

The Theory Grid and Grid Theorists

Jingde Cheng, Shinsuke Nara, Takahiro Koh, and Yuichi Goto

Department of Information and Computer Sciences, Saitama University, Japan
{cheng, nara, t_koh, gotoh}@aise.ics.saitama-u.ac.jp

**“In mathematics the art of proposing a question
must be held of higher value than solving it.”**

– G. F. L. P. Cantor, 1867.

Abstract

This paper proposes a novel research direction: to build the Theory Grid as cooperatively shared formal theories within a virtual organization, and then to implement Grid Theorists as the wisdom of crowds working based on the Theory Grid such that the grid theorists have the ability to find new theorems and propose new questions semi-automatically.

1. Introduction

Wos in 1988 proposed 33 basic research problems in automated reasoning [27, 28]. The 31st one is the problem of automated theorem finding (**ATF** for short): “What properties can be identified to permit an automated reasoning program to find new and interesting theorems, as opposed to proving conjectured theorems?”

The **ATF** problem is still completely open until now. The most important and difficult requirement of the problem is that, in contrast to proving conjectured theorems supplied by the user, it asks properties and/or criteria such that an automated reasoning program can use them to find some theorems in a field that must be evaluated by theorists of the 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.

Note that the **ATF** problem requires general properties and/or criteria that an automated reasoning program should be obedient to, but not some concrete theorems in a special field. Therefore, any of those automated reasoning programs [12, 20, 23, 29] to discover or prove some concrete theorems in a special field by heuristic search has nothing to do with the **ATF** problem. The **ATF** problem also requires that an automated reasoning program can use the properties and/or criteria to find “new and interesting” theorems, but not those known or trivial ones. Finally, from the description of the **ATF** problem, we can see that it

explicitly requires properties and/or criteria of an individual automated reasoning program.

Any scientific discovery was not completely based on only one scientist’s wisdom and experiences but must have achieved as a result of endeavors by many scientists. On the other hand, the concept of grid in Grid Computing discipline is defined as the controlled and coordinated resource sharing and problem solving in dynamic, multi-institutional virtual organizations [13-16]. The Grid Computing technologies and/or activities involve networking services, connections, and communication of an unlimited number of resources within a virtual organization, and therefore, the Grid Computing has been refereed as the world’s largest computer [13-16].

This paper considers a revised edition of the **ATF** problem: “What properties can be identified to permit **a crowd of automated reasoning programs** to find new and interesting theorems, as opposed to proving conjectured theorems?” We propose a novel research direction: to build the Theory Grid as cooperatively shared formal theories within a virtual organization, and then to implement Grid Theorists as the wisdom of crowds working based on the Theory Grid such that the grid theorists have the ability to find new theorems and propose new questions semi-automatically [10].

Our conjecture to motivate this new research direction is that maybe a crowd of automated reasoning programs can more easily find new and interesting theorems than an individual automated reasoning program. This new research direction is also motivated by our practices in a research to solve the original **ATF** problem where we found that automated theorem finding by forward deduction needs huge computing power and resources. Today, only the Grid Computing can provide such huge computing power and resources.

2. The Theory Grid and Grid Theorists

The *Theory Grid* is a formal theory infrastructure that coordinates various formal theories (represented

by and based on various formal logic systems) in a distributed way using standard, open, general-purpose protocols and interfaces to meet demands of its application programs for theorem discovery and/or question proposition [10].

The concept of Theory Grid differs from the concepts of Computational Grid and Data Grid in that the Theory Grid concerns coordinated formal theory sharing for theorem discovering and/or question proposing. From the viewpoint of architecture, the Theory Grid should be built on the top of a computational and/or data grid environment. Therefore, if an intrinsic function of a grid computing environment is to provide its users with standard, open, general-purpose protocols and interfaces for computational power and data sharing, then a similar intrinsic function of the Theory Grid is to provide its users with standard, open, general-purpose protocols and interfaces for formal theory sharing.

