Race against the Teens – Benchmarking Mechanized Math on Pre-university Problems

Takuya Matsuzaki[1,2], Hidenao Iwane[3,2], Munehiro Kobayashi[4], Yiyang Zhan[5], Ryoya Fukasaku[6], Jumma Kudo[6], Hirokazu Anai[3,7], and Noriko H. Arai[2]

[1] Nagoya University, Japan
[2] National Institute of Informatics, Japan
[3] Fujitsu Laboratories, Ltd., Japan
[4] University of Tsukuba, Japan
[5] Université Paris Diderot
[6] Tokyo University of Science, Japan
[7] Kyushu University, Japan

Abstract. This paper introduces a benchmark problem library for mechanized math technologies including computer algebra and automated theorem proving. The library consists of pre-university math problems taken from exercise problem books, university entrance exams, and the International Mathematical Olympiads. It thus includes problems in various areas of pre-university math and with a variety of difficulty. Unlike other existing benchmark libraries, this one contains problems that are formalized so that they are obtainable as the result of mechanical translation of the original problems expressed in natural language. In other words, the library is designed to support the integration of the technologies of mechanized math and natural language processing towards the goal of end-to-end automatic math problem solving. The paper also presents preliminary experimental results of our prototype reasoning component of an end-to-end system on the library. The library is publicly available through the Internet.

1 Introduction

One of the ultimate goals of automated theorem proving is to produce computer programs that allow a machine to conduct mathematical reasoning like human beings. It seems that a tacit understanding exists on how we should interpret this goal. First, the input of the programs is assumed to be expressed in some formal language, but not in a natural language. Second, the term "human beings" is used to mean gifted mathematicians rather than ordinary people. In this paper, we propose a different interpretation of the goal by providing a new problem library for benchmarking automated math reasoners, and showing experimental results on the problem set.

Though traditionally ignored in the framework of automated theorem proving and computer algebra, interpreting given problems is as important as solving them in mathematical reasoning; it took almost a century to determine the

```
(Find (x)
  (exists (A B C D E)
     (& (exists (F)
           (& (exists (U T S R Q P)
                 (& (= (polygon (list-of A B C D E F))
                       (polygon (list-of P Q R S T U)))
                    (is-regular-polygon (polygon (list-of P Q R S T U)))))
              (is-diagonal-of (seg A C) (polygon (list-of A B C D E F)))
              (exists (U T S R Q P)
                 (& (= (polygon (list-of A B C D E F))
                       (polygon (list-of P Q R S T U)))
                    (is-regular-polygon (polygon (list-of P Q R S T U)))))
              (is-diagonal-of (seg C E) (polygon (list-of A B C D E F)))))
        (exists (M N)
           (& (exists (r)
                 (& (= (/ (length-of (seg A M)) (length-of (seg A C)))
                       (/ (length-of (seg C N)) (length-of (seg C E))))
                    (= (/ (length-of (seg C N)) (length-of (seg C E))) r)
                    (on M (seg A C))
                    (= (/ (length-of (seg A M)) (length-of (seg A C)))
                       (/ (length-of (seg C N)) (length-of (seg C E))))
                    (= (/ (length-of (seg C N)) (length-of (seg C E))) r)
                    (on N (seg C E))
                    (= x r)))
              (~ (= M A)) (~ (= M C))
              (~ (= N C)) (~ (= N E))
              (points-colinear (list-of B M N)))))))
```

Fig. 1. Mechanical Translation Result of IMO 1982, Problem 5

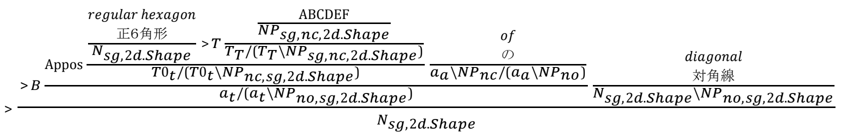
regular hexagon
正6角形
$N_{sg,2d.Shape}$
ABCDEF
$NP_{sg,nc,2d.Shape}$
$>T$
$T_T/(T_T\backslash NP_{sg,nc,2d.Shape})$
Appos
$T0_t/(T0_t\backslash NP_{nc,sg,2d.Shape})$
of
の
$a_a\backslash NP_{nc}/(a_a\backslash NP_{no})$
$>B$
$a_t/(a_t\backslash NP_{no,sg,2d.Shape})$
diagonal
対角線
$N_{sg,2d.Shape}\backslash NP_{no,sg,2d.Shape}$
$>$
$N_{sg,2d.Shape}$

Fig. 2. Syntactic/Semantic Analysis of Problem (CCG Derivation Tree)

language and axioms required to express the Jordan curve theorem, and this is exactly how long it took to solve the problem. This is also the case in curriculum math. Hence, it is fair to assume the problems are expressed in a natural language but not in a formal language if we want to seriously argue about whether or not a machine is as intelligent as high school graduates in math problem solving.

Given this situation, we developed a new problem library of real pre-university math problems. It is designed to cover various sub-areas of curriculum math and a diverse range of difficulty. The initial release of the data set includes more than 700 problems taken from three sources: popular high school math exercise book series, entrance examinations of seven top universities in Japan, and the past problems from International Mathematical Olympiads (IMO). Our choice of the three problem sources is motivated by the desire to measure the performance of mechanized math systems with high school students of different skill and intellectual levels as reference points.

