Doctoral Dissertation

Automated Forward Deduction Based on Strong Relevant Logics and Its Applications

強相関論理に基づいた 自動前向き演繹とその応用

Yuichi Goto

Information and Mathematical Sciences,
Graduate School of Science and Engineering,
Saitama University

Supervisor: Professor Jingde Cheng

December 2004

Abstract

A forward deduction engine is an indispensable component for a computing system with purposes of prediction and/or discovery. However, there is no practical forward deduction engine until now. This is perhaps because of the following two problems. First, conclusions of forward deduction engines based on classical mathematical logic or its various conservative extensions are not necessarily true and relevant to the given premises in the sense of conditional. Thus, the users of the forward deduction engines have to evaluate all conclusions by themselves. Second, any automated forward deduction is inefficient in the sense that it often deduces many intermediates, i.e. redundant conclusions or instances of theorems that have previously deduced or given. Thus a large amount of execution time and a large amount of main memory are needed to perform automated forward deduction. In order to implement a practical forward deduction engine, we have to solve the above two problems.

From the viewpoint of logic, Cheng qualitatively showed that classical mathematical logic, its various classical conservative extensions, and traditional relevant logics are not suitable to underlying forward deduction for prediction and/or discovery because their logical theorems include a lot of paradoxes of conditional, and showed that strong relevant logics are more hopeful candidates for the purpose. But, it is not clear that how 'bad' the classical mathematical logic, its various classical conservative extensions, and traditional relevant logics are, and how 'good' strong relevant logics are, since no quantitative analysis and discussion is reported until now. In this thesis, we present a quantitative analysis and discussion on implicational paradoxes in classical mathematical logic with the connective of implication and negation, as the first step of quantitative comparative study between classical mathematical logic and strong relevant logics. This comparative study showed that strong relevant logics are quantitatively suitable by far than classic mathematical logic to underlie forward deduction. The result showed that a practical forward deduction engine should be based on strong relevant logics.

In order to solve the performance problem of forward deduction engines, we investigated the relationship between the execution time of forward deduction engines and the amount of given premises and deduced conclusions. Based on the investigation, we proposed a parallelization model, which is a kind of master-slave model, for automated forward deduction, and showed the effectiveness of the model by implementing an automated forward deduction system for general-purpose entailment calculus, named EnCal, based on the model. The case study showed that the execution time of the parallelization version of EnCal gets shorter in proportion to the increase in the number of processors without depending on the number of deduced conclusions and given premises. Hence, improving the performance by parallel processing is effective for forward deduction engines.

In order to show usefulness of forward deduction engines based on strong relevant logics, we investigated some applications of automated forward deduction based on strong relevant logics. First, towards automating scientific discovery

processes, we investigated how to answer logic puzzles by automated forward deduction based on strong relevant logics. In this case study, we got answers to two logic puzzles by reasoning, but not proving. Our case study showed that automated forward deduction based on strong relevant logics may be useful as a tool for automating scientific discovery processes. Second, we investigated the problem of automated theorem finding in von Neumann-Bernays-Godel set theory (NBG set theory for short) by automated forward deduction based on strong relevant logics. In this case study, we deduced some NBG set theory theorems from axioms of NBG set theory, but no paradoxical theorem was been deduced. Our case study showed that it is in principle possible to find theorems of the NBG set theory by automated forward forward deduction based on strong relevant logics. Third, we investigated what role automated forward deduction based on temporal relevant logics can play in anticipatory systems with requirements of high reliability and high security. We showed that the high-performance automated forward deduction based on temporal relevant logics is necessary to implementation of an anticipatory reasoning engine. Lastly, we investigated what role automated forward deduction based on spatial relevant logics can play in geographic information systems. We proposed a new family of strong relevant logics, named spatial relevant logics, and showed that the high-performance automated forward deduction based on spatial relevant logics is necessary to implement a spatial reasoning engine.

This thesis is organized as follows. Chapter 1 presents the background, motivation and purpose of this research. Chapter 2 explains the notions and terminology used in this research. Chapter 3 gives a quantitative analysis on implicational paradoxes in classical mathematical logic from the viewpoint of forward deduction. Chapter 4 gives an analysis for execution time of forward deduction engines, proposes a parallelization model of automated forward deduction, and then presents a case study of a parallelization of EnCal. Chapter 5 presents some applications of automated forward deduction based on strong relevant logics. Concluding remarks are given in chapter 6.

Acknowledgments

Special thanks are due to my thesis supervisor Professor Jingde Cheng for his invaluable support and guidance through the hard moments of graduate school. I am also grateful to my dissertation committee: Professor Hitoshi Maekawa, Professor Yutaka Ohsawa, and Professor Norihiko Yoshida for their support, valuable feedback, and insightful ideas to this research.

List of Figures

List of Tables

Contents

Chapter 1

Introduction

1.1 Background and Motivation

Reasoning is the process of drawing new conclusions from some premises which are known facts and/or assumed hypotheses. A logically valid reasoning is a reasoning such that its process of drawing new conclusions from premises is justified based on some logical criterion in order to obtain correct conclusions. Therefore, a reasoning may be valid on a logical criterion but invalid on another. Automated reasoning is concerned with the execution of computer programs that assist in solving problems requiring reasoning.

Knowledge Engineering (KE) is a discipline concerned with constructing and maintaining knowledge bases to store knowledge of various domains in the real world and using automated reasoning based on the knowledge to solve problems in the domains that ordinarily require human reasoning. However, the current knowledge-based systems cannot reason about those situations and/or problems that have not been considered by their developers and/or users. A major cause of this inadequacy is that the systems cannot autonomously generate new and valid reasoning rules from those existing reasoning rules and facts that are programmed or inputted in the systems by their developers or users [27, 22].

On the other hand, although from 1950s many automated reasoning system for theorem proving have been developed and some difficult theorems have been automatically proved using the systems, at present there is no automated reasoning system can form some concept and/or find some theorem in a domain that are completely new and interesting to the scientists working on the domain [43, 44]. The problem of automated theorem finding, i.e. what properties can be identified to permit an automated reasoning program to find new and interesting theorems, as opposed to proving conjectured theorems? which was proposed by Wos 1988 as the thirty-first of 33 open research problems in automated reasoning, is still open until now [47, 48].

A reactive system is a computing system that maintains an ongoing interaction with its environment as opposed to computing some final value on termination [28, 29], such as computer operating systems, air plain and train traffic control systems and so on. Almost all reactive systems only can perform those operations in response to instructions explicitly issued by users or application programs, but

have no ability to do something actively and anticipatorily by themselves. From the viewpoint of information security engineering, in order to prevent attacks beforehand, it is to be desired that a reactive system can detect and predict omens of attacks anticipatorily and then take some actions to inform its users and perform some operations to defend attacks by itself. On the other hand, those reactive systems with highly reliable requirements also need some anticipatory mechanism to prevent disasters beforehand [12].

All the above applications need automated reasoning which certainly deduces correct conclusions if all premises are true. Reasoning can be classified into three forms, deduction, induction and abduction. Deduction is the process of deducing or drawing conclusions from some general principles already known or assumed. Induction is the process of inferring some general laws or principles from the observation of particular instances. Abduction is the process whereby a surprising fact is made explicable by the application to it of a suitable proposition. The deduction guarantees that the conclusions deduced in the process are true if all premises are true, but induction and abduction do not. Thus, automated forward deduction is an indispensable component for above computing systems with purposes of prediction and/or discovery.

However, there is no practical forward deduction engine until now. This is perhaps because of the following two problems. First, conclusions of forward deduction engines based on classical mathematical logic or its various conservative extensions are not necessarily true and relevant to the given premises in the sense of conditional. Thus, the users of the forward deduction engines have to evaluate all conclusions by themselves. Second, automated forward deduction is inefficient in the sense that it often deduces many intermediates, i.e. redundant conclusions or instances of theorems that have previously deduced or given. Thus a large amount of execution time and a large amount of main memory are needed to perform automated forward deduction. In order to implement a practical forward deduction engine, we have to solve the above two problems.

1.2 Earlier research

The origin of automated forward deduction is a computer program 'Logic Theory Machine' developed by Newell, Shaw and Simon in 1957 [16]. The purpose of that program is to emulate the process by which a person might seek proofs in the propositional calculus of *Principia Mathematica* written by Whitehead and Russell. Its algorithm is "British Museum algorithm" by which all possible proofs are generated until one leading to the desired result is reached. Logic Theory Machine was demonstrated on fifty-two theorems from the propositional calculus in *Principia Mathematica*; it proved thirty-eight of them [16]. However, from the viewpoint of computational complexity, it is impossible to process the large amount of well-formed formulas deduced by Logic Theory Machine, because Logic Theory Machine is based on classical mathematical logic. Consequently, the researchers at the time regarded this approach as the impossible one, and after that no one has researched on this direction.

On the other hand, in 1958 $\sim$ 1960, Hao Wang programmed a decision procedure to prove all of the theorems which the Logic Theory Machine proved and more 350 theorems of *Principia Mathematica*, and his program could prove all of theorems of the *Principia Mathematica* of first-order logic with equality [16]. In 1965, Abraham Robinson showed how to combine unification and satisfiability checking into a single rule called "resolution" [16]. According above two successes, the main direction of automated deduction work is based on resolution and backward deduction, up to now. However, both of them are method for proving, but not reasoning.

However, the failure of Logic Theory Machine is caused by classical mathematical logic rather than forward deduction. From the viewpoint of conditional, classical mathematical logic, CML for short, and its various conservative extensions have the well-known "implicational paradox problem." In CML, the notion of conditional is represented by the truth-functional extensional notion of material implication (denoted by $\rightarrow$ in this thesis) that is defined as $A \rightarrow B =_{df} \neg(A \wedge \neg B)$ or $A \rightarrow B =_{df} \neg A \vee B$. However, the material implication is intrinsically different from the notion of conditional in meaning (semantics).

Historically, implicational paradoxes have been studied many years. The main aim of Lewis's work beginning in 1912 on the establishment of modern modal logic was to find a satisfactory theory of implication which is better than CML in that it can avoid those implicational paradoxes, though his plan was not successful in the sense that some implicational paradoxes in terms of strict implication remained in modal logic [1, 2, 41]. Sugihara 1955 provided the first general characterization of implicational paradoxes [1]. Ackermann 1956 proposed the concept of "Rigorous implication" [1, 2, 41]. During 1957 $\sim$ 1959, Von wright, Geach , and Smiley suggested some informal criteria for the notion of entailment, i.e. so-called "Wright-Geach-Smiley criterion" for entailment [1]. However, it is hard until now to know exactly how to formally context of logic. During the 1950s $\sim$ 1970s, Anderson and Belnap extended the work of Ackermann and proposed variable-sharing as a necessary but not sufficient formal condition for the relevance between the antecedent and consequent of a logical entailment [1, 2, 41].

Relevant logics, RL for short, were constructed during the 1950s $\sim$ 1970s in order to find a mathematically satisfactory way of grasping the notion of entailment [1, 2, 19, 41]. The first one of such logics is Ackermann's logic system Π'. Anderson and Belnap modified and reconstructed Ackermann's logic system into an equivalent logic system, called "system E of entailment." Belnap proposed a logic system, called "system R of relevant implication." Another important relevant logic system is "system T of ticket entailment" or "system T of entailment shorn of modality" which is proposed by Anderson and Belnap. All those logic systems are usually called "entailment logic," "relevance logic," or "relevant logic" [1, 2, 19, 41].

In 1991, Cheng pointed out the new paradoxes in above relevant logics, and proposed strong relevant logics, SRL for short, which do not include the paradoxes [4, 7]. Cheng also showed that an entailment calculus based on the paradox-free relevant logics can underlie reasoning rule generation in knowledge-based systems and automated theorem finding, and proposed an automated forward deduction

system for general-purpose entailment calculus, named EnCal [6]. EnCal supports automated forward deduction for entailment calculi based on SRL as well as other logics.

1.3 Structure of this thesis

In this thesis, we investigate two problems at first: logic systems to underlie automated forward deduction and the performance of automated forward deduction, in order to implement a practical forward deduction engine.

From the viewpoint of logic, Cheng qualitatively showed that CML, its various classical conservative extensions, and RL are not suitable to underlying forward deduction for prediction and/or discovery because their logical theorems include a lot of paradoxes of conditional, and showed that SRL are more hopeful candidates for the purpose [4, 7]. But, it is not clear that how 'bad' the CML, its various classical conservative extensions, and RL are, and how 'good' SRL are, since no quantitative analysis and discussion is reported until now. In this thesis, we present a quantitative analysis and discussion on implicational paradoxes in CML with the connective of implication and negation, as the first step of quantitative comparative study between CML and SRL.

In order to solve the performance problem of the forward deduction engines, we investigate the relationship between the execution time of forward deduction engines and the amount of given premises and deduced conclusions. Based on the investigation, we propose a parallelization model, which is a kind of master-slave model, for automated forward deduction, and show the effectiveness of that model by implementing EnCal, based on the model.

In order to show usefulness of forward deduction engines based on SRL, we investigate some applications of automated forward deduction based on SRL. First, towards automating scientific discovery processes, we investigate how to answer logic puzzles by automated forward deduction based on SRL. Second, we investigate the problem of automated theorem finding in von Neumann-Bernays-Godel set theory (NBG set theory for short) by automated forward deduction based on SRL. Third, we investigate what role automated forward deduction based on temporal relevant logics can play in anticipatory systems with requirements of high reliability and high security. Lastly, we investigate what role automated forward deduction based on spatial relevant logics can play in geographic information systems.

This thesis is organized as follows. Chapter 2 explains the notions and terminology used in this research. Chapter 3 gives a quantitative analysis of paradoxes of implication in CML from the viewpoint of a logic system to underlie automated forward deduction. Chapter 4 gives an analysis for execution time of forward deduction engines, proposes a parallelization model of automated forward deduction, and then presents a case study of a parallelization of EnCal. Chapter 5 presents some applications of automated forward deduction based on strong relevant logics. Concluding remarks are given in chapter 6.

Chapter 2

Reasoning by forward deduction based on logics and its automation

2.1 Basic notions

Reasoning is the process of drawing new conclusions from given premises, which are already known facts or previously assumed hypotheses (Note that how to define the notion of 'new' formally and satisfactorily is still a difficult open problem until now). Therefore, reasoning is intrinsically ampliative, i.e. it has the function of enlarging or extending some things, or adding to what is already known or assumed. In general, a reasoning consists of a number of arguments (or inferences) in some order. An argument is a set of statements (or declarative sentences) of which one statement is intended as the conclusion, and one or more statements, called 'premises,' are intended to provide some evidence for the conclusion. An argument is a conclusion standing in relation to its supporting evidence. In an argument, a claim is being made that there is some sort of evidential relation between its premises and its conclusion: the conclusion is supposed to follow from the premises, or equivalently, the premises are supposed to entail the conclusion. Therefore, the correctness of an argument is a matter of the connection between its premises and its conclusion, and concerns the strength of the relation between them (Note that the correctness of an argument depends neither on whether the premises are really true or not, nor on whether the conclusion is really true or not). Thus, there are some fundamental questions: What is the criterion by which one can decide whether the conclusion of an argument or a reasoning really does follow from its premises or not? Is there the only one criterion, or are there many criteria? If there are many criteria, what are the intrinsic differences between them? It is logic that deals with the validity of argument and reasoning in general.

A logically valid reasoning is a reasoning such that its arguments are justified based on some logical validity criterion provided by a logic system in order to obtain correct conclusions (Note that here the term 'correct' does not necessarily mean 'true'.). Today, there are so many different logic systems motivated by

various philosophical considerations. As a result, a reasoning may be valid on one logical validity criterion but invalid on another.

Proving is the process of finding a justification for an explicitly specified statement from given premises, which are already known facts or previously assumed hypotheses. A proof is a description of a found justification. A logically valid proving is a proving such that it is justified based on some logical validity criterion provided by a logic system in order to obtain a correct proof.

The most intrinsic difference between reasoning and proving is that the former is intrinsically prescriptive and predictive while the latter is intrinsically descriptive and non-predictive. The purpose of reasoning is to find some new conclusion previously unknown or unrecognized, while the purpose of proving is to find a justification for some specified statement previously given. Proving has an explicitly given target as its goal while reasoning does not.

Discovery is the process to find out or bring to light of that which was previously unknown. For any discovery, both the discovered thing and its truth must be unknown before the completion of discovery process. Since reasoning is the only way to draw new conclusions from given premises, there is no discovery process that does not invoke reasoning.

Logic is a special discipline which is considered to be the basis for all other sciences, and therefore, it is a science prior to all others, which contains the ideas and principles underlying all sciences [23, 46]. Logic deals with what entails what or what follows from what, and aims at determining which are the correct conclusions of a given set of premises, i.e. to determine which arguments are valid. Therefore, the most essential and central concept in logic is the logical consequence relation that relates a given set of premises to those conclusions, which validly follow from the premises.

In general, a formal logic system L consists of a formal language, called the object language and denoted by $F(L)$, which is the set of all well-formed formulas of L, and a logical consequence relation, denoted by meta-linguistic symbol $\vdash_L$, such that $P \subseteq F(L)$ and $c \in F(L)$, $P \vdash_L c$ means that within the frame work of L, c is valid conclusion of premises P, i.e. c validly follows from P. For a formal logic system $(F(L), \vdash_L)$, a logical theorem t is a formula of L such that $\phi \vdash_L t$ where ϕ is empty set. We use $Th(L)$ to denote the set of all logical theorems of L. $Th(L)$ is completely determined by the logical consequence relation $\vdash_L$. According to the representation of the logical consequence relation of a logic, the logic can be represented as a Hilbert style formal system, a Gentzen natural deduction system, a Gentzen sequent calculus system, or other type of formal system.

Let $(F(L), \vdash_L)$ be a formal logic system and $P \subseteq F(L)$ be a non-empty set of sentences (i.e. closed well-formed formulas). A formal theory with premises P based on L, called a L-theory with premises P and denoted by $T_L(P)$, is defined as $T_L(P) =_{df} Th(L) \cup Th^e_L(P)$, and $Th^e_L(P) =_{df} \{\text{et}|P \vdash_L \text{et and et} \notin Th(L)\}$ where $Th(L)$ and $Th^e_L(P)$ are called the logical part and the empirical part of the formal theory , respectively, and any element of $Th^e_L(P)$ is called an empirical theorem of the formal theory.

In the literature of mathematical, natural, social, and human sciences, it is probably difficult, if not impossible, to find a sentence form that is more gener-

ally used for describing various definitions, propositions, and theorems than the sentence form of 'if ... then ...'. In logic, a sentence in the form of 'if ... then ...' is usually called a conditional proposition or simply conditional which states that there exists a relation of sufficient condition between the 'if' part and the 'then' part of the sentence. Scientists always use conditionals in their descriptions of various definitions, propositions, and theorems to connect a concept, fact, situation or conclusion to its sufficient conditions. The major work of almost all scientists is to discover some sufficient condition relations between various phenomena, data, and laws in their research fields.

In general, a conditional must concern two parts which are connected by the connective 'if ... then ...' and called the antecedent and the consequent of that conditional, respectively. The truth of a conditional depends not only on the truth of its antecedent and consequent but also, and more essentially, on a necessarily relevant and conditional relation between them. The notion of conditional plays the most essential role in reasoning because any reasoning form must invoke it, and therefore, it is historically always the most important subject studied in logic and is regarded as the heart of logic [1].

When we study and use logic, the notion of conditional may appear in both the object logic (i.e. the logic we are studying) and the meta-logic (i.e. the logic we are using to study the object logic). In the object logic, there usually is a connective in its formal language to represent the notion of conditional, and the notion of conditional, usually represented by a meta-linguistic symbol, is also used for representing a logical consequence relation in its proof theory or model theory. On the other hand, in the meta-logic, the notion of conditional, usually in the form of natural language, is used for defining various meta-notions and describing various meta-theorems about the object logic.