In general, a **formal logic system** \mathbf{L} consists of a formal language, called the **object language** and denoted by $F(\mathbf{L})$, which is the set of all **well-formed formulas** of \mathbf{L} , and a **logical consequence relation**, denoted by meta-linguistic symbol $\vdash_{\mathbf{L}}$, such that for $P \subseteq F(\mathbf{L})$ and $c \in F(\mathbf{L})$, $P \vdash_{\mathbf{L}} c$ means that within the framework of \mathbf{L} , c is a valid conclusion of premises P , or given P as premises, c as a valid conclusion follows from P . For a formal logic system $(F(\mathbf{L}), \vdash_{\mathbf{L}})$, a **logical theorem** t is a formula of \mathbf{L} such that $\emptyset \vdash_{\mathbf{L}} t$ where \emptyset is the empty set. We use $\text{Th}(\mathbf{L})$ to denote the set of all logical theorems of \mathbf{L} . $\text{Th}(\mathbf{L})$ is completely determined by the logical consequence relation $\vdash_{\mathbf{L}}$. 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. A formal logic system \mathbf{L} is said to be **explosive** if and only if $\{A, \neg A\} \vdash_{\mathbf{L}} B$ for any two different formulas A and B ; \mathbf{L} is said to be **paraconsistent** if and only if it is not explosive.

Let $(F(\mathbf{L}), \vdash_{\mathbf{L}})$ be a formal logic system and $P \subseteq F(\mathbf{L})$ be a non-empty set of **sentences** (i.e. **closed well-formed formulas**). A **formal theory** with premises P based on \mathbf{L} , called a **\mathbf{L} -theory with premises P** and denoted by $T_{\mathbf{L}}(P)$, is defined as $T_{\mathbf{L}}(P) =_{\text{df}} \text{Th}(\mathbf{L}) \cup \text{Th}_{\mathbf{L}}^{\circ}(P)$, and $\text{Th}_{\mathbf{L}}^{\circ}(P) =_{\text{df}} \{et \mid P \vdash_{\mathbf{L}} et \text{ and } et \notin \text{Th}(\mathbf{L})\}$ where $\text{Th}(\mathbf{L})$ and $\text{Th}_{\mathbf{L}}^{\circ}(P)$ are called the **logical part** and the **empirical part** of the formal theory, respectively, and any element of $\text{Th}_{\mathbf{L}}^{\circ}(P)$ is called an **empirical theorem** of the formal theory. A formal theory $T_{\mathbf{L}}(P)$ is said to be **directly inconsistent** if and only if there exists a formula A of \mathbf{L} such that both $A \in P$ and $\neg A \in P$ hold. A formal theory $T_{\mathbf{L}}(P)$ is said to be **indirectly inconsistent** if and only if it is not

directly inconsistent but there exists a formula A of \mathbf{L} such that both $A \in T_{\mathbf{L}}(P)$ and $\neg A \in T_{\mathbf{L}}(P)$. A formal theory $T_{\mathbf{L}}(P)$ is said to be **consistent** if and only if it is neither directly inconsistent nor indirectly inconsistent. A formal theory $T_{\mathbf{L}}(P)$ is said to be **explosive** if and only if $A \in T_{\mathbf{L}}(P)$ for arbitrary formula A of \mathbf{L} ; $T_{\mathbf{L}}(P)$ is said to be **paraconsistent** if and only if it is not explosive. An explosive formal theory is not useful at all. Therefore, any meaningful formal theory should be paraconsistent. Note that if a formal logic system \mathbf{L} is explosive, then any directly or indirectly inconsistent \mathbf{L} -theory $T_{\mathbf{L}}(P)$ must be explosive.

