rangle) = (|\Gamma^{\mathcal{CS4}+*}| - |\Box\Pi|,\ |\Gamma^{\mathcal{CS4}+*}| - |\neg\Box\Sigma|,\ deg(\Delta)).$$

We note that $|\Gamma^{\mathcal{CS4}+*}| - |\Box\Pi|$ and $|\Gamma^{\mathcal{CS4}+*}| - |\neg\Box\Sigma|$ are always non–negative.

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

	$\lvert\Gamma^{\mathcal{CS4}+*}\rvert - \lvert\Box\Pi\rvert$	$\lvert\Gamma^{\mathcal{CS4}+*}\rvert - \lvert\neg\Box\Sigma\rvert$	$deg(\Delta)$
$(\wedge)$	$=$	$=$	$<$
$(\vee)$	$=$	$=$	$<$
$(\supset)$	$=$	$=$	$<$
$(\neg\wedge)$	$=$	$=$	$<$
$(\neg\vee)$	$=$	$=$	$<$
$(\neg\supset)$	$=$	$=$	$<$
$(\neg\neg)$	$=$	$=$	$<$
$(T+)$	$=$	$=$	$<$
$(S4+)_s$	$=$	$<$	–
$(S4+)_t$	$<$	–	–

The table shows the following: $(S4+)_t$ strictly decreases $|\Gamma^{\mathcal{CS4}+*}| - |\Box\Pi|$. On the other hand, $(S4+)_s$ leaves $|\Gamma^{\mathcal{CS4}+*}| - |\Box\Pi|$ unchanged but strictly decreases $|\Gamma^{\mathcal{CS4}+*}| - |\neg\Box\Sigma|$. Others leave $|\Gamma^{\mathcal{CS4}+*}| - |\Box\Pi|$ and $|\Gamma^{\mathcal{CS4}+*}| - |\neg\Box\Sigma|$ unchanged but strictly decrease $deg(\Delta)$. Thus, we can obtain termination of $\mathcal{CS4}$+. □

Theorem 5.5.2 *The tableau system* $\mathcal{CS4}$+ *is a loop–free tableau system for* **S4**.

Proof) Similarly to the proof of Theorem 5.3.2. □

5.5.2 Upper bound of $\mathcal{C}$S4+

As we have seen already, termination of the proof–search procedure determined by $\mathcal{C}$**S4**+ is guaranteed by almost the same lexicographic order as Mouri's sequent system for **K4**. Therefore, upper bound of $\mathcal{C}$**S4**+ can be also estimated similarly to that of Mouri's sequent system. Actually, the present author has given a simple estimation of the upper bound of $\mathcal{C}$**S4**+ in [21] as follows:

Proposition 5.5.3 *Let $f_{\mathcal{C}\mathbf{S4}+}(\Gamma)$ denote the upper bound of $\mathcal{C}$**S4**+ for any finite set Γ. Then, $f_{\mathcal{C}\mathbf{S4}+}(\Gamma)$ can be estimated at $(2l)^{9l^2+3l+1} \cdot (2l+1) - 4l^2$, where $l = len(\Gamma)$. In other words, for the input Γ with the length of l, $\mathcal{C}$**S4**+ terminates in the steps of $(2l)^{9l^2+3l+1} \cdot (2l+1) - 4l^2$ at worst.*

5.6 Conclusion

In this chapter we discussed termination and the upper bound of each of proof–search procedures. Each of proof–search procedures determined by $\mathcal{S}$**KB**, $\mathcal{S}$**K4B** and $\mathcal{C}$**S4**+ has termination, that is, applications of rules of them never continue infinitely. In Section 5.2, we estimate the upper bound by using a certain order which guarantees termination of the proof–search procedure by Mouri's sequent system for **K**4. On the other hand, the upper bound of each of proof–search procedures determined by $\mathcal{S}$**KB** and $\mathcal{S}$**K4B** was also given.

5.7 Note

Proof–search procedures for **S4** are proposed also by Heuerding, Seyfried and Zimmermann [11] and Mouri [23]. Upper bound is not explicitly discussed in them.

On the other hand, the time complexity of decision procedure has been discussed in [33, 3, 19]. Since our result is different from time complexity, it could not be easy to compare our result with the results of [33, 3, 19]. It will be worth clarifying the relation between our result and them.

In [33, 3, 19], the time complexity of decision procedures has been discussed. In [33], it is shown that checking provability in **K** is in *EXPTIME–complete*. In [3], it is shown that Fitting's translation of **S4** to **KT** 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 [19], *NPTIME*–proof–search strategies for **K45** and **S5** are discussed.

On the other hand, in [17, 14, 16], the space complexity of decision procedures for modal logics are discussed.

Chapter 6

Proof–search procedure for temporal logic $\mathbf{K_t}$ based on sequent system

By utilizing techniques used in Chapters 3 and 4, we will introduce a sequent system for $\mathbf{K_t}$. Such a trial will give us a new approach to $\mathbf{K_t}$. However, the form of histories in our sequent system for $\mathbf{K_t}$ is so complicated that we need to make it simpler. It is a future work.

In this chapter, we give a proof–search procedure for temporal logic $\mathbf{K_t}$ based on sequent system. We will show its completeness by giving a way to construct counter–models. Our approach will also goes on tableau systems. This chapter is organized as follows: In Sections 6.1 and 6.2, we give a brief survey of temporal logic $\mathbf{K_t}$, its Kripke semantics for temporal logic introduced before sequent systems for $\mathbf{K_t}$. In Section 6.3, we introduce our a sequent system $\mathcal{S}\mathbf{K_t}$ for temporal logic, which gives us a proof–search procedure for temporal logic. In Sections 6.4 and 6.5, we introduce model graphs for $\mathcal{S}\mathbf{K_t}$ and show the completeness of $\mathcal{S}\mathbf{K_t}$ by using them. In Section 6.6, we show termination of our sequent system $\mathcal{S}\mathbf{K_t}$. In Section 6.7, some concluding remarks will be given.

6.1 Temporal logic $\mathbf{K_t}$

Temporal logics are widely used in computer science because reasoning about time is the most natural and intuitive. By possible world semantics, the flow of time is represented as s frame (W, R), where W is a set of *moments* of time and R is a binary relation on W. This semantics can easily represent reflexive, transitive, euclidean, symmetric and serial. Deferent restriction on R give other temporal logic. As for further information on temporal logics, see [30, 8, 7].

Here, we adopt two modal operators, called *necessity* operators, which are the future operator $[F]$ and the past operator $[P]$. These necessity operators are interpreted in the following:

$[F]A$ at all future times, A
$[P]A$ at all past times, A

Then, the *possibility* operator, denoted by $\langle F\rangle$ and $\langle P\rangle$, are $\neg[F]\neg$ and $\neg[P]\neg$, and represent "some time in the future" and "some time in the past", respectively. The minimal temporal logic $\mathbf{K_t}$ is the least normal modal logic containing the following axioms:

$$\begin{array}{ll} \mathbf{4F} & : \ [F]A \supset [F][F]A \\ \mathbf{4P} & : \ [P]A \supset [P][P]A \\ \mathbf{CF} & : \ A \supset [F]\langle P\rangle A \\ \mathbf{CP} & : \ A \supset [P]\langle F\rangle A \end{array}$$

Axioms **4F** and **4P** represent *transitivity* of time. On the other hand, axioms **CF** and **CP** represent the *conversion* of future and past.

A *Kripke frame* for temporal logics is a triple (W, R_F, R_P) where W is a non–empty set of possible worlds and R_F and R_P are binary relations on W, that is $R_F, R_P \subseteq W \times W$. In addition, let (W, R_F, R_P) be a Kripke frame and V be a mapping from propositional variables to 2^W, called a valuation on (W, R_F, R_P). A *Kripke model* for temporal logics is a quadruple (W, R_F, R_P, V). For a given Kripke model (W, R_F, R_P, V), a relation $\models$ between an element of W and a formula is defined inductively as follows:

$$\begin{array}{lll} w \models p & \textit{iff} & w \in V(p), \\ w \models A \wedge B & \textit{iff} & w \models A \text{ and } w \models B, \\ w \models A \vee B & \textit{iff} & w \models A \text{ or } w \models B, \\ w \models A \supset B & \textit{iff} & w \models A \text{ implies } w \models B, \\ w \models \neg A & \textit{iff} & w \not\models A, \\ w \models [F]A & \textit{iff} & \text{for all } w',\ wR_Fw' \text{ implies } w' \models A, \\ w \models [P]A & \textit{iff} & \text{for all } w',\ wR_Pw' \text{ implies } w' \models A. \end{array}$$

Proposition 6.1.1 *Let $(W, R_F, R_P, \models)$ be a model for temporal loigcs. Then the follwing holds.*

$$\begin{array}{lll} (1) \ \models [F]A \supset [F][F]A & \textit{iff} & \forall u, v, w(uR_Fv \text{ and } vR_Fw \text{ imply } uR_Fw), \\ (2) \ \models [P]A \supset [P][P]A & \textit{iff} & \forall u, v, w(uR_Pv \text{ and } vR_Pw \text{ imply } uR_Pw), \\ (3) \ \models A \supset [F]\langle P\rangle A & \textit{iff} & \forall u, v(uR_Fv \text{ implies } vR_Pu), \\ (4) \ \models A \supset [P]\langle F\rangle A & \textit{iff} & \forall u, v(uR_Pv \text{ implies } vR_Fu). \end{array}$$

By (3) and (4), R_F is a converse relation of R_P and thus it is enough to take either of R_F and R_P. Hence, by taking $R_F = R_P$, we can see from axioms of $\mathbf{K_t}$ that the form (W, R_T) suffices for a frame for $\mathbf{K_t}$, where R_T is transitive. Then, for any formula A, $[F]A$ and $[P]A$ are defined as follows:

$$\begin{array}{lll} w \models [F]A & \text{iff} & \text{for all } w,\ wR_Tw' \text{ implies } w' \models A, \\ w \models [P]A & \text{iff} & \text{for all } w',\ w'R_Tw \text{ implies } w' \models A. \end{array}$$

6.2 Sequent systems for $\mathbf{K_t}$

A sequent system for $\mathbf{K_t}$ is devised by Nishimura in [27], though it lacks cut elimination. This means that the subformula property does not hold in Nishimura's sequent system. For example, it is impossible to show that a sequent $p \to [F]\neg[P]\neg p$ is provable without any application of cut rule.

$$\dfrac{\dfrac{\dfrac{[P]\neg p \to [P]\neg p}{\to [P]\neg p, \neg[P]\neg p}}{\to \neg p, [F]\neg[P]\neg p} \quad \dfrac{p \to p}{\neg p, p \to}}{p \to [F]\neg[P]\neg p}$$

In [18], Maruyama, Tojo and Ono proposed sequent systems for temporal epistemic logics, which will be discussed in Section 6.8, by using Takano's method introduced in [34]. Let Σ, Π, Λ and Θ be sets of formulas. Rules of cut and for $[F]$ and $[P]$ of their systems are of the following form:

$$\dfrac{\Sigma \to \Lambda, A \quad A, \Pi \to \Theta}{\Sigma, \Pi \to \Lambda, \Theta}\ (AC)$$

where $A \in Sub(\Sigma \cup \Lambda \cup \Pi \cup \Theta)$

$$\dfrac{[F]\Sigma, \Sigma \to [P]\Lambda, [P]\Theta, A}{[F]\Sigma \to [P]\Lambda, \Theta, [F]A}\ (T1)'$$

where $[P]\Theta \subseteq Sub(\Sigma \cup \Lambda \cup \{A\})$

$$\dfrac{[P]\Sigma, \Sigma \to [F]\Lambda, [F]\Theta, A}{[P]\Sigma \to [F]\Lambda, \Theta, [P]A}\ (T2)'$$

where $[F]\Theta \subseteq Sub(\Sigma \cup \Lambda \cup \{A\})$

Though cut elimination theorem does not hold in these systems, they have the subformula property. Still, from our standpoint, there are some deficiencies in them, since cut rule amy be applied redundantly. Thus, it will be a challenge problem to introduce a "loop–free" sequent system for $\mathbf{K_t}$.

6.3 Our sequent system $\mathcal{S}\mathbf{K_t}$ for $\mathbf{K_t}$

First, we define *list*. Let e be an arbitrary element. List L is of the following form:

$$\begin{array}{rcll} L & := & [\,] & \text{(empty list)} \\ & | & e :: L & \text{(concatenation)} \end{array}$$

We take $[x, y, \cdots]$ as the abbreviations $x :: y :: \cdots :: [\,]$.

Instead of $\blacksquare$ for $\Box$, we introduce auxiliary symbols $\overline{[F]}$ and $\overline{[P]}$. Then, $\Box$ and $\overline{\Box}$ denote any of $[F]$ and $[P]$, and any of $\overline{[F]}$ and $\overline{[P]}$, respectively. Also, for a finite set Γ, $\Box\Gamma$ and $\overline{\Box}\Gamma$ denote $\{\Box A \mid A \in \Gamma\}$ and $\{\overline{\Box}A \mid A \in \Gamma\}$, respectively. A history h is an expression of the following form:

$$\begin{array}{rcll} h & := & \langle \epsilon \mid \epsilon \parallel \epsilon \mid \epsilon \rangle^0_C & \text{(initial history)} \\ & \mid & \langle [F]\Pi_1 \mid [P]\Pi_2 \parallel [F]\Sigma_1 \mid [P]\Sigma_2 \rangle^r_F & \text{(future history)} \\ & \mid & \langle [P]\Pi_2 \mid [F]\Pi_1 \parallel [P]\Sigma_2 \mid [F]\Sigma_1 \rangle^r_P & \text{(past history)} \end{array}$$

where r is natural number. Let $\mathcal{H}$ denote a sequence of histories, which is defined inductively as follows:

$$\begin{array}{rcll} \mathcal{H} & := & h :: [\,] & \\ & \mid & \langle [F]\Pi_1 \mid [P]\Pi_2 \parallel [F]\Sigma_1 \mid [P]\Sigma_2 \rangle^r_F :: \mathcal{H} & \text{(future concatenation)} \\ & \mid & \langle [P]\Pi_2 \mid [F]\Pi_1 \parallel [P]\Sigma_2 \mid [F]\Sigma_1 \rangle^r_P :: \mathcal{H} & \text{(past concatenation)} \end{array}$$

We call $\mathcal{H}$ a *history path*. Note that any history path cannot be the empty list. Rules of our sequent system, called $\mathcal{S}\mathbf{K_t}$, except transitional rules is given in Figure 6.1. We call each of $(\wedge \to)$, $(\to \wedge)$, $(\vee \to)$, $(\to \vee)$, $(\supset\to)$, $(\to\supset)$, $(\neg \to)$, $(\to \neg)$, $(cut)_L$ and $(cut)_R$ a *static rule*, and each of (Fsym), (Fforce), (Ftra), (Fturn), (Fnew), (Psym), (Pforce), (Ptra), (Pturn) and (Pnew) a *transitional rule*.

