計算機が大学入試数学問題を解く

A computer program that solves pre-university mathematical problems

松崎　拓也　(名古屋大学)

概　　要

In a joint work with many people, we have developed a computer system that solves pre-university level math problems written in natural language. The system is comprised of two parts. One is a language processing pipeline, which translates a math problem into a logical formula. The other is a computer algebra system that derives an answer from the translated problem. In the talk, I will mainly talk about the former part. The main obstacle in the translation from a natural language into a logical language is the flexibility of the natural language, which enables us to convey complex meaning in a concise expression but makes the sentences highly ambiguous for a machine. I will explain how we combat with it using both logical and statistical means.

1. はじめに

筆者らは2012年から日本語で書かれた入試数学問題を自動的に解く計算機プログラムの開発を続けている．これは現在の人工知能技術を総合することで大学入試問題に挑戦するという「ロボットは東大に入れるか」プロジェクトの一環として行われたものであるが，数学問題を自動的に解く試みは20世紀なかばの人工知能研究初期から続く古い課題であり，さらに，機械的な検証や演繹が可能となるように数学的課題を形式化するという問題意識自体は少なくとも19世紀末の数理論理学の創始までさかのぼれる．しかし，自然言語で書かれた問題を入力とし，問題テキストを自動的に分析することで得た問題内容の記述から自動演繹によって答えを導く，といういわゆる end-to-end 型の問題設定で数学問題の自動解答に取り組んだ例は極めて少ない．

この問題設定における主な技術的課題は２つある．ひとつめは，人間向けに書かれた問題テキストから，機械的な演繹のための入力となりうるレベルの精緻かつ正確な問題内容の記述を自動的に得ることである．これは簡単に言えば日本語で書かれた数学問題を論理式による記述へと正確に自動翻訳するという課題である．ふたつめは，そのように機械的に導出された，一般に非常に冗長かつ長大なものとなる論理式を入力として，計算コストの大きな自動演繹アルゴリズムを実行しなければならない点にある．これら２つの課題は自動的な問題解答を実現するためにはいずれも解決しなければならないものだが，本稿では主に前者の課題に関して述べる．

数学問題のように明確な内容をもつテキストの自動解析は，論理をモデルとした自然言語の研究において最も基本的な目標と考えられる．1980年代の第２次人工知能ブームの時代に，ひとつの目標としてそのようなことを考えた言語処理研究者は存在したものと想像されるが，実際に数学問題などのテキストからその意味内容を自動解析によって得たという報告は筆者の知る限り存在しない．このような基本的課題についての取り組みが極めて少ないことは意外なことである．おそらくその理由は，現実のテ

キストを対象とする場合，様々な言語現象を一度に扱う必要があるためではないかと思われる．テキストジャンルごとに現れる言語現象に偏りはあるものの，機械の目線で実際のテキストを眺めると，ごく短い文章であっても実に様々な言語手段が組み合わさって成り立っていることが分かる．このため，個別的にみたときに現象として「興味深い」かどうかとは無関係に，さまざまな現象をある程度広く解析できるようになるまでは，たとえ基本的な内容を持つテキストであれその解析は一歩も進まない．

このような対象を解析する基礎となるのは，種々の言語現象の組合せとしてテキストを解析するために，単語や文型ごとにその形と意味との対応を腑分けして整理した文法および辞書である．通常のソフトウェアの（あるいは科学理論の）開発と同様に，文法および辞書は内省だけではなく実験すなわち現実のテキストを実際に解析して結果を観察することを繰り返すことで徐々に構築されていく．この基本的な枠組みは1980年代から特に大きく変化したわけではないが，当時のマシン性能では一文の解析時間が数十分というオーダーであった．現実のテキストを解析できるレベルの文法・辞書の開発には，再帰テスト的に多数の文の解析を繰り返す必要があり，そのようなことは80年代には端的に不可能だったと思われる．

自動解答システムの開発，特にその言語処理部分の開発はまだ途上であるが，多少なりと現実のテキストの解析が可能になってきた背景にはマシン性能の向上に加え，論理的手法と統計的手法を合わせた言語分析手法および解析アルゴリズムの存在がある．本稿では，まず自動解答システムの概要について述べた後，システムの言語処理部分における論理的手法と統計的手法の関係について説明する．