To build the Theory Grid in a grid computing environment, it is necessary to generate and distribute various formal theories over the grid computing environment. From the viewpoint of relationships among various formal theories, the Theory Grid can be modeled as a semi-lattice such that its different elements represent different formal theories respectively, its partial order among elements represents the inclusion relation between two formal theories, and its minimum element represents a formal theory of the axiomatic set theory. There may be various types of the inclusion relation between two formal theories. For examples, a formal theory $T_{\mathbf{L}}(P)$ may be a subset of another formal theory $T_{\mathbf{L}}(P')$ if $P \subseteq P'$; a formal theory $T_{\mathbf{L}}(P)$ may be a subset of another formal theory $T_{\mathbf{L}}(P')$ if $\text{Th}(\mathbf{L}) \subset \text{Th}(\mathbf{L}')$; a formal theory $T_{\mathbf{L}}^k(P)$ may be a subset of another formal theory $T_{\mathbf{L}}^{k+1}(P)$ if $\text{Th}^k(\mathbf{L}) \subset \text{Th}^{k+1}(\mathbf{L})$ where k denotes the degree of conditional (see below).

Since a formal theory $T_{\mathbf{L}}(P)$ is generally an infinite set of formulas, even though premises P are finite, we have to find some method in automated theorem finding to limit the range of candidates for “new theorems” to a finite set of formulas. The strategy we adopted is to sacrifice the completeness to get the finite set of candidates. This is based on Cheng’s conjecture that almost all “new and interesting theorems” of a formal theory can be deduced from the premises of that theory by finite inference steps concerned with finite number of low degree entailments [7, 8].

For a formal logic system where the notion of conditional is represented by a primitive connective, say ‘ \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 $A \Rightarrow B$ or $Q(A \Rightarrow B)$ where Q is the quantifier prefix of $A \Rightarrow B$ is called a **first degree conditional** if and only if both A and B are zero degree formulas; a formula A is called a **first degree formula** if and only if it satisfies any one of the following conditions: (1) A is a first degree conditional, (2) A is in the form $+B$ where $+$ is a one-place connective such as negation and so on, and B is a first degree formula, (3) A is in the form

$B * C$ where $*$ is a non-implicational two-place connective such as conjunction or disjunction and so on such that both of B and C is first degree formulas, or one of B and C is a first degree formula and another is a zero degree formula, and (4) A is in the form $Q(B)$ where Q is the quantifier prefix and B is a first degree formula. More general, let $\text{deg}_{\rightarrow}(A)$ denote the degree of conditional in a formula A . $\text{deg}_{\rightarrow}(+A) =_{\text{df}} \text{deg}_{\rightarrow}(A)$ where $+$ is a one-place connective such as negation and so on, $\text{deg}_{\rightarrow}(A * B) =_{\text{df}} \max\{\text{deg}_{\rightarrow}(A), \text{deg}_{\rightarrow}(B)\}$ where $*$ is a non-implicational two-place connective such as conjunction or disjunction and so on, $\text{deg}_{\rightarrow}(A \Rightarrow B) =_{\text{df}} 1 + \max\{\text{deg}_{\rightarrow}(A), \text{deg}_{\rightarrow}(B)\}$, and $\text{deg}_{\rightarrow}(Q(A)) =_{\text{df}} \text{deg}_{\rightarrow}(A)$ where Q is the quantifier prefix of A . Let k be a natural number. A formula A is called a k^{th} **degree formula** if and only if $\text{deg}_{\rightarrow}(A) = k$; in particular, a k^{th} degree formula A is called a k^{th} **degree conditional** if it is a conditional.