initial sequent

$$\Gamma, A \to A, \Delta \ \mathcal{H}$$

static rule

$$\frac{\Gamma, A, B \to \Delta \ \mathcal{H}}{\Gamma, A \wedge B \to \Delta \ \mathcal{H}} \ (\wedge \to) \qquad \frac{\Gamma \to \Delta, A, B \ \mathcal{H}}{\Gamma \to \Delta, A \vee B \ \mathcal{H}} \ (\to \vee) \qquad \frac{\Gamma, A \to \Delta, B \ \mathcal{H}}{\Gamma \to \Delta, A \supset B \ \mathcal{H}} \ (\to \supset)$$

$$\frac{\Gamma \to \Delta, A \ \mathcal{H}}{\Gamma, \neg A \to \Delta \ \mathcal{H}} \ (\neg \to) \qquad \frac{\Gamma, A \to \Delta \ \mathcal{H}}{\Gamma \to \Delta, \neg A \ \mathcal{H}} \ (\to \neg)$$

$$\frac{\Gamma \to \Delta, A, B \ \mathcal{H} \quad \Gamma, B \to \Delta, A \ \mathcal{H} \quad \Gamma, A \to \Delta, B \ \mathcal{H}}{\Gamma \to \Delta, A \wedge B \ \mathcal{H}} \ (\to \wedge)$$

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

$$\frac{\Gamma, A, B \to \Delta \ \mathcal{H} \quad \Gamma, B \to \Delta, A \ \mathcal{H} \quad \Gamma \to \Delta, A, B \ \mathcal{H}}{\Gamma, A \supset B \to \Delta \ \mathcal{H}} \ (\supset \to)$$

$$\frac{\Box\Gamma, \overline{\Box}A, \overline{\Box}\Phi, \Lambda \to \Box\Delta, \overline{\Box}\Psi, \sharp\Omega, \sharp A, \Xi, A \ \mathcal{H} \quad A, \Box\Gamma, \overline{\Box}A, \overline{\Box}\Phi, \Lambda \to \Box\Delta, \overline{\Box}\Psi, \sharp\Omega, \Xi \ \mathcal{H}}{\Box\Gamma, \Box A, \overline{\Box}\Phi, \Lambda \to \Box\Delta, \overline{\Box}\Psi, \sharp\Omega, \Xi \ \mathcal{H}} \ (cut)_L$$

$$\frac{\Box\Gamma, \overline{\Box}\Phi, \Lambda \to \Box\Delta, \overline{\Box}A, \overline{\Box}\Psi, \sharp\Omega, \sharp A, \Xi, A \ \mathcal{H} \quad A, \Box\Gamma, \overline{\Box}\Phi, \Lambda \to \Box\Delta, \overline{\Box}A, \overline{\Box}\Psi, \sharp\Omega, \Xi \ \mathcal{H}}{\Box\Gamma, \overline{\Box}\Phi, \Lambda \to \Box\Delta, \Box A, \overline{\Box}\Psi, \sharp\Omega, \Xi \ \mathcal{H}} \ (cut)_R$$

$$\Lambda \equiv p_1, \cdots, p_n, \ \Xi \equiv q_1, \cdots, q_m$$

Figure 6.1: Sequent system for $\mathbf{K_t}$: $\mathcal{S}\mathbf{K_t}$

Now let us explain our transitional rules. Transitional rules are of the following form:

$$\frac{S_1^F\mathcal{H}_1^F \quad \cdots \quad S_n^F\mathcal{H}_n^F \qquad\qquad S_1^P\mathcal{H}_1^P \quad \cdots \quad S_m^P\mathcal{H}_m^P}{S\mathcal{H}}$$

where $i, j \geq 0$. Suppose that S is the following form:

$$p_1, \cdots, p_l, \overline{[F]}\Gamma_1, \overline{[P]}\Gamma_2 \to \overline{[F]}\Delta_1, \overline{[P]}\Delta_2, \sharp\Omega, q_1, \cdots, q_k.$$

We define upper sequents $(S_i^F\mathcal{H}_i^F)$, called *F–moments*, and $(S_j^P\mathcal{H}_j^P)$, called *P–moments*, depending on the form of $\mathcal{H}$.

First, we show conditions of transitional rules. Every formula and set appearing in conditions will be clear in the definition of transitional rules. Conditions (1), (2), (3) and (4) are given as follows:

(1) : $\Omega_i \subseteq \Omega$,
$[P]\Omega_i \subseteq Sub(\Gamma_1 \cup \Delta_2 \cup \{A_i\})$,
$[P](\Omega - \Omega_i) \cap Sub(\Gamma_1 \cup \Delta_2 \cup \{A_i\}) = \emptyset$

(2) : $\Omega_j \subseteq \Omega$,
$[F]\Omega_j \subseteq Sub(\Gamma_2 \cup \Delta_1 \cup \{B_j\})$,
$[F](\Omega - \Omega_j) \cap Sub(\Gamma_2 \cup \Delta_1 \cup \{B_j\}) = \emptyset$

(3) : for all $[F]A_i \in [F]\Delta_1''$,
$[F]A_i \in [F]\widehat{\Sigma}_1$,
$([P]\Omega_i \cup [P]\Delta_2) \subseteq [P]\widehat{\Sigma}_2$,
where Ω_i satisfies (1) for each A_i such that $[F]A_i \in [F]\Delta_1''$

(4) : for all $[P]B_j \in [P]\Delta_2''$,
$[P]B_j \in [P]\widehat{\Sigma}_2$,
$([F]\Omega_j \cup [F]\Delta_1) \subseteq [F]\widehat{\Sigma}_1$,
where Ω_j satisfies (2) for each B_j such that $[P]B_j \in [P]\Delta_2''$

The condition (1) says that Ω_i is determined by Γ_1, Δ_2 and A_i, and it is guaranteed that Ω_i is maximal by the last condition of (1). Hence, Ω_i is uniquely determined. Also about (2), similarly to (1). The condition (3) guarantees that, after an application of transitional rule, all formulas occurring in an upper sequent are included in a history. About (4), similarly to (3). We note also that Ω_i in (3) and Ω_j in (4) are determined by A_i and B_j, respectively.

1. **Case** $\mathcal{H} = [\langle \epsilon \mid \epsilon \mid\mid \epsilon \mid \epsilon \rangle^0_C]$:

F–moment :

(a) **Case** $\overline{[F]}\Delta_1 = \emptyset$: no F–moment

(b) **Case** $\overline{[F]}\Delta_1 \neq \emptyset$: Let $\overline{[F]}\Delta_1 = \{\overline{[F]}A_1, \cdots, \overline{[F]}A_n\}$.

i. **Case** $\overline{[F]}\Gamma_1 = \overline{[P]}\Gamma_2 = \emptyset$:

$$\left.\begin{array}{rcl} S^F_i & = & [F]\Gamma_1, \Gamma_1 \to A_i, [P]\Delta_2, [P]\Omega_i \\ \mathcal{H}^F_i & = & [\langle \epsilon \mid \epsilon \mid\mid [F]A_i \mid [P]\Delta_2, [P]\Omega_i \rangle^1_F] \end{array}\right) \text{(Fsym)}$$

where (1) $(1 \leq 1 \leq n)$.

ii. **Case** $\overline{[F]}\Gamma_1 \neq \emptyset$ or $\overline{[P]}\Gamma_2 \neq \emptyset$:

$$\left.\begin{array}{rcl} S^F_i & = & [F]\Gamma_1, \Gamma_1 \to A_i, [P]\Delta_2, [P]\Omega_i \\ \mathcal{H}^F_i & = & [\langle [F]\Gamma_1 \mid [P]\Gamma_2 \mid\mid [F]A_i \mid [P]\Delta_2, [P]\Omega_i \rangle^0_F] \end{array}\right) \text{(Ftra)}$$

where (1) $(1 \leq 1 \leq n)$.

P–moment :

(a) **Case** $\overline{[P]}\Delta_2 = \emptyset$: no P–moment

(b) **Case** $\overline{[P]}\Delta_2 \neq \emptyset$: Let $\overline{[P]}\Delta_2 = \{\overline{[P]}B_1, \cdots, \overline{[P]}B_m\}$.

i. **Case** $\overline{[F]}\Gamma_1 = \overline{[P]}\Gamma_2 = \emptyset$:

$$\left.\begin{array}{rcl} S^P_j & = & [P]\Gamma_2, \Gamma_2 \to [F]\Delta_1, [P]B_j, [F]\Omega_j \\ \mathcal{H}^P_j & = & [\langle \epsilon \mid \epsilon \mid\mid [P]B_j \mid [F]\Delta_1, [F]\Omega_j \rangle^1_P] \end{array}\right) \text{(Psym)}$$

where (2) $(1 \leq j \leq m)$.

ii. **Case** $\overline{[F]}\Gamma_1 \neq \emptyset$ or $\overline{[P]}\Gamma_2 \neq \emptyset$:

$$\left.\begin{array}{rcl} S^P_j & = & [P]\Gamma_2, \Gamma_2 \to [F]\Delta_1, [P]B_j, [F]\Omega_j \\ \mathcal{H}^P_j & = & [\langle [P]\Gamma_2 \mid [F]\Gamma_1 \mid\mid [P]B_j \mid [F]\Delta_1, [F]\Omega_j \rangle^0_P] \end{array}\right) \text{(Ptra)}$$

where (2) $(1 \leq j \leq m)$.

2. **Case** $\mathcal{H} = \langle [F]\Pi_1 \mid [P]\Pi_2 \parallel [F]\Sigma_1 \mid [P]\Sigma_2 \rangle^r_F :: \mathcal{H}'$:

F–moment :

(a) **Case** $\overline{[F]}\Delta_1 = \emptyset$: no F–moment

(b) **Case** $\overline{[F]}\Delta_1 \neq \emptyset$:

i. **Case** $[F]\Gamma_1 = [F]\Pi_1$ and $[P]\Gamma_2 = [P]\Pi_2$:

A. **Case** $\sharp B \in \sharp\Omega$ for some B such that $[P]B \in [P]\Gamma_2$: Let $\overline{[F]}\Delta_1 = \{\overline{[F]}A_1, \cdots, \overline{[F]}A_n\}$.

$$\left.\begin{array}{rcl} S^F_i & = & [F]\Gamma_1, \Gamma_1 \to A_i, [P]\Delta_2, [P]\Omega_i \\ \mathcal{H}^F_i & = & \langle [F]\Pi_1 \mid [P]\Pi_2 \parallel [F]A_i \mid [P]\Delta_2, [P]\Omega_i \rangle^0_F :: \mathcal{H} \end{array}\right) \quad \text{(Fforced)}$$

where (1) $(1 \leq n \leq n)$.

B. Otherwise : Let $\overline{[F]}\Delta_1 = \overline{[F]}\Delta'_1 \cup \overline{[F]}\Delta''_1$, where $[F]\Delta'_1 \cap [F]\Sigma_1 = \emptyset$ and $[F]\Delta''_1 \subseteq [F]\Sigma_1$. Let $\overline{[F]}\Delta'_1 = \{\overline{[F]}A_1, \cdots, \overline{[F]}A_n\}$.

$$\left.\begin{array}{rcl} S^F_i & = & [F]\Gamma_1, \Gamma_1 \to A_i, [P]\Delta_2, [P]\Omega_i \\ \mathcal{H}^F_i & = & \langle [F]\Pi_1 \mid [P]\Pi_2 \parallel [F]\Sigma_1, [F]A_i \mid [P]\Delta_2, [P]\Omega_i \rangle^{r+1}_F :: \mathcal{H}' \end{array}\right) \quad \text{(Fsym)}$$

where (1) $(1 \leq i \leq n)$.

ii. **Case** $[F]\Gamma_1 \supsetneq [F]\Pi_1$ or $[P]\Gamma_2 \subsetneq [P]\Pi_2$ (See Note 1) :
Let $\overline{[F]}\Delta_1 = \{\overline{[F]}A_1, \cdots, \overline{[F]}A_n\}$.

$$\left.\begin{array}{rcl} S^F_i & = & [F]\Gamma_1, \Gamma_1 \to A_i, [P]\Delta_2, [P]\Omega_i \\ \mathcal{H}^F_i & = & \langle [F]\Gamma_1 \mid [P]\Gamma_2 \parallel [F]A_i \mid [P]\Delta_2, [P]\Omega_i \rangle^0_F :: \mathcal{H} \end{array}\right) \quad \text{(Ftra)}$$

where (1) $(1 \leq i \leq n)$.

P–moment :

(a) **Case** $\overline{[P]}\Delta_2 = \emptyset$: no P–moment

(b) **Case** $\overline{[P]}\Delta_2 \neq \emptyset$:

i. **Case** some $\langle [P]\widehat{\Pi}_2 \mid [F]\widehat{\Pi}_1 \parallel [P]\widehat{\Sigma}_2 \mid [F]\widehat{\Sigma}_1 \rangle^{\hat{r}}_P \in \mathcal{H}$ such that $[F]\Gamma_1 \supseteq [F]\widehat{\Pi}_1$ and $[P]\Gamma_2 \subseteq [P]\widehat{\Pi}_2$: Let $\overline{[P]}\Delta_2 = \overline{[P]}\Delta'_2 \cup \overline{[P]}\Delta''_2$ such that $[P]\Delta''_2$ satisfying (4) and $\overline{[P]}\Delta'_2 \cap \overline{[P]}\Delta''_2 = \emptyset$. Let $\overline{[P]}\Delta'_2 = \{\overline{[P]}B_1, \cdots \overline{[P]}B_m\}$.

$$\left.\begin{array}{rcl} S^P_j & = & [P]\Gamma_2, \Gamma_2 \to [F]\Delta_1, [P]B_j, [F]\Omega_j \\ \mathcal{H}^P_j & = & \langle [P]\Gamma_2 \mid [F]\Gamma_1 \parallel [P]B_j \mid [F]\Delta_1, [F]\Omega_j \rangle^0_P :: \mathcal{H} \end{array}\right) \quad \text{(Pturn)}$$

where (2) $(1 \leq j \leq m)$.

ii. Otherwise : Let $\overline{[P]}\Delta_2 = \{\overline{[P]}B_1, \cdots, \overline{[P]}B_m\}$.

$$\left.\begin{array}{rcl} S^P_j & = & [P]\Gamma_2, \Gamma_2 \to [F]\Delta_1, [P]B_j, [F]\Omega_j \\ \mathcal{H}^P_j & = & \langle [P]\Gamma_2 \mid [F]\Gamma_1 \parallel [P]B_j \mid [F]\Delta_1, [F]\Omega_j \rangle^0_P :: \mathcal{H} \end{array}\right) \quad \text{(Pnew)}$$

where (2) $(1 \leq j \leq m)$.