Table 1. Number of Problems and Directives

		Chart	Univ	IMO	total
#Problems		288	245	212	745
#Directives	`Find`	473	438	110	1021
	`Draw`	28	16	0	44
	`Show`	78	73	134	285

Table 2. Subject Areas (IMO)

Algebra	57
Number Theory	38
Analysis	1
Geometry	105
Combinatorics	11

Although problems in the library are formalized in a formal language so that the automatic reasoning (AR) and computer algebra system (CAS) communities will find it appealing to challenge the problems, the formalization is designed so that the problems can be obtained as the result of mechanical translation of their originals. This might have sounded unrealistic in the previous century but is now within the range of contemporary research, thanks to the recent progress that has been made in deep linguistic processing (e.g., [3, 2, 9]). Fig. 1 presents an output from the translation module under development and Fig. 2 depicts a part of the process that derives the logical translation of a Japanese phrase "正6角形/regular-hexagon $ABCDEF$ の/of 対角線/diagonal", that corresponds to "diagonal(s) of the regular hexagon $ABCDEF$." Problems in the library were formalized manually according to the design of the aforementioned translation module. That is, they were translated manually into the formal language by word-by-word and sentence-by-sentence basis without any inference and paraphrasing.

The formalized problem set and the accompanying axioms are publicly available at `http://github.com/torobomath/benchmark`. The problems and the axioms are formulated in a higher-order language that is mostly compatible with the TPTP's typed higher-order format (THF) [13]. The data and the axioms are distributed both in the TPTP's THF syntax and an S-expression format. For readability, we use the S-expression format for presenting the data. Several basic elements such as logical connectives and quantifiers are renamed following the TPTP's convention.

The rest of the paper is structured as follows. We first describe how we collected and formalized curriculum math problems in Section 2. Several problems are shown in Section 3 to exemplify what aspects of mechanized math are necessary. Besides proof problems, the benchmark includes many "Find X"-type problems. Technical issues in formalizing such problems are discussed in Section 4. Section 5 provides an overview of a prototype solver system built on an integration of a simple logical inference system with computer algebra systems. Experimental results on the initial release of the data by our solver system are presented in Section 6. Finally, we conclude the paper and discuss future directions in Section 7.

Table 3. Subject Areas (Chart & Univ)

	Chart	Univ
Algebra	51	10
Linear Algebra	28	62
Geometry	136	65
Pre-Calculus	15	74
Calculus	42	33
Combinatorics	16	1

Table 4. Distribution of Theory Labels

	Chart	Univ	IMO
PA	84	0	42
RCF	174	245	115
ZF	30	0	55
RCF+PA	1	0	8
Transc	23	0	6
PA+Transc	5	0	1
Comb	0	0	10
other	1	0	30

Table 5. Statistics on the Syntactic Properties (min/avg/max/**median**)

	Todai Robot Project Math Benchmark				TPTP-THF
	Chart	**Univ**	**IMO**	**All**	
# of formulae	1/ 2/ 7/ **2**	1/ 2/ 8/ **2**	1/ 1/ 5/ **1**	1/ 2/ 8/ **1**	1/103/ 5639/**10**
# of atoms	14/80/485/**65**	14/125/652/**95**	11/85/2658/**65**	11/97/2658/**72**	1/819/64867/**88**
Avg atoms/formula	9/42/161/**38**	14/ 58/232/**54**	10/77/2658/**56**	9/57/2658/**48**	1/ 22/ 811/ **6**
# of symbols	3/16/ 31/**16**	6/ 19/ 34/**19**	4/19/1332/**12**	3/18/1332/**15**	1/ 45/ 1442/ **9**
# of variables	1/12/ 55/ **9**	1/ 17/ 72/**13**	0/ 9/ 35/ **8**	0/13/ 72/ **9**	0/154/11290/**19**
λ	0/ 4/ 22/ **3**	0/ 4/ 23/ **3**	0/ 1/ 9/ **1**	0/ 3/ 23/ **2**	0/ 22/ 385/ **2**
$\forall$	0/ 2/ 49/ **0**	0/ 2/ 24/ **0**	0/ 5/ 22/ **4**	0/ 3/ 49/ **0**	0/123/10753/ **9**
$\exists$	0/ 5/ 38/ **4**	0/ 9/ 50/ **6**	0/ 2/ 20/ **1**	0/ 6/ 50/ **4**	0/ 8/ 496/ **2**
# of connectives	11/67/416/**55**	13/105/476/**78**	11/77/2655/**58**	11/82/2655/**61**	0/574/51044/**52**
Max formula depth	8/20/ 50/**19**	12/ 25/ 59/**23**	9/28/1327/**20**	8/24/1327/**21**	2/ 36/ 359/**11**
Avg formula depth	0/ 5/ 9/ **4**	0/ 5/ 9/ **4**	0/ 5/ 9/ **5**	0/ 5/ 9/ **5**	0/ 5/ 9/ **6**

2 Pre-university Math Problems as a Benchmark for Mechanized Math Systems

In this section, we first describe the sources and the types of the benchmark problems. We then explain how we encoded the problems other than proof problems. Finally, the representation language is described.

2.1 The Problem Library

The initial release of the data set consists of 745 problem files containing 1,353 directives, and 1,897 axioms defining 1,040 symbols (functions, predicates, and constants). The problems were taken from three sources: "Chart-shiki" (**Chart**), Japanese university entrance exams (**Univ**), and International Mathematical Olympiads (**IMO**).