Let $(F(\mathbf{L}), \vdash_{\mathbf{L}})$ be a formal logic system and k be a natural number. The k^{th} **degree fragment** of \mathbf{L} , denoted by $\text{Th}^k(\mathbf{L})$, is a set of logical theorems of \mathbf{L} which is inductively defined as follows: (1) if A is a j^{th} ($j \leq k$) degree formula and an axiom of \mathbf{L} , then $A \in \text{Th}^k(\mathbf{L})$, (2) if A is a j^{th} ($j \leq k$) degree formula which is the result of applying an inference rule of \mathbf{L} to some members of $\text{Th}^k(\mathbf{L})$, then $A \in \text{Th}^k(\mathbf{L})$, and (3) Nothing else are members of $\text{Th}^k(\mathbf{L})$. Obviously, the definition of the k^{th} degree fragment of logic \mathbf{L} is constructive. The k^{th} degree fragment of logic \mathbf{L} can be constructed in principle by forward deductions based on the logic. Note that the k^{th} degree fragment of logic \mathbf{L} does not necessarily include all k^{th} degree logical theorems of \mathbf{L} because it is possible for \mathbf{L} that deductions of some k^{th} degree logical theorems of \mathbf{L} must invoke those logical theorems whose degrees are higher than k . On the other hand, the following holds obviously: $\text{Th}^0(\mathbf{L}) \subset \text{Th}^1(\mathbf{L}) \subset \dots \text{Th}^{k-1}(\mathbf{L}) \subset \text{Th}^k(\mathbf{L}) \subset \text{Th}^{k+1}(\mathbf{L}) \subset \dots$

Let $(F(\mathbf{L}), \vdash_{\mathbf{L}})$ be a formal logic system, $P \subset F(\mathbf{L})$ ($P \neq \emptyset$), and k and j be two natural numbers. A formula A is said to be j^{th} -**degree-deducible from P based on $\text{Th}^k(\mathbf{L})$** if and only if there is a finite sequence of formulas f_1, \dots, f_n such that $f_n = A$ and for all i ($i \leq n$) (1) $f_i \in \text{Th}^k(\mathbf{L})$, or (2) $f_i \in P$, or (3) f_i ($\text{deg}_{\rightarrow}(f_i) \leq j$) is the result of applying an inference rule to some members f_{j_1}, \dots, f_{j_m} ($j_1, \dots, j_m < i$) of the sequence. The set of all formulas which are j^{th} -degree-deducible from P based on $\text{Th}^k(\mathbf{L})$ is called the j^{th} **degree fragment of the formal theory with premises P based on $\text{Th}^k(\mathbf{L})$** , denoted by $\text{T}_{\text{Th}^k(\mathbf{L})}^j(P)$. A formula is said to be j^{th} -**degree-deductive from P based on $\text{Th}^k(\mathbf{L})$** if and only if it is j^{th} -degree-deducible from P based on $\text{Th}^k(\mathbf{L})$ but not $(j-1)^{\text{th}}$ -degree-deducible from P based on $\text{Th}^k(\mathbf{L})$. Note that in the above definitions, we do

not require ($j \leq k$). The notion of j^{th} -degree-deductive can be used as a metric to measure the difficulty of deducing an empirical theorem from given premises P based on logic \mathbf{L} . The difficulty is relative to the complexity of problem being investigated as well as the strength of underlying logic \mathbf{L} .

Conceptually, a **Grid Theorist** is a crowd of (an unlimited number of) automated reasoning programs working together on the Theory Grid in order to discover new and interesting theorems and/or propose new and interesting questions in a special field. A successful grid theorist should have the ability to discover new and interesting theorems and/or propose new and interesting questions by a systematic way that can be learnt by other grid theorists [10]. Thus, the fundamental questions we have to answer are: Can we implement such a successful grid theorist? How can we implement a successful grid theorist?

3. An Architecture

Figure 1 shows an architecture we designed for the Theory Grid and grid theorists for automated theorem finding and automated question proposing. The Theory Grid computing environment consists of two groups: the server group that includes a number of databases (knowledge-bases) and functional components to implement the Theory Grid, and the client group that includes an unlimited number of clients to behave grid theorists working on the Theory Grid. All databases (knowledge-bases), functional components, and clients are distributed over the Internet.