3. **Case** $\mathcal{H} = \langle [P]\Pi_2 \mid [F]\Pi_1 \parallel [P]\Sigma_2 \mid [F]\Sigma_1\rangle^r_P :: \mathcal{H}'$:

F–moment :

(a) **Case** $\overline{[F]}\Delta_1 = \emptyset$: no F–moment

(b) **Case** $\overline{[F]}\Delta_1 \neq \emptyset$:

i. **Case** some $\langle [F]\widehat{\Pi}_1 \mid [P]\widehat{\Pi}_2 \parallel [F]\widehat{\Sigma}_1 \mid [P]\widehat{\Sigma}_2\rangle^{\hat{r}}_F \in \mathcal{H}$ such that $[F]\Gamma_1 \subseteq [F]\widehat{\Pi}_1$ and $[P]\Gamma_2 \supseteq [P]\widehat{\Pi}_2$: Let $\overline{[F]}\Delta_1 = \overline{[F]}\Delta'_1 \cup \overline{[F]}\Delta''_1$ such that $[F]\Delta''_1$ satisfying (3) and $\overline{[F]}\Delta'_1 \cap \overline{[F]}\Delta''_1 = \emptyset$. Let $\overline{[F]}\Delta'_2 = \{\overline{[F]}A_1, \cdots, \overline{[F]}A_n\}$.

$$\left.\begin{array}{rcl} S^F_i & = & [F]\Gamma_1, \Gamma_1 \to A_i, [P]\Delta_2, [P]\Omega_i \\ \mathcal{H}^F_i & = & \langle [F]\Gamma_1 \mid [P]\Gamma_2 \parallel [F]A_i \mid [P]\Delta_2, [P]\Omega_i\rangle^0_F :: \mathcal{H} \end{array}\right) \text{ (Fturn)}$$

where (1) $(1 \leq i \leq n)$.

ii. Otherwise : Let $\overline{[F]}\Delta_1 = \{\overline{[F]}A_1, \cdots, \overline{[F]}A_n\}$.

$$\left.\begin{array}{rcl} S^F_i & = & [F]\Gamma_1, \Gamma_1 \to A_i, [P]\Delta_2, [P]\Omega_i \\ \mathcal{H}^F_i & = & \langle [F]\Gamma_1 \mid [P]\Gamma_2 \parallel [F]A_i \mid [P]\Delta_2, [P]\Omega_i\rangle^0_F :: \mathcal{H} \end{array}\right) \text{ (Fnew)}$$

where (1) $(1 \leq n \leq n)$.

P–moment :

(a) **Case** $\overline{[P]}\Delta_2 = \emptyset$: no P–moment

(b) **Case** $\overline{[P]}\Delta_2 \neq \emptyset$

i. **Case** $[F]\Gamma_1 = [F]\Pi_1$ and $[P]\Gamma_2 = [P]\Pi_2$:

A. **Case** $\sharp B \in \sharp\Omega$ for some A such that $[F]A \in [F]\Gamma_1$: Let $\overline{[P]}\Delta_2 = \{\overline{[P]}B_1, \cdots, \overline{[P]}B_m\}$.

$$\left.\begin{array}{rcl} S^P_j & = & [P]\Gamma_2, \Gamma_2 \to [F]\Delta_1, B_j, [F]\Omega_j \\ \mathcal{H}^P_j & = & \langle [P]\Pi_2 \mid [F]\Pi_1 \parallel [P]B_j \mid [F]\Delta_1, [F]\Omega_j\rangle^0_P :: \mathcal{H} \end{array}\right) \text{ (Pforced)}$$

where (2) $(1 \leq j \leq m)$.

B. Otherwise : Let $\overline{[P]}\Delta_2 = \overline{[P]}\Delta'_2 \cup \overline{[P]}\Delta''_2$, where $[P]\Delta'_2 \cap [P]\Sigma_2 = \emptyset$ and $[P]\Delta''_2 \subseteq [P]\Sigma_2$. Let $\overline{[P]}\Delta'_2 = \{\overline{[P]}B_1, \cdots, \overline{[P]}B_m\}$.

$$\left.\begin{array}{rcl} S^P_j & = & [P]\Gamma_2, \Gamma_2 \to [F]\Delta_1, B_j, [F]\Omega_j \\ \mathcal{H}^P_j & = & \langle [P]\Pi_2 \mid [F]\Pi_1 \parallel [P]\Sigma_2, [P]B_j \mid [F]\Delta_1, [F]\Omega_j\rangle^{r+1}_P :: \mathcal{H}' \end{array}\right) \text{ (Psym)}$$

where (2) $(1 \leq j \leq m)$.

ii. **Case** $[F]\Gamma_1 \subsetneq [F]\Pi_1$ or $[P]\Gamma_2 \supsetneq [P]\Pi_2$ (See Note 2) :
Let $\overline{[P]}\Delta_2 = \{\overline{[P]}B_1, \cdots, \overline{[P]}B_m\}$.

$$\left.\begin{array}{rcl} S^P_j & = & [P]\Gamma_2, \Gamma_2 \to [F]\Delta_1, B_j, [F]\Omega_j \\ \mathcal{H}^P_j & = & \langle [P]\Gamma_2 \mid [F]\Gamma_1 \parallel [P]B_j \mid [F]\Delta_1, [F]\Omega_j\rangle^0_P :: \mathcal{H} \end{array}\right) \text{ (Ptra)}$$

where (2) $(1 \leq j \leq m)$.

Note 1 : The more future we go, the more number of $[F]$–formulas occurring in the left hand side of sequents and the less number of $[P]$–formulas occurring in the left hand side of sequents. Hence, when the above case does not hold, this case always holds.

Note 2 : The more past we go, the less number of $[F]$–formulas occurring in the left hand side of sequents and the more number of $[P]$–formulas occurring in the left hand side of sequents. Hence, when the above case does not hold, this case always holds.

In order to emphasize 'or'–branch, double lines are used in transitional rules (similarly to $\mathcal{C}$**S4**+ in Section 3.3).

For a given sequent $\Gamma \to \Delta$, the proof–search procedure determined by $\mathcal{S}\mathbf{K_t}$ goes similarly to that of $\mathcal{S}$**KB** by starting from $\Gamma \to \Delta\ [\langle \epsilon \mid \epsilon \parallel \epsilon \mid \epsilon \rangle^0_C]$. We note that a cut formula is always taken from one of $[F]$–formulas or $[P]$–formulas occurring in the lower sequent of $(cut)_L$ or $(cut)_R$. The subformula property in the strict sense does not hold in $\mathcal{S}\mathbf{K_t}$, but we can see that for a given sequent $\Gamma \to \Delta\ [\langle \epsilon \mid \epsilon \parallel \epsilon \mid \epsilon \rangle^0_C]$, any formula occurring in all proofs of $\Gamma \to \Delta\ [\langle \epsilon \mid \epsilon \parallel \epsilon \mid \epsilon \rangle^0_C]$ is in $(\Gamma \cup \Delta)^{\mathcal{S}\mathbf{KB}*}$, where

$$
\begin{array}{l}
(\Gamma \cup \Delta)^{\mathcal{S}\mathbf{K_t}*} = \\
\quad Sub(\Gamma \cup \Delta) \cup \\
\quad \{\overline{[F]}A \mid [F]A \in Sub(\Gamma \cup \Delta)\} \cup \\
\quad \{\overline{[P]}A \mid [P]A \in Sub(\Gamma \cup \Delta)\} \cup \\
\quad \{\sharp A \mid [F]A \in Sub(\Gamma \cup \Delta) \ \text{ or } \ [P]A \in Sub(\Gamma \cup \Delta)\}.
\end{array}
$$

6.4 Model graph for $\mathcal{S}\mathbf{K_t}$

For the purpose of giving a way to construct counter–models of formulas mot provable in $\mathcal{S}\mathbf{K_t}$, similarly as model graphs for $\mathcal{C}$**S4**+ in Section 3.4, we will define model graphs for $\mathcal{S}\mathbf{K_t}$. In constructing model graphs, we have to keep the following in mind:

- the axioms **4F** and **4P** makes the accessibility relation transitive, and
- the axioms **CF** and **CP** guarantees that the relations R_F and R_P are each the converse of the other, but not that R_T is not symmetric.

These make the construction of counter–models more complicated than those of **KB** and **K4B**.

Lemma 6.4.1 *Each static rule of* $\mathcal{S}\mathbf{K_t}$ *is invertible in* $\mathcal{S}\mathbf{K_t}$.

Proof) Similarly to Theorem 4.2.1. □

For sequents $\Gamma_1 \to \Delta_1\ \mathcal{H}$ and $\Gamma_2 \to \Delta_2\ \mathcal{H}$ with the same sequence of histories, if both $\Gamma_1 \subseteq \Gamma_2$ and $\Delta_1 \subseteq \Delta_2$, we say that $\Gamma_1 \to \Delta_1\ \mathcal{H}$ *is in* $\Gamma_2 \to \Delta_2\ \mathcal{H}$. Then, we write $\Gamma_1 \to \Delta_1\ \mathcal{H}\ \subseteq\ \Gamma_2 \to \Delta_2\ \mathcal{H}$ to denote it. A sequent $\Gamma \to \Delta\ \mathcal{H}$ is *closed with respect to*

a rule (r) if whenever (an instance of) the lower sequent of (r) is in $\Gamma \to \Delta\ \mathcal{H}$, so is (a corresponding instance of) at least one of the upper sequents of (r).

Lemma 6.4.2 (saturation lemma) *Suppose that a sequent* $\Gamma \to \Delta\ \mathcal{H}$ *is not provable in* $\mathcal{S}\mathbf{K_t}$. *Then, there is an effective procedure to construct some* $\mathcal{S}\mathbf{K_t}$*–saturated* $(\Gamma \to \Delta)^s\ \mathcal{H}$ *with* $\Gamma \to \Delta \subseteq (\Gamma \to \Delta)^s$, *where* $a((\Gamma \to \Delta)^s),\ s((\Gamma \to \Delta)^s) \subseteq (\Gamma \cup \Delta)^{\mathcal{S}\mathbf{K_t}*}$.

Proof) Similarly to Theorem 4.2.2. □

Lemma 6.4.3 *If a sequent* $\Gamma \to \Delta\ \mathcal{H}$ *is closed with respect to all static rules of* $\mathcal{S}\mathbf{K_t}$ *then* $\Gamma \to \Delta\ \mathcal{H}$ *is subformula–complete.*

Proof) Clear. □

Definition 6.4.4 *Let* W *be a nonempty set and* R_T *be a binary relation on* W, *that is* $R \subseteq W \times W$. *Then, a* $\mathbf{K_t}$**–model graph** *for a sequent* $\Gamma \to \Delta\ [\langle \epsilon \mid \epsilon \mid\mid \epsilon \mid \epsilon \rangle^0_C]$ *is a finite* $\mathbf{K_t}$*–frame* (W, R_T) *such that each member of* W *is a* $\mathcal{S}\mathbf{K_t}$*–saturated sequent* $w\mathcal{H}$ *with* $a(w), s(w) \subseteq (\Gamma \cup \Delta)^{\mathcal{S}\mathbf{K_t}*}$ *and*

1. $\Gamma \to \Delta \subseteq w_0$ *for some* $w_0\mathcal{H}_0 \in W$,
2. *if* $[F]A \in s(w_i)$ *then there exists some* $w_j\mathcal{H}_j \in W$ *with* $(w_i\mathcal{H}_i)R_T(w_j\mathcal{H}_j)$ *and* $A \in s(w_j)$,
3. *if* $(w_i\mathcal{H}_i)R_T(w_j\mathcal{H}_j)$ *and* $[F]A \in a(w_i)$ *then* $A \in a(w_j)$,
4. *if* $[P]A \in s(w_i)$ *then there exists some* $w_j\mathcal{H}_j \in W$ *with* $(w_j\mathcal{H}_j)R_T(w_i\mathcal{H}_i)$ *and* $A \in s(w_j)$,
5. *if* $(w_j\mathcal{H}_j)R_T(w_i\mathcal{H}_i)$ *and* $[P]A \in a(w_i)$ *then* $A \in a(w_j)$.

As long as no confusion occurs , we call a $\mathbf{K_t}$–model graph simply as a model graph in the following.

Lemma 6.4.5 (satisfiability lemma) *If* (W, R_T) *is a model graph for* $\Gamma \to \Delta\ \langle \epsilon \mid \epsilon \mid\mid \epsilon \mid \epsilon \rangle^0$ *then there exists a* $\mathbf{K_t}$*–model* $(W, R_T, \models)$ *such that* $w \not\models \Gamma_* \supset \Delta^*$ *for some* $w\mathcal{H} \in W$.

Proof) Similarly to Theorem 4.2.5. □

6.5 Completeness of $\mathcal{S}\mathbf{K_t}$

In this section, we will show soundness and completeness of $\mathcal{S}\mathbf{K_t}$. We first need to give an interpretation of $\overline{[F]}A, \overline{[P]}A$ and $\sharp A$. For a given Kripke model $(W, R_T, \models)$, we define an interpretation of $\overline{[F]}A, \overline{[P]}A$ and $\sharp A$ as follows:

$$\begin{array}{lll} w \models \overline{[F]}A & \textit{iff} & w \models [F]A, \\ w \models \overline{[P]}A & \textit{iff} & w \models [P]A, \\ w \models \sharp A & \textit{iff} & w \models A. \end{array}$$

This means that $\overline{[F]}$ and $\overline{[P]}$ have the same semantics as $[F]$ and $[P]$, respectively, and $\sharp$ has no semantical role, though each of them plays a different syntactical role. We can show soundness of $\mathcal{S}\mathbf{KB}$ straightforwardly.

Theorem 6.5.1 *The sequent system $\mathcal{S}\mathbf{K_t}$ is sound with respect to $\mathbf{K_t}$–frames.*

Proof) Similarly to the proof of Theorem 4.3.6. □

The sketch of proof of completeness of $\mathcal{S}\mathbf{K_t}$ goes similarly to those of $\mathcal{S}\mathbf{KB}$ and $\mathcal{S}\mathbf{K4B}$. That is, we will show completeness of $\mathcal{S}\mathbf{K_t}$ by showing how to construct a counter–model of a given unprovable sequent. When $(w_i\mathcal{H}_i)R_T(w_j\mathcal{H}_j)$, $(w_i\mathcal{H}_i)$ is called a *past successor* of $(w_j\mathcal{H}_j)$, and $(w_j\mathcal{H}_j)$ is called a *future successor* of $(w_i\mathcal{H}_i)$.

Lemma 6.5.2 *If a sequent $\Gamma \to \Delta$ $[\langle \epsilon \mid \epsilon \mid\mid \epsilon \mid \epsilon \rangle^0_C]$ is not provable in $\mathcal{S}\mathbf{K_t}$ then there exists a finite $\mathbf{K_t}$–model $(W, R_T, \models)$ such that $w_0\mathcal{H}_0 \not\models \Gamma_* \supset \Delta^*$ for some $w_0\mathcal{H}_0 \in W$.*

Proof) This proof goes as follows:

1. construction of model graphs,
2. termination of the construction,
3. justification of the fact that our construction gives $\mathbf{K_t}$–model graphs.

Suppose that $\Gamma \to \Delta\ \mathcal{H}_0$ is not provable in $\mathcal{S}\mathbf{K_t}$, where $\mathcal{H}_0$ denotes $[\langle \epsilon \mid \epsilon \mid\mid \epsilon \mid \epsilon \rangle^0_C]$. We give a way to construct a finite model graph (W, R_T) for $\Gamma \to \Delta\mathcal{H}_0$. Let $(W, R_T) = (\emptyset, \emptyset)$ at the beginning. The first step is to create $\mathcal{S}\mathbf{K_t}$–saturated $w_0\mathcal{H}_0$ with $\Gamma \to \Delta \subseteq w_0$, where $a(w_0), s(w_0) \subseteq (\Gamma \cup \Delta)^{\mathcal{S}\mathbf{K_t}*}$. This is possible by Lemma 6.4.2. We add $(w_0\mathcal{H}_0)$ to W.