"Chart-shiki" is a popular problem book series containing more than ten thousand problems in total. In the first release of the data set, the **Chart** division consists of arithmetic problems and various types of geometry problems (including those involving calculus and linear algebra) (Table 3). Every problem in "Chart-shiki" is marked with one to five stars by the editors of the book series according to its difficulty. We sampled the problems so that their levels of difficulty would be uniformly distributed.

The **Univ** division of the data consists of the past entrance exams of seven top Japanese national universities. Unlike in most countries, in Japan each na-

IMO 1982, Problem 5

The diagonals AC and CE of the regular hexagon $ABCDEF$ are divided by the inner points M and N, respectively, so that $\frac{AM}{AC} = \frac{CN}{CE} = r$. Determine r if B, M, and N are collinear.

```
;;-------------------------------------------------------------
(def-directive problem_IMO_1982_2
  (Find (r)
    (exists (A B C D E F M N)
        (& (is-regular-polygon (polygon (list-of A B C D E F)))
           (on M (seg A C))
           (on N (seg C E))
           (~ (= M A)) (~ (= M C))
           (~ (= N C)) (~ (= N E))
           (= (/ (length-of (seg A M)) (length-of (seg A C)))
              (/ (length-of (seg C N)) (length-of (seg C E))))
           (= (/ (length-of (seg A M)) (length-of (seg A C)))
              r)
           (colinear B M N)))))

(def-answer problem_IMO_1982_2
   (lambda r (= r (/ 1 3))))
;;-------------------------------------------------------------
```

Fig. 3. Problem File Example (IMO 1982, Problem 5)

tional university prepares its entrance exam by itself. As a result, several hundreds of brand-new problems are produced every year for the entrance exams. In the first release, the **Univ** division includes the problems that were manually classified as 'most likely expressible' in the first-order theory of real-closed fields (RCF) (Table 3). Two hundred more **Univ** problems involving transcendental functions and integers arithmetic (often as a mixture with reals) are currently under preparation for the second release of the data set.

The **IMO** division consists of about 2/3 of the past IMO problems. The initial release includes all of the geometry and real algebra problems, and some of the problems in number theory, function equations, and combinatorics.

Each problem is labeled by its subject domain name such as geometry or calculus (Table 2 and Table 3), and also by its formal theory name. The problems that are naturally expressible (by humans) in the theories of RCF or Peano Arithmetic (PA) are labeled so, and the rest of the problems are tentatively labeled ZF, standing for Zermelo-Fraenkel Set Theory. We scrutinized the problems labeled ZF and classified them into several groups such as 'RCF+PA' (mixture of integer and real arithmetics) and 'Transc' (problems involving transcendental functions that cannot be reformulated in RCF), though they are not formal theories (Table 4). Table 5 lists statistics for the formalized problems. For reference, it also lists those for the typed-higher order format (THF) problems in TPTP version 6.1.0.

Table 6. Logical Translations in ZF Set Theory and Peano Arithmetic

a) There are infinitely many prime numbers greater than 4.
ZF: $\|\{n \in \mathbb{N} \mid \mathrm{prime}(n) \wedge n > 4\}\| = \omega$ PA: $\forall N \exists n (n > N \wedge \mathrm{prime}(n) \wedge n > 4)$
b) There is an even number of prime numbers less than 4.
ZF: $\exists k \in \mathbb{N}(\mathrm{even}(k) \wedge \|\{n \mid \mathrm{prime}(n) \wedge n < 4\}\| = k)$ PA: $\exists k(\mathrm{even}(k) \wedge \mathrm{num\_of}(\mathrm{prime\_less\_than}(4)) = k)$
c) There are two prime numbers less than 4.
ZF: $\|\{n \in \mathbb{N} \mid \mathrm{prime}(n) \wedge n < 4\}\| = 2$ PA$_1$: $\exists n_1 \exists n_2 \begin{pmatrix} \mathrm{prime}(n_1) \wedge \mathrm{prime}(n_2) \wedge n_1 < 4 \wedge n_2 < 4 \wedge n_1 \neq n_2 \\ \wedge \forall m((\mathrm{prime}(m) \wedge m < 4) \rightarrow (m = n_1 \vee m = n_2)) \end{pmatrix}$ PA$_2$: $\mathrm{num\_of}(\mathrm{prime\_less\_than}(4)) = 2$

2.2 A Formalization of Curriculum Math Problems

We formalize a problem as a pair of a *directive* and its *answer*. By surveying the problems, we identified three major types of directives:

- `Show` $[\phi]$ is a proof problem to prove ϕ.
- `Find`(v) $[\phi(v)]$ is a problem to find all values for v that satisfy condition $\phi(v)$.
- `Draw`(v) $[\phi(v)]$ requests a geometric object v defined by $\phi(v)$ be drawn.

`Show` directives must be familiar to the reader, though the set of problems requiring proofs is a minority in curriculum math. Table 1 shows that students are asked to find some values more frequently than to prove propositions. The answer to a `Find` problem, `Find`$(v)[\phi(v)]$, is expected to be a characteristic function $f(v)$ that returns `true` if v satisfies $\phi(v)$. The answer to a `Draw` problem `Draw`$(v)[\phi(v)]$ should be the geometric object v expressed as a characteristic function on R^2. Fig. 3 is an example of a `Find` problem taken from IMO 1982.