From the viewpoint of object logic, there are two classes of conditionals [11]. One class is empirical conditionals and the other class is logical conditionals. For a logic, a conditional is called an empirical conditional of the logic if its truth-value, in the sense of that logic, depends on the contents of its antecedent and consequent and therefore cannot be determined only by its abstract form (i.e. from the viewpoint of that logic, the relevant relation between the antecedent and the consequent of that conditional is regarded to be empirical); a conditional is called a logical conditional of the logic if its truth-value, in the sense of that logic, depends only on its abstract form but not on the contents of its antecedent and consequent, and therefore, it is considered to be universally true or false (i.e. from the viewpoint of that logic, the relevant relation between the antecedent and the consequent of that conditional is regarded to be logical). A logical conditional that is considered to be universally true, in the sense of that logic, is also called an entailment of that logic. Indeed, the most intrinsic difference between various different logic systems is to regard what class of conditionals as entailments.

An *entailment calculi* is a formalization of a logical system L such that the notion of conditional (entailment) is represented in L by a primitive connective and all logical theorems of L are represented in the form of entailment.

For a formal logic system where the notion of conditional is represented by primitive connective entailment '$\Rightarrow$', a formula is called a zero degree formula if

and only if there is no occurrence of '$\Rightarrow$' in it; a formula of the form '$A \Rightarrow B$' is called a first degree conditional if and only if both A and B are zero degree formula; a formula A is called a first degree formula if and only if it satisfies one of the following conditions [11]:

1. A is a first degree conditional,
2. A is in the form $+B$ ($+$ is a one-place connective such as negation and so on) where B is a first degree formula,
3. A is in the form $B * C$, ($*$ is a non-implicational two-place connective such as conjunction or disjunction and so on), where both of B and C is a first degree formulas, or one of B and C are a first degree formula and the another is a zero degree formula.

Let k be a natural number. A formula of the form '$A \Rightarrow B$' is called a k^{th} degree conditional if and only if both A and B are $(k-1)^{th}$ degree formulas, or either formula A or B is a $(k-1)^{th}$ degree formula and the another is a $j^{th}(j < k-1)$ degree formula; a formula is called k^{th} degree formula if and only if it satisfies one of the following conditions [11]:

1. A is a k^{th} degree conditional,
2. A is in the form $+B$ ($+$ is a one-place connective such as negation and so on) where B is a k^{th} degree formula,
3. A is in the form $B * C$, ($*$ is a non-implicational two-place connective such as conjunction or disjunction and so on), where both of B and C is a k^{th} degree formulas, or one of B and C are a k^{th} degree formula and the another is a $j^{th}(j < k)$ degree formula.

Let $(F(L), \vdash_L)$ be a formal logic system and k be a natural number. The k^{th} degree fragment of L, denoted by $Th^k(L)$, is a set of logical theorems of L that is inductively defined as follows (in the terms of Hilbert-style formal systems) [11]:

1. if A is a $j^{th}(j \leq k)$ degree formula and an axiom of L, then $A \in Th^k(L)$,
2. if A is a $j^{th}(j \leq k)$ degree formula that is the result of applying an inference rule of L to some members of $Th^k(L)$, then $A \in Th^k(L)$,
3. nothing else is a member of $Th^k(L)$, i.e. only those obtained from repeated applications of 1. and 2. are members of $Th^k(L)$.

Let $(F(L), \vdash_L)$ be a formal logic system, $P \subset F(L)$, and k and j be two natural numbers. A formula A is said to be j^{th}-degree-deducible from P based on $Th^k(L)$ if and only if there is an finite sequence of formulas $f_1, \ldots, f_n$ such that $f_n = A$ and for all $i(i \leq n)$ (1) $f_i \in Th^k(L)$, or (2) $f_i \in P$, or (3) f_i whose degree is not higher than j is the result of applying an inference rule to some members $f_{j_1}, \ldots, f_{j_m}(j_1, \ldots, j_m < i)$ of the sequence. If $P \neq \phi$, then the set of all formulas which are j^{th}-degree-deducible from P based on $Th^k(L)$ is called the j^{th} degree fragment of the formal theory with premises P based on $Th^k(L)$, denoted by $T^j_{Th^k(L)}(P)$ [11].

2.2 EnCal: an automated forward deduction system for general-purpose entailment calculus

Cheng proposed an automated forward deduction system for general-purpose entailment calculus, named EnCal [6]. It supports an automated forward deduction for entailment calculi based on SRL as well as other logics. It provides its users with the following major facilities. For a formal logic system L which may be a propositional logic, a first-order predicate logic, or a second-order predicate logic, a non-empty set P of formulas as premises, inference rules of logic system L and natural number k and j (usually, $k, j \leq 5$) as limit of degree which is the degree of nested entailment (denoted by '$\Rightarrow$' in this thesis), all specified by the user, EnCal can

1. reason out all logical theorem schemata of the $Th^k(L)$,
2. verify whether or not a formula is a logical theorem schema of the $Th^k(L)$, if yes, then give the proof,
3. reason out all empirical theorems of the j^{th} degree fragment of L-theory with premises P based on $Th^k(L)$,
4. verify whether or not a formula is an empirical theorem of the j^{th} degree fragment of L-theory with premises P based on $Th^k(L)$, if yes, then give the proof [6].

EnCal consists of the following major parts.

- EnCal-P: EnCal-P is a pattern-driven implementation of the inference rule of Modus Ponens for propositional logics. It can reason out logical theorem schemata of the k^{th} degree fragment of L.

- EnCal-Q: EnCal-Q is an extension of EnCal-P to deal with first order predicate logics. It has the ability to deal with individual quantifiers and variables. EnCal-Q can reason out logical theorem schemata of the k^{th} degree fragment of L.

- EnCal-Q2: EnCal-Q2 is an extension of EnCal-Q to deal with second order predicate logics. It has the ability to deal with predicate quantifiers and variables as well as individual quantifiers and variables. It can reason out logical theorem schemata of the k^{th} degree fragment of L.

- EnCal-E: In contrast to the EnCal-P and EnCal-Q , which are tools for reasoning about logical entailments, EnCal-E is a tool for reasoning out empirical entailments with logical theorem schema generated by EnCal-P and EnCal-Q.

- EnCal-E2: EnCal-E2 is an extension of EnCal-E to deal with second order theories.

- EnCal-T: EnCal-T is a tool kit for the user to edit input data for EnCal, transform the reasoning results into various forms specified by the user, and provide the user with various set operations on the reasoning results.

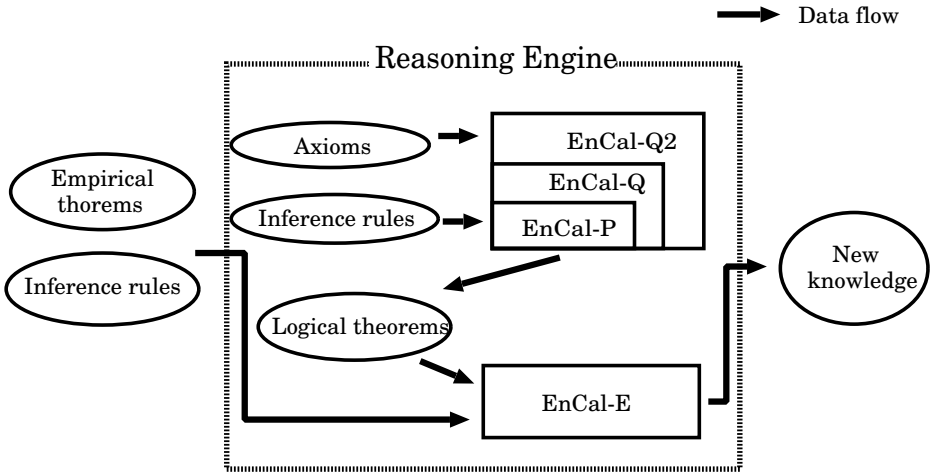

Figure 2.1: The relationship among the parts of EnCal

Figure 2.1 presents the relationship among the parts of EnCal. These parts were implemented by Cheng with LISP [6]. EnCal-P, EnCal-Q, EnCal-E were also implemented by Programming Language C, or C++ by author and et al. [26, 35, 34]

Chapter 3

A quantitative analysis of implicational paradoxes in classical mathematical logics

3.1 Introduction

Forward reasoning and/or deduction based on some fundamental logic system is indispensable to any computing system to discover new knowledge or predict future incidents. In principle, any logic system can be used as a fundamental logic to underlie forward reasoning and/or deduction processes with a certain purpose, if the logical consequence relation defined by the logic system is correspond or suitable to the purpose of the reasoning and/or deduction processes. However, from the viewpoint of practice, only those logic systems, which define logical consequence relations correspond or suitable to the purposes of reasoning and/or deduction processes required by problem solving in the real world, should be used as the underlying logic systems for the forward reasoning and/or deduction processes.

From the viewpoint of logic, Cheng qualitatively showed that classical mathematical logic, CML for short, its various classical conservative extensions, and traditional relevant logics, RL for short, are not suitable to underlying forward deduction for prediction and/or discovery because their logical theorems include a lot of paradoxes of conditional, and showed that strong relevant logics, SRL for short, are more hopeful candidates for the purpose [4, 7]. But, it is not clear that how 'bad' the CML, its various classical conservative extensions, and RL are, and how 'good' SRL are, since no quantitative analysis and discussion is reported until now.

In order to show that SRL are quantitatively suitable to underlying forward deduction, as the first step, we quantitatively investigate the implicational paradoxes in CML.

The rest of this chapter is organized as follows: section 3.2 gives a very simple explanation about CML and implicational paradoxes. Section 3.3 gives a very simple explanation about RL and SRL. Section 3.4 gives the method quantitative analysis on implicational paradoxes in CML, and section 3.5 shows an quantita-

tive analysis on implicational paradoxes in CML. Section 3.6 discusses about our analysis. Summary is given in section 3.7.

3.2 Classical mathematical logic and its various extensions

CML was established in order to provide formal languages for describing the structures with which mathematicians work, and the methods of proof available to them; its principal aim is a precise and adequate understanding of the notion of mathematical proof.

In CML, the notion of conditional, which is intrinsically intensional but not truth-functional, is represented by the truth-functional extensional notion of material implication (denoted by $\rightarrow$ in this thesis) that is defined as $A \rightarrow B =_{df} \neg(A \wedge \neg B)$ or $A \rightarrow B =_{df} \neg A \vee B$, where $\wedge$, $\vee$, and $\neg$ denote the notion of conjunction, disjunction, and negation, respectively. However, the material implication is intrinsically different from the notion of conditional in meaning (semantics). It is no more than an extensional truth-function of its antecedent and consequent but does not require that there is a necessarily relevant and conditional relation between its antecedent and consequent, i.e. the truth-value of the formula $A \rightarrow B$ depends only on the truth-values of A and B, though there could exist no necessarily relevant and conditional relation between A and B. It is this intrinsic difference in meaning between the notion of material implication and the notion of conditional that leads to the well-known "implicational paradox problem" in CML. The problem is that if one regards the material implication as the notion of conditional and regards every logical theorem of CML as an entailment or valid reasoning form, then a great number of logical axioms and logical theorems of CML, such as $A \rightarrow (B \rightarrow B)$, $B \rightarrow (\neg A \vee A)$, and so on, present some paradoxical properties and therefore they have been referred to in the literature as "implicational paradoxes" [1, 2, 20, 30, 41]. Because all implicational paradoxes are logical theorems of any CML-theory $T_{CML}(P)$, for a conclusion of a reasoning from a set P of premises based on CML, we cannot directly accept it as a correct conclusion in the sense of conditional, even if each of the given premises is regarded to be true and the conclusion can be regarded to be true in the sense of material implication.

Note that any classical conservative extension or non-classical alternative of CML where the classical account of validity is adopted as the logical validity criterion and the notion of conditional is directly or indirectly represented by the material implication has the similar problems as the above problems in CML [7].

Consequently, in the framework of CML, its various classical conservative extensions, even if a reasoning is valid in the sense of CML, neither the necessary relevance between its premises and conclusion nor the truth of its conclusion in the sense of conditional can be guaranteed necessarily.

3.3 Relevant logics

RL were constructed during the 1950s in order to find a mathematically satisfactory way of grasping the elusive notion of relevance of antecedent to consequent in conditionals, and to obtain a notion of implication which is free from the so-called "paradoxes" of material and strict implication [1, 2, 20, 30, 41]. Some major RL are "system E of entailment", "system R of relevant implication", and "system T of ticket entailment." Anderson and Belnap proposed variable-sharing as a necessary but not sufficient formal condition for the relevance between the antecedent and consequent of an entailment. The underlying principle of these relevant logics is the relevance principle , i.e. for any entailment provable in E, R, or T, its antecedent and consequent must share a sentential variable. Variable-sharing is a formal notion designed to reflect the idea that there be a meaning-connection between the antecedent and consequent of an entailment [1, 2, 20, 30, 41].

However, although RL have rejected those paradoxes of implication, there still exist some logical axioms or theorems in the logics, which are not so natural in the sense of conditional. Such logical axioms or theorems, for instance, are $(A \wedge B) \Rightarrow A$, $A \Rightarrow (A \vee B)$, and so on, where $\Rightarrow$ denotes the primitive intensional connective in the logics to represent the notion of conditional. Cheng named these logical axioms or theorems "conjunction-implicational paradoxes" and "disjunction-implicational paradoxes" [7].

In order to establish a satisfactory logic calculus of conditional to underlie relevant reasoning, the present Cheng has proposed some SRL , named Rc, Ec, and Tc [7, 10]. The logics require that the premises of an argument represented by a conditional include no unnecessary and needless conjuncts and the conclusion of that argument includes no unnecessary and needless disjuncts. As a modification of R, E, and T, Rc, Ec, and Tc reject all conjunction-implicational paradoxes and disjunction-implicational paradoxes in R, E, and T, respectively. Since the SRL are free of not only implicational paradoxes but also conjunction-implicational and disjunction-implicational paradoxes, in the framework of SRL, if a reasoning is valid, then both the relevance between its premises and its conclusion and the validity of its conclusion in the sense of conditional can be guaranteed in a certain sense of strong relevance.

The logical connectives, axiom schemata, and inference rules of SRL are as follows [7, 10]:

Primitive logical connectives:

$\Rightarrow$: entailment

$\neg$: negation

$\wedge$: extensional conjunction

Defined logical connectives:

$\otimes$: intensional conjunction, $A \otimes B =_{df} \neg(A \Rightarrow \neg B)$

$\oplus$: intensional disjunction, $A \oplus B =_{df} \neg A \Rightarrow B$

$\Leftrightarrow$: intensional equivalence, $A \Leftrightarrow B =_{df} (A \Rightarrow B) \otimes (B \Rightarrow A)$

$\vee$: extensional disjunction, $A \vee B =_{df} \neg(\neg A \wedge \neg B)$

$\rightarrow$: material implication, $A \rightarrow B =_{df} \neg(A \wedge \neg B)$ or $\neg A \vee B$

$\leftrightarrow$: extensional equivalence, $A \leftrightarrow B =_{df} (A \rightarrow B) \wedge (B \rightarrow A)$

Quantifiers:

$\forall$: universal quantifier

$\exists$: existential quantifier

These quantifiers are not independent and can be defined as follows:

$$\begin{aligned} \forall x A &=_{df} \neg\exists x \neg A, \\ \exists x A &=_{df} \neg\forall x \neg A. \end{aligned}$$

Axiom schemata

E1: $A \Rightarrow A$

E2: $(A \Rightarrow B) \Rightarrow ((C \Rightarrow A) \Rightarrow (C \Rightarrow B))$

E2': $(A \Rightarrow B) \Rightarrow ((B \Rightarrow C) \Rightarrow (A \Rightarrow C))$

E3: $(A \Rightarrow (A \Rightarrow B)) \Rightarrow (A \Rightarrow B)$

E3': $(A \Rightarrow (B \Rightarrow C)) \Rightarrow ((A \Rightarrow B) \Rightarrow (A \Rightarrow C))$

E3": $(A \Rightarrow B) \Rightarrow ((A \Rightarrow (B \Rightarrow C)) \Rightarrow (A \Rightarrow C))$

E4: $(A \Rightarrow ((B \Rightarrow C) \Rightarrow D) \Rightarrow ((B \Rightarrow C) \Rightarrow (A \Rightarrow D))$

E4': $(A \Rightarrow B) \Rightarrow (((A \Rightarrow B) \Rightarrow C) \Rightarrow C)$

E4": $((A \Rightarrow A) \Rightarrow B) \Rightarrow B$

E4"': $(A \Rightarrow B) \Rightarrow ((B \Rightarrow C) \Rightarrow (((A \Rightarrow C) \Rightarrow D) \Rightarrow D))$

E5: $(A \Rightarrow (B \Rightarrow C)) \Rightarrow (B \Rightarrow (A \Rightarrow C))$

E5': $A \Rightarrow ((A \Rightarrow B) \Rightarrow B)$

N1: $(A \Rightarrow (\neg A)) \Rightarrow (\neg A)$

N2: $(A \Rightarrow (\neg B)) \Rightarrow (B \Rightarrow (\neg A))$

N3: $(\neg(\neg A)) \Rightarrow A$

C1: $(A \wedge B) \Rightarrow A$

C2: $(A \wedge B) \Rightarrow B$

C3: $((A \Rightarrow B) \wedge (A \Rightarrow C)) \Rightarrow (A \Rightarrow (B \wedge C))$

C4: $(LA \wedge LB) \Rightarrow L(A \wedge B)$, where $LA =_{df} (A \Rightarrow A) \Rightarrow A$

D1: $A \Rightarrow (A \vee B)$

D2: $B \Rightarrow (A \vee B)$

D3: $((A \Rightarrow C) \wedge (B \Rightarrow C)) \Rightarrow ((A \vee B) \Rightarrow C)$

DCD: $(A \wedge (B \vee C)) \Rightarrow ((A \wedge B) \vee C)$

C5: $(A \wedge A) \Rightarrow A$

C6: $(A \wedge B) \Rightarrow (B \wedge A)$

C7: $((A \Rightarrow B) \wedge (B \Rightarrow C)) \Rightarrow (A \Rightarrow C)$

C8: $(A \wedge (A \Rightarrow B)) \Rightarrow B$

C9: $\neg(A \wedge \neg A)$

C10: $A \Rightarrow (B \Rightarrow (A \wedge B))$

IQ1: $\forall x(A \Rightarrow B) \Rightarrow (\forall xA \Rightarrow \forall xB)$

IQ2: $(\forall xA \wedge \forall xB) \Rightarrow \forall x(A \wedge B)$

IQ3: $\forall xA \Rightarrow A[t/x]$ (if x may appear free in A and t is free for x in A, i.e. free variables of t do not occur bound in A)

IQ4: $\forall x(A \Rightarrow B) \Rightarrow (A \Rightarrow \forall xB)$ (if x does not occur free in A)

IQ5: $\forall x_1 \cdots \forall x_n(((A \Rightarrow A) \Rightarrow B) \Rightarrow B)$ $(0 \leq n)$

Inference Rules:

$\Rightarrow$E: "from A and $A \Rightarrow B$ to infer B" (Modus Ponens)

$\wedge$I: "from A and B infer $A \wedge B$" (Adjunction)

$\forall$I: "if A is an axiom, so is $\forall xA$" (Generalization of axioms)

Thus, various relevant logic systems may now defined as follows, where we use 'A | B' to denote any choice of one from two axiom schemata A and B.

Te = {E1, E2, E2', E3| E3" } + $\Rightarrow$E

Ee = {E1, E2 | E2', E3 | E3', E4 | E4'} + $\Rightarrow$E

Ee = {E2', E3, E4"} + $\Rightarrow$E

Ee = {E1, E3, E4"'} + $\Rightarrow$E

Re = {E1, E2 | E2', E3 | E3', E5 | E5'} + $\Rightarrow$E

Ten = Te + {N1, N2, N3 }

Een = Ee + {N1, N2, N3 }

Ren = Re + {N2, N3 }

T = Ten + {C1 $\sim$ C3, D1 $\sim$ D3, DCD } + $\wedge$I

E = Een + {C1 $\sim$ C4, D1 $\sim$ D3, DCD } + $\wedge$I

R = Ren + {C1 $\sim$ C3, D1 $\sim$ D3, DCD } + $\wedge$I

Tc = Ten + {C3, C5 $\sim$ C10}

Ec = Een + {C3 $\sim$ C10}

Rc = Ren + {C3, C5 $\sim$ C10}

TcQ = Tc + {IQ1 $\sim$ IQ4 } + $\forall$I

EcQ = Ec + {IQ1 $\sim$ IQ5 } + $\forall$I

RcQ = Rc + {IQ1 $\sim$ IQ4 } + $\forall$I

Here, Te, Ee, and Re are the purely implicational fragments of T, E, and R, respectively, and the relationship between Ee and Re is known as Re = Ee + $A \Rightarrow LA$; Ten, Een, and Ren are the implication-negation fragments of T, E, and R, respectively; TcQ, EcQ, and RcQ are predicate strong relevant logics proposed by Cheng [11].