We give a way of immediate future and immediate past successors of any node $w_k\mathcal{H}_k$ (possibly this is $w_0\mathcal{H}_0$). Since $w_k\mathcal{H}_k$ is not provable in $\mathcal{S}\mathbf{K_t}$,

$$\overline{[F]}\Gamma_1, \overline{[P]}\Gamma_2 \to \overline{[F]}\Delta_1, \overline{[P]}\Delta_2, \sharp\Omega\ \mathcal{H}_k$$

must not be provable in $\mathcal{S}\mathbf{K_t}$, where $\overline{[F]}\Gamma_1$ contains all $\overline{[F]}$–formulas in $a(w_k)$, $\overline{[P]}\Gamma_2$ contains all $\overline{[P]}$–formulas in $a(w_k)$, $\overline{[F]}\Delta_1$ contains all $\overline{[F]}$–formulas in $s(w_k)$, $\overline{[P]}\Delta_2$ contains all $\overline{[P]}$–formulas in $s(w_k)$ and $\sharp\Omega$ contains all $\sharp$–formulas in $s(w_k)$.

Our construction depends on the form of $\mathcal{H}_k$.

1. **Case** $\mathcal{H}_k = [\langle \epsilon \mid \epsilon \mid\mid \epsilon \mid \epsilon \rangle^0_C]$: In this case, $w_k\mathcal{H}_k$ denotes $w_0\mathcal{H}_0$. If both $\overline{[F]}\Delta_1$ and $\overline{[P]}\Delta_2$ are empty, then $(\{w_0\mathcal{H}_0\}, \emptyset)$ is the desired model graph since this is a $\mathbf{K_t}$–frame and all of properties of model graphs are satisfied.

   F–successor :

   (a) **Case** $\overline{[F]}\Delta_1 = \emptyset$: Do nothing. No F-successor.

   (b) **Case** $\overline{[F]}\Delta_1 \neq \emptyset$: Let $\overline{[F]}\Delta_1 = \{\overline{[F]}A_1, \cdots, \overline{[F]}A_n\}$.

   i. **Case** $\overline{[F]}\Gamma_1 = \overline{[P]}\Gamma_2 = \emptyset$: By using (Fsym), we can obtain

   $$\begin{aligned} u_i &= ([F]\Gamma_1, \Gamma_1 \to A_i, [P]\Delta_2, [P]\Omega_i)^s \\ \mathcal{H}^F_i &= [\langle \epsilon \mid \epsilon \mid\mid [F]A_i \mid [P]\Delta_2, [P]\Omega_i \rangle^1_F]. \end{aligned}$$

   Add $(u_i\mathcal{H}^F_i)$ to W. Put $(w_0\mathcal{H}_0)R_T(u_i\mathcal{H}^F_i)$.

   ii. **Case** $[F]\Gamma_1 \supsetneq [F]\Pi_1$ or $[P]\Gamma_2 \subsetneq [P]\Pi_2$: By using (Ftra), we can obtain

   $$\begin{aligned} u_i &= ([F]\Gamma_1, \Gamma_1 \to A_i, [P]\Delta_2, [P]\Omega_i)^s \\ \mathcal{H}^F_i &= [\langle [F]\Gamma_1 \mid [P]\Gamma_2 \mid\mid [F]A_i \mid [P]\Delta_2, [P]\Omega_i \rangle^0_F]. \end{aligned}$$

   Add $(u_i\mathcal{H}^F_i)$ to W. Put $(w_0\mathcal{H}_0)R_T(u_i\mathcal{H}^F_i)$.

   P–successor :

   (a) **Case** $\overline{[P]}\Delta_2 = \emptyset$: Do nothing. No P-successor.

   (b) **Case** $\overline{[P]}\Delta_2 \neq \emptyset$: Let $\overline{[P]}\Delta_2 = \{\overline{[P]}B_1, \cdots, \overline{[P]}B_m\}$.

   i. **Case** $\overline{[F]}\Gamma_1 = \overline{[P]}\Gamma_2 = \emptyset$: By using (Psym), we can obtain

   $$\begin{aligned} v_j &= ([P]\Gamma_2, \Gamma_2 \to [F]\Delta_1, [P]B_j, [F]\Omega_j)^s \\ \mathcal{H}^P_j &= [\langle \epsilon \mid \epsilon \mid\mid [P]B_j \mid [F]\Delta_1, [F]\Omega_j \rangle^1_P]. \end{aligned}$$

   Add $(v_j\mathcal{H}^P_j)$ to W. Put $(v_j\mathcal{H}^P_j)R_T(w_0\mathcal{H}_0)$.

   ii. **Case** $[F]\Gamma_1 \supsetneq [F]\Pi_1$ or $[P]\Gamma_2 \subsetneq [P]\Pi_2$: By using (Ptra), we can obtain

   $$\begin{aligned} v_j &= ([P]\Gamma_2, \Gamma_2 \to [F]\Delta_1, [P]B_j, [F]\Omega_j)^s \\ \mathcal{H}^P_j &= [\langle [P]\Gamma_2 \mid [F]\Gamma_1 \mid\mid [P]B_j \mid [F]\Delta_1, [F]\Omega_j \rangle^0_P]. \end{aligned}$$

   Add $(v_j\mathcal{H}^P_j)$ to W. Put $(v_j\mathcal{H}^P_j)R_T(w_0\mathcal{H}_0)$.

2. **Case** $\mathcal{H}_k = \langle [F]\Pi_1 \mid [P]\Pi_2 \parallel [F]\Sigma_1 \mid [P]\Sigma_2\rangle^r_F :: \mathcal{H}'_k$: If both $\overline{[F]}\Delta_1$ and $\overline{[P]}\Delta_2$ are empty then, do nothing. In this case, $w_k\mathcal{H}_k$ is an end–node.

F–successor :

(a) **Case** $\overline{[F]}\Delta_1 = \emptyset$: Do nothing. No F-successor.

(b) **Case** $\overline{[F]}\Delta_1 \neq \emptyset$:

i. **Case** $[F]\Gamma_1 = [F]\Pi_1$ and $[P]\Gamma_2 = [P]\Pi_2$:

A. **Case** $\sharp B \in \sharp\Omega$ for some B such that $[P]B \in [P]\Gamma_2$: Let $\overline{[F]}\Delta_1 = \{\overline{[F]}A_1, \cdots, \overline{[F]}A_n\}$. By using (Ffroced), we can obtain

$$\begin{aligned} u_i &= ([F]\Gamma_1, \Gamma_1 \to A_i, [P]\Delta_2, [P]\Omega_i)^s \\ \mathcal{H}^F_i &= \langle [F]\Pi_1 \mid [P]\Pi_2 \parallel [F]A_i \mid [P]\Delta_2, [P]\Omega_i\rangle^0_F :: \mathcal{H}_k. \end{aligned}$$

Add $(u_i\mathcal{H}^F_i)$ to W. Put $(w_k\mathcal{H}_k)R_T(u_i\mathcal{H}^F_i)$.

B. **Case** there is no such $\sharp B$: Let $\overline{[F]}\Delta_1 = \overline{[F]}\Delta'_1 \cup \overline{[F]}\Delta''_1$, where $[F]\Delta'_1 \cap [F]\Sigma_1 = \emptyset$ and $[F]\Delta''_1 \subseteq [F]\Sigma_1$. Let $\overline{[F]}\Delta'_1 = \{\overline{[F]}A_1, \cdots, \overline{[F]}A_n\}$. By using (Fsym), we can obtain

$$\begin{aligned} u_i &= ([F]\Gamma_1, \Gamma_1 \to A_i, [P]\Delta_2, [P]\Omega_i)^s \\ \mathcal{H}^F_i &= \langle [F]\Pi_1 \mid [P]\Pi_2 \parallel [F]\Sigma_1, [F]A_i \mid [P]\Delta_2, [P]\Omega_i\rangle^{r+1}_F :: \mathcal{H}'_k. \end{aligned}$$

Add $(u_i\mathcal{H}^F_i)$ to W. Put $(w_k\mathcal{H}_k)R_T(u_i\mathcal{H}^F_i)$.

Let $[F]\Gamma'_1$ be the set of all $[F]$–formulas in $a(u_i)$ and $[P]\Gamma'_2$ be the set of all $[P]$–formulas in $a(u_i)$. If $[F]\Gamma'_1 = [F]\Pi_1$, $[P]\Gamma'_2 = [P]\Pi_2$ and $r+1 \geq 2$ then for $w'\mathcal{H}'$ such that $w_k\mathcal{H}_k$ is an immediate past successor of $w_k\mathcal{H}_k$, put $(w_k\mathcal{H}_k)R_T(w'\mathcal{H}')$.

C. **Case** neither (Fforced) nor (Fsym) is applicable : In this case, $w_k\mathcal{H}_k$ is an end–node. If $r = 0$ then put $(w_k\mathcal{H}_k)R_T(w_k\mathcal{H}_k)$. If $r \geq 1$ then for $w'\mathcal{H}'$ such that $w_k\mathcal{H}_k$ is an immediate past successor of $w_k\mathcal{H}_k$, put $(w_k\mathcal{H}_k)R_T(w'\mathcal{H}')$.

ii. **Case** $[F]\Gamma_1 \supsetneq [F]\Pi_1$ or $[P]\Gamma_2 \subsetneq [P]\Pi_2$: Let $\overline{[F]}\Delta_1 = \{\overline{[F]}A_1, \cdots, \overline{[F]}A_n\}$. By using (Ftra), we can obtain

$$\begin{aligned} u_i &= ([F]\Gamma_1, \Gamma_1 \to A_i, [P]\Delta_2, [P]\Omega_i)^s \\ \mathcal{H}^F_i &= \langle [F]\Gamma_1 \mid [P]\Gamma_2 \parallel [F]A_i \mid [P]\Delta_2, [P]\Omega_i\rangle^0_F :: \mathcal{H}_k. \end{aligned}$$

Add $(u_i\mathcal{H}^F_i)$ to W. Put $(w_k\mathcal{H}_k)R_T(u_i\mathcal{H}_i)$.

P–successor :

(a) **Case** $\overline{[P]}\Delta_2 = \emptyset$: Do nothing. No P-successor.

(b) **Case** $\overline{[P]}\Delta_2 \neq \emptyset$:

i. **Case** some $\langle [P]\widehat{\Pi}_2 \mid [F]\widehat{\Pi}_1 \mid\mid [P]\widehat{\Sigma}_2 \mid [F]\widehat{\Sigma}_1\rangle^{\widehat{r}}_P \in \mathcal{H}_k$ such that $[F]\Gamma_1 \supseteq [F]\widehat{\Pi}_1$ and $[P]\Gamma_2 \subseteq [P]\widehat{\Pi}_2$: Let $\overline{[P]}\Delta_2 = \overline{[P]}\Delta'_2 \cup \overline{[P]}\Delta''_2$ such that $[P]\Delta''_2$ satisfying (4) and $\overline{[P]}\Delta'_2 \cap \overline{[P]}\Delta''_2 = \emptyset$.

A. **Case** $\overline{[P]}\Delta'_2 = \emptyset$: Let $\widehat{w}\widehat{\mathcal{H}} \in W$ be a node such that $\widehat{\mathcal{H}} = \langle [P]\widehat{\Pi}_2 \mid [F]\widehat{\Pi}_1 \mid\mid [P]\widehat{\Sigma}_2 \mid [F]\widehat{\Sigma}_1\rangle^{\widehat{r}}_P :: \widehat{\mathcal{H}}'$ for some $\widehat{\mathcal{H}}'$. (Such $\widehat{w}\widehat{\mathcal{H}}$ must be in W.) Let $\sharp\widehat{\Omega}$ be the set of all $\sharp$–formulas in $s(\widehat{w})$.

1) **Case** for all $\widehat{A}$ such that $[F]\widehat{A} \in [F]\widehat{\Pi}_1$, $\sharp\widehat{A} \notin \sharp\Omega$, and for all B such that $[P]B \in [P]\Gamma_2$, $\sharp B \notin \sharp\widehat{\Omega}$: Put $(\widehat{w}\widehat{\mathcal{H}})R_T(w_k\mathcal{H}_k)$.

2) **Case** Otherwise : Let $\overline{[P]}\Delta''_2 = \{\overline{[P]}B_1, \cdots, \overline{[P]}B_m\}$. By using (Pturn), we can obtain

$$\begin{array}{rcl} v_j & = & ([P]\Gamma_2, \Gamma_2 \to [F]\Delta_1, [P]B_j, [F]\Omega_j)^s \\ \mathcal{H}^P_j & = & \langle [P]\Gamma_2 \mid [F]\Gamma_1 \mid\mid [P]B_j \mid [F]\Delta_1, [F]\Omega_j\rangle^0_P :: \mathcal{H}_k. \end{array}$$

Add $(v_j\mathcal{H}^P_j)$ to W. Put $(v_j\mathcal{H}^P_j)R_T(w_k\mathcal{H}_k)$.

B. **Case** $\overline{[P]}\Delta'_2 \neq \emptyset$: Let $\overline{[P]}\Delta'_2 = \{\overline{[P]}B_1, \cdots \overline{[P]}B_m\}$. By using (Pturn), we can obtain

$$\begin{array}{rcl} v_j & = & ([P]\Gamma_2, \Gamma_2 \to [F]\Delta_1, [P]B_j, [F]\Omega_j)^s \\ \mathcal{H}^P_j & = & \langle [P]\Gamma_2 \mid [F]\Gamma_1 \mid\mid [P]B_j \mid [F]\Delta_1, [F]\Omega_j\rangle^0_P :: \mathcal{H}_k. \end{array}$$

Add $(v_j\mathcal{H}^P_j)$ to W. Put $(v_j\mathcal{H}^P_j)R_T(w_k\mathcal{H}_k)$.

ii. Otherwise : Let $\overline{[P]}\Delta_2 = \{\overline{[P]}B_1, \cdots, \overline{[P]}B_m\}$. By using (Pnew), we can obtain

$$\begin{array}{rcl} v_j & = & ([P]\Gamma_2, \Gamma_2 \to [F]\Delta_1, [P]B_j, [F]\Omega_j)^s \\ \mathcal{H}^P_j & = & \langle [P]\Gamma_2 \mid [F]\Gamma_1 \mid\mid [P]B_j \mid [F]\Delta_1, [F]\Omega_j\rangle^0_P :: \mathcal{H}_k. \end{array}$$

Add $(v_j\mathcal{H}^P_j)$ to W. Put $(v_j\mathcal{H}^P_j)R_T(w_k\mathcal{H}_k)$.

3. **Case** $\mathcal{H}_k = \langle [P]\Pi_2 \mid [F]\Pi_1 \mid\mid [P]\Sigma_2 \mid [F]\Sigma_1\rangle^r_P :: \mathcal{H}'_k$: Similarly to the above case.