To use a problem set in the above format as benchmark data, we need a *rule* to judge whether a system's output is acceptable or not. It is clear for the `Show` directives: true or false. We regard a `Draw` directive as a variant of `Find` problems for which the system is supposed to find a formula that defines the geometric object. Then, what is "to solve a `Find` problem?" Roughly speaking, a solver is supposed to give a *correct* solution in its *simplest* form. We will discuss the properties an answer formula for a `Find` problem has to satisfy in Section 4.

2.3 Representation Language

We formalized all the problems in a single theory on the basis of ZF regardless of their context. In formality, it is a typed lambda calculus with parametric polymorphism. This is again due to the fully automatic, end-to-end task setting. Table 6 demonstrates why a higher-order language is appropriate as the target language. Mechanical translation assumes a systematic correspondence between the syntactic structures of the input and output languages; the results of the

```
;; tangent(S1, S2, P) <-> geometric objects
;; S1 and S2 are tangent at point P
(def-pred
  tangent :: Shape -> Shape -> Point => Bool)

(axiom
  def_tangent_line_and_circle
  (p q c r P)
  (<-> (tangent (line p q) (circle c r) P)
       (& (on P (line p q))
          (perpendicular (line c P) (line p q))
          (= (distance^2 P c) (^ r 2)))))
```

```
;; maximum(S, m) <->
;; m is the maximum element of set S
(def-pred
  maximum :: (SetOf R) -> R => Bool)

(axiom
  def_maximum
  (set max)
  (<-> (maximum set max)
       (& (elem max set)
          (forall (v)
              (-> (elem v set)
                  (<= v max))))))
```

Fig. 4. Type Definitions and Axioms

mechanical translations of a), b), and c) into ZF in Table 6 are expected to have the same or at least a similar structure thanks to the set builder notation such as $\{n \in \mathbb{N} \mid \text{prime}(n) \wedge n > 4\}$, which is expressed using λ-abstraction in the dataset. However, the expressions of the three sentences in PA must be different since the concept of finiteness cannot be expressed in first-order logic. Meanwhile, the expressibility of ZF allows almost word-by-word translations for all sentences. Parametric polymorphism is utilized to have polymorphic lists and sets in the language and define various operations on them while keeping the axioms and the lexicon (i.e., the mapping table from words to their semantic representations) concise.

We believe the vast majority of our benchmark problems can be eventually expressed in first-order logic. To mechanically fill the gap between the heavy-duty language and the relatively simple content is however a mandatory step to connect natural language processing and automated reasoning together for end-to-end automatic problem solving.

Since our mechanical translator is still under development, the problems were formalized manually at the current stage. Operators, all majored in computer science and/or mathematics, were trained to translate the problems as faithfully as possible to the original natural language statements following the NLP design. The sets of new symbols and their defining axioms were introduced in parallel with the problem formalization, to match the problem formula as close as possible to the problem text.

In the language, we currently have 31 types including `Bool`(ean), `Z` (integers), `Q` (rational numbers), `R`(eals), `C`(omplex numbers), `ListOf`(α) (polymorphic lists), `SetOf`(α) (polymorphic sets), `Point` (in 2D and 3D spaces), `Shape` (sets of `Point`s), `Equation` (in real domain), and so on. The types are somewhat redundant in that we can represent, e.g., `Equation` simply by a function of type $\mathtt{R} \to \mathtt{R}$ by regarding $f : \mathtt{R} \to \mathtt{R}$ as representing $f(x) = 0$. The abundance of types, however, helped a lot in organizing the axioms and debugging the formalized problems. Fig. 4 presents an excerpt from an axiom file that includes two type definitions (two `def-pred`s) and two axioms.

All in all, the language shall be understood as a conservative extension of ZF set theory. It thus has some overlap with previous efforts toward formalizing a large part math, such as Mizar's math library [5]. However, some essential parts of the system (e.g., the definition of the real numbers and arithmetic) are left undefined although nothing prevents the users of the problem library from doing so. Instead of writing all the inference rules explicitly, we delegated computer algebra systems to take care of it. Although it is not within our current research focus, full formalization of the system (maybe by embedding it into an existing formalized math library) is an interesting future direction.

2.4 Related Work

Development of a well-designed benchmark is doubtlessly a crucial part of AR. The most notable example is the "Thousands of Problems for Theorem Provers" (TPTP, [12]), which covers various domains and several problem formats including CNF, first-order formula with quantifiers, and typed higher-order logic. Previous efforts have also accumulated benchmarks for various branches of AR, such as SAT [6], satisfiability modulo theory [1], inductive theorem proving [4], and geometry problems [11]. However, the current study is the first attempt to offer a large collection of curriculum math problems including not only proof problems but also `Find` and `Draw` problems with a wide range of difficulties as a benchmark for AR technologies.

3 Problem Samplers

We provide several sample problems taken from the first release of the library.

Hokkaido University, 2011, Science Course, Problem 3 (2)

Let ℓ be the trajectory of $(t+2, t+2, t)$ for t ranging over the real numbers. O$(0,0,0)$, A$(2,1,0)$, and B$(1,2,0)$ are on a sphere S, centered at C(a,b,c). Determine the condition on a, b, c for which S intersects with ℓ.

In the data set, the above problem is formalized as shown in Fig. 5.