The strong relevance principle, SRP for short, is the one of principles in SRL [45]: every sentential variable in a well-formed formula A occurs at least once as an antecedent part and at least once as a consequent part. The definition of an antecedent part and a consequent part is as follows, let A, B and C be well-formed formulas,

1. A is a consequent part of A,
2. if $\neg B$ is a consequent part (antecedent part) of A, then B is an antecedent part (consequent part) of A,
3. if $B \Rightarrow C$ is a consequent part (antecedent part) of A, then B is an antecedent part (consequent part) of A, and C is consequent (antecedent part) of A,
4. if $B \wedge C$ or $B \vee C$ is a consequent part (antecedent part) of A, then both B and C are consequent parts (antecedent parts) of A.

If A is a theorem of Rc, Ec, or Tc, then A satisfies SRP [45]. If A is a theorem of Ren, Een, or Ten, then A satisfies SRP [1]. On the other hand, implicational paradoxes in CML do not satisfy SRP, because SRP is a principle which formally guarantees the relationship between antecedent and consequent. Therefore, we can distinguish implicational paradoxes from axioms and logical theorems in CML by whether a well-formed formula satisfies SRP or not.

3.4 The method of quantitative analysis

The purpose of this thesis is to investigate implicational paradoxes in CML, we therefore focus on the axiomatic system of CML with only implication and negation, and discuss about implicational paradoxes on the schemata of well-formed formulas [25].

In this thesis, a schema of a well-formed formula is defined as the formula $A \in F(L)$ obtained by applying following operation to the well-formed formula.

Operation

1. Let $\{O\} = \{o_1, o_2, \cdots\}$ denote a set of symbols with order relation which is not included the vocabulary of logic system L.
2. $i \leftarrow 1$
3. Continue following operations until all sentential variables in a well-formed formula A are replaced with elements of $\{O\}$.

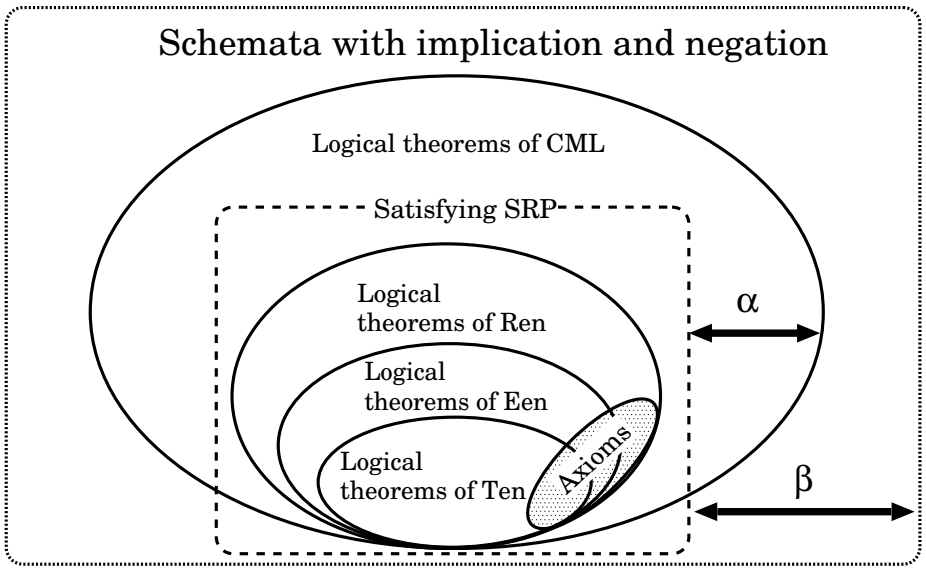

Figure 3.1: The relationship among sets of schemata

(a) Replace all sentential variables, which are i^{th} kind from the head of the A, with a i^{th} element of $\{O\}$.

(b) $i \leftarrow i + 1$.

In this thesis, 'pattern variables' denotes the elements of $\{O\}$.

A schema of a well-formed formula A is a k^{th} degree schema if and only if a certain well-formed formula B is a k^{th} degree formula if A is obtained from B by the above operation. A schema A is a k^{th} degree axiom schema or a k^{th} degree logical theorem schema of a logic system L if and only if a certain well-formed formula B is a k^{th} degree formula, and is also an axiom or a logical theorem of L if A is obtained from B by the above operation. k^{th} degree schemata fragment of formal logic system L is a set which includes all j^{th} degree schemata of L $(1 \leq j \leq k)$, and denoted by $F_k(L)$. k^{th} degree logical theorem schemata fragment of formal logic system L is the set which consists of all j^{th} degree axiom schemata and logical theorem schemata of L $(1 \leq j \leq k)$, and denoted by $Th_k(L)$. Note that $Th_k(L)$ is a different set of $Th^k(L)$ which is defined at chapter 2. $FS_k(CML)$ denotes the set which consists of all schemata satisfying SRP in $F_k(CML)$.

Figure 3.1 shows the relationship among a set of logical theorems in CML, a set of logical theorems satisfying SRP in CML, and sets of logical theorems in Ren, Een, or Ten. Note that the vocabulary of Ren, Een, and Ten has not only entailment $\Rightarrow$ as a primitive logical connective, but also material implication $\rightarrow$ as a defined logical connective [7]. Actually, in viewpoint from syntax, the relationship between well-formed formulas of RL, denoted by WFF_{RL}, and that of CML, denoted by WFF_{CML}, is

$$\mathrm{WFF}_{CML} \subset \mathrm{WFF}_{RL}. \tag{3.1}$$

However, in this thesis, we regard material implication $\rightarrow$ in CML and entailment $\Rightarrow$ in RL as a same connective to represent the notion of conditional.

$F_k(CML)$ is as same as $F_k(Ren)$, $F_k(Een)$, and $F_k(Ten)$. Let $ThS_k(CML)$ denote the set which is $FS_k(CML) \wedge Th_k(CML)$. $IP_k(CML)$ denotes the set of all implicational paradoxes in $Th_k(CML)$, i.e. $IP_k(CML) = Th_k(CML) - ThS_k(CML)$. There is following relationship among $Th_k(CML)$, $ThS_k(CML)$, $Th_k(Ren)$, $Th_k(Een)$, and $Th_k(Ten)$:

$$Th_k(Ten) \subset Th_k(Een) \subset Th_k(Ren) \subseteq ThS_k(CML) \subset Th_k(CML). \tag{3.2}$$

Note that there is no report about whether $Th_k(Ren)$ and $ThS_k(CML)$ are same or not. We can know how good SRL and RL are than CML, by comparing the number of elements of $IP_k(CML)$ with $ThS_k(CML)$ because $ThS_k(CML)$ is super set or same set of $Th_k(Ren)$, and is super set of $Th_k(Een)$ and $Th_k(Ten)$.

However, it is very difficult, not impossible, to obtain all elements of $Th_k(CML)$ and $ThS_k(CML)$ when k is a large number. In syntax approach, that is, deduction, we cannot stop the deduction process because we cannot know the number of all elements of $Th_k(CML)$ before we obtain all elements. On the other hand, in semantic approach, e.g. the tableau method, huge computational resource is needed because the number of formulas which should be checked is huge.

From the relationship between the difference between the number of elements of $F_k(CML)$ and $FS_k(CML)$ and the degree of implication, we analogize the relationship between the difference between the number of elements of $Th_k(CML)$ and $ThS_k(CML)$ and the degree of implication. It is possible to regard that a certain schema consists of a certain permutation of pattern variables and a certain connection of logical connectives. On a certain connection of logical connectives, we regard following conditions as limitations of the possible kinds of permutations of pattern variables: (1) a schema is an element of $F_k(CML)$, (2) a schema is an element of $FS_k(CML)$, (3) a schema is an element of $Th_k(CML)$, and (4) a schema is an element of $ThS_k(CML)$. For example, while a certain connection of logical connectives is $\gamma \Rightarrow (\gamma \Rightarrow \gamma)$ (γ means the place to put pattern variables), on limitation (1) $\sim$ (4), the possible kinds of permutations are respectively 5 kinds('aaa', 'aab', 'aba', 'abb', and 'abc'), 1 kind ('aaa'), 3 kinds ('aaa', 'aba' and 'abb'), and 1 kind('aaa'). Note that 'a, b, c' denote pattern variables. The difference between the number of elements of $F_k(CML)$ and $FS_k(CML)$ is caused by the difference between the severeness of limitation (1) and (2), and similarly, that of $Th_k(CML)$ and $ThS_k(CML)$ is caused by the difference between the severeness of limitation (3) and (4). The difference between the severeness of limitations becomes clear as the number of places to put pattern variables in a schema becomes large. Let m be the kinds of pattern variables which occur in a schema, and n be the number of places to put pattern variables in the schema.

- m is equal or less than n if a schema is an element of $F_k(CML)$.
- m is equal or less than $n-1$ if a schema is an element of $Th_k(CML)$.
- m is equal or less than $n/2$ if a schema is an element of $FS_k(CML)$ or $ThS_k(CML)$.

The possible kinds of permutations of pattern variables increase explosively in proportion to the increasement of the kinds of pattern variables in a schema. The number of places to put pattern variables in a schema becomes large as the degree of implications becomes large. Hence, the difference between the number of elements of $F_k(CML)$ and $FS_k(CML)$ becomes larger and larger as the degree of implication becomes large, and similarly, the difference between the number of elements of $Th_k(CML)$ and $ThS_k(CML)$ becomes larger and larger, too. That is, these differences have same tendency against the increasement of the degree of implication.

3.5 Quantitative analysis

In this section, we calculated the number of elements of $F_k(CML)$ and $FS_k(CML)$.

The number of elements of $F_k(CML)$ is calculated with the number of kinds of schemata without pattern variables and the number of elements of their equivalence classes. A formula is a schema without pattern variables if and only if a certain schema whose all pattern variables are removed. Let A and B denote schemata of well-formed formulas. Let $R(A, B)$ denote a relation that both schemata without pattern variables of A and B are same. A set of schemata of well-formed formulas can divide into equivalence classes because $R(A, B)$ is an equivalence relation.

'Label' denotes the places of pattern variables in a schema without pattern variables. 'Label number' denotes the number of labels in a schema without pattern variables. The number of elements of an equivalence classes of a schema without pattern variables is the possible kinds of permutations of pattern variables, while label number is i.

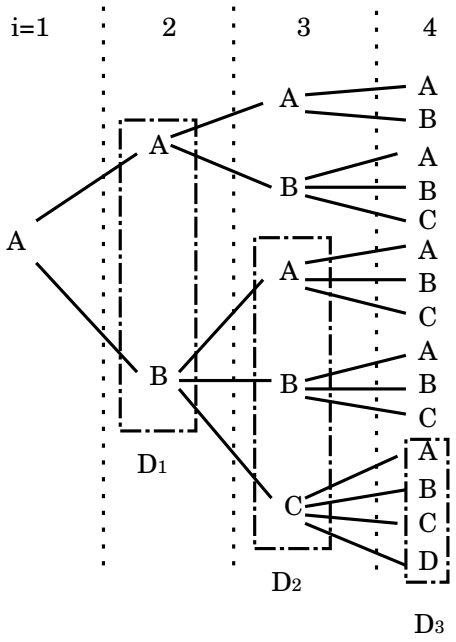

Figure 3.2: The permutations of pattern variables and the label numbers

Figure 3.2 is the relationship between label numbers and the number of permutations of pattern variables. In the figure, pattern variables are represented by capital alphabet. Let i denotes the label number. Di in the figure defined as follows,

$$D_i = i + 1. \tag{3.3}$$

Let $p[i]$ denote the number of permutations of pattern variables in label number i. $p[i]$ is defined as follows,

$$\begin{aligned} p[1] &= 1, \\ p[i] &= \sum_{k=1}^{i-1} \left(S_{i-1}^{k} \cdot D_k \right), (i > 1). \end{aligned} \tag{3.4}$$

The coefficient $S_n^k,\ (1 \leq n,\ 1 \leq k)$ in eq. (3.4) is the stirling numbers of the second kind. The stirling numbers of second kind is defined as follows,

$$\begin{aligned} S_n^k &= 0,\ (n < k), \\ S_n^1 &= 1, \\ S_n^n &= 1, \\ S_n^k &= S_{n-1}^{k-1} + S_{n-1}^{k} \cdot k,\ (k < n). \end{aligned} \tag{3.5}$$

Let $L[k][i]$ denote the number of kinds of schemata without pattern variables while the degree of implications is k and label numbers is i. $L[k][i]$ is defined as follows,

$$\begin{aligned} L[k][i] &= 0,\ (k+1 \leq i \leq 2^k), \\ &= \sum_{j=k}^{2^{(k-1)}} \left(\sum_{e=0}^{k-1} L[k-1][j] \cdot L[e][h] \cdot 4 \right),\ (e \neq k-1,\ j+h=i), \\ &= \sum_{j=k}^{2^{(k-1)}} \left(\sum_{e=0}^{k-1} L[k-1][j] \cdot L[e][h] \cdot 2 \right),\ (e = k-1,\ j+h=i). \end{aligned} \tag{3.6}$$

The initial values of above equation are as follows,

$$\begin{aligned} L[0][1] &= 2, \\ L[1][2] &= 8. \end{aligned} \tag{3.7}$$

From eq. (3.4) and eq. (3.6), the number of elements of $F_k(CML)$ is defined as follows,

$$\sum_{i=1}^{k} \left(\sum_{j=i+1}^{2^i} L[i][j] \cdot p[j] \right). \tag{3.8}$$

On the other hand, we can calculate the number of elements of $FS_k(CML)$ from following the equivalence relation. Let $F'_k(CML)$ denote the set of schemata without pattern variables in $F_k(CML)$. We can divide $F'_k(CML)$ into equivalence classes by using the degree of implications k, label numbers i, the number of labels a which are antecedent parts of the schema without pattern variables, the number

of antecedent parts for short, and the number of labels c which are consequent parts of the schema without pattern variables, the number of consequent parts for short. Note that k is zero or a natural number and $k+1 \leq i \leq 2^k$, $a+c=i$. The number of kinds of elements satisfying SRP on a certain schema without pattern variables is calculated by the number of antecedent parts and the number of consequent parts on the schema without pattern variables.

Let $K=(k,i,a,c,s)$ denote the representative where the degree of implications k, label numbers i, the number of antecedent parts a, the number of consequent parts c and the number of elements of this equivalence class s. s is a value which totaled the product of the number of elements of all order pair (M,V) in $F'_k(CML)$. Those order pairs fill the following relations, if other representative $M=(m,n,o,p,q)$ and $V=(v,w,x,y,z)$ are not same K.

$$\begin{aligned} k &= max(m,v)+1 \\ i &= n+w \\ a &= p+x \\ c &= o+y \\ s &= s+q\cdot z \end{aligned} \tag{3.9}$$

The initial values are $(0,0,0,1,1)$ and $(0,1,0,1,1)$ when the degree of implication is 0, and are $(1,2,1,1,4)$, $(1,2,2,0,2)$, and $(1,2,0,2,2)$ when the degree of implication is 1. The number of elements of the equivalence classes is calculated from the initial values by applying the relation of eq. (3.9).

While the number of antecedent parts is a, and the number of consequent parts is c, the number of kinds of elements satisfying SRP on a certain schema without pattern variables is calculated as follows,

$$\sum_{j=1}^{min(a,c)} Q\left(j, min(a,c)\right)\cdot j!\cdot Q\left(j, max(a,c)\right). \tag{3.10}$$

$Q(p,n)$ is defined as follows,

$$\begin{aligned} Q(p,n) &= 0,\ (n<p) \\ Q(p,p) &= 1, \\ Q(p,p+1) &= \sum_{j=1}^{p} j, \\ Q(p,p+m) &= \sum_{j=1}^{p}(j\cdot\delta[m-1][j]),\ (1<m). \end{aligned} \tag{3.11}$$

$\delta[i][j]$ is defined as follows,

$$\begin{aligned} \delta[1][j] &= \sum_{k=j}^{p} k,\ (1\leq j\leq p) \\ \delta[i][j] &= \sum_{k=j}^{p}(k\cdot\delta[i-1][k]),\ (1<i,\ 1\leq j\leq p). \end{aligned} \tag{3.12}$$

Table 3.1: The number of elements of $F_k(CML)$ and $FS_k(CML)$

degree	$F_k(CML)$ (a)	$FS_k(CML)$ (b)
1	1.60×10^1	4.00×10^0
2	2.26×10^3	2.60×10^2
3	1.67×10^8	8.90×10^6
4	2.92×10^{19}	5.15×10^{17}
5	1.63×10^{45}	6.31×10^{42}
6	4.29×10^{103}	2.13×10^{100}
7	1.02×10^{235}	3.09×10^{230}
8	8.15×10^{527}	5.61×10^{521}

Table 3.2: The number of elements of $Th_k(CML)$ and $ThS_k(CML)$

degree	$Th_k(CML)$ (c)	$ThS_k(CML)$ (d)	$(a-b)/b$	$(c-d)/d$
1	2.00×10^0	2.00×10^0	3.00	0.00
2	3.14×10^2	9.80×10^2	7.68	2.20
3	4.19×10^7	2.44×10^6	17.78	7.13

We can calculate the number of elements of $FS_k(CML)$ from eq. (3.9) and (3.10). Table 3.1 shows the number of elements of $F_k(CML)$ and $FS_k(CML)$ $(1 \leq k \leq 8)$. Figure 3.3 shows the relationship between the degree of implication, and the difference between the number of elements of $F_k(CML)$ and $FS_k(CML)$. The y-axis shows the value of $(a-b)/b$, where a and b mean the number of elements of $F_k(CML)$ and $FS_k(CML)$ respectively in table 3.1. The x-axis shows the degree of implication. Fig. 3.3 shows that the difference between the elements of $F_k(CML)$ and $FS_k(CML)$ becomes larger as the degree of implication becomes large.

We also get the number of elements of $Th_k(CML)$ and $ThS_k(CML)$ $(k = 1, 2, 3)$ by scripts based on the tableau method and the definition of SRP. Table 3.2 is the number of elements of $Th_k(CML)$ and $ThS_k(CML)$ $(k = 1, 2, 3)$. In table 3.2, degree means the degree of implication. $Th_k(CML)$ and $ThS_k(CML)$ mean the number of elements of them. $(a-b)/b$ means the value of $(a-b)/b$ where a and b denote the number of elements of $F_k(CML)$ and $FS_k(CML)$ respectively, that is, the value denote β in fig. 3.1. $(c-d)/d$ means the value of $(c-d)/d$ where c and d denote the number of elements of $Th_k(CML)$ and $ThS_k(CML)$ respectively, that is, the value denote α in fig. 3.1.

The table shows that the number of elements of $IP_3(CML)$, the set of all implicational paradoxes in $Th_3(CML)$, is 7.13 times as many as that of $ThS_3(CML)$. It also shows that the difference between the number of elements of $Th_k(CML)$ and $ThS_k(CML)$ becomes large as the degree of implication increases although that of $Th_k(CML)$ and $ThS_k(CML)$ is less than that of $F_k(CML)$ and $FS_k(CML)$.