Next we show that this construction terminates. As we can see, nodes are generated by saturation and applications of transitional rules. From proof of Lemma 6.4.2, we can see that $\mathcal{S}\mathbf{K_t}$–saturation always terminates. In Section 6.6, we will show that applications of transitional rules of $\mathcal{S}\mathbf{K_t}$ never continue infinitely.

Let W consist of all nodes generated in the construction. Let R_T be the transitive closure of R_T. To show that (W, R_T) gives a $\mathbf{K_t}$–model graph, we need the following property:

Property Let $(w\mathcal{H})R_T(w'\mathcal{H}')$ such that $(w'\mathcal{H}')$ is an immediate future successor of $(w\mathcal{H})$.

1. if $[F]A \in a(w)$ then $[F]A \in a(w')$ and $A \in a(w')$,

2. if $[P]A \in a(w')$ then $[P]A \in a(w)$ and $A \in a(w)$.

We show only Property 1, since we can show Property 2 similarly to Property 1. Suppose that $[F]A \in a(w)$.

- When $w'\mathcal{H}'$ is generated from $w\mathcal{H}$ by using (Fsym), (Fforced), (Ftra), (Fturn) or (Fnew), we can easily see that $A \in a(w')$. In addition, once $[F]$–formula appear in the left hand of a sequent, they disappear by any application of (Fsym), (Fforced), (Ftra), (Fturn) or (Fnew). Therefore, $[F]A \in a(w')$.

- When $w\mathcal{H}$ is generated by from $w'\mathcal{H}'$ by using application of (Psym), (Pforced), (Ptra), (Pturn) or (Pnew), since $[F]A \in a(w)$, $[F]A \in Sub(a(w') \cup s(w'))$ must hold. Since w' is subformula–complete, $[F]A \in a(w')$ or $[F]A \in s(w')$.

  Assuming that $[F]A \in s(w')$, since $w\mathcal{H}$ is generated from $w'\mathcal{H}'$ by using application of (Psym), (Pforced), (Ptra), (Pturn) or (Pnew), $[F]A \in s(w)$. Since $w\mathcal{H}$ is unprovable, this is a contradiction. Therefore, $[F]A \in a(w')$.

  On the other hand, since w' is closed with respect to $(cut)_L$, and is subformula–complete, $A \in a(w')$ or $A \in s(w')$. Assuming that $A \in s(w')$, $\sharp A \in s(w')$. Since $[F]A \in a(w)$, by application of (Psym), (Pforced), (Ptra), (Pturn) or (Pnew), $\sharp A$ become $[F]A \in s(w)$. Since $w\mathcal{H}$ is unprovable, this is a contradiction. Therefore, $A \in a(w')$.

Now, recall the property of $\mathbf{K_t}$–model graph.

1. Clear.

2. Clear from the construction.

3. Suppose that $(w_i\mathcal{H}_i)R_T(w_j\mathcal{H}_j)$ and $[F]A \in a(w_i)$.

   - When $(w_j\mathcal{H}_j)$ is a future successor of $(w_i\mathcal{H}_i)$ or $(w_i\mathcal{H}_i)$ is a past successor of $(w_j\mathcal{H}_j)$, by using Property 1. repeatedly, we can see $A \in a(w_j)$.
   - When $(w_j\mathcal{H}_j)R_T(w_i\mathcal{H}_i)$ also holds, from the construction, $[F]A \in a(w_j)$, $A \in a(w_j)$ hold.
   - When $w_j\mathcal{H}_j = w_i\mathcal{H}_i$, from the construction, $[F]A \in a(w_j)$ and $A \in a(w_j)$ hold.

4. Clear from the construction.

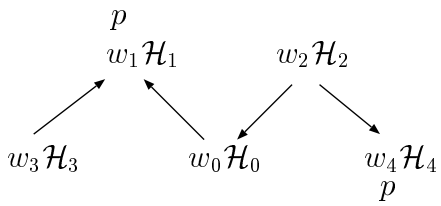
5. Similarly to 3 by using Property 2.

Therefore, we can obtain a $\mathbf{K_t}$–model such that $w_0\mathcal{H}_0 \not\models \Gamma_* \supset \Delta^*$ by using Lemma 6.4.5. □

Thus, we can obtain completeness of $\mathcal{S}\mathbf{K_t}$.

Theorem 6.5.3 *The sequent system $\mathcal{S}\mathbf{K_t}$ is complete with respect to $\mathbf{K_t}$–frames.*

Example 1 As an example, the following figure is a $\mathbf{K_t}$–model graph for $[F]p \to [F]\neg p \vee [P]p$.



$w_0 = \neg p, [F]p\overline{[F]}p \to [F]\neg p \vee [P]p, [F]\neg p, [P]p, \overline{[F]}\neg p, \overline{[P]}p, \sharp p, p$
$\mathcal{H}_0 = [\langle \epsilon \mid \epsilon \mid\mid \epsilon \mid \epsilon \rangle^0_C]$

$w_1 = p, [F]p, \overline{[F]}p \to \neg p, [P]p, \overline{[P]}p$
$\mathcal{H}_1 = [\langle [F]p \mid \epsilon \mid\mid [F]\neg p \mid [P]p \rangle^0_F]$

$w_2 = \neg p \to [F]\neg p, \overline{[F]}\neg p, p$
$\mathcal{H}_2 = [\langle \epsilon \mid [F]p \mid\mid [P]p \mid [F]\neg p \rangle^0_P]$

$w_3 = \to p$
$\mathcal{H}_3 = [\langle \epsilon \mid [F]p \mid\mid [P]p \mid \epsilon \rangle^0_P, \langle [F]p \mid \epsilon \mid\mid [F]\neg p \mid [P]p \rangle^0_F]$

$w_4 = p \to \neg p$
$\mathcal{H}_4 = [\langle \epsilon \mid \epsilon \mid\mid [F]\neg p \mid \epsilon \rangle^0_F, \langle \epsilon \mid [F]p \mid\mid [P]p \mid [F]\neg p \rangle^0_P]$

Figure 6.2: A counter–model for $[F]p \supset [F]\neg p \vee [P]p$

It is easily seen that $w_0[\langle \epsilon \mid \epsilon \mid\mid \epsilon \mid \epsilon \rangle^0_C] \not\models [F]p \supset [F]\neg p \vee [P]p$.

Example 2 The following is an proof of $p \to [F]\neg[P]\neg p\, [\langle \epsilon \mid \epsilon \mid\mid \epsilon \mid \epsilon \rangle^0_C]$. Let $\mathcal{H}_0$ denote $[\langle \epsilon \mid \epsilon \mid\mid \epsilon \mid \epsilon \rangle^0_C]$ and $\mathcal{H}_1$ denote $[\langle \epsilon \mid \epsilon \mid\mid [F]\neg[P]\neg p \mid [P]\neg p \rangle^1_F]$.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{[P]\neg p \to [P]\neg p\mathcal{H}_1}{\to \neg[P]\neg p, [P]\neg p\mathcal{H}_1}}{p, \overline{[P]}\neg p \to \overline{[F]}\neg[P]\neg p, \sharp\neg[P]\neg p, \sharp\neg p\mathcal{H}_0}}{\overline{[P]}\neg p, p \to \overline{[F]}\neg[P]\neg p, \sharp\neg[P]\neg p, \sharp\neg p, \neg p\mathcal{H}_0} \quad \dfrac{\neg p, \overline{[P]}\neg p, p \to \overline{[F]}\neg[P]\neg p, \sharp\neg[P]\neg p\mathcal{H}_0}{\overline{[P]}\neg p, p \to \overline{[F]}\neg[P]\neg p, \sharp\neg[P]\neg p, p\mathcal{H}_0}}{[P]\neg p, p \to \overline{[F]}\neg[P]\neg p, \sharp\neg[P]\neg p\mathcal{H}_0}}{p \to \overline{[F]}\neg[P]\neg p, \sharp\neg[P]\neg p, \neg[P]\neg p\mathcal{H}_0} \quad \dfrac{\dfrac{\dfrac{\dfrac{\to \neg[P]\neg p, [P]\neg p\mathcal{H}_1}{[P]\neg p \to [P]\neg p\mathcal{H}_1}}{p \to \overline{[F]}\neg[P]\neg p, \overline{[P]}\neg p, \sharp\neg p\mathcal{H}_0}}{p \to \overline{[F]}\neg[P]\neg p, \overline{[P]}\neg p, \sharp\neg p, \neg p\mathcal{H}_0} \quad \dfrac{p \to \overline{[F]}\neg[P]\neg p, \overline{[P]}\neg p, p\mathcal{H}_0}{\neg p, p \to \overline{[F]}\neg[P]\neg p, [P]\neg p\mathcal{H}_0}}{p \to \overline{[F]}\neg[P]\neg p, [P]\neg p\mathcal{H}_0}}{\neg[P]\neg p, p \to \overline{[F]}\neg[P]\neg p\mathcal{H}_0}}{p \to [F]\neg[P]\neg p\mathcal{H}_0}$$

6.6 Termination of $\mathcal{S}\mathbf{K_t}$

In this section, we show termination of proof–search of $\mathbf{K_t}$, that is applications of the rules of $\mathcal{S}\mathbf{K_t}$ never continue infinitely. Before that, we define the modal depth of $[F]A, [P]A, \overline{[F]}A$

and $\overline{[P]}A$ as follows:

$$\begin{aligned} mdep([F]A) &= 1 + mdep(A), \\ mdep([P]A) &= 1 + mdep(A), \\ mdep(\overline{[F]}A) &= 0, \\ mdep(\overline{[P]}A) &= 0. \end{aligned}$$

The length Ll of lists is defined inductively as follows:

$$\begin{aligned} L\ell([\,]) &= 0, \\ L\ell(e :: L) &= 1 + L\ell(L). \end{aligned}$$

Theorem 6.6.1 *The proof–search procedure determined by* $\mathcal{SK}_{\mathbf{t}}$ *always terminates for any given sequent, no matter how rules are applied.*

Proof) To show termination of $\mathcal{SK}_{\mathbf{t}}$, we need to define the degree of each sequent. Let T be any of C, F and P. The notation $\widehat{\Box}$ denotes $[F]$ when $\Box$ denotes $[P]$, otherwise $\widehat{\Box}$ denotes $[P]$. Then, history is of the following form:

$$\langle \Box\Pi_1 \mid \widehat{\Box}\Pi_2 \mid\mid \Box\Sigma_1 \mid \widehat{\Box}\Sigma_2 \rangle^r_T.$$

Then, history paths $\mathcal{H}$ can be written like the following form:

$$\begin{array}{l} \langle \Box\Pi_1 \mid \widehat{\Box}\Pi_2 \mid\mid \Box\Sigma_1 \mid \widehat{\Box}\Sigma_2 \rangle^r_T :: [\,] \text{ or} \\ \langle \Box\Pi_1 \mid \widehat{\Box}\Pi_2 \mid\mid \Box\Sigma_1 \mid \widehat{\Box}\Sigma_2 \rangle^r_T :: \mathcal{H}' \text{ for some history path } \mathcal{H}'. \end{array}$$

Let $c_1 = |(\Gamma \cup \Delta)^{\mathcal{SK}_{\mathbf{t}}*}|$ and $c_2 = 2 \cdot 2^{4 \cdot c_1}$. For a sequent $\Gamma \to \Delta\ \mathcal{H}$, we define the degree $D(\Gamma \to \Delta\ \mathcal{H})$, which is a quadruple of positive integers, as follows:

$$D(\Gamma \to \Delta\ \mathcal{H}) = (c_2 - L\ell(\mathcal{H}),\ c_1 - |\Box\Sigma_1|,\ mdep(\Gamma \cup \Delta),\ \ell(\Gamma \cup \Delta)),$$

where the leftmost history of $\mathcal{H}$ is $\langle \Box\Pi_1 \mid \widehat{\Box}\Pi_2 \mid\mid \Box\Sigma_1 \mid \widehat{\Box}\Sigma_2 \rangle^r_T$. We note that $c_2 - L\ell(\mathcal{H})$ and $c_1 - |\Box\Sigma_1|$ are positive.

Now, we can show that every application of rules of $\mathcal{SK}_{\mathbf{t}}$ decreases strictly the value of the degree D. Since the lexicographic order $\ll$ is well–order, we can obtain termination of proof–search. The following tables shows that each application of the rules in search decreases the quadruple with respect to the lexicographic order:

	$c_2 - L\ell(\mathcal{H})$	$c_1 - \lvert\Box\Sigma_1\rvert$	$mdep(\Gamma \cup \Delta)$	$\ell(\Gamma \cup \Delta)$
$(\wedge \rightarrow)$	$=$	$=$	$=$	$<$
$(\rightarrow \wedge)$	$=$	$=$	$=$	$<$
$(\vee \rightarrow)$	$=$	$=$	$=$	$<$
$(\rightarrow \vee)$	$=$	$=$	$=$	$<$
$(\supset \rightarrow)$	$=$	$=$	$=$	$<$
$(\rightarrow \supset)$	$=$	$=$	$=$	$<$
$(\neg \rightarrow)$	$=$	$=$	$=$	$<$
$(\rightarrow \neg)$	$=$	$=$	$=$	$<$
$(cut)_L$	$=$	$=$	$<$	$-$
$(cut)_R$	$=$	$=$	$<$	$-$
(Fsym)	$=$	$<$	$-$	$-$
(Ftra)	$<$	$-$	$-$	$-$
(Fforced)	$<$	$-$	$-$	$-$
(Fturn)	$<$	$-$	$-$	$-$
(Fnew)	$<$	$-$	$-$	$-$
(Psym)	$=$	$<$	$-$	$-$
(Ptra)	$<$	$-$	$-$	$-$
(Pforced)	$<$	$-$	$-$	$-$
(Pturn)	$<$	$-$	$-$	$-$
(Pnew)	$<$	$-$	$-$	$-$

From the above table, we can see the following:
Case (Fsym) : (Fsym) leave $c_2 - L\ell(\mathcal{H})$ unchanged but strictly decrease $c_1 - |\Box\Sigma_1|$.

Case (Ftra) : (Ftra) strictly decrease $c_2 - L\ell(\mathcal{H})$.

Case (Fforced) : Similarly to (Ftra).

Case (Fturn) : Similarly to (Ftra).

Case (Fnew) : Similarly to (Ftra).

(Psym), (Ptra), (Pforced), (Pturn) and (Pturn) are discussed similarly to the above. In addition, both $(cut)_L$ and $(cut)_R$ leave $c_2 - L\ell(\mathcal{H})$ and $c_1 - |\Box\Sigma_1|$ unchanged but strictly decrease $mdep(\Gamma \cup \Delta)$. Others leave $c_2 - L\ell(\mathcal{H})$, $c_1 - |\Box\Sigma_1|$ and $mdep(\Gamma \cup \Delta)$ unchanged but strictly decrease $\ell(\Gamma \cup \Delta)$. Thus, we can obtain our theorem. □

We note that termination of $\mathcal{S}\mathbf{K}_\mathbf{t}$ shows that the construction in Lemma 6.5.2 also terminates.