It is not difficult to obtain an equivalent formula in the language of first-order RCF by rewriting the predicates and functions using their defining axioms. However, it results in a formula including 22 variables and 22 atoms, that is way above the ability of existing RCF-QE solvers to deal with. It is not very surprising seeing that the time complexity of RCF-QE, a key step in the solution process, is doubly exponential in the number of variables in a given formula. We enhanced existing RCF-QE algorithms to overcome the difficulty. Fortunately, our prototype system successfully solved this problem. We will explain the enhancement in detail in Section 5.

Chart-shiki, Math 3+C, Problem 09CBCE011

Consider $0 < \frac{|ax-y|}{\sqrt{1+a^2}} < \frac{2\sqrt{2}}{x+y}$ for $x > 0$ and $y > 0$. Prove that there are only finitely many pairs of positive integers (x, y) that satisfy the above inequalities when a is a rational number.

```
;; FILE: Univ-Hokkaido-2011-Ri-3.lsp
(def-directive
  hokudai_2011_Ri_3_2
   (Find (abc)
     (exists (a b c O A B C l S)
       (& (= abc (list-of a b c))
          (line-type l)
          (= l (shape-of-cpfun (lambda p (exists (t) (= p (point (+ t 2) (+ t 2) t))))))
          (sphere-type S)
          (= O (point 0 0 0)) (= A (point 2 1 0)) (= B (point 1 2 0))
          (on O S) (on A S) (on B S)
          (= C (point a b c))
          (= C (center-of S))
          (intersect l S)))))))
```

Fig. 5. Hokkaido University, 2011, Science Course, Problem 3 (2)

In the data set, the above problem is formalized as follows:

$$\forall a \in \mathbb{Q} \exists n \in \mathbb{Z}(n > 0 \land n = |\{(x,y) \in \mathbb{Z}^2 \mid P(\texttt{int2real}(x), \texttt{int2real}(y))\}|)$$

where $P = \lambda(x,y) \in \mathbb{R}^2(x > 0 \land y > 0 \land 0 < \frac{|ax-y|}{\sqrt{1+a^2}} < \frac{2\sqrt{2}}{x+y})$. Several $\in$'s preceding to domain names in the formula signify their types. Despite the seeming mixture of reals, integers, and rational numbers, we can easily find an equivalent formula in the language of PA. The mechanization of processes such as this is one of our ongoing research topics.

IMO 2012, Problem 2

Let $n \geq 3$ be an integer, and let $a_2, a_3, \ldots, a_n$ be positive real numbers such that $a_2 a_3 \cdots a_n = 1$. Prove that $(1+a_2)^2(1+a_3)^3 \ldots (1+a_n)^n > n^n$.

In the data set, this problem is formalized using a higher-order function `prod_from_to :: (Z → R) → Z → Z → R`, which corresponds to $\Pi_{\text{from}}^{\text{to}}$ in the common notation. This problem apparently requires some kind of inductive reasoning but the domain includes both real numbers and integers. Problems of this type are abundant in curriculum math. We believe they will prove to be new and interesting and challenging problems for automated inductive reasoning, both theoretically (e.g., formalizing them in a suitable local theory other than ZF) and practically.

IMO 2003, Problem 1

S is the set $\{1, 2, 3, \ldots, 1000000\}$. Show that for any subset A of S with 101 elements we can find 100 distinct elements x_i of S, such that the sets $\{a + x_i \mid a \in A\}$ are all pairwise disjoint.

It is straightforward to translate the above-mentioned problem in ZF:

$$\forall A \begin{pmatrix} A \subset S \land |A| = 101 \\ \quad \to \exists X\,(X \subset S \land |X| = 100 \land \text{pairwise\_disjoint}(\ \{\{a+x \mid a \in A\} \mid x \in X\})) \end{pmatrix}$$

Table 7. Preference Hierarchy on Answer Form

Directive type	Syntactic condition on the answer formula
$\texttt{Find}(v : \texttt{R})[\phi(v, \boldsymbol{p})]$	1. $\bigvee_i \left(\bigwedge_j (v\ \rho_{ij}\ \alpha_{ij}) \wedge \psi_i(\boldsymbol{p})\right)$
	2. $\bigvee_i \left(\bigwedge_j f_{ij}(v)\ \rho_{ij}\ 0 \wedge \psi_i(\boldsymbol{p})\right)$
	3. $\bigvee_i \left(\exists(n : \texttt{Z}). \left(\bigwedge_j (v\ \rho_{ij}\ \alpha_{ij}(n)) \wedge (n\ \rho_i\ \gamma_i)\right) \wedge \psi_i(\boldsymbol{p})\right)$
	4. $\bigvee_i \left(\exists(r : \texttt{R}). \left(\bigwedge_j (v\ \rho_{ij}\ \alpha_{ij}(r)) \wedge (r\ \rho_i\ \gamma_i)\right) \wedge \psi_i(\boldsymbol{p})\right)$
$\texttt{Find}(v : \texttt{Z})[\phi(v, \boldsymbol{p})]$	1. $\bigvee_i \left(\bigwedge_j (v\ \rho_{ij}\ \alpha_{ij}) \wedge \psi_i(\boldsymbol{p})\right)$
	2. $\bigvee_i (\exists(n : \texttt{Z}). (v = \alpha_i(n) \wedge (n\ \rho_i\ \gamma_i)) \wedge \psi_i(\boldsymbol{p}))$
$\texttt{Find}(v : \texttt{SetOf(Point)})[\phi(v, \boldsymbol{p})]$	$\bigvee_i (v = \{(x, y) \mid \xi_i(x, y)\} \wedge \psi_i(\boldsymbol{p}))$

($\rho_* \in \{=, <, \leq, \geq, >\}$; α_*, $\alpha_*(\cdot)$, γ_*, $\xi_*(\cdot, \cdot)$: first-order terms not including v, x, y; $f_{ij}(v)$: first-order term; $\psi_i(\boldsymbol{p})$: quantifier-free first-order formula)

where $S = \{n \in \mathbb{N} \mid 1 \leq n \leq 1000000\}$. Moreover, it can be expressed in PA, too. However, the effort to reformulate it in PA does little help in solving it.