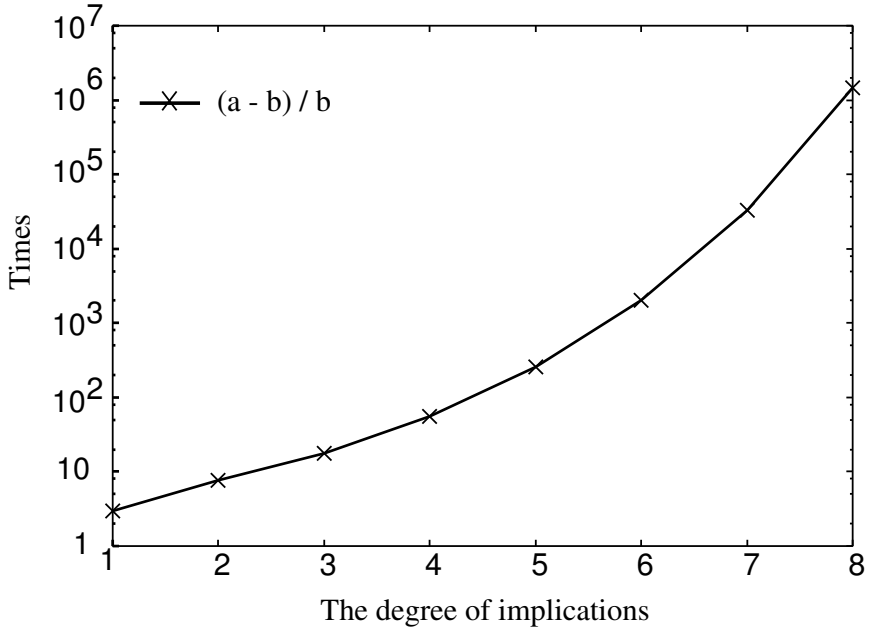

Figure 3.3: The degree and the difference between $F_k(CML)$ and $FS_k(CML)$

Thus we can think that the number of elements of $IP_k(CML)$ is more than 7.13 times that of $ThS_k(CML)$ when k is more than 3.

3.6 Discussion

The difference between the number of elements of $Th_k(Ren)$ and $IP_k(CML)$ must be larger than that of $ThS_k(CML)$ and $IP_k(CML)$ because eq. 3.2 shows $Th_k(Ren)$ is a superset of $Th_k(Een)$ and $Th_k(Ten)$, and $Th_k(Ren)$ is a same or subset of $ThS_k(CML)$. Thus $Th_3(CML)$ has the number of implicational paradoxes more than 7.13 times that of elements of $Th_3(Ren)$, $Th_3(Een)$, or $Th_3(Ten)$. Moreover, we can think that the number of elements of $IP_k(CML)$ becomes larger than that of $Th_k(Ren)$, $Th_k(Een)$, or $Th_k(Ten)$ as the k becomes large.

Implicational paradoxes spoil the validity of forward deduction, and unnecessarily lengthen the execution time of automated forward deduction. A conclusion may not have the relationship between premises and the conclusion if it is deduced from premises including implicational paradoxes or another conclusions deduced from premises including implicational paradoxes. The process of forward deduction which deduces such conclusion is not valid. Consequently, on automated forward deduction, it is waste to process implicational paradoxes and conclusions deduced from premises which include implicational paradoxes.

Thus, automated forward deduction based on SRL or RL guarantees the valid deduction process, and its execution time is shorter than the execution time of automated forward deduction based on CML although both deduce same conclusions from same premises. Therefore, SRL or RL is quantitatively more suitable

by far than CML, as a logic system underlying forward deduction.

3.7 Summary

We have investigated the number of implicational paradoxes in axiomatic system of CML with only implication and negation. The investigation shows that the number of elements of $IP_3(CML)$, the set of all implicational paradoxes in $Th_3(CML)$, is 7.13 times as many as that of $ThS_3(CML)$. The difference between the number of elements of $Th_k(CML)$ and $ThS_k(CML)$ must become larger and larger like that of $F_k(CML)$ and $FS_k(CML)$, as the degree of implication becomes large. Thus, the number of elements of $IP_k(CML)$ must be more than 7.13 times that of $ThS_k(CML)$ when k is more than 3.

Implicational paradoxes spoil the validity of forward deduction, and unnecessarily lengthen the execution time of automated forward deduction. Consequently, SRL and RL are quantitatively more suitable by far than CML, as a logic system underlying forward deduction because the logic systems include no implicational paradoxes. Therefore automated deduction should be based on SRL by far than CML in order to implement a practical forward deduction engine.

Chapter 4

Improving the performance of automated forward deduction

4.1 Introduction

Automated forward deduction is an indispensable component of many application systems for prediction and/or discovery. The performance of automated forward deduction is crucial to its applicability. Since a forward deduction system working for prediction and/or discovery has no explicitly specified proposition or theorem given previously as goal, it often deduces many redundant intermediates, i.e. instances of those that have previously deduced. Thus a large amount of execution time and a large amount of main memory are needed to perform automated forward deduction.

In order to solve the performance problem of forward deduction engines, we investigate the relationship between the execution time of a forward deduction engine and the amount of given premises and deduced conclusions. Based on the investigation, we propose a parallelization model which is a kind of master-slave model for automated forward deduction, and show the effectiveness of that model by implementing EnCal based on the model.

The rest of this chapter is organized as follows: section 4.2 gives an analysis for execution time of automated forward deduction. Section 4.3 presents a parallelization model of automated forward deduction, and section 4.4 presents the model of parallelization version of EnCal in order to show a case study of improving the performance of automated forward deduction by parallel processing. Section 4.5 shows our implementation of the parallelization version of EnCal and our experiments on a shared-memory parallel computer and clusters of PCs. Section 4.6 discusses our experimental results. Summary is given in Section 4.7.

4.2 Computational complexity of automated forward deduction

Automated forward deduction is a process of deducing new and unknown conclu-

sions automatically by applying inference rules to premises and previously deduced conclusions repeatedly until some previously specified condition is satisfied. An automated forward deduction consists of 3 parts as follows:

1. Initialization part: it takes in premises, some termination conditions, and inference rules.
2. Forward deduction part: about each inference rule, it repeats following processes until it deduces no new conclusion.
   (a) Matching process: it seeks and picks up some premises, which are to be applied to an inference rule, from a set of premises. Then it matches the premises to an inference rule.
   (b) Deduction process: it applies an inference rule to the premises which were matched at the matching process.
   (c) Duplication checking process: it compares a conclusion which was deduced at the deduction process with all previously deduced conclusions and premises in order to check whether it is a duplicate or not.
   (d) Adding process: it adds the conclusion which was judged to be new at the duplication checking process to the set of premises.
3. Outputting part: it outputs all new conclusions into files.

In general, in order to get new conclusions, forward deduction engines perform the above four processes in forward deduction part repeatedly for each inference rule, until some termination conditions are satisfied. The each process in the forward deduction part depends on the result of previous process prior to it.

We present some equations about the execution time and the amount of data of forward deduction engines if all inference rules apply to all given premises. Let n be the number of previously given premises, i be the number of inference rules, r be the number of premises required by an inference rule, in this assumption all inference rules require r premises, and τ_m be the execution time of judging whether an inference rule can apply to some premises or not and matching the inference rule to the premises at the matching process. The execution time at the matching process is

$$n^r \cdot i \cdot \tau_m. \tag{4.1}$$

Let τ_d be the execution time of deducing a conclusion. The execution time at the deduction process is at most

$$n^r \cdot i \cdot \tau_d. \tag{4.2}$$

Let τ_c be the execution time of comparing a deduced conclusion at the deduction process with a previously deduced conclusion or a given premise at the duplication checking process. The execution time at the duplication checking process is at most

$$\begin{aligned} & n^r \cdot n \cdot i \cdot \tau_c + (1 \cdot \tau_c + 2 \cdot \tau_c + \ldots + (i \cdot n^r - 1) \cdot \tau_c) \\ = \; & (n^{r+1} \cdot i + \sum_{k=1}^{i \cdot n^r - 1} k) \cdot \tau_c \\ = \; & \frac{1}{2} \left\{ i^2 \cdot n^{2r} + 2 \cdot i \cdot n^{r+1} - i \cdot n^r \right\} \cdot \tau_c. \end{aligned} \tag{4.3}$$

Let τ_a be the execution time of adding a conclusion into a set of premises. The execution time at the adding process is at most

$$n^r \cdot i \cdot \tau_a. \tag{4.4}$$

In this assumption, the total execution time of above four processes is

$$\begin{aligned} & n^r \cdot i \cdot (\tau_m + \tau_d + \tau_a) + \frac{1}{2}\left\{i^2 \cdot n^{2r} + 2 \cdot i \cdot n^{r+1} - i \cdot n^r\right\} \cdot \tau_c \\ \approx \quad & O(i^2 \cdot n^{2r}). \end{aligned} \tag{4.5}$$

On the other hand, the number of deduced conclusions in forward deduction is huge. Let n be the number of premises, i be the number of inference rules, and r be the number of premises required by an inference rule. The number of deduced conclusions at deduction process is at most

$$n^r \cdot i. \tag{4.6}$$

Let R_1 denote the set of the conclusions which are deduced from given premises. The number of conclusions which are deduced from R_1 as premises at deduction process is at most

$$n^{2 \cdot r} \cdot i^{r+1}. \tag{4.7}$$

R_j denotes the set of conclusions which are deduced from R_{j-1} at deduction process $(2 \leq j)$. The number of conclusions which are deduced from R_{j-1} is at most

$$n^{j \cdot r} \cdot i^{\sum_k^{j-1} k}. \tag{4.8}$$

In the adding process, deduced new conclusions are added into the set of premises. Then processing of forward deduction is again repeated using the conclusions as premises. Thus, even the number of premises is few, the number of deduced conclusions becomes large easily.

We therefore can regard the execution time of a forward deduction engine as $O(N^{2r})$, where N denotes the number of data which includes both finally deduced conclusions and given premises, because the number of data N is rather large than the number of given inference rules i. A useful forward deduction engine must get enough effective conclusions in an acceptable time. Thus we must improve the performance of a forward deduction engine.

The performance of a forward deduction engine can be improved by aspects. One is shortening the execution time of each process in a forward deduction engine. Another is reducing the processing load. The first aspect focuses on the execution time of each process, i.e. shortening τ_m, τ_d, τ_c and τ_a. It is expected that it can shorten the execution time of a forward deduction engine at a constant rate without the increasement in the number of deduced conclusions and given premises. The second aspect focuses on the processing load: the order of the execution time become less than $O(N^{2r})$ where N is the number of deduced conclusions and given premises and r is the number of premises required by an inference rule. This aspect can be classified into two approaches. One is to narrow down the range of data

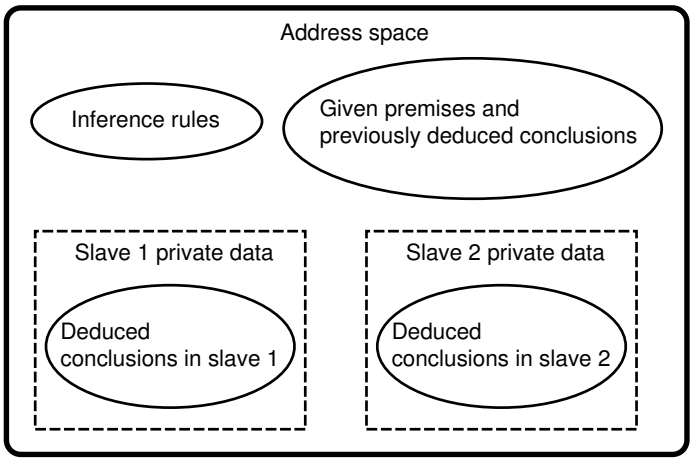

Figure 4.1: Data arrangement of parallelization model of automated forward deduction

being processed. As at a certain time, it is not necessarily to process all data at a certain process of a forward deduction engine. It is better that only certain data which must be processed is done. It is also expected that this approach can shorten the execution time at a constant rate without the increasement in the number of deduced conclusions and given premises. Other is reducing the processing load on one processor by parallel processing. It is expected that this approach can shorten the execution time in proportion to the number of using processors.

We focus on reducing the processing load on one processor by parallel processing. This approach is flexible to the increasement in the number of deduced conclusions and given premises since it can increase the number of processors.

4.3 A parallelization model of automated forward deduction

We summarize the processing features of automated forward deduction.

1. The each process in the forward deduction part depends on the results of previous process prior to it.

2. Previously deduced conclusions and given premises are accessed frequently at the matching process and the duplication checking process.

3. Independently of other sets, matching process, deduction process, and adding process can process the set which consists of a certain inference rule and a certain set of premises.

4. The certainty to detect the duplication in duplication checking process decreases, if it cannot compare a deduced conclusion with all previously deduced conclusions and given premises.

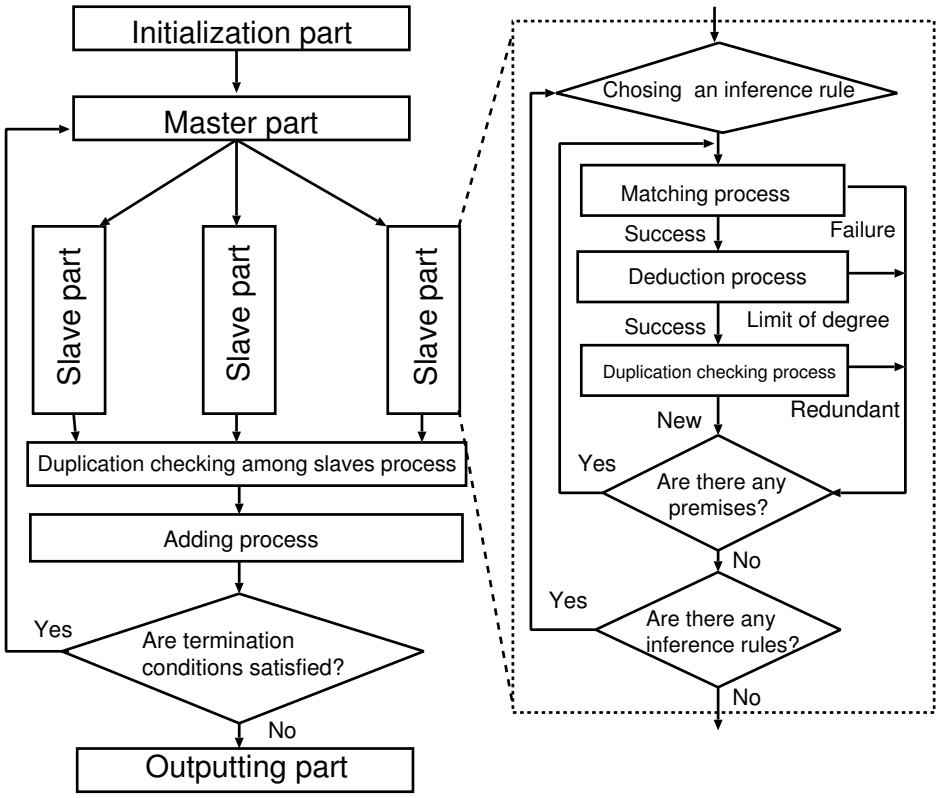

Figure 4.2: A parallelization model of automated forward deduction

We design a parallelization model of automated forward deduction based on master-slave model. Master-slave model based on agenda parallelism paradigm [3] is a suitable model for parallel automated forward deduction because of above features 1, 3 and 4.

Figure 4.1 shows the data arrangement of parallelization model of automated forward deduction. In fig. 4.1, 'slave 1 private data' means data which other slaves cannot access. 'slave 2 private data' is so. The inference rules, previously deduced conclusions, and given premise are shared by slaves because of the above feature 2. The deduced conclusions in a slave are put on the slave's address space, because the writing to a memory occurs.

Figure 4.2 shows the model of parallelization version of automated forward deduction. The model consists of 4 parts: initialization part, master part, slave part, and outputting part, where matching process, deduction process, duplication checking process, and adding process are as same as the processes of sequential automated forward deduction.

1. Initialization part: it takes in premises, some termination conditions, and inference rules.
2. Master part: it evenly divides all sets of given premises and previously deduced conclusions to all slaves. Note that the sets are requires inference rules as premises. It changes a slave part after dividing premises and conclusions to slaves.

3. Slave part: the following processes are repeated independently of other slaves.
   (a) Matching process.
   (b) Deduction process.
   (c) Duplication checking process.
4. Duplication checking among slaves process: it detects and reduces the duplicate which is not detected at the duplication checking process in slave part.
5. Adding process.
6. Outputting part.

It is necessary for efficient duplication check to be able to access all previously deduced conclusions. However, the set of deduced conclusions at a slave is not referred by other slaves. Hence, this model needs the duplication checking among slaves process.

We present some equations about the execution time of duplication checking among slaves process if all slaves deduces same number of conclusions. Let m be the number of deduced conclusions in a slave, and p be the number of slaves. Let τ_c be the execution time of comparing a deduced conclusion at the deduction process with a previously deduced conclusion at duplication checking among slaves process or duplication checking process. The execution time of duplication checking among slaves process is at most

$$\sum_{k=1}^{p-1} m^2 \cdot \tau_c = (p-1) \cdot m^2 \cdot \tau_c. \tag{4.9}$$

Let n be the number of previously given premises, i be the number of inference rules, r be the number of premises required by an inference rule, in this assumption all inference rules require r premises, and p be the number of slaves. In this model, the number of deduced conclusions in a slave is at most n/p. Thus the execution time at a slave part is as follows from eq. (4.5) and eq. (4.9),

$$\begin{aligned} &\frac{n^r}{p} \cdot i\,(\tau_m + \tau_d + \tau_a) + \frac{1}{2p}\left\{ i^2 \cdot \frac{n^{2r}}{p} + 2 \cdot i \cdot n^{r+1} - i \cdot n^r \right\} \cdot \tau_c \\ &+(p-1)\frac{n^{2\cdot r} \cdot i^2}{p^2} \cdot \tau_c. \end{aligned} \tag{4.10}$$

From eq. (4.5) and eq. (4.10), the speed-up ratio against the increasement of the number of slaves is presented as follows,

$$\begin{aligned} &\frac{n^r \cdot i \cdot (\tau_m + \tau_d + \tau_a) + \frac{1}{2}\left\{ i^2 \cdot n^{2r} + 2 \cdot i \cdot n^{r+1} - i \cdot n^r \right\} \cdot \tau_c}{\frac{n^r}{p} \cdot i\,(\tau_m + \tau_d + \tau_a) + \frac{1}{2p}\left\{ i^2 \cdot \frac{n^{2r}}{p} + 2 \cdot i \cdot n^{r+1} - i \cdot n^r \right\} \cdot \tau_c + (p-1)\frac{n^{2\cdot r} \cdot i^2}{p^2} \cdot \tau_c} \\ \approx\ &\frac{\frac{1}{2}\left\{ i^2 \cdot n^{2r} + 2 \cdot i \cdot n^{r+1} - i \cdot n^r \right\} \cdot \tau_c}{\frac{1}{2p}\left\{ i^2 \cdot \frac{n^{2r}}{p} + 2 \cdot i \cdot n^{r+1} - i \cdot n^r \right\} \cdot \tau_c + (p-1)\frac{n^{2\cdot r} \cdot i^2}{p^2} \cdot \tau_c} \end{aligned}$$

$$\approx \frac{\frac{1}{2}\{i^2 \cdot n^{2r}\} \cdot \tau_c}{\frac{1}{p}\{i^2 \cdot n^{2r}\} \cdot \tau_c}$$
$$= \frac{p}{2}. \tag{4.11}$$

When n^{2r} is large, we can expect that this model is effective for improving performance of automated forward deduction.

4.4 Parallelization version of EnCal

In following sections, we present a case study of improving the performance of EnCal with parallel processing. In this case study, we focus on EnCal-P that reasons out all logical theorem schemata , LTSs for short, of the k^{th} degree fragment of a propositional logic since it is most basic function of EnCal.

An automated forward deduction by EnCal consists of 3 parts as follows.

1. Initialization part: it takes in premises, limits of degree k and j and inference rules.
2. Forward deduction part: about each inference rule, it repeats following processes until it deduces no new LTSs.
   (a) Matching process: it seeks and picks up some LTSs from a set of previously deduced LTSs or premises to apply an inference rule. Then it matches the LTS to the inference rule.
   (b) Deduction process: it applies the inference rule to the LTSs which were matched at the matching process.
   (c) Duplication checking process: it compares a conclusion which was deduced at the deduction process with all previously deduced LTSs in order to check whether it is duplicate or not.
   (d) Adding process: it adds the conclusion which was judged to be new at the duplication checking process as new LTS to the set of premises if the its degree of nest of entailment is i^{th} degree ($1 \leq i \leq k$).
3. Outputting part: it outputs all new LTSs to a file.

At present, the inference rule of EnCal is modus ponens only. Modus ponens is that B is deduced from $A \Rightarrow B$ and A. The termination condition of EnCal-P is to deduce all logical theorem schema in the k^{th} degree fragment of a logic system L, denoted by $Th^k(L)$, from given axioms, where k is a natural number.

The forward deduction algorithm in EnCal is as follows. Let n be the number of previously deduced LTSs and given premises, and $\{P\} = \{P_0, P_1, \ldots, P_{n-1}\}$ be the set of premises and previously deduced LTSs.