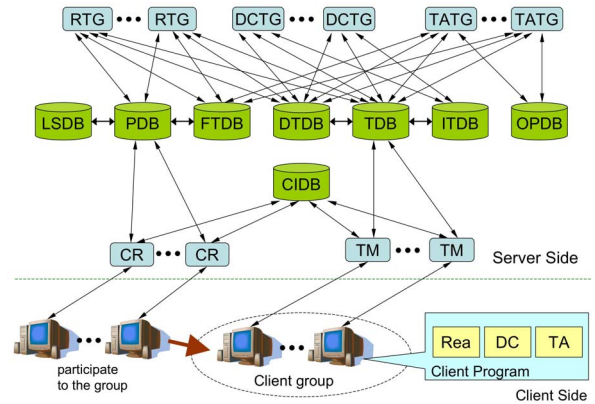


Figure 1 An Architecture of the Theory Grid and Grid Theorists

The databases (knowledge-bases) of the server group are as follows:

Logic System Database (LSDB): The database managing various fragments of underlying logic systems and various inference rules.

Formal Theory Database (FTDB): The database managing previously known empirical premises, empirical theorems, and empirical inference rules of formal theories.

Open Problem Database (OPDB): The database managing previously known open problems.

Deduced Theorem Database (DTDB): The database managing new empirical theorems deduced by clients.

Intermediate Theorem Database (ITDB): The database managing those empirical theorems which are intermediate results need to be checked.

Client Information Database (CIDB): The database managing information about all clients.

Task Database (TDB): The database managing information about all reasoning tasks, duplication check tasks, and theorem analysis tasks.

Project Database (PDB): The database managing information about the current project of automated theorem finding, e.g., a fragment of underlying logic system, inference rules, nest degrees of logic connectives, and so on, which is specified by an organization of automated theorem finding.

Although in Figure 1 we show only one for each type of the above databases, there may be a number of databases of each type in a large-scale grid computing environment to implement the Theory Grid.

The functional components of the server group are as follows:

Client Register (CR): A client register CR registers clients such that it gets setting information, a fragment of the underlying logic system and inference rules from the Project Database PDB, sends a client reasoning program, its setting information, the fragment of the underlying logic system and the inference rules to a client, and sends information about registered clients to the Client Information Database CIDB. There may be a number of client registers.

Reasoning Task Generator (RTG): A reasoning task generator RTG generates reasoning tasks continuously and passes the tasks into Task Database TDB. A reasoning task is defined by premises, inference rules, and nest degrees of logic connectives. There may be a number of reasoning task generators.

Duplication Check Task Generator (DCTG): A duplication check task generator DCTG generates duplication check tasks continuously and passes the tasks into Task Database TDB. A duplication check task is defined by an empirical theorem (or schema) and a set of intermediate theorems. There may be a number of duplication check task generators.

Theorem Analysis Task Generator (TATG): A theorem analysis task generator TATG generates

theorem analysis tasks continuously and passes the tasks into Task Database TDB. A theorem analysis task is defined by a previously known theorem or a previously known open problem and a set of deduced theorems. There may be a number of theorem analysis task generators.

Task Manager (TM): A task manager TM manages reasoning tasks, duplication check tasks, and theorem analysis tasks such that it takes tasks from TDB, certifies clients, assigns tasks to a client, receives results of tasks from a client, and sends the results to TDB. There may be a number of task managers.

A client is basically a “worker” with a general-purpose reasoning engine and a general-purpose pattern matching checker. An authenticated client may perform any of three types of tasks, i.e., reasoning tasks, duplication check tasks, and theorem analysis tasks. For a reasoning task, it receives the reasoning task from a task manager, reasons about empirical theorems of the target formal theory with the specified fragment of the underlying logic system, inference rules, and premises, and sends the deduced empirical theorems to the task manager. For a duplication check tasks, it receives the duplication check task from a task manager, checks whether the intermediate theorems are duplicated against previously deduced empirical theorems or not, and sends the check results to the task manager. For a theorem analysis task, it receives the theorem analysis task from a task manager, checks whether one of the deduced theorems is a previously known theorem (in this case, the theorem is rediscovered) or a previously known open problem (in this case, the problem is resolved) or not, and sends the analysis results to the task manager. The number of clients is not limited.