2. 入試数学問題に対する自動解答システム

自動解答システムは問題テキストを論理式に翻訳する言語処理部と，入力された論理式をもとに演繹を行い解答を導く演繹処理部からなる．この翻訳＋演繹という構成に関しては大きな議論の余地はないと思われるが，実際に自動解答プログラムを実装するためには「問題を解く」という情報処理プロセスの仕様を明確にしておく必要がある．

入試数学問題には，大きく分けて「…を証明せよ」「…を求めよ」「…を図示せよ」という3つのタイプの問題がある．このうち，証明問題はそのまま伝統的な自動演繹の課題とみなせる．また,「…を図示せよ」という問題は図示すべき図形を表す簡単な表式が求まれば画像として表示することはさほど難しくないため,「…を求めよ」というタイプの問題に帰着する．では，入試数学問題において「何かを求める」とはどういうことだろうか？

まず分かるのは,「xを求めよ」という問題に対する答えはほぼ必ず「$x = \cdots$」という形をしていることである.「xの範囲を求めよ」という問題の答えは例えば「$0 < x < 1$」のような形になるが，これもまた「xの範囲 $= 0 < \square < 1$」という形の略記と考えられる．それでは答えである「$x = \cdots$」の右辺はどのような形をしているだろうか．入試問題を多数観察すると，そこに現れるのは以下のようなごく限られた要素のみであることが分かる：

- 数（整数，実数，複素数）および変数
- $+, -, \cdot$（積）, $\div$, べき乗
- ベクトルおよび行列

- sin, cos, tan, log, exp, $\sqrt{}$
- 等号・不等号
- 列挙．例えば $x = 1, -1$．場合分け．例えば $x = \begin{cases} 1 & (a > 0) \\ -1 & (a \leq 0) \end{cases}$

では，これらの要素をどのように組み合わせたものが「正解」だろうか．これは論理的にははっきりしている．すなわち,「…という x を求めよ」という問題文を翻訳した論理式を，全体として x に関する条件とみなして $\Phi(x)$ とし，答えの右辺を ϕ とすれば

$$\Phi(x) \Leftrightarrow x = \phi$$

すなわち問題で表されている x に関する条件と答えが同値となることが「正解」の条件である．

一方で，問題文の翻訳結果である論理式には，限量子$\forall$および$\exists$を用いた，$\forall x.\phi(x)$ や $\exists x.\phi(x)$ といった表現が出現し，また $\lambda x.\phi(x)$ （$\phi(x)$ を満たすような x の集合）といった表現（ラムダ抽象）が含まれる．しかし，これらを答えに用いることは許されない．例えば「$x^2 + y^2 = 1$ と $y = x + a$ が共有点をもつ a の範囲を求めよ」という問題に対して「$x^2 + y^2 = 1$ かつ $y = x + a$ をみたす x, y が存在する」ことを表す

$$\text{答} : \exists x \exists y (x^2 + y^2 = 1 \wedge y = x + a)$$

は正解である $-\sqrt{2} \leq a \leq \sqrt{2}$ と論理的には等価であるにも関わらず認められない．また,「$x^2 = 1$ の正の解を求めよ」に対して「$x^2 = 1$ かつ $x > 0$ となる x」を表す

$$\text{答} : \lambda x.(x^2 = 1 \wedge x > 0)$$

も同様に答えとしては認められない．

以上の観察から,「…を求めよ」という問題を解く，とは，ラムダ抽象や限量子を含む論理式（高階述語論理式）と同値な「$x = \cdots$」という形の式で，かつ右辺には数や変数，四則演算といった上で列挙した要素のみを含むものを探すという課題であることが分かった．では，そのような答えの式「$x = \cdots$」をどのように探せばよいだろうか．実は，この課題は,「方程式の答えを求める」という特別なケースを除くと，自動定理証明を中心とするこれまでの自動演繹の研究においてはほぼ手付かずだった．もしもヤマ勘でたまたま正解となる「$x = \cdots$」という式を見つけられれば，それが正解であることを自動証明する問題へと帰着するが，そのようなことは通常，人にも計算機にも不可能である．もう一つの問題は，演繹のスタートとなる問題の論理表現 $\Phi(x)$ が「どのような対象についての」論理式なのか，いいかえれば，どの公理系のもとで演繹を行うのか，という問題である．