4 What Constitutes an Answer to a Find Problem?

In Subsection 2.2, the properties an answer formula for a `Find` problem has to satisfy for it to be regarded as acceptable (correctness and simplicity) were briefly discussed. Now we will discuss these in detail. In [14], Sutcliffe et al. proposed the conditions which answers of answer-extraction problems have to satisfy. Our definition of 'answer' encompasses theirs in spirit and covers more complicated cases beyond the extraction of a finite number of answers.

The definition of the *correctness* of an answer is straightforward. Given a problem $\texttt{Find}(x)[\psi(x, \boldsymbol{p})]$, where $\boldsymbol{p}$ stands for zero or more free parameters, an answer formula $\phi(x, \boldsymbol{p})$ must satisfy:

$$\forall x \forall \boldsymbol{p} (\psi(x, \boldsymbol{p}) \leftrightarrow \phi(x, \boldsymbol{p})). \tag{1}$$

An example of a correct answer formula $\phi'(x, \boldsymbol{p})$ is provided for each `Find` problem in the library. If $\phi'(x, \boldsymbol{p})$ is used instead of $\psi(x, \boldsymbol{p})$, the proof task for (1) should generally be easy.

The *simplicity* of an answer is harder to define. Suppose that you are given a problem, $\texttt{Find}(v : \mathbb{R})[v^2 = a]$, in a math test. Then, $\lambda v.(v^2 = a)$ is of course not an acceptable answer. However, test-takers are expected to answer, for example,

$$\lambda v. \left((a \geq 0 \wedge v = a^{1/2}) \vee (a \geq 0 \wedge v = -a^{1/2})\right).$$

An answer to a problem asking to find all real numbers v satisfying a formula $\phi(v)$ in the first-order language of RCF is called *simple* when it is in the form $\lambda v.\psi(v)$ satisfying the following conditions.

- $\psi(v)$ is a quantifier-free formula in disjunctive normal form, and
- each dual clause in $\psi(v)$ consists of atoms of the form of $v\ \rho\ \alpha$ or $\beta\ \rho\ 0$, where $\rho \in \{=, <, >, \leq, \geq\}$, and α and β are first-order terms not including v and comprises numbers, variables (i.e., parameters) and functions in $\{+,\ -,\ \cdot,\ /,\ \hat{}\,(\text{power})\}$.

The aforementioned syntactic conditions for a problem classified in RCF should be acceptable because RCF allows quantifier elimination [15]. Furthermore, the statistics tell us that almost all pre-university math problems have explicit solutions (i.e., in the form of $x = \alpha, \beta > x > \gamma$, etc.)

For problems other than those expressible in RCF, we tried our best to capture a loose, common understanding in the form of acceptable answers by examining the model answers (for humans) to the benchmark problems. Our tentative definition of 'simple answers' is as follows:

- Simplicity of the sub-language: an answer formula should be in a language consisting of Boolean connectives, equality and inequalities, numbers, variables, and the four arithmetic operations and power calculations, sin, cos, tan, exp, log, 'type coercion functions' such as `int_to_real`, and *a minimal use of lambda abstractions and quantifications.*
- Explicitness: whenever possible within the above restriction imposed on the language, the answer to a problem of the form $\mathtt{Find}(x)[\phi(x)]$ should be given using atoms such as $x = \alpha$ and $x > \alpha$, where α does not include x.

Note that we need quantification in general unless the problem is expressible in a theory that allows quantifier elimination. For instance, in the sub-language defined above, there is no way to express the answer to "Determine all positive numbers v that are divisible by three and also by two," other than, e.g., $\exists k(v = 6k \wedge k > 0)$. As for "*minimal use of* $\lambda, \forall, \exists$", we define the preference of answer form tentatively (Table 7). The answer-check routine compares a solver's answer and the model answer in the data set, and checks whether the solver's answer ranks equal (or higher) in the hierarchy.

5 Prototype Solver

While developing the benchmark data set, we also developed a prototype math problem solver system (overviewed in Fig. 6). Given a formalized problem, the system first rewrites it iteratively using the axioms and several equivalence-preserving transformation rules such as beta-reduction, extensional equality between functions ($\lambda x.M = \lambda x.N \Leftrightarrow \forall x(M = N)$), variable elimination by substitution ($\forall x(x = f \rightarrow \phi(x)) \Leftrightarrow \phi(f)$, and ($\exists x(x = f \wedge \phi(x)) \Leftrightarrow \phi(f)$ where x does not occur free in f). In the course of the rewriting process, several types of terms, such as multiplication and division of polynomials and integration, are evaluated (simplified) by CASs such as Mathematica 9.0 and Maple 18. Once the input is rewritten to a formula in the language of RCF, quantifier-elimination (QE) algorithms are invoked; we utilized the RCF-QE algorithm implemented in SyNRAC [8]. When QE is proceeded successfully, the remaining tasks, solving equations and inequalities in many cases, will be taken care by the CASs. When the input is rewritten in the language of PA, we apply the `Reduce` command of Mathematica.