Note that the above all databases (knowledge-bases) may depend on the target formal theory representing the target area of automated theorem finding and automated question proposing, while the above all functional components are independent of any formal theory. While Figure 1 shows a general architecture of the Theory Grid and grid theorists for automated theorem finding and automated question proposing in any formal theory. On the other hand, to build the Theory Grid as worldwide cooperatively shared formal theories within a large-scale virtual organization, we have to prepare different formal theories and databases (knowledge-bases) for different disciplines at first, then define and maintain the partial order among the formal theories we mentioned in Section 2, and finally organize various formal theories as the whole of the Theory Grid.

Based on the Theory Grid computing environment presented above, the grid computing approach to

automated theorem finding and automated question proposing may consist of the following steps:

(1) The principal investigator (or the principal group of investigators) decides the underlying logic system at first, then prepares the various fragments of the underlying logic system, prepare various fragments of the target formal theory based on the underlying logic system, prepares databases (knowledge-bases) LSDB, FTDB, and OPDB, defines the current project, prepares databases PDB, and finally sends call for participation to the world.

(2) Any individual or organization want to contribute to the project submits registration data of computer(s) to a client register anytime anywhere and then gets all necessary setting information, the programs of a general-purpose reasoning engine and a general-purpose pattern matching checker, the fragment of the underlying logic system, and the inference rules to work as a client or clients.

(3) Task generators, task managers, and clients work cooperatively to deduce empirical theorems by forward deduction or generate questions by induction and abduction, and present candidates of new empirical theorems and new questions to the principal investigator (or the principal group of investigators).

(4) For those candidates of new empirical theorems and new questions, the principal investigator (or the principal group of investigators) evaluates their significance and/or interestingness.

4. Scientific Challenges

To implement the Theory Grid and grid theorists, and ultimately find new theorems and propose new questions semi-automatically, there are many scientific challenges and technical issues we have to solve.

A scientific and theoretical fundamental problem, which is intrinsically important to automated theorem finding and automated question proposing, is to adapt what logic system as the fundamental logic system to underlie representing and reasoning about formal theories [6-8, 10]. Because classical mathematical logic and its various classical conservatives extensions adopt the classical account of validity (i.e., an argument is valid if and only if it is impossible for all its premises to be true while its conclusion is false) as the logical validity criterion, and do not take the relevance between the premises and conclusion of an argument into account, they accept a lot of implicational paradoxes (i.e., those implicational formulas whose antecedents are not relevant to consequents at all) as their logical theorems [2, 3]. As a result, classical mathematical logic and its various classical conservatives extensions are explosive and a

reasoning based on any of them is not necessarily relevant, i.e., the conclusion of the reasoning may be not relevant to its premises at all. Although the traditional relevant logics have rejected the implicational paradoxes, their logical axioms or theorems still include a lot of conjunction-implicational paradoxes and disjunction-implicational paradoxes, and have the problem similar to that of classical mathematical logic [7-9, 11]. Therefore, we can say that at present the only hopeful way for automated theorem finding and automated question proposing is to adopt the family of strong relevant logics, which require the strong relevance between the premises and conclusion of a valid argument, as the fundamental logic system to underlie representing and reasoning about formal theories. However, the strong relevant logics themselves have some open problems and there are many challenges for us to apply the strong relevant logics and their various extensions to representing and reasoning about various formal theories.

An intrinsic difference between automated theorem finding and automated theorem proving is that once a theorem is founded by automated theorem finding programs, there has been a proof of that theorem, i.e., the trace of finding the theorem from its premises. However, this is true only if we can certainly get a correct trace of an automated theorem finding process. For automated theorem finding in a single formal theory, it is not so difficult to record and backtrack such a trace, while for automated theorem finding concerning a group of formal theories, it is very important to correctly define and maintain the partial order among the formal theories we mentioned in Section 2.