Algorithm 1 Forward deduction

1. $n \leftarrow$ the number of premises.

2. $p \leftarrow 0$
3. $k \leftarrow$ the limit of degree.
4. **do**
5. $\quad n^{'} \leftarrow n$
6. $\quad$ **for** ($i \leftarrow 0$, $i < n$, $i \leftarrow i + 1$)
7. $\quad\quad$ **for** ($j \leftarrow p$, $j < n$, $j \leftarrow j + 1$)
8. $\quad\quad\quad$ Matching(P_i,P_j):
   If it can apply Modus Ponens to between P_i and P_j,
   return SUCCESS. If no, return FAILURE.
9. $\quad\quad\quad$ **if** Matching(P_i,P_j) returns SUCCESS
10. $\quad\quad\quad\quad$ **then** Deduction(P_i,P_j):
    it applies Modus Ponens to P_i and P_j.
11. $\quad\quad\quad\quad$ Duplication_check(C):
    If a conclusion C which was deduced at Deduction(P_i,P_j) is
    duplicate, return DUPLICATE. If no, return NEW.
12. $\quad\quad\quad\quad$ **if** Duplication_check(C) returns NEW
13. $\quad\quad\quad\quad\quad$ **then** Adding(C):
    it adds an a conclusion C into $\{P\}$ if the degree of C is smaller
    than k. After that $n' \leftarrow n^{'} + 1$.
14. $\quad$ **for** ($i \leftarrow p$, $i < n$, $i \leftarrow i + 1$)
15. $\quad\quad$ **for** ($j \leftarrow 0$, $j < p$, $j \leftarrow j + 1$)
16. $\quad\quad\quad$ Matching(P_i,P_j)
17. $\quad\quad\quad$ **if** Matching(P_i,P_j) returns SUCCESS
18. $\quad\quad\quad\quad$ **then** Deduction(P_i,P_j)
19. $\quad\quad\quad\quad$ Duplication_check(C)
20. $\quad\quad\quad\quad$ **if** Duplication_check(C) returns NEW
21. $\quad\quad\quad\quad\quad$ **then** Adding(C)
22. $\quad p \leftarrow n$
23. $\quad n \leftarrow n^{'}$
24. **while** (new LTSs are deduced).

The algorithm of Duplication_check(C) is as follows. Let Comp(A, B) be a function which compares A with B to judge whether B is duplicate of A. If B is duplicate, then Comp(A, B) returns DUPLICATE.

Algorithm 2 Duplication_check(C)

1. **for** ($i \leftarrow 0$, $i < n$, $i \leftarrow i + 1$)
2. $\quad$ Comp(P_i, C):
   C is a conclusion which was deduced at Deduction process.
3. $\quad$ **if** Comp(P_i, C) returns DUPLICATE
4. $\quad\quad$ **then return** DUPLICATE
5. **return** NEW.

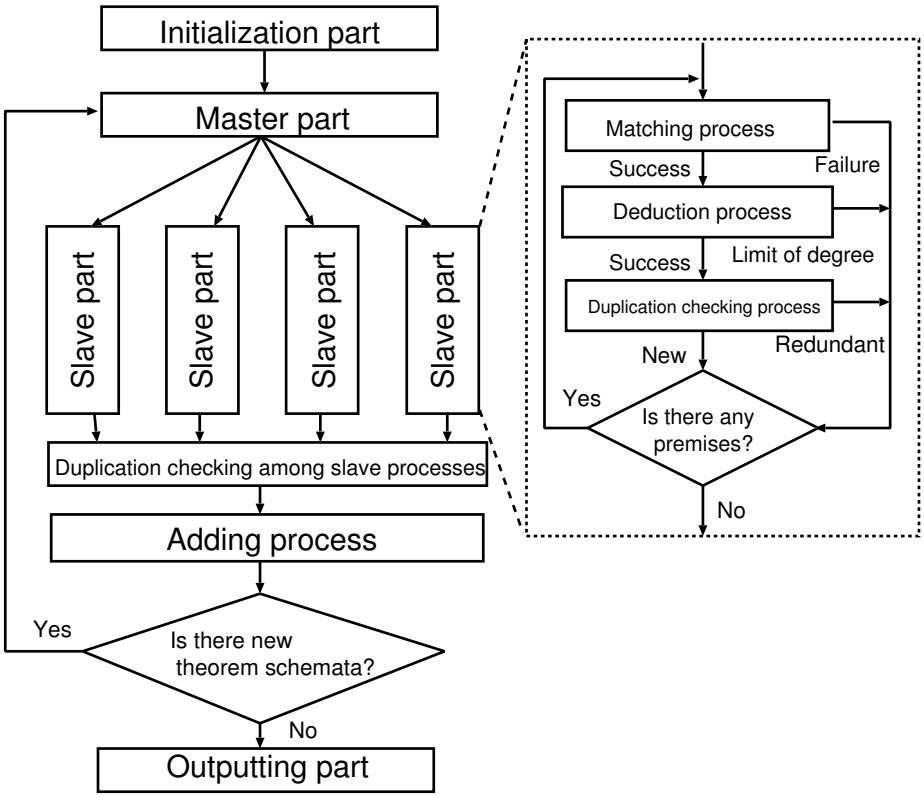

Figure 4.3: The model of parallelization version of EnCal

The major portion of the execution time of EnCal as same as automated forward deduction is spent at the duplication checking process. Let N be the number of given premises and new LTSs which are deduced finally. The number of premises required by modus ponens is two. The number of times of processing Matching(A, B) is N^2. The number of times of processing Duplication_check(A) is at most N^2 and at least N, since the number of deduced conclusions and intermediates is at most N^2 and at least N. The number of times of processing Comp(A, B) is at most

$$\sum_{k=1}^{N^2-1} k = \frac{\{N^2(N^2-1)\}}{2}, \tag{4.12}$$

and at least

$$\sum_{k=1}^{N-1} k = \frac{\{N(N-1)\}}{2}. \tag{4.13}$$

Thus the calculated amount of EnCal approaches at most $O(N^4)$ and at least $O(N^2)$.

We design the parallelization version of EnCal based on the parallelization model in section 4.3. Figure 4.3 shows the model of parallelization version of EnCal. On this parallelization of EnCal, let p be the number of processors, and N be the number of given premises and new LTSs which are deduced finally. If deduced conclusions or intermediates are evenly deduced on each slave, the number

of times of processing Comp(A,B) at the duplication check process on one slave is at most

$$\frac{\sum_{k=1}^{N^2-1} k}{p} = \frac{\{N^2(N^2-1)\}}{2p}, \tag{4.14}$$

and at least

$$\frac{\sum_{k=1}^{N-1} k}{p} = \frac{\{N(N-1)\}}{2p}. \tag{4.15}$$

The number of times of processing Comp(A,B) at the duplication checking among slaves process is at most

$$p \cdot \sum_{k=1}^{N^2/p-1} k = \frac{\{N^2(N^2-p)\}}{2p}, \tag{4.16}$$

and at least

$$p \cdot \sum_{k=1}^{N/p-1} k = \frac{\{N(N-p)\}}{2p}. \tag{4.17}$$

Thus the theoretical speed up ratio is approximated as follows,

$$\begin{aligned} \text{Speed up ratio} &\approx \frac{\frac{\{N^2(N^2-1)\}}{2}}{\frac{\{N^2(N^2-1)\}}{2p} + \frac{\{N^2(N^2-p)\}}{2p}} \text{(at most)} \\ &\approx \frac{\frac{\{N(N-1)\}}{2}}{\frac{\{N(N-1)\}}{2p} + \frac{\{N(N-p)\}}{2p}} \text{(at least)} \\ &\approx \frac{p}{2} \end{aligned} \tag{4.18}$$

In the parallelization version of EnCal based on master-slave model, its theoretical value shows that the processing load on one processor decreases in proportion to the increasement in used processors.

4.5 Implementation and results

Table 4.1: The execution time on Sun Enterprise 6000 (sec.)

Logic systems	1 processor	2 processors	4 processors	8 processors	16 processors
Te(4)	3499	1641	922	562	381
Ee(4)	12347	6100	3370	2021	1290
Re(4)	85962	41146	21825	12668	8236

We implemented the parallelization version of EnCal based on master-slave model on a shared-memory parallel computer and a cluster of PCs, and got the execution time of deducing 4^{th} degree fragment from axioms of some logic systems, in order to investigate the effectiveness of its model.

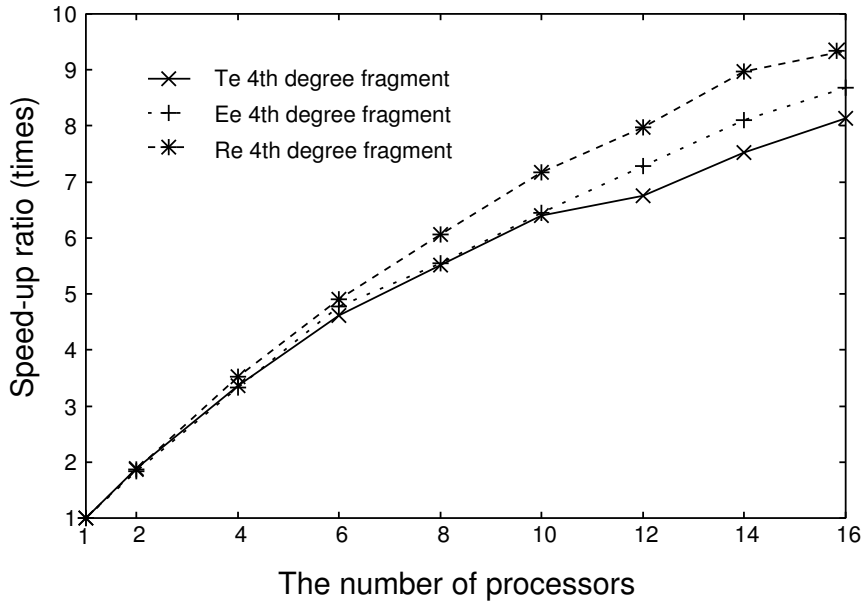

Figure 4.4: Speed-up ratio on Sun Enterprise 6000

We implemented the parallelization version of EnCal with C and OpenMP [36, 38] on the Sun Enterprise 6000 (Ultra SPARC 168 MHz x 16, 4 Gbyte main memory). Table 4.1 shows the execution time on Sun Enterprise 6000. Te(4) denotes the 4^{th} degree fragment of relevant logic system T with entailment. The number of conclusions in Te(4) is 10,649. Ee(4) denotes the 4^{th} degree fragment of relevant logic system E with entailment. The number of conclusions in Ee(4) is 15,519. Re(4) denotes the 4^{th} degree fragment of relevant logic system R with entailment. The number of conclusions in Re(4) is 35,027. Table 4.1 shows that the execution time gets shorter in proportion to the increasement in the number of processors without depending on the number of deduced conclusions. Figure 4.4 shows the relation between the number of processors and the speed-up ratio against the execution time on 1 processor. Figure 4.4 shows the same tendency as the theoretical value $p/2$, p is the number of processors, acquired in sec. 4.4 was shown.

We also implemented the parallelization version of EnCal on an 8-node dual Pentium III 1GHz PC SMP cluster (i840 chipset, 1GB RDRAM main memory per node, Linux 2.2.16). The nodes on the PC SMP cluster are interconnected through a 100Base-TX Ethernet switch. MPICH-SCore [31] was used as a communication library. We used an intranode MPI library for the PC SMP cluster. All routines were written in C. Table 4.2 shows the execution time on the cluster of PCs. The column of "1 processor / 1 node" is a case of deducing by 1 processor per 1 node. The column of "2 processors / 1 node" is a case of deducing by 2 processor per 1 node. In Table 4.2, the execution time gets shorter in proportion to the increasement in the number of processors without depending on the number of deduced conclusions. Figure 4.5 shows the relation between the number of processors and the speed-up ratio against the execution time on 1 processor, using 2 processors

Table 4.2: The execution time on a clusters of PCs (sec.)

Logic systems		1 processor	2 processors	4 processors	8 processors	16 processors
Te(4)	1 processor / 1 node	905	497	272	157	————
	2 processors / 1 node	————	497	273	157	107
Ee(4)	1 processor / 1 node	4706	2413	1285	666	————
	2 processors / 1 node	————	2418	1287	666	406
Re(4)	1 processor / 1 node	31317	15840	8530	4620	————
	2 processors / 1 node	————	15860	8570	4642	2709

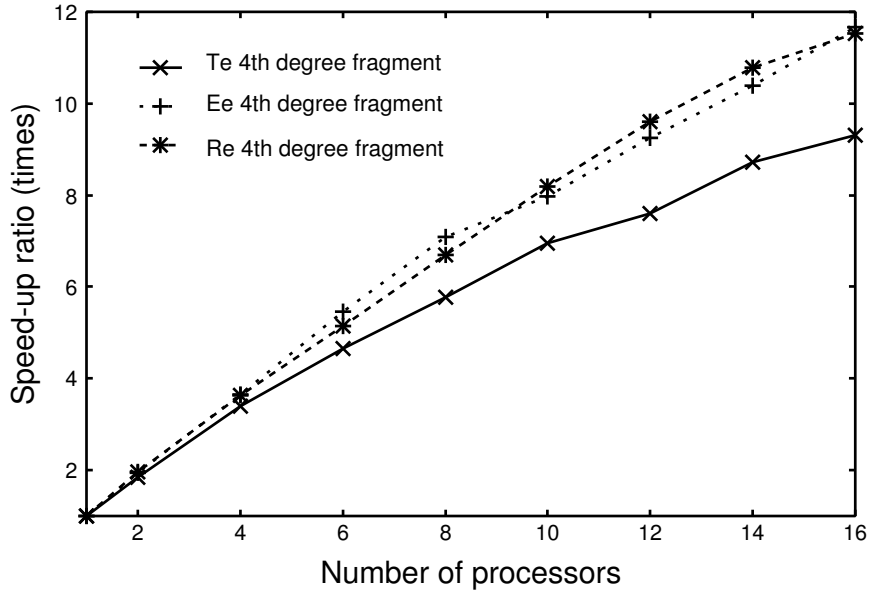

Figure 4.5: Speed-up ratio on a clusters of PCs (2 CPU / 1 node)

per node. Figure 4.5 shows the same tendency as the theoretical value $p/2$, p is the number of processors, acquired in sec. 4.4 was shown.

Thus our experiments shows the model of the parallelization version of EnCal is effective for improving the performance of EnCal, independent of the difference in the hardware and the number of deduced conclusions.

4.6 Discussion

The reason why the speed-up ratio of the implementation on the Sun Enterprise 6000 and that of the PC SMP cluster are not same is that these implementations are based on same the parallelization model in sec. 4.3, but not consists of same functions, same library, same programing codes.

Improving the performance by parallel processing is effective for not only En-

Cal but also other forward deduction engines. Our experiments showed that the speed-up ratio increases in proportion to the increasement of the number of processors, independent of the difference in the cases and the number of deduced conclusions. Although the inference rule of EnCal changes from modus ponens into other inference rules, or the premises change from Ren, Een, and Ten into other axioms of a certain logic system, the speed-up ratio must increase in proportion to the increasement of the number of processors. Only the number of deduced conclusions in a slave and the execution time of processes in automated forward deduction change when inference rules and premises are changed. The speed-up ratio does not change if the execution time of processes changes. In our parallelization model, the execution time of the parallelization one is longer than that of sequential one, if the number of deduced conclusions in a slave changes. Let n be the number of previously deduced conclusions and given premises, and p be the number of processors (slaves), and r be the number of premises required an inference rule. If the number of deduced conclusions on a certain slave is m_i, and the number of that on other slaves is m_j ($m_j < m_i$, $i \neq j$), then the speed-up ratio against the increasement of the number of slaves is presented as follows, from eq. (4.5) and eq. (4.10),

$$\begin{aligned} &\approx \frac{\frac{1}{2}\left\{i^2 \cdot n^{2r} + 2 \cdot i \cdot n^{r+1} - i \cdot n^r\right\} \cdot \tau_c}{\frac{1}{2}\left\{i^2 \cdot m_i^2 + 2 \cdot i \cdot m_i - i \cdot 1\right\} \cdot \tau_c + (p-1)(m_i \cdot m_j) \cdot \tau_c} \\ &= \frac{\frac{1}{2}\left\{i^2 \cdot n^{2r} + 2 \cdot i \cdot n^{r+1} - i \cdot n^r\right\} \cdot \tau_c}{\frac{1}{2}\left\{i^2 \cdot m_i^2 + (2 \cdot i \cdot (p-1) \cdot m_j) m_i - i \cdot 1\right\} \cdot \tau_c}. \end{aligned} \tag{4.19}$$

When n is enough large, the execution time of the parallelization one is longer than that of the sequential one if m_i is n^r. However, in our parallelization model, the number of deduced conclusions in a slave part is at most n/p, that is, m_i never becomes n^r. Thus, our parallelization model of automated forward deduction is effective for improving the performance of automated forward deduction.

4.7 Summary

We presented a model of a parallelization version of automated forward deduction based on master-slave model and implemented EnCal based on the model on a shared-memory parallel computer and a clusters of PCs, as a case study to investigate the effectiveness of parallel processing for improving the performance of automated forward deduction. Our experiments showed that the execution time on both cases gets shorter in proportion to the increasement in the number of processors without depending on the number of deduced conclusions and given premises. Hence, improving the performance by parallel processing is effective for forward deduction engines.

We therefore showed that it is possible to implement a practical forward deduction engine by using our parallelization model.

Chapter 5

Applications of automated forward deduction based on strong relevant logics

5.1 Introduction

We have investigated the issue of the logic systems underlying forward deduction and the issue of the performance of automated forward deduction in chapter 3 and chapter 4, in order to implement a practical forward deduction engine. For these investigation, we showed that a practical forward deduction engine can be implemented by automated forward deduction based on SRL and based on the parallelization model proposed by us.

In this chapter, in order to show usefulness of automated forward deduction based on SRL, we investigate some applications of it.

5.2 Answering logic puzzles by reasoning

5.2.1 Logic puzzle and scientific discovery

Scientific discovery is indispensable to developing science. In generally, huge cost and/or long time are necessary to find a new knowledge or fact on scientific discovery. However, it is not always succeed although huge cost and/or long time are spent to find new knowledge or facts. It is possible to reduce cost and/or time to find a new one if all processes or one of the processes of scientific discovery can be automated. Since there is no discovery process that does not invoke reasoning, to automate the reasoning process is a first step of automating scientific discovery.

Automated reasoning in scientific discovery is forward rather than backward, because there is no given target as goal in scientific discovery, and is need high performance because it deal with huge amount of premises to draw a new conclusion as new knowledge. Moreover, the automated reasoning should guarantee the valid reasoning process from the viewpoint of scientific reasoning as well as our everyday reasoning because the advantage of automated scientific discovery

is 'automation'. If we must check whether a drawn conclusion is correct or not, then there is no point in automating the reasoning process. These requirements can be satisfied by automated forward deduction based on SRL and based on the parallelization model proposed by us.

In this section, we show a case study to use automated forward deduction based on SRL as a forward deduction engine for scientific discovery. In this case study, we get answers to two logic puzzles by automated forward deduction based on SRL.

Logic puzzle is a puzzle which has following feature:

1. it consists of premises, a question, and an answer,
2. both its premises and answer are true and have the relationship among them,
3. the answer can be proven from premises logically in finite steps.

In almost logic puzzles, its question is key information to gets its answer. If an answer of a certain question is 'Yes' or 'No', then to solve the puzzle is to find the path to the question or the one's negation from premises. On the other hand, if an answer of a certain question is the consequent of question, to solve the puzzle is to deduce a conditional and the antecedent of the conditional, e.g. 'if A then B' and 'A' where A is the question and B is the answer, from premises. That is, the question of a logic puzzle is the answer of the logic puzzle itself.

The approach to get an answer of a logic puzzle is classified into the proving approach and the reasoning approach. The farther is an approach to gets an answer of a logic puzzle with premises and a question by proving, e.g. backward reasoning or reductio ad absurdum. The later is an approach to deduce an answer from premises by forward reasoning. To solve a logic puzzle on the reasoning approach is to gets a set of conclusions which includes an answer of the logic puzzle, but does not include the negation of answer and contradictions.