6.7 Conclusion

Our work in this chapter shows whether histories can accomplish the alternative loop–checking, or a possibility of histories to avoid loop–checking. By enhancing histories used

in Chapters 3 and 4. we gave a sequent system, called $\mathcal{S}\mathbf{K_t}$, for the temporal logic $\mathbf{K_t}$. We showed its completeness by giving a way of construction of counter–model like that given in the previous chapters. It also bring us the finite model property for $\mathbf{K_t}$. The sequent system for $\mathcal{S}\mathbf{K_t}$ gives us a proof–search procedure, which decides whether a given formula provable or not, and gives us a proof of if when it is provable. In addition, we can obtain a counter–model when is the given formula not provable. Termination of the proof–search procedure determined by $\mathcal{S}\mathbf{K_t}$ was shown. Since the proof–search procedure determined by $\mathcal{S}\mathbf{K_t}$ do not need to check the repetition of sequents, it do not cause inefficiency by that. Therefore, the proof–search procedure determined by $\mathcal{S}\mathbf{K_t}$ seems to be relatively feasible.

6.8 Note

Temporal logics are very useful for formalizing various notions which appear in computer science, and they have been investigated recently. Temporal logics are used by combining with other logics, for example epistemic logics, called temporal epistemic logics. The development of temporal epistemic logics have been paid much attention recently. In [18], Maruyama, Tojo and Ono introduced sequent systems for logics of belief and knowledge for multi–agent models with $[F]$ and $[P]$ as a temporal operators. The proof–search procedure determined by each of them needs to check the repetition of sequents. Termination of their sequent systems has not discussed explicitly. In [36] M. Wooldridge, C. Dixon and M. Fisher introduced tableau systems for temporal epistemic logics with the unary "next" operator $\bigcirc$ and the binary "until" operator $\mathcal{U}$ as temporal operators. In the paper, they gave a model–search procedure for these logics, which constructs a model for a given formula when it is satisfiable.

Chapter 7

Related works

In this chapter, we refer related works. The dependency relations between our works and related works are shown in Figure 7.1. Each number of the arrows in the figure denotes a detailed road map to our sequent system $\mathcal{S}\mathbf{K_t}$.

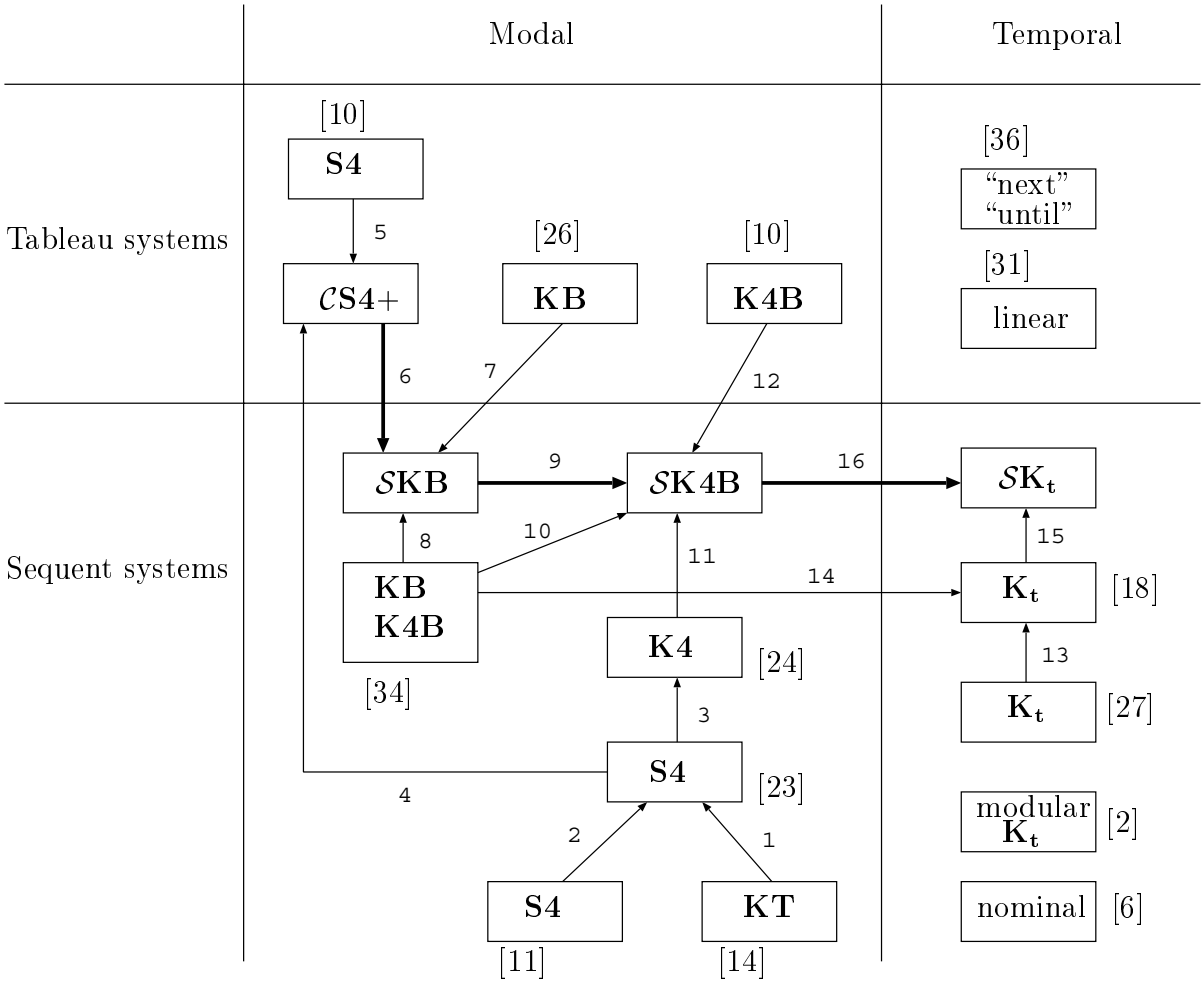


Figure 7.1: Related works

Now, we explain the detailed road map to $\mathcal{S}\mathbf{K_t}$, though our investigation went along bold arrows from $\mathcal{C}\mathbf{S4}+$ to $\mathcal{S}\mathbf{K_t}$, that is, in the order of Arrows 6, 9, 15, In the following, the notation (@i) denotes Arrow i.

In [23], to avoid loops caused by axiom **T**, the technique of using an extra moral operator as a marked version of $\Box$, denoted $\blacksquare$ in [23], was imported from [14] (@1). Histories were introduced in [11] to avoid loops caused by axioms **T** and **4**. In [23], a sequent system for **S4** was introduced by combining $\blacksquare$ and histories (@1, 2). As for avoiding loops caused by axiom **T**, $\blacksquare$ gains an advantage over histories because we need to check sets of formulas if we adopt histories. In [24], a proof–search procedure for **K4** was given by using history (@3), and a construction of counter–models of **K4** was also proposed.

In [10], tableau systems for some modal logics are shown. By applying $\blacksquare$ and histories to a tableau system for **S4** shown in [10], $\mathcal{C}\mathbf{S4}+$ was introduced (@4). In the proof of completeness of $\mathcal{C}\mathbf{S4}+$, the techniques of $\blacksquare$, histories and model graphs were combined (@4, 5).

In [26], a tableau system for **KB** is proposed, though the formulation of **KB** with tableau systems was an open problem in [10]. The sequent system $\mathcal{S}\mathbf{KB}$ was introduced by using the technique of using $\sharp$, which is based on the idea in [26] (@7). Model graphs for tableau systems were imported into sequent systems (@6, 7), and model graphs became available also in sequent systems. In [34], sequent systems for **KB** and **K4B** are proposed, in which the subformula property holds by restricting cut formulas. Our $\mathcal{S}\mathbf{KB}$ utilized the idea of [34] (@8).

Model graphs for sequent system were applied also to the proof of completeness of $\mathcal{S}\mathbf{K4B}$ (@9). The idea of restriction of cut formulas of [34] was introduced in our $\mathcal{S}\mathbf{K4B}$ (@10). Moreover, histories were imported from [24] (@11), and we referred to the way of construction of model graphs from [10] (@12).

In [27], a sequent system for $\mathbf{K_t}$ is proposed, though the subformula property does not hold in the sequent system. In [18], by introducing the idea of restriction of cut formulas of [34] to the sequent system in [27], a sequent system for $\mathbf{K_t}$ was given, in which the subformula property holds (@13, 14). By using all techniques so far, we proposed a sequent system $\mathcal{S}\mathbf{K_t}$ (@15, 16).

As further information, in [36, 31, 2, 6], temporal logics with different formulation from ours are discussed. In [36], a tableau system, which has "next" operator and "until" operator as tense operators, are proposed. In [31], a linear temporal logic is discussed with a tableau system. In [2], a labelled sequent system for a temporal logic are proposed, which is has neither the axiom **4F** nor the **4P**. In [6], nominal tense logics are discussed with sequent systems.

Chapter 8

Conclusions and further work

In this chapter, we summarize the results of this thesis and mention further studies.

8.1 Conclusion

1. To facilitate approaching the temporal logic $\mathbf{K_t}$, we first introduced some techniques in systems for the modal logics **S4**, **KB** and **K4B**, and gave proof–search procedures for them.

   (a) Proof–search procedure for **S4** (Chapter 3)
   We gave a proof–search procedure for **S4** based on the tableau system $\mathcal{C}$**S4**+, in which histories was introduced. Moreover, we gave a way of construction of counter-models of **S4**, which were given by model graphs, in the proof of completeness of $\mathcal{C}$**S4**+.

   (b) Proof–search procedure for **KB** and **K4B** (Chapter 4)
   In the first part of this chapter, we gave a proof–search procedure for **KB**. It is determined by the sequent system $\mathcal{S}$**KB**. On the other hand, in the second part of this chapter, we gave a proof–search procedure for **K4B**, which is determined by the sequent system $\mathcal{S}$**K4B**. Both proof–search procedures avoid redundant applications of the cut rule. Model graphs for sequent systems $\mathcal{S}$**KB** and $\mathcal{S}$**K4B** are introduced, by which we can show completeness of these systems.

   (c) Termination and upper bounds of proof–search procedures for **KB**, **K4B** and **S4** (Chapter 5)
   Termination of proof–search procedures for each of **K4**, **KB**, **K4B** and **S4** was shown. It guarantees that $\mathcal{S}$**KB**, $\mathcal{S}$**K4B** and $\mathcal{C}$**S4**+ are loop–free systems. In addition, we estimated upper bounds of them. Estimation of upper bound would contribute to implementation of proof–search procedures.

2. Proof–search procedure for $\mathbf{K_t}$ (Chapter 6)
   In this chapter, by enhancing histories to sequences of histories, we gave a proof–search procedure for $\mathbf{K_t}$ based on our sequent system $\mathcal{S}\mathbf{K_t}$. The proof–search procedure avoids redundant applications of the cut rule. In the proof of completeness of $\mathcal{S}\mathbf{K_t}$, we gave a way of construction of counter–models for a unprovable formula with model graphs.

Our work investigated a possibility of histories to avoid loop–checking. By making checking histories the alternative loop–checking, we put histories to trial on $\mathbf{K_t}$. Actually, a simpler sequent system for $\mathbf{K_t}$ had been proposed before $\mathcal{S}\mathbf{K_t}$ was introduced. However, the present author found that the simper system had a crucial deficiency in histories because of alternating applications of rules on operators $[F]$ and $[P]$. That is why histories was enhanced. Eventually, such an approach to $\mathbf{K_t}$ brought us a complicated system $\mathcal{S}\mathbf{K_t}$, though we did not need loop–checking.

8.2 Further work

We can say that our work of **S4**, **KB** and **K4B** are acceptable. However, though we mentioned alternating applications of rules on operators $[F]$ and $[P]$, it is not clear whether that would really cause loops nor not in Maruyama, Tojo and Ono's system for $\mathbf{K_t}$ introduced in Section 6.2. In other words, it is not evident whether such a problem is inevitable or not. This should be clarified.

Further work is as follows:

- First of all, we need to simplify the sequent system $\mathcal{S}\mathbf{K_t}$. Transitional rules of it are so complicated that we require to refine them. In order to do it, one way would be to analyze histories. Also, another new techniques might give us a practical system for $\mathbf{K_t}$. In contrast, we need to clarify whether loop–free systems for $\mathbf{K_t}$ must intrinsically be complicated or not. This issue is not only the most interesting but also the most important.

  After we succeed in simplifying $\mathcal{S}\mathbf{K_t}$, the following will be practical further work:

  – Fusion of temporal logic and other logics
    It will be possible to combine $\mathcal{S}\mathbf{K_t}$ with other logics. Since temporal logics and epistemic logics are very useful for formalizing various notions appearing in computer science, for example it will be possible to improve Maruyama, Tojo and Ono's sequent systems for temporal epistemic logics with $\mathcal{S}\mathbf{K_t}$.

– Implementation

Indeed, Mouri constructed a proof assistant system xpe (X window system Proof Editor) and implemented proof–search procedures for $\mathbf{K}4, \mathbf{S4}$ etc. on xpe. (See [25] for the details.) It will be possible to implement our proof–search procedures for $\mathbf{KB}, \mathbf{K4B}$ and $\mathbf{K_t}$ on xpe. By implementation, it will be also possible to assess efficiency of them.

- As another further work, the conversion of $\mathcal{S}\mathbf{KB}$ and $\mathcal{S}\mathbf{K4B}$ is desired, though we discussed in Sections 4.2.4 and 4.3.4. Since it is still hard to read proofs converted by ways discussed in the sections, we need simplification of proofs. It will make more available our proof–search procedures determined by $\mathcal{S}\mathbf{KB}$ and $\mathcal{S}\mathbf{K4B}$.

Bibliography

[1] F. Baader and T. Nipkow, Term Rewriting and All That, Cambridge University Press, 1998.

[2] N. Bonnette and R. Goré *A Labelled sequent system for tense logic K_t*, in Proceedings of the Australian Joint Conference on Artificial Intelligence, LNAI 1052, pp. 71–82, Springer Verlag, 1998.

[3] S. Cerrito and M. C. Mayer, *A polynomial translation of S4 into T and contraction–free tableaux for S4*, Logic Journal of the IGPL, 5(2):287–303, Oxford University Press 1997.

[4] M. C. Fitting, *Tableau methods of proof for modal logics*, Norte Dame Journal of Formal Logic 13, pp.237–247, 1972.

[5] M. C. Fitting, Proof of Methods for Modal and Intuitionistic Logics, D.Reidel Publishing Co., Dordrecht, 1983.

[6] S. Demri, *Sequent calculi for nominal tense logics: a step towards mechanization?*, in Proceedings of Analytic Tableaux and Related Methods (TABLEAUX '99), LNAI 1617, pp. 140–154, Springer Verlag, 1999.

[7] D. Gabbay, I. Hodkinson and M. Reynolds, Temporal Logic Mathematical Foundations and Computational Aspects, volume 1, Oxford University Press, 1994.