As mentioned in Section 3, the first-order formulas generated by mechanical translation are much larger than expected [7, 10]. We enhanced the RCF-QE

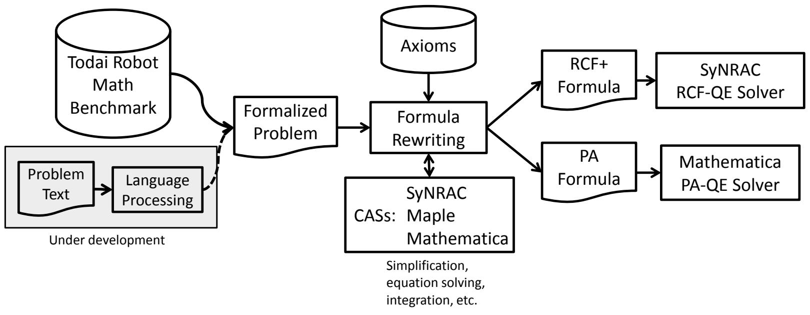

Fig. 6. System Overview

Table 8. Overall Results

		Succeeded		Failed		
		Success %	Time (sec) Min/Med/Avg/Max	Timeout	Wrong	Other
Chart	RCF	63.8% (111/174)	13/18.0/ 37.4/ 343	10.9%	1.7%	23.6%
	PA	57.1% (48/ 84)	12/17.0/ 20.3/ 172	0.0%	0.0%	42.9%
	Other	10.0% (3/ 30)	13/14.0/ 17.7/ 26	0.0%	0.0%	90.0%
	All	56.3% (162/288)	12/17.0/ 32.0/ 343	6.6%	1.0%	36.1%
Univ	All (RCF only)	58.0% (142/245)	12/26.5/ 85.5/1417	15.5%	2.9%	23.7%
IMO	RCF	16.5% (19/115)	14/25.0/ 51.8/ 197	29.6%	0.9%	53.0%
	PA	4.8% (2/ 42)	25/29.5/ 29.5/ 34	16.7%	0.0%	78.6%
	Other	3.6% (2/ 55)	17/24.5/ 24.5/ 32	12.7%	0.0%	83.6%
	All	10.8% (23/212)	14/25.0/ 47.5/ 197	22.6%	0.5%	66.0%

algorithms by numerous techniques to handle them: choice of the computation order of sub-formulas, specialized QE algorithms for restricted input formulas, simplification of the intermediate formulas by utilizing the interim results, and so on. Additionally, we developed an algorithm for computing *the area enclosed by a set of curves* and an extended RCF-QE command to reduce some of the problems involving trigonometric functions to RCF-QE problems.

6 Experiments

The prototype system was run on the benchmark problems with a time limit of 3600s per problem (including the time spent on checking the correctness of the answers). Table 8 shows the number of successfully solved problems, minimum, median, average, and maximum (wallclock) time spent on solved problems, number of failures due to timeout, wrong answers (disproofs for `Show` or wrong answers for `Find` or `Draw` directives), and those not solved due to various reasons (the column headed 'Other'). Approximately one-third of the 'Other' cases were

Table 9. Breakdown of Results on **Chart** RCF Problem by Number of Stars

# of Stars	Succeeded		Failed		
	Success %	Time (sec) Min/Med/Avg/Max	Timeout	Wrong	Other
1	82.4% (28/34)	13/17.0/20.4/ 65	2.9%	0.0%	14.7%
2	79.4% (27/34)	16/18.0/28.1/230	2.9%	2.9%	14.7%
3	57.6% (19/33)	15/17.0/36.1/341	6.1%	0.0%	36.4%
4	47.4% (18/38)	15/19.0/62.1/343	23.7%	2.6%	26.3%
5	54.3% (19/35)	16/28.0/53.6/279	17.1%	2.9%	25.7%

Table 10. Breakdown of Results on **Univ** RCF Problems by University

University	# of All Problems	RCF Problems %	Overall Success %	Success % on RCF Problems
Hokkaido	72	44.4% (32/ 72)	25.0% (18/ 72)	56.3% (18/32)
Tohoku	80	52.5% (42/ 80)	30.0% (24/ 80)	57.1% (24/42)
Tokyo	160	38.8% (62/160)	18.8% (30/160)	48.4% (30/62)
Nagoya	72	41.7% (30/ 72)	20.8% (15/ 72)	50.0% (15/30)
Osaka	64	37.5% (24/ 64)	32.8% (21/ 64)	87.5% (21/24)
Kyoto	88	43.2% (38/ 88)	33.0% (29/ 88)	76.3% (29/38)
Kyushu	96	36.5% (35/ 96)	18.8% (18/ 96)	51.4% (18/35)

due to a failure in the problem reformulation phase; i.e., for those problems, the system could not find an equivalent formula expressible in either RCF or PA. Explicitly wrong answers were due to bugs in our formula rewriting system module and/or malfunctions of Maple's equation/inequality solving command.