高校数学の問題には，様々な数学的対象が入り混じって現れる．特に有理数（および整数・自然数）と実数がともに現れる問題は多い．このような様々な対象をまとめて扱うための公理系は，実質的には集合論しか存在しない．しかしもちろん集合論における演繹の探索空間は極めて膨大であり，入試数学で問われるレベルの内容に対し，通常の自動定理証明で行われるような力任せの自動探索で答えとなる式が見つかることは望むべくもない．

我々はここである特定の言語で記述できる問題に着目した．それは実閉体（Real closed field, RCF）の一階の言語で記述できる問題である．実閉体の一階の言語に対しては限量子消去アルゴリズム (RCF-QE) が存在する．これはすなわち問題が実閉体の一階の言語で記述できさえすれば，原理的には（ほぼ）答えが求まることを意味する．高校数学の問題のうち，幾何の問題の大部分と実数に関する代数の問題，さらに微積分の問題の一部は，その問題の中心となる部分が実閉体の一階の論理式で記述でき，そのような問題は国立大学 2 次試験問題のおよそ 4 割を占める [1]．実閉体の一階の論理式から限量子を消去することで残るのは，連立方程式・不等式の論理和であり，高校数学の問題であればこれらを解いて「$x = \cdots$」という形の答えを得ることはほぼ可能であることが期待できる．

もちろん実際の入試問題の大多数は実閉体の言語で直接記述されている訳ではない．解くために必要な演繹の主要部分が実閉体に関するものとみなせる問題であっても，幾何の問題や，場合によっては自然数と実数が入り混じった問題として与えられるものが多数存在する．このため，これらの問題を論理式へと翻訳するステップにおけるターゲット言語はやはり集合論とせざるを得ない．翻訳の結果得られる集合論の論理式から実閉体の一階の論理式への変換は発見的（ヒューリスティック）な方法によって行われることになる．以上の検討から得られた「集合論への翻訳 + 実閉体の言語への変換 + 限量子消去による演繹」という 3 ステップの構成は，翻訳のターゲット言語に関する「なんでも書けるように」という要求と，演繹部における「処理が決定可能である」という要求の 2 つを，発見的な手段による論理式の変換で半ば無理やりつないだものと見ることもできる．集合論で記述された問題を実閉体の言語による記述へ「落とし込む」変換部分は，一般的には極めて多様な演繹を必要としうるが，幸いにして実際の入試問題の場合はいくつかの規則に基づく等価な書き換えによってその大部分が実閉体の論理式へと変換できる．

以上をまとめると，一階の実閉体の論理式として表現できる「…を求めよ」タイプの問題に対し，実閉体の限量子消去（RCF-QE）を演繹部の中心とした以下のような手順が考えられる．

Step 1 問題文を機械的に翻訳し，集合論の（高階）論理式を得る

Step 2 そこから高階の要素（ラムダ抽象）および実数の四則および等号・不等号以外の要素を取り除き，一階の実閉体の論理式を得る

Step 3 RCF-QE アルゴリズムによって限量子を消去する

Step 4 残った連立方程式・不等式を解く

図 1 は，これを処理フローとして表し，各ステップにおける中間結果の例を示したものである．

この処理フローをプログラムとして実装したものが現在の解答システムの中心部である．RCF-QE アルゴリズムの実装としては Iwane らによる SyNRAC [2] を用いている．解答システムでは，RCF によって記述可能な問題に対する RCF-QE 以外にも，整数の問題や超越関数を含む問題など，いくつかのタイプの問題に対して発見的解法に基づく演繹処理（以下，ソルバーと呼ぶ）が実装されている．これらの発見的解法に基づくソルバーを用いた解答も，上記の手順を拡張することで行っている．すなわち，