[8] R. Goldblatt, Logics of Time and Computation, Second Edition, CLSI Lecture Note No.7, Center for the Study of Language and Information, Stanford University, 1992.

[9] R. Goré, *Cut–free sequent and tableau systems for propositional Diodorean modal logics*, Studia Logica 53:433–457, 1994.

[10] R. Goré, *Tableau methods for modal and temporal logics*, in Handbook of Tableau Methods, eds. by Marcello D'Agostino, Dov M.Gabby, Reiner Hähnle, Joachim Posegga, Kluwer Academic Publishers, 1999.

[11] A. Heuerding, M. Seyfried and H. Zimmermann, *Efficient loop–check for backward proof search in some non–classical propositional logics*, in Proceedings of the

5th Workshop on Theorem Proving with Analytic Tableaux and Related Methods (TABLEAUX '96), LNCS 1071, pp. 210–225, Springer Verlag, 1996.

[12] K. J. J. Hintikka, *Form and content in quantification theory*, Acta Philosophica Fennica, 8:3–55, 1955.

[13] J. Hudelmaier, *On a contraction free sequent calculus for the modal Logic S4*, in Proceedings of the 3rd Workshop on Theorem Proving with Analytic Tableaux and Related Methods. Technical Report TR–94/5, Department of Computing, Imperial College of Science, Technology and Medicine, London, 1994.

[14] J. Hudelmaier, *Improved decision procedures for the modal logics K, T and S4*, in Proceeding of the 9th International Workshop, Computer Science Logic '95, LNCS 1092, pp. 320–334, Springer Verlag, 1996.

[15] T. Kobayashi, Theorem proving in non–classical logics, Master's thesis, Department of Computer and Information Sciences, Ibaraki University, 1994.

[16] M. Kracht, *Notes on the space requirements for checking satisfiability in modal logics*, in Proceeding of the 4th International Workshop, Advances in Modal Logic (AiML 2002), pp. 205–221, CSLI.

[17] R. E. Ladner, *The computational complexity of provability in systems of modal propositional logic*, SIAM Journal on Computing, 6(3):467–480, 1977.

[18] A. Maruyama, S. Tojo and H. Ono, *Decidability of temporal epistemic logics for multi–agent models*, in Proceeding of the ICLP'01 workshop on Computational Logics in Multi–Agent systems, pp. 31–40, 2001.

[19] F. Massacci, *Single step tableaux for modal logics, computational properties, complexity and methodology*, Journal of Automated Reasoning, 24(3):319–364, 2000.

[20] T. Matsumoto and H. Ono, *Theorem prover for modal logics with tableau method*, in Proceedings of International Workshop, Rewriting in Proof and Computation (RPC '01), pp. 42–59, 2001.

[21] T. Matsumoto, *A tableau system for modal logic S4 with an efficient proof–search procedure*, in Proceedings of the 5th JSSST Workshop on Programming and Programming Languages (PPL2003), pp. 75–86, 2003.

[22] M. Mouri, *Theorem provers with counter models and xpe*, Bulletin of the Section of Logic, 30(2):79–86, 2001.

[23] M. Mouri, *An efficient construction of counter–models for modal logic S4*, in Proceedings of International Workshop, Rewriting in Proof and Computation (RPC '01), pp. 247–256, 2001.

[24] M. Mouri, *Constructing counter-models for modal logic K4 from refutation trees*, Bulletin of the Section of Logic, 31(2):81–90, 2002.

[25] M. Mouri, the URL is of xpe is: `http://www.jaist.ac.jp/~mouri/`.

[26] L. A. Nguyen, *Analytic tableau systems and interpolation for the modal logics KB, KDB, K5, KD5*, Studia Logica 69, pp. 41–57, 2001.

[27] H. Nishimura, *A study of some tense logics by Gentzen's sequential method*, Publications of the Research Institute for Mathematical Science, Kyoto University 16, pp. 343–353, 1980.

[28] H. Ono, *Proof–theoretical methods in nonclassical logic – an introduction*, Theories of Types and Proofs, eds. by M. Takahashi et al, MSJ Memoir 2, Mathematical Society of Japan, pp. 207-254, 1998.

[29] L. Pinto and R. Dyckhoff, *Loop–free construction of counter–models for intuitionistic propositional logic*, Symposia Gaussiana, Conf. A (M. Behara, R. Fritsch and R.G. Lintz, editors), Walter de Gruyter & Co., Berlin, pp. 225–232, 1995.

[30] N. Rescher and A. Urquhart, Temporal Logic, Springer Verlag, 1971.

[31] P. Schmitt and J. Goubault-Larrecq, *A Tableau system for linear–time temporal logic*, in Proceedings of the 3rd Workshop Tools and Algorithms for the Construction and Analysis of Systems (TACAS '97), LNCS 1217, pp. 130–144, Springer Verlag, 1997.

[32] R. M. Smullyan, First-Order Logic, Springer Verlag, 1968. Revised edition, Dover Press, 1994.

[33] E. Spaan, Complexity of Modal Logics, PhD thesis, Department of Mathematics and Computer Science, University of Amsterdam, 1993.

[34] M. Takano, *Subformula property as a substitute for cut–elimination in modal propositional logics*, Mathematica Japonica 37, pp. 1129–1145, 1992.

[35] M. Takano, *A modified subformula property for the modal logics K5 and KD5*, Bulletin of the Section of Logic, 30(2):115–122, 2001.

[36] M. Wooldridge, C. Dixon and M. Fisher, *A tableau–based proof method for temporal logics of knowledge and belief*, Journal of Applied Non–Classical Logics 8, pp. 225–258, 1998.

[37] H. Wang, *A survey of mathematical logic*, Studies in Logic and the Foundations of Mathematics, North–Holland, Amsterdam, 1963.

Publications

Refereed Paper in Journal

[1] T. Matsumoto, *Time complexity of a proof–search procedure for K4*, Bulletin of the Section of Logic, (32)4:201–211, (Dec. 2003).

International Conference Paper

[2] T. Matsumoto and H. Ono, *Theorem prover for modal logics with tableau method*, Proceedings of International Workshop, Rewriting in Proof and Computation (RPC '01), pp. 42–59, (Oct. 2001).

Refereed Domestic Conference Paper

[3] T. Matsumoto, *A tableau system for modal logic S4 with an efficient proof-search procedure*, Proceedings of the 5th JSSST Workshop on Programming and Programming Languages (PPL2003), pp. 75–86, (Mar. 2003).

Domestic Conference Papers

[4] A. Maruyama and T. Matsumoto, *Tableau system and theorem prover for temporal logic Kt*, Proceedings of the 35th MLG meeting, pp. 30–32, (Jan. 2002).

[5] T. Matsumoto, *A model tableau system for S4*, Proceedings of the 36th MLG meeting, pp. 16–17, (Dec. 2002).

[6] T. Matsumoto, *Improved sequent system for KB*, Proceedings of the 37th MLG meeting, pp. 26–29, (Dec. 2003).

Domestic Conference

[7] T. Matsumoto, *Improved sequent system for KB*, the 23rd TRS meeting, (Sep. 2003).

Appendix A

Implementation of our proof–search procedure

In this chapter, we will briefly show an implementation of our proof–search procedure based on tableau system. It covers classical logic, modal logics **K**, **KD**, **KT**, **K4** and **S4**, and simply a prototype of the implementation. The proof–search procedure is implemented in Standard ML of New Jersey (abbreviated SML/NJ). Standard ML of New Jersey is available in the following:

`http://www.smlnj.org//index.html`

The detail is omitted. As for further information on SML/NJ, see this web site. The reader is supposed to be familiar with SML/NJ.

A.1 Introduction

The proof–search procedure is constituted of the following six files.

`sources.cm`	:	File of list of all source files
`load.sml`	:	File to start up
`utility.sml`	:	File in which utility functions are defined
`formula.sml`	:	File in which the data structure of formulas is defined
`tableau_tree.sml`	:	File in which the data structure of tableaus is defined
`tableau_prover.sml`	:	File in which the main function is defined

The source codes are shown in Section A.5.

A.2 Syntax

The data structure of formula is defined as follows:

```
datatype formula = AND   of formula * formula
                 | OR    of formula * formula
                 | IMPLY of formula * formula
                 | NOT   of formula
                 | BOX   of formula
                 | FOR   of string
                 | FALSE ;
```

For example, the following formulas:

- `(OR (FOR "p", NOT (FOR "p")))`
- `(IMPLY (BOX (FOR "A"),BOX (BOX (FOR "A"))))`

denote $(p \vee \neg p)$ and $(\Box A \supset \Box\Box A)$, respectively.
The data structure of tableau is defined as follows:

```
type name_of_rule = string ;

type node = formula list *
            formula list *
            formula list *
            formula list *
            string list  *
            string list ;

type history = formula list * formula list ;

datatype tree = Node of name_of_rule * node * history * tree list
              | Leaf of formula list ;
```

For example, a tableau for $p \vee \neg p$ is denoted as follows:

```
Node
   ("NOT-OR",([NOT (OR (FOR "p",NOT (FOR "p")))],[],[],[],[],[]),([],[]),
    [Node
       ("NOT",([NOT (NOT (FOR "p"))],[],[],[],[],["p"]),([],[]),
        [Node ("p",([],[],[],[],["p"],["p"]),([],[]),[Leaf [FALSE]])])])
```

This tableau is as follows:

$$\dfrac{\dfrac{\dfrac{\neg(p \vee \neg p)\ \langle\epsilon|\epsilon\rangle}{\neg\neg p, \neg p\ \langle\epsilon|\epsilon\rangle}\ (\neg\vee)}{p, \neg p\ \langle\epsilon|\epsilon\rangle}\ (\neg\neg)}{f}\ (f)$$

Moreover, a tableau for $\Box p \supset \Box\Box p$ is denoted as follows:

```
Node
    ("NOT-IMPLY",
     ([NOT (IMPLY (BOX (FOR "A"),BOX (BOX (FOR "A"))))],[],[],[],[],[]),
     ([],[]),
     [Node
        ("T",([],[FOR "A"],[],[BOX (FOR "A")],[],[]),([],[]),
         [Node
            ("S4t",([],[],[FOR "A"],[BOX (FOR "A")],["A"],[]),([],[]),
             [Node
                ("T",([],[FOR "A"],[],[FOR "A"],[],[]),
                 ([FOR "A"],[BOX (FOR "A")]),
                 [Node
                    ("S4s",([],[],[FOR "A"],[FOR "A"],["A"],[]),
                     ([FOR "A"],[BOX (FOR "A")]),
                     [Node
                        ("T",([],[FOR "A"],[],[],[],["A"]),
                         ([FOR "A"],[FOR "A",BOX (FOR "A")]),
                         [Node
                            ("F",([],[],[FOR "A"],[],["A"],["A"]),
                             ([FOR "A"],[FOR "A",BOX (FOR "A")]),
                             [Leaf [FALSE]])])])])])])])
```

This tableau is as follows:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\neg(\Box A \supset \Box\Box A)\ \langle\epsilon|\epsilon\rangle}{\Box A, \neg\Box\Box A\ \langle\epsilon|\epsilon\rangle}\ (\neg\supset)}{\blacksquare A, A, \neg\Box\Box A\ \langle\epsilon|\epsilon\rangle}\ (T)}{\Box A, \neg\Box A\ \langle\Box A|\neg\Box\Box A\rangle}\ (S4)_t}{\blacksquare A, A, \neg\Box A\ \langle\Box A|\neg\Box\Box A\rangle}\ (T)}{\Box A, \neg A\ \langle\Box A|\neg\Box\Box A, \neg\Box A\rangle}\ (S4)_s}{\blacksquare A, A, \neg A\ \langle\Box A|\neg\Box\Box A, \neg\Box A\rangle}\ (T)}{f}\ (f)$$

A.3 Hello, good–bye

A.3.1 Starting up

After invoking SML/NJ, type `use "load.sml"` to load all functions for our proof–search procedure.

```
Standard ML of New Jersey, Version 110.0.3, January 30, 1998 [CM; autoload enabled]
- use "load.sml" ;


val it = () : unit
-
```

A.3.2 Quitting

As we know, type as follows:

```
- [CTRL-D]
%
```

A.4 Proof–search

A.4.1 Selecting logic

We can select logic from **K**, **KD**, **KT**, **K4** and **S4**. Our targeting logic is determined by a variable `Logic`, which can be the following value:

`CPC`	:	classical logic
`K`	:	modal logic **K**
`KD`	:	modal logic **KD**
`KT`	:	modal logic **KT**
`K4`	:	modal logic **K4**
`S4`	:	modal logic **S4**

The dafault value of `Logic` is `CPC`. To select other logic, type `Logic :=` (*logic*) as follows:

```
- !Logic ;
val it = CPC : logic
- Logic := S4 ;
val it = () : unit
- !Logic ;
val it = S4 : logic
-
```

A.4.2 Sample dialogue

For proof–saerch, we use the function `prove`. If a proof–search succeeds, `prove` returns a tableau, otherwise, it returns an exception `UNPROVABLE`. A sample dialogue for proof–search is shown in the following:

```
- !Logic ;
val it = CPC : logic
- prove (OR (FOR "p", NOT (FOR "p"))) ;
val it =
  Node
    ("NOT-OR",([NOT (OR (FOR "p",NOT (FOR "p")))],[],[],[],[],[]),([],[]),
     [Node
        ("NOT",([NOT (NOT (FOR "p"))],[],[],[],[],["p"]),([],[]),
         [Node ("F",([],[],[],[],["p"],["p"]),([],[]),[Leaf [FALSE]])])])
  : tree
- val axiom_k = IMPLY (BOX (IMPLY (FOR "A",FOR "B")),IMPLY (BOX (FOR "A"),BOX (FOR "B"))) ;
val axiom_k =
  IMPLY (BOX (IMPLY (FOR "A",FOR "B")),IMPLY (BOX (FOR "A"),BOX (FOR "B")))
  : formula
- prove axiom_k ;

uncaught exception UNPROVABLE
  raised at: tableau_prover.sml:71.22-71.32
- Logic := K ;
val it = () : unit
- prove axiom_k ;
val it =
  Node
    ("NOT-IMPLY",
     ([NOT
         (IMPLY
            (BOX (IMPLY (FOR "A",FOR "B")),
             IMPLY (BOX (FOR "A"),BOX (FOR "B"))))],[],[],[],[],[]),([],[]),
     [Node
        ("NOT-IMPLY",
         ([NOT (IMPLY (BOX (FOR "A"),BOX (FOR "B")))],
          [IMPLY (FOR "A",FOR "B")],[],[],[],[]),([],[]),
         [Node
            ("K",([],[FOR "A",IMPLY (FOR "A",FOR "B")],[],[FOR "B"],[],[]),
             ([],[]),
             [Node
                ("IMPLY",([IMPLY (FOR "A",FOR "B")],[],[],[],["A"],["B"]),
                 ([],[]),
                 [Node
                    ("F",([],[],[],[],["A"],["A","B"]),([],[]),[Leaf [FALSE]]),
                  Node
                    ("F",([],[],[],[],["B","A"],["B"]),([],[]),
                     [Leaf [FALSE]])])])])]) : tree
-
```

A.5 Source codes

Source codes are shown in the following.