In this paper, we only claim “to find new theorems and propose new questions semi-automatically” rather than “to find new and interesting theorems and propose new and interesting questions automatically.” This is because it is very difficult, if not impossible, to establish a general criterion to evaluate the interest of a theorem and/or question to scientists. What we can do at present is to let grid theorists provide scientists with candidates for new and interesting theorems and/or questions and remain the final evaluation for scientists to decide.

5. Concluding Remarks

The most primitive idea to find a systematic methodology of scientific discovery appeared in Aristotle’s Posterior Analytics [1]. The modern works on systematic methodology of scientific discovery started at least from Popper’s work [22]. Historically,

scientific discovery methodology is the most important subject studied in history of science and philosophy of science [17, 18, 21, 22, 26], and just recently, it became the research subject of some cognitive scientists and computer scientists who believe that the process of a scientific discovery can be described and modeled in a normal way and therefore it can be simulated by computer programs automatically [19, 24, 25].

Wos's **ATF** problem can be regarded as an attempt to find a systematic methodology in automated reasoning area. From the 1970s, there have been some works on mathematical theorem discovery and proof [12, 20, 23, 29]. But, as we have mentioned in Section 1, these works have nothing to do with the **ATF** problem.

A computational grid with distributed storage systems and execution platforms intends to provide users a powerful computing environment for large-scale computational tasks [13-16]. A data grid, usually constructed based on a computational grid, intends to provide users a global infrastructure for cooperative sharing of information in distributed data resources [13-16]. A knowledge-based grid intends to provide users a high-level, abstract reference architecture for knowledge discovery and data mining on data grids [4-6, 30]. While the Theory Grid proposed in this paper intends to provide scientists a global, computational infrastructure for scientific discovery cooperatively performed on formal theories.

The position of this paper in Computer Science is to show a novel research direction to find a systematic methodology for automated theorem finding and automated question proposing using the wisdom of crowds coordinately shared by grid computing technologies. We will discuss technical issues in the Theory Grid and grid theorists in other papers.

Acknowledgement

The work presented in this paper was supported in part by a grant from Dai Nippon Printing Co., Ltd., Japan.