The process to solve a logic puzzle on reasoning approach is a particular case of scientific discovery which is satisfied following conditions:

1. there are already enough premises to deduce new knowledge or facts,
2. it guarantees to be able to get new knowledge or facts,
3. the new knowledge or fact is previously given as a question and/or an answer,
4. the amount of premises is small.

The essential difference between the process to solve a logic puzzle on the reasoning approach and scientific discovery is whether a new knowledge or fact is previously given or not. Other differences are also important, but not essential.

Thus we conclude that automated forward deduction based on SRL is an useful tool to automate the reasoning process in scientific discovery if it can solve a logic puzzle on the reasoning approach with automated forward deduction based on SRL.

5.2.2 Experiments and Results

We got answers to two logic puzzles on the reasoning approach [24]. The logic puzzles are 'Three goddesses' and 'Angel and devil' [37]. In this case study, we used EnCal-E and 3^{rd} degree fragment of a predicate strong relevant logic EcQ as a forward deduction engine. EnCal-E is a tool for reasoning out all empirical theorems of the j^{th} degree fragment of L-theory with premises P based on k^{th} degree fragment of a formal logic system L.

This case study is consists of following procedure:

1. to transform the premises and an answer of a logic puzzle into first order theory,
2. to deduce empirical theorems from the transformed premises by EnCal-E,
3. to find out the transformed answer or the negation of the answer, and contradictions from deduced empirical theorems.

In this case study, it can find out the answer or negation of answer, and contradictions automatically, because we chose logic puzzles whose the transformed answer does not include two-place logical connectives.

We show the results of this case study below.

Logic puzzle: Three goddesses

> Three goddesses say as follows:
>
> Athena: The most beautiful goddess is not Aphrodite.
> Aphrodite: The most beautiful goddess is not Hera.
> Hera: I am the most beautiful goddess.
>
> Only the most beautiful goddess says truth. Who is the most beautiful goddess?
> *Answer: Aphrodite.*

The premises of this logic puzzle are transformed as follows:

- a, b, and c denote Athena, Aphrodite, and Hera respectively.
- $a \neq b$, $b \neq c$, and $c \neq a$
- $Most(x)$ denotes that x is the most beautiful goddess.
- $\forall x \forall y((x \neq y) \Rightarrow (Most(x) \Rightarrow \neg Most(y)))$ denotes that if x is the most beautiful goddess then y is not.
- $Most(a) \Rightarrow \neg Most(b)$ and $\neg Most(a) \Rightarrow Most(b)$ denote that the most beautiful goddess is not Aphrodite.

- $Most(b) \Rightarrow \neg Most(c)$ and $\neg Most(b) \Rightarrow Most(c)$ denotes that the most beautiful goddess is not Hera.
- $Most(c) \Rightarrow Most(c)$ and $\neg Most(c) \Rightarrow \neg Most(c)$ denote that the most beautiful goddess is Hera.

The number of deduced empirical theorems from above premises by EnCal-E was 146. The theorems which do not include two-place logical connectives were $\neg Most(a)$, $Most(b)$, and $\neg Most(c)$. These results are not contradictory to the answer of above logic puzzle.

Logic puzzle: Angel and devil

> Only angels and devils occur in Erika's dream. Angels always say truth. Devils always say falsehood. Yesterday, two women occurred in Erika's dream. The one said 'If I am an angel, then she is an angel, too.' Can we know whether they are angels or devils?
> *Answer: Yes, we can. They are angels.*

The premises of this logic puzzle are transformed as follows:

- $Devil(x)$ denotes x is a devil.
- $Angel(x)$ denotes x is an angel.
- a and b means two women.
- $\forall x(Angel(x) \Rightarrow \neg Devil(x))$ denote that if x is an angel then x is not a devil.
- $\forall x(Devil(x) \Rightarrow \neg Angel(x))$ denote that if x is a devil then x is not an angel.
- $Angel(a) \Rightarrow (Angel(a) \Rightarrow Angel(b))$ and $Devil(a) \Rightarrow \neg(Angel(a) \Rightarrow Angel(b))$ denote that 'If I am an angel then, she is an angel, too'

The number of deduced empirical theorems from above premises by EnCal-E was 168. The theorems which do not include two-place logical connectives were $\neg Devil(a)$, $Angel(a)$, $\neg Devil(b)$, and $\neg Devil(b)$. These results are not contradictory to the answer of above logic puzzle.

5.2.3 Discussion

In this case study, we tried to solve two logic puzzles by automated forward deduction based on SRL on the reasoning approach. We could solve both of them. This case study showed that automated forward deduction based on SRL is essentially possible to automate the reasoning process in scientific discovery.

By the way, we can narrow down the required premises enough to deduce a new knowledge or fact with automated forward deduction, semi-automatically. The process to solve a logic puzzle on the reasoning approach is a particular case of scientific discovery. One of the important differences between the process to solve a logic puzzle and scientific discovery is that there are already enough premises to deduce new knowledge or facts as an answer and/or a question. In scientific discovery, it is difficult to provide all premises enough to deduce new knowledge or facts, because we cannot know which premises are necessary to deduce new one before we get the new one. Thus we should provide all premises enough to deduce new knowledge or facts by trial and error. However, the cost and/or time to spent the process of trial and error are reduced by automated forward deduction base on SRL. Since forward deduction based on SRL guarantees that the conclusions are true if all premises are true, we know that there are incorrect premises if a conclusion is false, in the framework of the strong relevance. By using above behavior, we can narrow down the required premises by using reductio ad absurdum with automated forward deduction base on SRL, semi-automatically.

5.2.4 Summary

We investigated the usefulness of automated forward deduction based on SRL in order to automate the reasoning process in scientific discovery. We tried to solve two logic puzzles on the reasoning approach by automated forward deduction based on SRL, as a particular case of scientific discovery. This case study showed that automated forward deduction based on SRL is essentially possible to automate the reasoning process in scientific discovery.

5.3 Automated theorem finding in NBG set theory

5.3.1 Automated theorem finding by forward deduction

Wos in 1988 proposed 33 basic research problems in automated reasoning [47]. The thirty-first one is the problem of automated theorem finding, ATF for short, which is: "what properties can be identified to permit an automated reasoning program to find new and interesting theorems, as opposed to proving conjectured theorems?" [47, 48] The field of automated reasoning is an outgrowth of the field of automated theorem proving. In fact, the dominant activity in automated reasoning is still that of proving some conjectured theorems. Nevertheless, the problem of ATF asks for criteria that an automated reasoning program can use to find interesting theorems, in contrast to proving conjectured theorems supplied by the users [48].

ATF is still a completely open problem up to now. The most important and difficult requirement of the problem is that, in contrast to proving conjectured theorems supplied by the users, it asks for criteria that an automated reasoning program can use to find some theorems in a field that must be evaluated by

theorists of that field as new and interesting theorems. The significance of solving the problem is obvious because an automated reasoning program satisfying the requirement can provide great assistance for scientists in various fields.

Cheng argued that ATF can be regarded as the automation of deduction operations of an agent [5]. From the viewpoint of logic, Cheng also pointed out that CML and/or its various conservative extensions, and traditional relevant logics, RL for short, are not suitable logic systems to underlie ATF, and proposed approach to the ATF problem that one should use SRL for mathematical knowledge representation and reasoning [5]. Since SRL are free of not only implicational paradoxes in CML and/or its various extensions, but also conjunction-implicational and disjunction-implicational paradoxes in RL, the valid reasoning process is guaranteed. Thus, SRL can satisfy the essential requirements for the logic system to be used as the fundamental logics to underlie ATF [5]. We agree with Cheng's argument from the viewpoint of a quantitative suitable logic system to underlie automated forward deduction.

On the other hand, in generally, an interesting theorem is complete and difficult by contrast axioms or definitions in a certain field. If completeness and difficulty of the theorem come from the length of the path which is from axioms and definitions as premises to the theorem as conclusion, then automated forward deduction in ATF is need high performance because of large amount of data. The amount of data processed in ATF becomes larger as the length of the path becomes longer.

We conclude that a forward deduction engine which is automated forward deduction based on SRL, and is implemented based on the parallelization model proposed by us is necessary to perform ATF by forward deduction.

5.3.2 Case study of automated theorem finding in NBG set theory

We tried to perform ATF in von Neumann-Bernays-Godel set theory , NBG set theory for short, by automated forward deduction based on SRL, as a case study.

NBG set theory is one axiom system for set theory that can in turn be expressed within the language of the first-order predicate calculus, which can be programmed on a digital computer. On the other hand, all extant mathematics can be formulated within the language of set theory.

In this case study, axioms and definitions of NBG set theory that we took are listed in a book entitled "Automated Development of Fundamental Mathematical Theories" [39]. We transformed them to the formulas with only entailment and negation connectives of SRL by adopting equivalent formulas that include defined intensional connectives. They are shown as follows, where every axiom and definition are represented with intensional connectives of SRL, and existential quantifier in the formula is represented universal quantifier by quantifier equivalence formula $\exists x M(x) \equiv \neg\forall x \neg M(x)$.

Axiom A-1: Sets are classes

$$\forall x(M(x) \Rightarrow CLS(x)). \tag{5.1}$$

Here, $CLS(x)$ means x is a class, $M(x)$ means x is a set.

Axiom A-2: Elements of classes are sets

$$\forall x \forall y (x \in y \Rightarrow M(x)). \tag{5.2}$$

Here, $(x \in y)$ means x belongs to y.

Axiom A-3: Extensionality

$$\forall x \forall y (\forall u (M(u) \Rightarrow (u \in x \Leftrightarrow u \in y)) \Rightarrow (x = y)). \tag{5.3}$$

Axiom A-4: Existence of unordered pair

$$\forall x \forall y (M(x) \otimes M(y) \Rightarrow \exists z (M(z) \otimes \forall u (M(u) \Rightarrow (u \in z \Leftrightarrow (u = x \oplus u = y))))). \tag{5.4}$$

Definition of unordered pair

$$\forall x \forall y \forall u (M(x) \otimes M(y) \otimes M(u) \Rightarrow (u \in \{x, y\} \Leftrightarrow (u = x \oplus u = y))). \tag{5.5}$$

Here, $\{x, y\}$ means x and y is an unordered pair.

Definition of singleton set

$$\forall x (M(x) \Rightarrow (\{x\} = \{x, x\})). \tag{5.6}$$

Here, $\{x\}$ means singleton set.

Definition of ordered pair

$$\forall x \forall y (M(x) \otimes M(y) \Rightarrow (< x, y >= \{\{x\}, \{x, y\}\})). \tag{5.7}$$

Here, $< x, y >$ means x and y is an ordered pair.

Axiom B-1: Elementhood relation

$$\exists z \forall x \forall y (M(x) \otimes M(y) \Rightarrow (< x, y > \in z \Leftrightarrow x \in y)) \tag{5.8}$$

Axiom B-2: Binary intersection

$$\forall x \forall y \exists z \forall u (M(u) \Rightarrow (u \in z \Leftrightarrow (u \in x \otimes u \in y))). \tag{5.9}$$

Axiom B-3: Complement

$$\forall x \exists y \forall u (M(u) \Rightarrow (u \in y \Leftrightarrow \neg (u \in x))). \tag{5.10}$$

Axiom B-4: Domain

$$\forall x \exists y \forall z (M(z) \Rightarrow (z \in y \Leftrightarrow \exists u (< u, z > \in x))). \tag{5.11}$$

Axiom B-5: Cartesian product

$$\forall x \exists y \forall z \forall u (M(z) \otimes M(u) \Rightarrow (< z, u > \in y \Leftrightarrow u \in x)) \tag{5.12}$$

Axiom B-6: Inverse

$$\forall x \exists y \forall z \forall u (M(z) \otimes M(u) \Rightarrow (< z, u > \in y \Leftrightarrow < u, z > \in x)). \tag{5.13}$$

Axiom B-7: Rotate

$$\forall x \exists y \forall z \forall u \forall v (M(z) \otimes M(u) \otimes M(v) \Rightarrow (< z, < u, v >> \in y \Leftrightarrow < u, < v, z >> \in x)). \tag{5.14}$$

Axiom B-8: Flip

$$\forall x \exists y \forall z \forall u \forall v (M(z) \otimes M(u) \otimes M(v) \Rightarrow (< z, < u, v >> \in y \Leftrightarrow < z, < v, u >> \in x)). \tag{5.15}$$

Definition of $\subseteq$ (subclass)

$$\forall x \forall y ((x \subseteq y) \Leftrightarrow \forall u ((u \in x) \Rightarrow (u \in y))). \tag{5.16}$$

Definition of $\subset$ (proper subclass)

$$\forall x \forall y (x \subset y \Leftrightarrow (x \subseteq y \otimes \neg (x = y))). \tag{5.17}$$

Definition of EMPTY

$$\forall x (EMPTY(x) \Leftrightarrow \forall u (M(u) \Rightarrow \neg (u \in x)). \tag{5.18}$$

Definition of DISJOINT

$$\forall x \forall y (DISJOINT(x, y) \Leftrightarrow \forall u (M(u) \Rightarrow \neg (u \in x \otimes u \in y))). \tag{5.19}$$

Definition of SINGVAL

$$\begin{aligned} &\forall x (SINGVAL(x) \Leftrightarrow \\ &\forall u \forall v \forall w (M(u) \otimes M(v) \otimes M(w) \otimes < v, u > \in x \otimes < w, u > \in x \Rightarrow (v = w))). \end{aligned} \tag{5.20}$$

Axiom C-1: Infinity

$$\exists x (\neg EMPTY(x) \otimes \forall z (z \in x \Rightarrow \exists u (u \in x \otimes z \subset u))). \tag{5.21}$$

Axiom C-2: Sum class

$$\forall x (M(x) \Rightarrow \exists y (M(y) \otimes \forall u \forall v (M(u) \otimes M(v) \otimes u \in v \otimes v \in x \Rightarrow u \in y))). \tag{5.22}$$

Axiom C-3: Power class

$$\forall x (M(x) \Rightarrow \exists y (M(y) \otimes \forall u (M(u) \otimes u \subseteq x \Rightarrow u \in y))). \tag{5.23}$$

Axiom C-4: Replacement

$$\forall z \forall x(SINGVAL(z) \otimes M(x) \Rightarrow \\ \exists y(M(y) \otimes \forall u(M(u) \otimes u \in y \Rightarrow \exists v(M(v) \otimes v \in x \otimes < u, v > \in z)))). \tag{5.24}$$

Axiom D: Regularity

$$\forall x(\neg EMPTY(x) \Rightarrow \exists u(M(u) \otimes u \in x \otimes DISJOINT(u, x))). \tag{5.25}$$

Axiom E: Universal choice

$$\exists z(SINGVAL(z) \otimes \forall x(M(x) \otimes EMPTY(x) \Rightarrow \exists y(M(y) \otimes y \in x \otimes < y, x > \in z))). \tag{5.26}$$

We used EnCal in the experiments. EnCal runs on one PC with Pentium IV (3.2GHz CPU, 2Gbyte memory) and RedHat Lunix 9 (Linux Kernel is 2.4.20-8).

The procedures of the experiments that include the followings:

1. represent primitive NBG set theory axioms and definitions with only entailment and negation connectives,
2. reason out the logical theorem schemata of k^{th} $(k = 2, 3)$ degree fragment of predicate strong relevant logics with only entailment and negation connectives by using EnCal-Q, and
3. deduce j^{th} $(j = 2, 3)$ degree empirical theorems with EnCal-E by taking the formulas obtained from procedure (1) as empirical premise and procedure (2) as logical premises.

In this case study, the experiments mainly were carried out below 3^{rd} degree, for the larger the number of degree is, the more execution time of EnCal takes and enormous the experimental results deduced are. We carried out our experiments on EQen which is a predicate relevant logic system E with only entailment and negation.

We divided the axiom schema sets of EQen as follows:

- EQenSet1 = {E1, E2', E3, E4", N1, N2, N3, IQ1, IQ3, IQ4},
- EQenSet2 = Set1 + {E4"'},
- EQenSet3 = Set2 + {E4'},
- EQenSet4 = Set3 + {E3'},

Table 5.1: The data and results of the experiments

	Premises		Conclusions	
	LP	EP	2^{nd} emp (ET)	3^{rd} emp (ET)
EQenSet1.L3	2566	26	47 (5760s)	163 (30780s)
EQenSet2.L3	2568	26	47 (8040s)	163 (45960s)
EQenSet3.L3	2587	26	47 (6720s)	163 (40740s)
EQenSet4.L3	23775	26	47 (64020s)	163 (397560s)

where 'E*', 'N*', and 'IQ*' are axiom schemata of SRL in sec. 3.3.

Adopting the different axiom schema sets, We deduced their logical theorem schemata of 3^{rd} degree fragment, denoted by EQenSet1.L3, EQenSet2.L3, EQenSet3.L3, and EQenSet4.L3.

The experimental results are showed in table 5.1. In the table, the leftest column shows the name of logical premises used in the experiments; LP denotes the number of logical premises, which are logical axiom schemata or logical theorem schemata reasoned out; EP stands for the number of empirical premises, which are the axioms and definitions of NBG set theory; 2^{nd} emp and 3^{rd} emp mean the number of 2^{nd} and 3^{rd} degree NBG theorems obtained respectively; ET means the execution time of EnCal-E. All the digits mean the number of formulas in every field. The same set of empirical theorems are deduced from every set of logical premises.

Here we give an example for an empirical theorem in 3rd degree results deduced based on EQen's 3rd degree fragment as follows:

$$\forall x(\forall u(M(u) \Rightarrow ((u \in x) \Rightarrow \neg DISJOINT(u, x))) \Rightarrow EMPTY(x)).$$

5.3.3 Discussion

Our experiments showed that all the empirical theorems deduced from the every set of logical premises are same. The reason is that only same logical premises which every set of logical premises has are used to deduce the empirical theorems. The other logical premises were not used in these deduction. Having to process the data which is unnecessary to deduce new conclusions is the nature of forward reasoning for discovery or prediction.

This experiments showed that it is in principle possible to find theorems of the NBG set theory by automated forward deduction based on SRL. The evaluation of results of ATF by automated forward deduction is classified into three kinds: rediscovery of a known theorem, finding the answer of open problems, and finding a new theorem. If we can rediscover a known theorem in ATF by automated

forward deduction, then we can expect to get new theorems with the logic system underlying the forward deduction, and inference rules and premises used in ATF. Moreover if the argument of the known theorem is new proof of the theorem, to rediscover known theorem is to find the new proof of the theorem. If we can find the answer of open problems in ATF by automated forward deduction, we can regard the program for the ATF as a tool to provide great assistance for scientists in the fields. To find new theorems is one of goals of ATF. In this experiments, we can rediscovery known theorems, and no paradoxical theorem was deduced. Thus, we can expect to deduce new theorems by automated forward deduction based on SRL.

On the other hand, our experiments showed that it takes long execution time if the number of logic premises or the number of empirical theorems is large. Thus an high-performance forward deduction engine is needed to deduce new theorems in ATF by automated forward deduction.

5.3.4 Summary

We tried to perform ATF in NBG set theory by automated forward deduction based on SRL, as a case study. In this case study, we deduced some NBG set theory theorems from axioms of NBG set theory, but no paradoxical theorem was deduced. Our results showed that it is in principle possible to find theorems of the NBG set theory by automated forward deduction based on SRL.

5.4 Anticipatory reasoning in anticipatory reasoning-reacting systems

5.4.1 Temporal relevant logics

The concept of an anticipatory system first proposed by Rosen in 1980s [42]. Rosen considered that "an anticipatory system is one in which present change of state depends upon future circumstance, rather than merely on the present or past" and defined an anticipatory system as "a system containing a predictive model of itself and/or its environment, which allows it to change state at an instant in accord with the model's prediction to a latter instant" [42]. Until now, philosophical discussions on anticipatory systems and their characteristics are still being continued by scientists from various disciplines [14, 17, 18, 32, 33].

On the other hand, from the viewpoints of software reliability engineering and information security engineering, what we need is really useful systems with anticipatorily predictive capability to take anticipation for forestalling disasters and attacks rather than the philosophical definition and intension of an anticipatory system. In order to develop anticipatory systems useful in the real world, Cheng has proposed a new type of reactive systems, named "Anticipatory Reasoning-Reacting Systems," as a certain class of anticipatory systems [12].