A.5.1 `sources.cm`

```
(*
 *
 *  sources.cm
 *
 *)

Group is

    utility.sml
    formula.sml
    tableau_tree.sml
    tableau_prover.sml

(* ---------- EOF ---------- *)
```

A.5.2 `load.sml`

```
(*
 *
 *  load.sml
 *
 *)

Compiler.Control.Print.printDepth  := 5000 ;
Compiler.Control.Print.printLength := 5000 ;
Compiler.Control.Print.stringDepth := 5000 ;
Compiler.Control.Print.linewidth   := 5000 ;

CM.make() ;
open Prover ;

(* ---------- EOF ---------- *)
```

A.5.3 `utility.sml`

```
(*
 *
 *  utility.sml
 *
 *)

structure Utility =
    struct
    fun is_inset x ys = List.exists (fn y => x = y) ys ;

    fun is_subset [] [] = true
      | is_subset [] ys = true
      | is_subset xs [] = false
      | is_subset (x::xs) ys = (is_inset x ys) andalso (is_subset xs ys) ;

    fun is_prop_subset xs ys = not (is_subset ys xs) ;

    fun add x xs = if (is_inset x xs) then xs else (x :: xs) ;

    fun elim (x, [])     = []
      | elim (x, y::ys) = if x = y then elim (x, ys) else y :: (elim (x, ys)) ;

    end ; (* ----- End of structure ----- *)

(* ---------- EOF ---------- *)
```

A.5.4 formula.sml

```
(*
 *
 *  formula.sml
 *
 *)

structure Formula =
   struct

   datatype formula =
      AND   of formula * formula
    | OR    of formula * formula
    | IMPLY of formula * formula
    | NOT   of formula
    | BOX   of formula
    | FOR   of string
    | FALSE ;

   fun len (AND (A, B))   = 1 + (len A) + (len B)
    | len (OR (A, B))     = 1 + (len A) + (len B)
    | len (IMPLY (A, B)) = 1 + (len A) + (len B)
    | len (NOT A)         = 1 + (len A)
    | len (BOX A)         = 1 + (len A)
    | len (FALSE)         = 1
    | len (FOR A)         = 1 ;

   fun get_longest As =
      let
      fun longest (A, la, [])    = A
        | longest (A, la, B::Bs) =
          let
          val lb = len B
          in
          if (la > = lb) then
              longest (A, la, Bs)
          else
              longest (B, lb, Bs)
          end
```

```
        val top = hd As
        in
        longest (top, len top, tl As)
        end ;

    end ; (* ----- End of structure ----- *)

(* ---------- EOF ---------- *)
```

A.5.5 tableau_tree.sml

```
(*
 *
 *   tableau_tree.sml
 *
 *)

structure Tree =
    struct

    open Formula ;

    (* logics *)
    datatype logic =
        CPC
      | K
      | KD
      | KT
      | K4
      | S4  ;

    val Logic = ref CPC ; (* default logic *)

       type node =
        formula list * (* list of formulas to decompose *)
        formula list * (* list of active positive modalized formulas *)
        formula list * (* list of inactive positive modalized formulas *)
        formula list * (* list of negative modalized fromulas *)
        string list  * (* list of positive literals *)
        string list ;  (* list of negative literals *)

    type name_of_rule = string ;
    type history = formula list * formula list ;

    datatype tree =
        Node of name_of_rule * node * history * tree list
      | Leaf of formula list ;

    val InitNode = ([], [], [], [], [], []) : node ;
```

```
    fun node_to_set (remains, active_posmod, inactive_posmod, negmod, poss, negs) =
        remains @
        (map (fn A => BOX A) (active_posmod @ inactive_posmod)) @
        (map (fn A => NOT (BOX A)) negmod) @
        (map (fn A => FOR A) poss) @
        (map (fn A => NOT (FOR A)) negs) ;

    end ; (* ----- End of structure ----- *)

(* ---------- EOF ---------- *)
```

A.5.6 tableau_prover.sml

```
(*
 *
 *  tableau_prover.sml
 *
 *)


structure Prover =
   struct

   open Utility ;
   open Tree ;

   exception NO_RULE ;
   exception UNPROVABLE ;
   exception T_SATURATED ;

   (* history *)
   val InitHist = ([], []) : history ;

   fun judge [] negs = false
     | judge poss [] = false
     | judge (pos::poss) negs = (is_inset pos negs) orelse (judge poss negs) ;

   fun classical_rules A =
      case A of
      (AND (B, C))         => ("AND", [[B, C]])
        | (OR (B, C))          => ("OR", [[B], [C]])
        | (IMPLY (B, C))       => ("IMPLY", [[NOT B], [C]])
        | (NOT (AND (B, C)))   => ("NOT-AND", [[NOT B], [NOT C]])
        | (NOT (OR (B, C)))    => ("NOT-OR", [[NOT B, NOT C]])
        | (NOT (IMPLY (B, C))) => ("NOT-IMPLY", [[B, NOT C]])
        | (NOT (NOT B))        => ("NOT", [[B]])
        | _ => raise NO_RULE ;

   fun make_node []
      (anode as (remains, active_posmod, inactive_posmod, negmod, poss, negs)) = anode
     | make_node (A::As)
```

```
    (anode as (remains, active_posmod, inactive_posmod, negmod, poss, negs)) =
    case A of
      (BOX B) =>
       if (is_inset B active_posmod) then make_node As anode
       else make_node As
       (remains, B::active_posmod, inactive_posmod, negmod, poss, negs)
     | (NOT (BOX B)) =>
       if (is_inset B negmod) then make_node As anode
       else make_node As
       (remains, active_posmod, inactive_posmod, B::negmod, poss, negs)
     | (FOR B) =>
       if (is_inset B poss) then make_node As anode
       else make_node As
       (remains, active_posmod, inactive_posmod, negmod, B::poss, negs)
     | (NOT (FOR B)) =>
       if (is_inset B negs) then make_node As anode
       else make_node As
       (remains, active_posmod, inactive_posmod, negmod, poss, B::negs)
     |  _ =>
       if (is_inset A remains) then
       make_node As anode
       else
       make_node As
       (A::remains, active_posmod, inactive_posmod, negmod, poss, negs) ;

fun apply_tableau
    (anode as ([], active_posmod, inactive_posmod, negmod, poss, negs)) hist =
    (if (judge poss negs) then
     Node ("F", anode, hist, [Leaf [FALSE]])
     else
     (case (!Logic) of
         CPC => raise UNPROVABLE
       | K   => tableau_modal_rule_K_KD
         (active_posmod, inactive_posmod, negmod, poss, negs) hist
       | KT  => tableau_modal_rule_KT
         (active_posmod, inactive_posmod, negmod, poss, negs) hist
       | KD  => tableau_modal_rule_K_KD
         (active_posmod, inactive_posmod, negmod, poss, negs) hist
       | K4  => tableau_modal_rule_K4
```

```
        (active_posmod, inactive_posmod, negmod, poss, negs) hist
      | S4  => tableau_modal_rule_S4
        (active_posmod, inactive_posmod, negmod, poss, negs) hist))
  | apply_tableau
    (anode as ((A::As), active_posmod, inactive_posmod, negmod, poss, negs)) hist =
    (if (judge poss negs) then
        Node ("F", anode, hist, [Leaf [FALSE]])
     else
        (let
         val target = get_longest (A::As)
         val (rule, newbranches) = ((classical_rules target)
                    handle NO_RULE => raise UNPROVABLE)
         val remain = elim (target, A::As)
         in
         Node (rule, anode, hist,
           map (fn b =>
               apply_tableau (make_node b
                     (remain, active_posmod, inactive_posmod, negmod, poss, negs)) hist)
           newbranches)
         end))
and
    (* K+ and KD+ *)
    tableau_modal_rule_K_KD
    (active_posmod, inactive_posmod, negmod, poss, negs) hist =
    (let
     val event = ([NOT (hd negmod)]
           handle Empty =>
           if (!Logic) = KD then [] else raise UNPROVABLE)
     val new_posmod_event = active_posmod @ inactive_posmod @ event
     val newnode =
        if (null new_posmod_event) then
        raise UNPROVABLE
        else
        make_node (new_posmod_event) InitNode
     val rule = (case (!Logic) of
           K  => "K"
         | KD => "KD"
         | KT => "K"
         | _  => raise NO_RULE)
```

```
    in
    Node (rule, ([], active_posmod, inactive_posmod, negmod, poss, negs), hist,
          [apply_tableau newnode hist])
    handle UNPROVABLE =>
        let
        (* elimination of loop by (KD) *)
        val new_negmod = (tl negmod) handle Empty => raise UNPROVABLE
        in
        (tableau_modal_rule_K_KD
         (active_posmod, inactive_posmod, new_negmod, poss, negs) hist)
        end
    end)
and
  (* T *)
  tableau_modal_rule_T (active_posmod, inactive_posmod, negmod, poss, negs) hist =
  (if (null active_posmod) then
      raise T_SATURATED
   else
      let
      val active = get_longest active_posmod
      val new_active_posmod = elim (active, active_posmod)
      val new_node =
          make_node [active]
          ([], new_active_posmod, add active inactive_posmod, negmod, poss, negs)
      in
      Node ("T", ([], active_posmod, inactive_posmod, negmod, poss, negs), hist,
        [apply_tableau new_node hist])
      end)
and
  (* KT+ *)
  tableau_modal_rule_KT
  (anode as (active_posmod, inactive_posmod, negmod, poss, negs)) hist =
  (tableau_modal_rule_T anode hist
   handle T_SATURATED => tableau_modal_rule_K_KD anode hist)
    and
  (* (K4+)t, (K4)s *)
  tableau_modal_rule_K4
  (anode as (active_posmod, inactive_posmod, negmod, poss, negs))
  (hist as (valid, invalid)) =
```

```
    (let
     val event = (hd negmod) handle Empty => raise UNPROVABLE
     val new_node =
     make_node ((NOT event)::active_posmod) ([], active_posmod, [], [], [], [])
 in
     (if (is_prop_subset valid active_posmod) then
      (* (K4+)t *)
      let
          val new_hist = (active_posmod, [event])
      in
          Node ("K4t", ([], active_posmod, inactive_posmod, negmod, poss, negs), hist,
            [apply_tableau new_node new_hist])
      end
      else
      if (is_inset event invalid) then
          tableau_modal_rule_K4
          (active_posmod, inactive_posmod, tl negmod, poss, negs) hist
      else
          (* (K4+)s *)
          let
          val new_hist = (valid, add event invalid)
          in
          Node ("K4s", ([], active_posmod, inactive_posmod, negmod, poss, negs), hist,
                [apply_tableau new_node new_hist])
          end)
      handle UNPROVABLE =>
          tableau_modal_rule_K4
          (active_posmod, inactive_posmod, tl negmod, poss, negs) hist
  end)
 and
(* (S4+)t, (S4)s *)
tableau_modal_rule_S4
(anode as (active_posmod, inactive_posmod, negmod, poss, negs))
(hist as (valid, invalid)) =
(tableau_modal_rule_T anode hist
 handle T_SATURATED =>
     let
       val event = (hd negmod) handle Empty => raise UNPROVABLE
       val new_active_posmod = inactive_posmod
```

```
      val new_node = make_node [NOT event] ([], new_active_posmod, [], [], [], [])
    in
    (if (is_prop_subset valid inactive_posmod) then
     (* (S4+)t *)
     let
         val new_hist = (new_active_posmod, [event])
     in
         Node ("S4t", ([], active_posmod, inactive_posmod, negmod, poss, negs), hist,
           [apply_tableau new_node new_hist])
     end
     else
     if (is_inset event invalid) then
         tableau_modal_rule_S4
         (active_posmod, inactive_posmod, tl negmod, poss, negs) hist
     else
         (* (S4+)s *)
         let
         val new_hist = (valid, add event invalid)
         in
         Node ("S4s", ([], active_posmod, inactive_posmod, negmod, poss, negs), hist,
               [apply_tableau new_node new_hist])
         end)
     handle UNPROVABLE =>
         tableau_modal_rule_S4
         (active_posmod, inactive_posmod, tl negmod, poss, negs) hist
    end) ;

  fun prove A = apply_tableau (make_node [NOT A] InitNode) InitHist ;

  end ; (* ----- End of structure ----- *)

(* ---------- EOF ---------- *)
```

A.5.7 `example.sml`

Eamples of formulas are given in this file.

```
(*
 *
 *   example.sml
 *
 *)

local
    open Formula
in
    val A = FOR "A"
    val B = FOR "B"
    val C = FOR "C"
    val D = FOR "D"

    (* formulas *)
    val AK = IMPLY (BOX (IMPLY (A, B)), IMPLY (BOX A, BOX B))   (* axiom K *)
    val AD = IMPLY (BOX A, NOT (BOX (NOT A)))                   (* axiom D *)
    val AT = IMPLY (BOX A, A)                                   (* axiom T *)
    val A4 = IMPLY (BOX A, BOX (BOX A))                         (* axiom 4 *)
    val f1 = OR (A, NOT A)                                      (* exclusive middle *)
    val f2 = IMPLY (IMPLY (IMPLY (A, B), A), A)                 (* Peirce *)
    val f3 = OR (NOT (BOX (NOT (BOX A))), BOX A)                (* unprovable *)
   (* --- f3 causes loops in S4 and K4 --- *)

end ;

(* ---------- EOF ---------- *)
```

Appendix B

Theorem provers

In this chapter, we will give brief survey of some theorem provers. We can get information on theorem provers from the following site:

`http://www-formal.stanford.edu/clt/ARS/ars-db.html`

In this site, a database of existing mechanized reasoning systems is shown. It is possible to get a lot of informative knowledge on mechanized reasoning from the site. Next, we will briefly show the following three theorem provers:

- The Logics Workbench (LWB)
- The Stanford Temporal Prover (STeP)
- X window system Proof Editor (xpe)

B.1 The Logics Workbench

The Logics Workbench (LWB) is developed at the University of Bern in Switzerland, and it covers classical and non-classical propositional logics. The session with the LWB goes interactively. The LWB is available from the following site:

`http://www.lwb.unibe.ch/`

B.2 The Stanford Temporal Prover

The Stanford Temporal Prover (STeP) is developed by the REACT research group of Stanford University to support the computer–aided formal verification of reactive, real–time and hybrid systems based on their temporal specification. The STeP is available from the following site:

`http://www-step.stanford.edu/`

B.3 X window system Proof Editor

The X window system Proof Editor (xpe) is a proof assistant system developed by Mouri of Tokyo Denki University. The xpe has a graphical user interface and is compatible with a proof figure macros, called proof.sty, for LaTeX. We can edit proofs with the xpe, and formulas are input into the xpe in LaTeX. Proof–search procedures for classical logic, some modal logics and some substructural logics are implemented on the xpe. The xpe facilitates writing proofs in LaTeXand is available from the following site:

`http://www.jaist.ac.jp/~mouri/`

The proof figure macros proof.sty is developed by Tatsuta of National Institute of Informatics in Japan and available from the following URL:

`http://research.nii.ac.jp/~tatsuta/proof-sty.html`