References

- [1] Aristotle, "Organon," about 350 B.C.
- [2] A. R. Anderson and N. D. Belnap Jr., "Entailment: The Logic of Relevance and Necessity," Vol. I. Princeton University Press, 1975.
- [3] A. R. Anderson, N. D. Belnap Jr., and J. M. Dunn, "Entailment: The Logic of Relevance and Necessity," Vol. II. Princeton University Press, 1992.
- [4] F. Berman, "From TeraGrid to Knowledge Grid," *Communications of the ACM*, Vol. 44, No. 11, pp. 27-28, 2001.
- [5] M. Cannataro and D. Talia, "The Knowledge Grid," *Communications of the ACM*, Vol. 46, No. 1, pp. 89-93, 2003.
- [6] M. Cannataro and D. Talia, "Semantics and Knowledge Grids: Building the Next-Generation Grid," *IEEE-CS Intelligent Systems*, Vol. 19, No. 1, pp. 56-63, 2004.
- [7] J. Cheng, "Entailment Calculus as the Logical Basis of Automated Theorem Finding in Scientific Discovery," in "Systematic Methods of Scientific Discovery - Papers from the 1995 Spring Symposium," AAI Technical Report SS-95-03, pp. 105-110, 1995.
- [8] J. Cheng, "EnCal: An Automated Forward Deduction System for General-Purpose Entailment Calculus," in N. Terashima and E. Altman (Eds.), "Advanced IT Tools, IFIP World Conference on IT Tools, IFIP96 - 14th World Computer Congress," pp. 507-514, Chapman & Hall, 1996.
- [9] J. 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," *Frontiers in Artificial Intelligence and Applications*, Vol. 61, pp. 136-159, IOS Press, 2000.
- [10] J. Cheng, "Theory Grid and Grid Theorists (Position Paper)," *Proc. IPSJ SIGGN 2005 Workshop on Groupware and Network Services*, IPSJ Symposium Series Vol. 2005, No. 14, pp. 57-58, 2005.
- [11] J. Cheng, "Strong Relevant Logic as the Universal Basis of Various Applied Logics for Knowledge Representation and Reasoning," in J. Henno, H. Jaakkola, H. Kangassalo, and Y. Kiyoki (Eds.), "Information Modelling and Knowledge Bases XVII," *Frontiers in Artificial Intelligence and Applications*, Vol. 136, pp. 310-320, IOS Press, 2006.
- [12] S. Colton, "Automated Theory Formation in Pure Mathematics," Springer, 2002.
- [13] I. Foster, "What is the Grid? A Three Point Checklist," *GRID Today*, Vol. 1, No. 6, 2002.
- [14] I. Foster and C. Kesselman (Eds.), "The Grid: Blueprint for a New Computing Infrastructure," Morgan Kaufmann, 1999.
- [15] I. Foster and C. Kesselman (Eds.), "The Grid 2: Blueprint for a New Computing Infrastructure," Morgan Kaufmann, 2003 (2nd Edition).
- [16] J. Joseph and C. Fellenstein, "Grid Computing," Prentice Hall, 2004.
- [17] I. Lakatos, "Proofs and Refutations: The Logic of Mathematical Discovery," Edited by J. Worrall and E. Zahar, Cambridge University Press, 1976.
- [18] I. Lakatos, "The Methodology of Scientific Research Programmes," Edited by J. Worrall and G. Currie, Cambridge University Press, 1978.
- [19] P. Langley, H. A. Simon, G. L. Bradshaw, and J. M. Zytkow, "Scientific Discovery - Computational Explorations of the Creative Processes," MIT Press, 1987.
- [20] D. B. Lenat, "AM: An Artificial Intelligence Approach to Discovery in Mathematics as Heuristic Search," Ph. D. Thesis, Stanford University, 1976.
- [21] T. Nickles (ed.), "Scientific Discovery, Logic, and Rationality," D. Reidel, 1980.
- [22] K. R. Popper, "The Logic of Scientific Discovery (Translation of 'Logik der Forschung', Verlag von Julius Springer, 1935, Vienna)," Hutchinson Education, 1959.
- [23] A. Quaife, "Automated Development of Fundamental Mathematical Theorems," Kluwer Academic, 1992.
- [24] J. Shrager and P. Langley, "Computational Approaches to Scientific Discovery," in J. Shrager and P. Langley (eds.), "Computational Models of Scientific Discovery and Theory Formation," pp. 1-25, Morgan Kaufmann, 1990.
- [25] H. A. Simon, "What is a Systematic Method of Scientific Discovery?" in "Systematic Methods of Scientific Discovery - Papers from the 1995 Spring Symposium," AAI Technical Report SS-95-03, pp. 1-2, 1995.
- [26] J. Trusted, "The Logic of Scientific Inference," Macmillan Press, 1979.
- [27] L. Wos, "Automated Reasoning: 33 Basic Research Problems," Prentice-Hall, 1988.
- [28] L. Wos, "The Problem of Automated Theorem Finding," *Journal of Automated Reasoning*, Vol. 10, No. 1, pp. 137-138, 1993.
- [29] Wen-tsun Wu, "Mathematics Mechanization: Mechanical Geometry Theorem-Proving, Mechanical Geometry Problem-Solving, and Polynomial Equations-Solving," Science Press / Kluwer Academic Publishers, 2000.
- [30] H. Zhouge, "The Knowledge Grid," World Scientific, 2004.