An anticipatory reasoning-reacting system, ARRS for short , is a computing system containing a controller C with capabilities to measure and monitor the

behavior of the whole system, a traditional reactive system RS, a predictive model PM of RS and its external computing environment, and an anticipatory reasoning engine ARE such that according to predictions by ARE based on PM, C can order and control RS to carry out some operations with a high priority [12]. Since anticipatory reasoning based on the predictive model is the only way for the system to reason out predictions, both the predictive model PM and the anticipatory reasoning engine ARE must be based on a sound logical basis.

Cheng has also proposed temporal relevant logics which are obtained by introducing temporal operators and related axiom schemata and inference rules into SRL [8, 9]. Cheng has argued that temporal relevant logics are suitable logic systems to underlie PM and ARE by far than the temporal classical logics which are one of conservative extension of CML [12].

The temporal operators and related axiom schemata and inference rules are defined as follows [8, 9]:

Temporal operators:

- G: future-tense always or henceforth operator
  GA means " It will always be the case in the future from now that A".
- H: past-tense always operator
  HA means " It has always been the case in the past up to now that A".
- F: future-tense sometime or eventually operator
  FA means " It will be the case at least once in the future from now that A".
- P: past-tense sometime operator
  HA means " It has been the case at least once in the past up to now that A".

These temporal operators are not independent and can be defined as follows:

$$\begin{aligned} GA &=_{df} \neg F\neg A, \\ HA &=_{df} \neg P\neg A, \\ FA &=_{df} \neg G\neg A, \\ PA &=_{df} \neg H\neg A. \end{aligned}$$

Related axiom schemata

T1: $G(A \Rightarrow B) \Rightarrow (GA \Rightarrow GB)$

T2: $H(A \Rightarrow B) \Rightarrow (HA \Rightarrow HB)$

T3: $A \Rightarrow G(PA)$

T4: $A \Rightarrow H(FA)$

T5: $GA \Rightarrow G(GA)$

T6: $(FA \wedge FB) \Rightarrow F(A \wedge FB) \vee F(A \wedge B) \vee F(FA \wedge B)$

T7: $(PA \wedge PB) \Rightarrow P(A \wedge PB) \vee P(A \wedge B) \vee P(PA \wedge B)$

T8: $GA \Rightarrow FA$

T9: $HA \Rightarrow PA$

T10: $FA \Rightarrow F(FA)$

T11: $(A \wedge HA) \Rightarrow F(HA)$

T12: $(A \wedge GA) \Rightarrow P(GA)$

Inference Rules:

TG: "from A to infer GA and HA" (Temporal Generalization)

Cheng could obtain some minimal or weakest temporal relevant logics as follows [8, 9]:

$$
\begin{aligned}
T_0\mathrm{T} &= \mathrm{T} + \{\mathrm{T1} \sim \mathrm{T4}\} + \mathrm{TG} \\
T_0\mathrm{Tc} &= \mathrm{Tc} + \{\mathrm{T1} \sim \mathrm{T4}\} + \mathrm{TG} \\
T_0\mathrm{E} &= \mathrm{E} + \{\mathrm{T1} \sim \mathrm{T4}\} + \mathrm{TG} \\
T_0\mathrm{Ec} &= \mathrm{Ec} + \{\mathrm{T1} \sim \mathrm{T4}\} + \mathrm{TG} \\
T_0\mathrm{R} &= \mathrm{R} + \{\mathrm{T1} \sim \mathrm{T4}\} + \mathrm{TG} \\
T_0\mathrm{Rc} &= \mathrm{Rc} + \{\mathrm{T1} \sim \mathrm{T4}\} + \mathrm{TG}
\end{aligned}
$$

Here, Tc, Ec, Rc, TcQ, EcQ, and RcQ are SRL proposed by Cheng [7, 10].

Note that the minimal or weakest temporal classical logic K_t = all axiom schemata for CML + $\rightarrow$ E + $\{\mathrm{T1} \sim \mathrm{T4}\}$ + TG. Other characteristic axiom schemata such as T5 $\sim$ T12 that correspond to various assumptions about time can be added to T_0T, T_0Tc, T_0E, T_0Ec, T_0R, and T_0Rc respectively to obtain various temporal relevant logics. These logics can be extended into various first-order predicate logics by adding various symbols of quantifiers, individual variables, individual constants, individual functions, and individual predicates into their formal (object) languages, and introducing those usual axiom schemata and inference rules relative to quantifiers.

5.4.2 Automated forward deduction based on temporal relevant logics

An anticipatory reasoning engine ARE is indispensable components of ARRS. Anticipatory reasoning is a reasoning to draw new, previously unknown and/or unrecognized conclusions about some future event or events whose occurrence and truth are uncertain at the point of time when the reasoning is being performed.

An ARE must be a forward deduction engine. Reasoning can be classified into forward reasoning and backward reasoning. Forward reasoning is to infer new conclusions from known facts or assumed hypotheses. Backward reasoning is to find out the path which is from known facts or hypotheses to given goal or sub-goal. Anticipatory reasoning is forward rather than backward because when we perform anticipatory reasoning we cannot know some future event or events, whose occurrence and truth are uncertain at the time point of the reasoning is being performed, as a goal or sub-goal. Reasoning can be classified into three

forms, deduction, induction and abduction. For an ARRS, the conclusions deduced by the ARE must be definitely correct if the premises are correct. This can be guaranteed by only deduction. Therefore, an ARE must be a forward deduction engine.

From the philosophical viewpoint, the notion of anticipation itself is intrinsically time dependent. The earlier anticipatory reasoning draws conclusions, or the farther the future event is predicted by anticipatory reasoning, the higher is its degree of anticipation. To be a computing system useful in various applications in the real world, an anticipatory system must have the ability of anticipatory reasoning with some certain degree of anticipation to predict the occurrence and truth of some future event or events. On the other hand, from the viewpoints of software reliability engineering and information security engineering, a practical anticipatory system must be able to perform any anticipatory reasoning to get enough effective conclusions anticipatorily within an acceptable time in order to satisfy the requirements of high reliability and high security from applications. Since the most intrinsic characteristic of an anticipatory system is its ability of taking anticipation, an anticipatory system that cannot satisfy the requirements of anticipation and timeliness is useless at all in practices in the real world. Therefore, for an anticipatory system with requirements of high reliability and high security, its functioning is both anticipation-critical and time-critical.

Thus, we face a dilemma. On the one hand, as forward reasoning, anticipatory reasoning should deal with a lot of intermediates, which are usually involved in any forward reasoning, in order to get effective conclusions anticipatorily. On the other hand, anticipatory reasoning should be performed as efficiently as possible in order to keep a high degree of anticipation.

It may be possible to implement an anticipatory reasoning engine by using automated forward deduction based on temporal relevant logics, and the parallelization model proposed by us.

5.4.3 Summary

We have investigated what role automated forward deduction based on temporal relevant logics can play in anticipatory systems with requirements of high reliability and high security. We showed that the high-performance automated forward deduction based on temporal relevant logics is necessary to implementation of an anticipatory reasoning engine.

5.5 Spatial reasoning in geographic information systems

5.5.1 Spatial relevant logics

Spatial knowledge, i.e. shape, size, distance, orientation, relative position, connectivity, etc, plays an important role in our cognition and understanding of the

world. There are many applications that need means for representing and reasoning about spatial knowledge, such as robotics, motion planning, machine vision, solid modeling, spatial database systems, geographic information systems, distributed systems, natural language understanding, etc.

In the area of geographic information systems, until now, almost all existing methodologies for representing and reasoning about spatial knowledge are somehow based on CML or its various conservative extensions [21]. This approach, however, may be suitable to searching and describing a formal proof of a previously specified statement, under the condition that we have complete and consistent knowledge, but not necessarily suitable to forming a new concept and discovering a new statement, in particular, in the case that our knowledge is incomplete and inconsistent. This is because the aim, nature, and role of classical mathematical logic is descriptive and non-predictive rather than prescriptive and predictive.

We propose a new family of relevant logic systems, named Spatial Relevant Logic, as the fundamental logic system to underlie representing and reasoning about geographic knowledge [13]. The logics are obtained by introducing region connection predicates and axiom schemata of RCC [15, 40], point position predicates and axiom schemata, and point adjacency predicates and axiom schemata into SRL.

Let $\{r_1, r_2, r_3, \cdots\}$ be a countably infinite set of individual variables, called region variables. Atomic formulas of the form $C(r_1, r_2)$ are read as "region r_1 connects with region r_2." Let $\{p_1, p_2, p_3, \cdots\}$ be a countably infinite set of individual variables, called point variables. Atomic formulas of the form $I(p_1, r_1)$ are read as "point p_1 is in region r_1." Atomic formulas of the form $B(p_1, p_2, p_3)$ are read as "points p_1, p_2, p_3 is on a straight line and point p_1 is between point p_2 and point p_3." Note that here we use a many-sorted language.

The region connection predicates, position predicates, axiom schemata, and inference rules are as follows:

Primitive binary predicate:

C: connection
$C(r_1, r_2)$ means 'region r_1 connects with region r_2'.

I: be in
$I(p_1, r_1)$ means 'point p_1 is in region r_1'.

Arc: arc
$Arc(p_1, p_2)$ means 'p_1 is adjacent to p_2'.

$Path$: path
$Path(p_1, p_2)$ means 'there is a directed path from p_1 to p_2'.

B: be between
$B(p_1, p_2, p_3)$ means 'points p_1, p_2, p_3 is on a straight line and point p_1 is between point p_2 and point p_3'.

Defined Binary Predicates:

$DC(r_1, r_2) =_{df} \neg C(r_1, r_2)$
$DC(r_1, r_2)$ means 'r_1 is disconnected from r_2'.

$P(r_1, r_2) =_{df} \forall r_3\,(C(r_3, r_1) \Rightarrow C(r_3, r_2))$
$P(r_1, r_2)$ means 'r_1 is a part of r_2'.

$PP(r_1, r_2) =_{df} P(r_1, r_2) \wedge (\neg P(r_2, r_1))$
$PP(r_1, r_2)$ means 'r_1 is a proper part of r_2'.

$EQ(r_1, r_2) =_{df} P(r_1, r_2) \wedge P(r_2, r_1)$
$EQ(r_1, r_2)$ means 'r_1 is identical with r_2'.

$O(r_1, r_2) =_{df} \exists r_3\,(P(r_3, r_1) \wedge P(r_3, r_2))$
$O(r_1, r_2)$ means 'r_1 overlaps r_2'.

$DR(r_1, r_2) =_{df} \neg O(r_1, r_2)$
$DR(r_1, r_2)$ means 'r_1 is discrete from r_2'.

$PO(r_1, r_2) =_{df} O(r_1, r_2) \wedge (\neg P(r_1, r_2)) \wedge (\neg P(r_2, r_1))$
$PO(r_1, r_2)$ means 'r_1 partially overlaps r_2'.

$EC(r_1, r_2) =_{df} C(r_1, r_2) \wedge (\neg O(r_1, r_2))$
$PO(r_1, r_2)$ means 'r_1 is externally connected to r_2'.

$TPP(r_1, r_2) =_{df} PP(r_1, r_2) \wedge \exists r_3\,(EC(r_3, r_1) \wedge EC(r_3, r_2))$
$TPP(r_1, r_2)$ means 'r_1 is a tangential proper part of r_2'.

$NTPP(r_1, r_2) =_{df} PP(r_1, r_2) \wedge (\neg\exists r_3\,(EC(r_3, r_1) \wedge EC(r_3, r_2)))$
$NTPP(r_1, r_2)$ means 'r_1 is a nontangential proper part of r_2'.

Axiom schemata

RCC1: $\forall r_1 \forall r_2\,(C(r_1, r_2) \Rightarrow C(r_2, r_1))$

RCC2: $\forall r_1\,(C(r_1, r_1))$

PRCC1: $\forall p_1 \forall r_1 \forall r_2\,((I(p_1, r_1) \wedge DC(r_1, r_2)) \Rightarrow \neg I(p_1, r_2))$

PRCC2: $\forall p_1 \forall r_1 \forall r_2\,((I(p_1, r_1) \wedge P(r_1, r_2)) \Rightarrow I(p_1, r_2))$

PRCC3: $\forall p_1 \forall r_1 \forall r_2\,((I(p_1, r_1) \wedge PP(r_1, r_2)) \Rightarrow I(p_1, r_2))$

PRCC4: $\forall p_1 \forall r_1 \forall r_2\,((I(p_1, r_1) \wedge EQ(r_1, r_2)) \Rightarrow I(p_1, r_2))$

PRCC5: $\forall r_1 \forall r_2\,(O(r_1, r_2) \Rightarrow \exists p_1\,(I(p_1, r_1) \wedge I(p_1, r_2)))$

PRCC6: $\forall p_1 \forall r_1 \forall r_2\,((I(p_1, r_1) \wedge DR(r_1, r_2)) \Rightarrow \neg I(p_1, r_2))$

PRCC7: $\forall r_1 \forall r_2 (PO(r_1, r_2) \Rightarrow$
$\exists p_1 (I(p_1, r_1) \wedge I(p_1, r_2)) \wedge \exists p_2 (I(p_2, r_1) \wedge \neg I(p_2, r_2)) \exists p_3 (\neg I(p_3, r_1) \wedge I(p_3, r_2)))$

PRCC8: $\forall p_1 \forall r_1 \forall r_2\,((I(p_1, r_1) \wedge EC(r_1, r_2)) \Rightarrow \neg I(p_1, r_2))$

PRCC9: $\forall p_1 \forall r_1 \forall r_2\,((I(p_1, r_1) \wedge TPP(r_1, r_2)) \Rightarrow I(p_1, r_2))$

PRCC10: $\forall p_1 \forall r_1 \forall r_2\,((I(p_1, r_1) \wedge NTPP(r_1, r_2)) \Rightarrow I(p_1, r_2))$

PAC1: $\forall p_1 \forall p_2 (Arc(p_1, p_2) \Rightarrow Path(p_1, p_2))$

PAC2: $\forall p_1 \forall p_2 \forall p_3 (Path(p_1, p_2) \wedge Path(p_2, p_3) \Rightarrow Path(p_1, p_3))$

PPC1: $\forall p_1 \forall p_2 (\neg B(p_1, p_1, p_2))$

PPC2: $\forall p_1 \forall p_2 \forall p_3 (B(p_1, p_2, p_3) \Rightarrow B(p_1, p_3, p_2))$

PPC3: $\forall p_1 \forall p_2 \forall p_3 (B(p_1, p_2, p_3) \Rightarrow \neg B(p_2, p_1, p_3))$

PPC4: $\forall p_1 \forall p_2 \forall p_3 \forall p_4((B(p_1, p_2, p_3) \wedge B(p_4, p_1, p_3)) \Rightarrow B(p_4, p_2, p_3))$

PPC5: $\forall p_1 \forall p_2 \forall p_3 \forall p_4((B(p_1, p_2, p_3) \wedge B(p_4, p_1, p_2)) \Rightarrow B(p_4, p_2, p_3))$

We can now obtain some spatial relevant logics as follows:

RTcQ = TcQ +
{ RCC1, RCC2, PRCC1 $\sim$ PRCC10, PAC1, PAC2, PPC1 $\sim$ PPC5 }

REcQ = EcQ +
{ RCC1, RCC2, PRCC1 $\sim$ PRCC10, PAC1, PAC2, PPC1 $\sim$ PPC5 }

RRcQ = RcQ +
{ RCC1, RCC2, PRCC1 $\sim$ PRCC10, PAC1, PAC2, PPC1 $\sim$ PPC5 }

Here, TcQ, EcQ, and RcQ are strong relevant predicate logics.

5.5.2 Automated forward deduction based on spatial relevant logics

Here we give an example to show that the spatial relevant logic can underlie representing and reasoning about spatial knowledge. The example is to investigate what can be deduced based on a spatial relevant logic from a GIS about countries and their cities in the world, and what will happen when a logical theorem of CML, which is an implicational paradox, is added for deduction as a 'valid' logical theorem.

Let us suppose that our GIS includes the following knowledge:
$\forall r_1 \forall r_2((C(r_1, r_2) \wedge PP(r_2, \text{Europe})) \Rightarrow PP(r_1, \text{Europe}))$, $C(\text{France}, \text{Germany})$, $C(\text{France}, \text{Italy})$, $I(\text{Paris}, \text{France})$, $I(\text{Berlin}, \text{Germany})$, $I(\text{Rome}, \text{Italy})$, $PP(\text{France}, \text{Europe})$, $I(\text{Tokyo}, \text{Japan})$, and $I(\text{Beijing}, \text{China})$.

Based on 2^{nd} degree fragment of REcQ and limit the degree of nested '$\wedge$' to 1, we deduced 226 formulas from the above GIS. All the 226 formulas are relevant to the GIS in the sense of conditional as well as facts. Some new facts in the deduced 226 formulas are: $I(\text{Paris}, \text{Europe})$, $PP(\text{Germany}, \text{Europe})$, $PP(\text{Italy}, \text{Europe})$, $I(\text{Berlin}, \text{Europe})$, and $I(\text{Rome}, \text{Europe})$.

On the other hands, besides 2^{nd} degree fragment of REcQ, we added a logical theorem of CML '$(A \wedge B) \Rightarrow (A \Rightarrow B)$', which is an implicational paradox, as a 'valid' logical theorem in our deduction, but kept other things as the same as the above deduction. In this case, we deduced a lot of irrelevant conditionals such as $I(\text{Berlin}, \text{Germany}) \Rightarrow PP(\text{France}, \text{Europe})$, $I(\text{Tokyo}, \text{Japan}) \Rightarrow C(\text{France}, \text{Italy})$, $PP(\text{France}, \text{Europe}) \Rightarrow I(\text{Beijing}, \text{China})$ and so on. All these irrelevant conditionals were not included in the results of the above deduction using 2^{nd} degree fragment of REcQ only.

The spatial relevant logics have the following possible applications: first, as conservative extensions of SRL satisfying the strong relevance principle, the spatial relevant logics can underlie relevant reasoning as well as truth-preserving reasoning in the sense of conditional, ampliative reasoning, paracomplete reasoning, and paraconsistent reasoning. Moreover, the logics can be used for reasoning about relative relations among points as well as regions. Therefore, they can be used

to represent and specify geographic relations in a geographic information system based on incomplete or even inconsistent information at first, and then to reason (again, not proving) about the unspecified new geographic relations. This is a very useful way in modeling, designing, developing a geographic information system. Probably, at present, the family of spatial relevant logics is the only one to have such application.

Second, once we modeled a geographic information system and specified its desirable properties with the formal language of the spatial relevant logics, we can verify the properties based on the logics, even if there are some incompleteness and inconsistency.

For the first and second applications, an automated reasoning and verifying engine based on the spatial relevant logics is indispensable. We may implement a spatial reasoning engine by using automated forward deduction based on spatial relevant logics and the parallelization model proposed by us.

5.5.3 Summary

We have proposed spatial relevant logic as the fundamental logic system to underlie representing and reasoning about geographic knowledge. We may implement a spatial reasoning engine for geographic information systems by using automated forward deduction based on spatial relevant logics and the parallelization model proposed by us.

Chapter 6

Conclusions

6.1 Contributions

In this thesis, in order to implement a practical forward deduction engine, we investigated two problems of implementing automated forward deduction: the one is about logic systems to underlie automated forward deduction and other is about the performance of automated forward deduction. On the other hand, in order to show usefulness of automated forward deduction based on strong relevant logics, SRL for short, we also investigated some applications of automated forward deduction based on SRL.

We presented a quantitative analysis and discussion on implicational paradoxes in classical mathematical logic, CML for short, with the connective of implication and negation, as the first step of quantitative comparative study between CML and SRL. Our analysis results showed that the difference between the number of the set of $1^{st} \sim k^{th}$ degree logical theorem schemata of CML and that of its subset of $1^{st} \sim k^{th}$ degree logical theorem schemata satisfying strong relevant principle must be larger and larger as k becomes large. On the other hand, our analysis results showed that the number of implicational paradoxes in the set of $1^{st} \sim 3^{rd}$ degree logical theorem schemata of CML is 7.13 times as many as the number of logical theorem schemata of its subset of $1^{st} \sim 3^{rd}$ degree logical theorem schemata satisfying strong relevant principle. Thus the number of implicational paradoxes in the set of $1^{st} \sim k^{th}$ degree logical theorem schemata of CML must be more than 7.13 times the number of logical theorem schemata of its subset of $1^{st} \sim k^{th}$ degree logical theorem schemata satisfying strong relevant principle when k is more than 3. Implicational paradoxes spoil the validity of forward deduction, and unnecessarily lengthen the execution time of automated forward deduction. Consequently, as the logic systems underlying forward deduction, SRL and traditional relevant logics, RL for short, are quantitatively more suitable by far than CML and its various conservative extensions because both SRL and RL include no implicational paradoxes in their theorems. We therefore showed that automated deduction should be based on SRL, in order to implement a practical forward deduction engine.