Overall, the performances for the **Chart**, **Univ**, and **IMO** divisions seem to well reflect the inherent differences in their difficulty levels. Table 9, Table 10, and Table 11 show further analysis of the results obtained for the three divisions. Table 9 lists the performance figures for the RCF problem subsets in the **Chart** division that are rated level 1 to 5 in the exercise books. We see a clear difference between those rated level 1 or 2, and 4 or 5, especially in the percentages of the problems that had a timeout. Table 10 lists the performance figures for each university from which the exam problems were taken. Although average scores etc. of the entrance exams are not published, a statistical analysis undertaken by major prep schools tells us that the average score of successful applicants to the top universities is around 30-60% depending on schools and departments. Hence, it is very plausible that a machine will come to have the ability to pass the entrance math exams of top universities if it is able to cover areas other than RCF.

Finally, Table 11 lists the results on **IMO** problems taken from different time periods. Human and Machine Efficiency in the table shows the ratio between the attained points (by all contestants in a year and by our system, respectively)

Table 11. Results for **IMO** Problems by Decade

Years	Human Efficency	Machine Efficiency	Succeeded	Failed		
				Timeout	Wrong	Other
1959-69	58.23%	21.11%	26.3% (15/57)	22.8%	1.8%	49.1%
1970-79	46.57%	7.00%	13.3% (4/30)	26.7%	0.0%	60.0%
1980-89	44.35%	1.85%	3.1% (1/32)	31.2%	0.0%	65.6%
1990-99	38.27%	3.33%	5.7% (2/35)	11.4%	0.0%	82.9%
2000-13	34.31%	1.19%	1.9% (1/54)	22.2%	0.0%	75.9%

and all possible points[8]. It seems that the IMO problems are getting harder year by year not only for human participants but more so for our system.

We believe that these experimental results support our decision on the library organization, and encourage us to further proceed toward the goal of end-to-end math problem solving with the monolithic logical language based on ZF.

7 Conclusion and Prospects

In this paper, we introduced a benchmark problem library for mechanized math technologies. The library consists of curriculum math problems taken from exercise problem books, university entrance exams, and International Mathematical Olympiads. Unlike other existing benchmark libraries, this one contains problems that are formalized so that they are obtainable as the result of mechanical translation of the original problems expressed in natural language. Preliminary experimental results we obtained for our prototype system on the benchmark show that its performance is comparable to that of candidates for admission to top universities, at least for problems in real-closed fields.

Our future plan includes the expansion of the library with more problems on integer arithmetic, transcendental functions, combinatorics, and a mixture of real and integer arithmetics as well as development of the natural language processing module for an end-to-end system.

References

1. Barrett, C., Stump, A., Tinelli, C.: The Satisfiability Modulo Theories Library (SMT-LIB). `www.SMT-LIB.org` (2010)
2. Bos, J.: Wide-coverage semantic analysis with boxer. In: Bos, J., Delmonte, R. (eds.) Semantics in Text Processing. STEP 2008 Conference Proceedings. pp. 277–286. Research in Computational Semantics, College Publications (2008)
3. Clark, S., Curran, J.R.: Wide-coverage efficient statistical parsing with CCG and log-linear models. Computational Linguistics 33 (2007)

[8] The statistics were taken from the official IMO website: `https://www.imo-official.org/results_year.aspx`

4. Dennis, L.A., Gow, J., Schürmann, C.: Challenge problems for inductive theorem provers v1.0. Tech. Rep. ULCS-07-004, University of Liverpool, Department of Computer Science (2007)
5. Grabowski, A., Korniłowicz, A., , Naumowicz, A.: Mizar in a nutshell. Journal of Formalized Reasoning 3(2), 153245 (2010)
6. Hoos, H.H., Stützle, T.: SATLIB: An online resource for research on SAT. pp. 283–292. IOS Press (2000)
7. Iwane, H., Matsuzaki, T., Arai, N., Anai, H.: Automated natural language geometry math problem solving by real quantier elimination. In: Proceedings of the 10th International Workshop on Automated Deduction (ADG2014). pp. 75–84 (2014)
8. Iwane, H., Yanami, H., Anai, H., Yokoyama, K.: An effective implementation of symbolic-numeric cylindrical algebraic decomposition for quantifier elimination. Theor. Comput. Sci. 479, 43–69 (2013)
9. Kwiatkowksi, T., Zettlemoyer, L., Goldwater, S., Steedman, M.: Inducing probabilistic CCG grammars from logical form with higher-order unification. In: Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing. pp. 1223–1233. Association for Computational Linguistics (2010)
10. Matsuzaki, T., Iwane, H., Anai, H., Arai, N.H.: The most uncreative examinee: A first step toward wide coverage natural language math problem solving. In: Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence. pp. 1098–1104 (2014)
11. Quaresma, P.: Thousands of geometric problems for geometric theorem provers (TGTP). In: Schreck, P., Narboux, J., Richter-Gebert, J. (eds.) Automated Deduction in Geometry, Lecture Notes in Computer Science, vol. 6877, pp. 169–181. Springer Berlin Heidelberg (2011)
12. Sutcliffe, G.: The TPTP Problem Library and Associated Infrastructure: The FOF and CNF Parts, v3.5.0. Journal of Automated Reasoning 43(4), 337–362 (2009)
13. Sutcliffe, G., Benzmüller, C.: Automated reasoning in higher-order logic using the TPTP THF infrastructure. Journal of Formalized Reasoning 3(1), 1–27 (2010)
14. Sutcliffe, G., Stickel, M., Schulz, S., Urban, J.: Answer extraction for TPTP `http://www.cs.miami.edu/~tptp/TPTP/Proposals/AnswerExtraction.html`
15. Tarski, A.: A Decision Method for Elementary Algebra and Geometry. University of California Press, Berkeley (1951)