In order to solve the performance problem of forward deduction engines, we presented a model of a parallelization version of automated forward deduction based

on master-slave model and implemented a parallelization version of an automated forward deduction system for general-purpose entailment calculus, named EnCal, based on the model on a shared-memory parallel computer and a clusters of PCs as a case study to investigate the effectiveness of parallel processing for improving the performance of automated forward deduction. Our experiments showed that the execution time on both cases gets shorter in proportion to the increasement in the number of processors without depending on the number of deduced conclusions and given premises. Hence, improving the performance by parallel processing is effective for not only EnCal but also forward deduction engines. We therefore showed that it is in principle possible to implement a practical forward deduction engine by using the our parallelization model of automated forward deduction.

We investigated the usefulness of automated forward deduction based on SRL in scientific discovery. We tried to solve two logic puzzles on the reasoning approach by automated forward deduction based on SRL as a particular case of scientific discovery and succeeded to solve those puzzles. This case study showed that automated forward deduction based on SRL is an useful tool to automate the reasoning process in scientific discovery.

We investigated the problem of automated theorem finding by forward deduction and tried to find theorems in von Neumann-Bernays-Godel set theory, NBG set theory for short, by automated forward deduction based on SRL as a case study. In this case study, we deduced some NBG set theory theorems from axioms of NBG set theory, but no paradoxical theorem was deduced. Our results showed that it is in principle possible to find theorems of the NBG set theory by automated forward deduction based on SRL.

We investigated what role automated forward deduction based on temporal relevant logics can play in anticipatory systems with requirements of high reliability and high security. We showed that the high-performance automated forward deduction based on temporal relevant logics is necessary to implementation of an anticipatory reasoning engine.

We proposed spatial relevant logics as the fundamental logic system to underlie representing and reasoning about geographic knowledge, and showed that a spatial reasoning engine based on spatial relevant logics for geographic information systems can be implemented following our approaches.

6.2 Future works

The ultimate goal of this research is to implement a practical forward deduction engine and to apply the forward deduction engine into applications. There is a gap between the goal and our contributions in this thesis.

First, we investigated the implicational paradoxes in the axiomatic system of CML with only implication and negation, as the first step of quantitative comparative study between CML and SRL. This is only a quantitative comparative study between CML and the intersection between SRL and RL. A future work should be to investigate whether SRL is quantitative suitable logic systems to underlie automated forward deduction than CML and RL, by quantitative comparative studies

between RL and SRL, or CML and SRL.

Second, we showed that the parallelization model proposed by us is an effective model for improving the performance of automated forward deduction. In the model, all inference rules and previously deduced conclusions and given premises are shared by all slaves. However, in order to implement a forward deduction engine for scientific discovery, automated theorem finding, and geographic information systems, dividing inference rules, previously deduced conclusions and given premises into each address space of a slave is better than sharing them, because among of given premises and deduced conclusions may be huge in those applications. A future work should be to investigate the parallelization model whose data is divided into each address space of a slave.

Third, at present, EnCal cannot deal with various extensions of SRL, such as temporal relevant logics or spatial relevant logics. Hence we have to improve EnCal to deal with those logics. On the other hand, we have implemented only a parallelization version of EnCal-P, but never done that of EnCal-Q, EnCal-Q2, EnCal-E, and EnCal-E2. We therefore have to implement parallelization versions of those.

Publications

Refereed papers published in journals or books (first author)

- Yuichi GOTO, Shinsuke NARA, and Jingde CHENG: Efficient Anticipatory Reasoning for Anticipatory Systems with Requirements of High Reliability and High Security, International Journal of Computing Anticipatory Systems, Vol. 14, pp. 156-171, CHAOS, December 2004.

Refereed papers published in journals or books (co-author)

- Jingde CHENG, and Yuichi GOTO: A Strong Relevant Logic Approach to the Calculus of Fuzzy Conditionals, in D. Ruan, P. D'hondt, and E. E. Kerre (Eds.), "Computational Intelligent Systems for Applied Research," pp. 66-74, World Scientific, September 2002.
- Jingde CHENG, Naoki AKIMOTO, Yuichi GOTO, Masato KOIDE, Kouichi NANASHIMA, and Shinsuke NARA: HILBERT: An Autonomous Evolutionary Information System for Teaching and Learning Logic, in C. P. Constantinou and Z. C. Zacharia (Eds.), "Computer Based Learning in Science, Volume I, New Technologies And Their Applications in Education," pp. 245-254, University of Cyprus, July 2003.
- Jingde CHENG, and Yuichi GOTO: Representing and Reasoning about Spatial Knowledge Based on Spatial Relevant Logic, in S. Wang, K. Tanaka, et al. (Eds.), "ER Workshops 2004, CoMoGIS, Shanghai, China, November 8-12, 2004, Proceedings," Lectures Notes in Computer Science, Vol. 3289, pp. 114-126, Springer-Verlag, November 2004.

Refereed papers published in international conference proceedings (first author)

- Yuichi GOTO, Daisuke TAKAHASHI, and Jingde CHENG: Parallel Forward Deduction Algorithms of General-Purpose Entailment Calculus on Shared-Memory Parallel Computers, Proceedings of the 2nd ACIS International Conference on Software Engineering, Artificial Intelligence, Networking & Parallel/Distributed Computing (SNPD'01), pp. 168-175, August 2001.
- Yuichi GOTO, Shinsuke NARA, Daisuke TAKAHASHI, and Jingde CHENG: Improving Performance of Automated Forward Deduction System EnCal on Shared-Memory Parallel Computers, Proceedings of

the 3rd International Conference on Parallel and Distributed Computing, Applications and Technologies (PDCAT'02), pp. 63-68, September 2002.

- Yuichi GOTO, Takahiro KOH, and Jingde CHENG: A Comparative Study on Paradoxical Conditionals in Classical Mathematical Logic, Relevant Logics, and Strong Relevant Logics, Proceedings of the 1st International Conference on Knowledge Economy and Development of Science and Technology (International Workshop on Research Directions and Challenge Problems in Advanced Information Systems Engineering) (KEST'03), pp. 69-75, September 2003.

Refereed papers published in international conference proceedings (co-author)

- Shinsuke NARA, Yuichi GOTO, Daisuke TAKAHASHI, and Jingde CHENG: Parallel Forward Deduction System for General-Purpose Entailment Calculus on Clusters of PCs, Proceedings of the IASTED International Conference on Networks, Parallel and Distributed Processing, and Applications (NPDPA'02), pp. 359-364, October 2002.
- Ru LU, Feng SHANG, Yuichi GOTO, and Jingde CHENG: A Digital Reference Room for E-learning, Proceedings of the 1st International Conference on Knowledge Economy and Development of Science and Technology (International Workshop on Research Directions and Challenge Problems in Advanced Information Systems Engineering) (KEST'03), pp. 17-21, September 2003.
- Shinsuke NARA, Yuichi GOTO, and Jingde CHENG: Efficient Forward Deduction for Discovery and Prediction by Parallel Processing, Proceedings of the 1st International Conference on Knowledge Economy and Development of Science and Technology (International Workshop on Research Directions and Challenge Problems in Advanced Information Systems Engineering) (KEST'03), pp. 76-82, September 2003.
- Xiaobin WANG, Yuichi GOTO, Shinsuke NARA, and Jingde CHENG: Automated Theorem Finding by Forward Deduction Based on Strong Relevant Logic: A Case Study in NBG Set Theory, Proceedings of the 1st International Conference on Knowledge Economy and Development of Science and Technology (International Workshop on Research Directions and Challenge Problems in Advanced Information Systems Engineering) (KEST'03), pp. 83-91, September 2003.

Refereed papers published in local conference proceedings (first author)

- Yuichi GOTO, Xin LIU, Masoto KOIDE, Daisuke TAKAHASHI, and Jingde CHENG: Can E-Voting and E-Questionnaire Become E-services? Proceedings of the 2001 IPSJ SIGSE Winter Workshop, pp. 63-64, Itou, Japan, January 2002 (in Japanese).

- Yuichi GOTO, Masato KOIDE, Keigo NAGAHAMA, and Jingde CHENG: ENQUETE-BAISE: An E-Questionnaire Server for Ubiquitous Questionnaire, Proceedings of the 2004 IPSJ Symposium on Informatics, pp. 81-84, Tokyo, Japan, January 2004 (in Japanese).

Refereed papers published in local conference proceedings (co-author)

- Kouichi NANASHIMA, Yuichi GOTO, and Jingde CHENG: Construction of Reactive Systems with System Buses, Proceedings of the 2001 IPSJ SIGSE Winter Workshop, pp. 15-16, Itou, Japan, January 2002 (in Japanese).
- Shinsuke NARA, Yuichi GOTO, and Jingde CHENG: Improving the Performance of Reasoning Engine by Parallel Processing, Proceedings of the 10th IPSJ SIGMPS Symposium on Mathematical Modeling and Problem Solving - Problem Solving by Parallel and Distributed Procedure (SPSPDP 2003), pp. 101-108, Kyoto, Japan, October 2003 (in Japanese).
- Kouichi NANASHIMA, Yuichi GOTO, and Jingde CHENG: On the Implementation of Soft System Buses : Towards Persistent Systems for Ubiquitous Computing, Proceedings of the 2004 IPSJ Symposium on Informatics, pp. 77-80 Tokyo, Japan, January 2004 (in Japanese).

Bibliography

[1] Alan R. ANDERSON and Nuel D. BELNAP Jr.: *Entailment: The Logic of Relevance and Necessity, Vol. 1*, Princeton University Press , 1975.

[2] Alan R. ANDERSON, Nuel D. BELNAP Jr. and J. Michael DUNN: *Entailment: The Logic of Relevance and Necessity, Vol. 2*, Princeton University Press, 1992.

[3] Nicholas CARRIERO and David GELERNTER: *How to Write Parallel Programs: A First Course*, MIT Press, 1990.

[4] Jingde CHENG: Logical Tool of Knowledge Engineering: Using Entailment Logic rather than Mathematical Logic, Proc. 19th ACM Annual Computer Science Conference, pp. 228–238, 1991.

[5] Jingde CHENG: Entailment Calculus as the Logical Basis of Automated Theorem Finding in Scientific Discovery, in V.-P. Raul (Ed.), *Systematic Methods of Scientific Discovery: Papers from the 1995 Spring Symposium*, AAAI Press - American Association for Artificial Intelligence, pp. 105–110, 1995.

[6] Jingde CHENG: EnCal: An Automated Forward Deduction System for General–Purpose Entailment Calculus, in N. Terashima and E. Altman (Eds.), *Advanced IT Tools, Proceedings of the 14th WCC, Canberra*, Chapman & Hall, pp. 507–517, 1996.

[7] Jingde CHENG: The Fundamental Role of Entailment in Knowledge Representation and Reasoning, *Journal of Computing and Information*, Vol. 2, No. 1, pp. 853–873, 1996.

[8] Jingde CHENG: Temporal Relevant Logic as the Logic Basis for Reasoning about Dynamics of Concurrent Systems, *Proc. 1998 IEEE-SMC Annual International Conference on Systems, Man and Cybernetics*, San Diego, California,USA, pp. 794–799, 1998.

[9] Jingde CHENG: Temporal Relevant Logic: What Is and Why Study It?, *Abstract of the IUHPS/DLMPS 11th International Congress of Logic, Methodology and Philosophy of Science*, p. 253, 1999.

[10] Jingde CHENG: A Strong Relevant Logic Model of Epistemic Processes in Scientific Discovery, in E. Kawaguchi, H. Kangassalo, H. Jaakkola, and I. A.

Hamid (Eds.), *Information Modelling and Knowledge Bases XI*, IOS Press, pp. 136–159, 2000.

[11] Jingde CHENG: Mathematical Knowledge Representation and Reasoning Based on Strong Relevant Logic, in R. Trappl (Ed.), *Cybernetics and Systems 2002, Proceedings of 16th European Meeting on Cybernetics and Systems Research, Vol. II*, Austrian Society for Cybernetic Studies, pp. 789–794, 2002.

[12] Jingde CHENG: Anticipatory Reasoning-Reacting Systems, *Proc. International Conference on Systems,Development and Self-organization (ICSDS'2002)*, Beijing, China, pp. 161–165, 2002.

[13] Jingde CHENG and Yuichi GOTO: Representing and Reasoning about Spatial Knowledge Based on Spatial Relevant Logic, in S. Wang, K. Tanaka, et al. (Eds.), *Conceptual Modeling for Advanced Application Domains, ER 2004 Workshops CoMoGIS, CoMWIM, ECDM, CoMoA, DGOV, and eCOMO, Shanghai, China, November 2004, Proceedings*, Lecture Notes in Computer Science, Springer-Verlag, pp. 114–126, 2004.

[14] Ron CHIRSLEY: Some Foundational Issues Concerning Anticipatory Systems, *International Journal of Computing Anticipatory Systems*, Vol. 11, pp. 3–18, 2002.

[15] A. G. COHN, B. BENNETT, J. M. GOODAY, and N. GOTTS: RCC: a calculus for Region based Qualitative Spatial Reasoning, *GeoInformatica*, Vol. 1, pp. 275–316, 1997.

[16] Martin DAVIS: The Early History of Automated Deduction, in A. Robinson, and A. Voronkov (Eds.), *Handbook of Automated Reasoning*, Vol. 1, MIT press, pp. 5–15, 2001.

[17] Daniel M. DUBOIS: Introduction to Computing Anticipatory Systems, *International Journal of Computing Anticipatory Systems*, Vol. 2, pp. 3–14, 1998.

[18] Daniel M. DUBOIS: Review of Incursive, Hyperincursive and Anticipatory Systems - Foundation of Anticipation in Electromagnetism, in D. M. Dubois (Ed.), *Computing Anticipatory Systems: CASYS'99 - Third International Conference*, AIP Conference Proceedings 517, The American Institute of Physics, pp. 3–30, 2000.

[19] J. Michael DUNN: Relevance Logic and Entailment, in D. Gabbay, and F. Guenthner (Eds.), *Handbook of Philosophical Logic*, Kluwer Academic, pp. 117–224, 1986.

[20] J. Michael DUNN and G. RESTALL: Relevance Logic, in D. Gabbay, and F. Guenthner (Eds.), *Handbook of Philosophical Logic, 2nd Edition*, Vol. 6, Kluwer Academic, pp. 1–128, 2002.

[21] M. J. EGENHOFER, and R. G. GOLLEDGE: *Spatial and Temporal Reasoning in Geographic Information Systems*, New York: Oxford University Press, 1998.

[22] Oscar N. GARCIA and Chien YI-TZW: *Knowledge-Based Systems: Fundamentals and Tools*, IEEE Computer Society Press, 1991.

[23] Kurt GODEL: Russell's Mathematical Logic, in P. A. Schilpp, (Ed.), *The Philosophy of Bertrand Russell*, Open Court Publishing Company, pp. 123–153, 1944.

[24] Yuichi GOTO: Knowledge Discovery with Strong Relevant Logics and EnCal: Automated Forward Deduction System for General-Purpose Entailment Calculus, Master Thesis, Graduate School of Science and Engineering, Saitama University, 2003 (in Japanese).

[25] Yuichi GOTO, Takahiro KOH and Jingde CHENG: A Comparative Study on Paradoxical Conditionals in Classical Mathematical Logic, Relevant Logics, and Strong Relevant Logics, *Proc. of the 1st International Conference on Knowledge Economy and Development of Science and Technology (International Workshop on Research Directions and Challenge Problems in Advanced Information Systems Engineering) (KEST'03)*, Honjo, Japan, pp. 69–75, 2003.

[26] Yuichi GOTO, Shinsuke NARA, and Jingde CHENG: Efficient Anticipatory Reasoning for Anticipatory Systems with Requirements of High Reliability and High Security, *International Journal of Computing Anticipatory Systems*, Vol. 14, pp. 156-171, CHAOS, December 2004.

[27] Frederick HAYES-ROTH, Donald A. WATERMAN and Douglas B. LENAT: *Building expert systems*, Addison-Wesley, 1983.

[28] Zohar MANNA and Amir PNUELI: *The temporal logic of reactive and concurrent systems*, Springer-Verlag, 1992.

[29] Zohar MANNA and Amir PNUELI: *Temporal verification of reactive systems*, Springer-Verlag, 1995.

[30] E. D. MARES and R. K. MEYER: Relevant Logics, in L. Goble, (Ed.), *The Blackwell Guide to Philosophical Logic*, Blackwell, pp. 280–308, 2001.

[31] MPICH-SCore.: PC Clusters Consortium.
http://pdswww.rwcp.or.jp/home-j.html.

[32] Mihai NADIN: Anticipation - A Spooky Computation, *International Journal of Computing Anticipatory Systems*, Vol. 6, 2000.

[33] Mihai NADIN: Anticipatory Computing, *Ubiquity - The ACM IT Magazine and Forum*, Vol. 1, 2000. Issue 40.

[34] Shinsuke NARA, Yuichi GOTO, Daisuke TAKAHASHI and Jingde CHENG: Parallel Forward Deduction System for General-Purpose Entailment Calculus on Clusters of PCs, *Proc. of the IASTED International Conference on Networks, Parallel and Distributed Processing, and Applications (NPDPA'02)*, Tsukuba, Japan, pp. 359–364, 2002.

[35] Kazunori NISHI, Jingde CHENG and Kazuo USHIJIMA: Improving the Performance of Automated Forward Deduction System EnCal, *Proc. International Symposium on High Performance Computing (ISHPC'97)*, Lecture Notes in Computer Science, Vol. 1336, Springer-Verlag, pp. 371–380, 1997.

[36] Omni.: RWCP OpenMP compiler project.
http://www.hpcc.jp/Omni/.

[37] Hirokazu ONODA: *Logic puzzle collection to ask and answer each other*, Kodansha, 2002 (in Japanese).

[38] OpenMP.: Simple, Portable, Scalable SMP Programming.
http://www.openmp.org/.

[39] Art QUAIFE: *Automated Development of Fundamental Mathematical Theories*, Kluwer Academic, 1992.

[40] David A. RANDELL and Zhan CUI and Anthony G. COHN: A Spatial Logic based on Regions and Connection, *Proc. of 3rd International Conference on Knowledge Representation and Reasoning*, pp. 165–176, 1992.

[41] Stephen READ: *Relevant Logic: A Philosophical Examination of Inference*, Blackwell, 1988.

[42] Robert ROSEN: *Anticipatory Systems - Philosophical, Mathematical and Methodological Foundations*, Pergamon Press, 1985.

[43] Herbert A. SIMON: Why Should Machines Learn?, in R. S. Michalski, J. G. Carbonell, and T. M. Mitchell(Eds.), *Machine Learning: An Artificial Intelligence Approach*, Springer, Berlin, Heidelberg, pp. 25–37, 1984.

[44] Herbert A. SIMON: What is a Systematic Method of Scientific Discovery?, *Systematic Methods of Scientific Discovery - Papers from 1995 Spring Symposium, Technical Report SS-9503*, AAAI Press, pp. 1–2, 1995.

[45] Takao TAGAWA, Junya OHBORI, Jingde CHENG, and Kazuo USHIJIMA: Strong Relevance Principle on Relevant Logics, Transactions of the Japanese Society for Artificial Intelligence, Vol. 13, No. 3, pp. 387–394, 1998 (in Japanese).

[46] Albert TARSKI: *Introduction to Logic and to the Methodology of the Deductive Sciences, 4th Edition, Revised*, Oxford University Press, 1994.

[47] Larry WOS: *Automated Reasoning: 33 Basic Research Problems*, Prentice-Hall, 1988.

[48] Larry WOS: The Problem of Automated Theorem Finding, *Journal of Automated Reasoning*, Vol. 10, No. 1, pp. 137–138, 1993.

Index