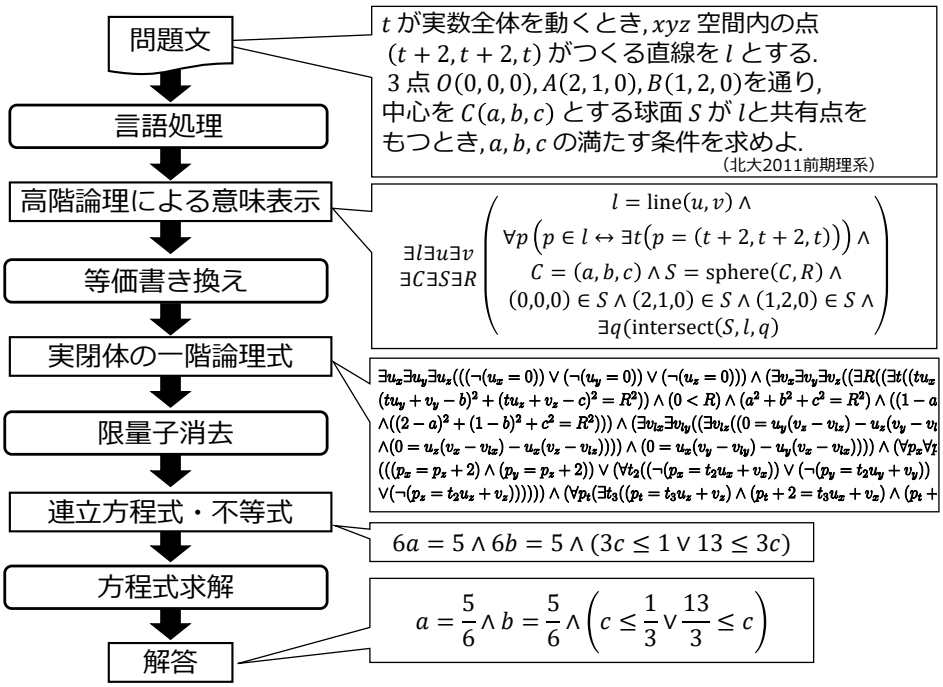

図 1: 実閉体の限量子消去を中心とした解答フロー

Step 2 の等価書き換えの過程で，あるソルバーが受け付ける入力形式が得られた場合に，そのソルバーを起動する．例えば，全ての変数・定数が整数である論理式が得られれば，その時点で整数問題ソルバーを起動する．

3. 解答システムにおける言語処理

日本語で書かれた数学問題テキストから論理式への変換において中核となるのは，組合せ範疇文法 [3, 4] と呼ばれる枠組みで記述された形式文法を用いた文解析である．この節では，組合せ範疇文法について概説したのち，言語処理部における論理的モデリングと統計的モデリングの関係について説明する．

3.1. 組合せ範疇文法

組合せ範疇文法は，組合せ規則と呼ばれる少数の文法規則と，多数の語彙項目からなる．語彙項目とは，単語の表層形 w に対して統語範疇 C と意味関数 f を対応づけた 3 つ組 $w :: C : f$ である．以下に動詞「(曲線が)(点を) 通る」と普通名詞「直線」の語彙項目を例として示す：

$$\text{通る} :: (S\backslash NP_{ga})\backslash NP_o : \lambda y.\lambda x.(y \in x)$$
$$\text{直線} :: N : \lambda x.\mathrm{line}(x)$$

統語範疇は，S(文)，NP_c(格 c の名詞句)，N(普通名詞) といった基本範疇，あるいは，それらを前向き・後ろ向きのスラッシュを用いて再帰的に組み合わせた $\mathbf{X}/\mathbf{Y}$ または $\mathbf{X}\backslash\mathbf{Y}$ ($\mathbf{X}, \mathbf{Y}$ は統語範疇) という形の複合範疇のいずれかである．複合範疇 $\mathbf{X}\backslash\mathbf{Y}$ は,「左側に範疇 $\mathbf{Y}$ をもつ句を取って全体として範疇 $\mathbf{X}$ の句となるもの」を表す．同様に，複合範疇 $\mathbf{X}/\mathbf{Y}$ は,「右側に範疇 $\mathbf{Y}$ をもつ句を取って全体として範疇 $\mathbf{X}$ の句となるもの」を表す．例えば上記の「通る」の統語範疇 $(S\backslash NP_{ga})\backslash NP_o$ は左に NP_{ga} すなわ

ちガ格の名詞句と NP_o すなわちヲ格の名詞句をとって全体として文を作るような単語であることを表す.

意味関数は単語や句の意味を表す高階論理式（型付きλ式）である．例えば，普通名詞「直線」には,「x は直線である」を表す一項述語 line を用いて「直線であるモノの集合」を意味する $\lambda x.\mathrm{line}(x)$ が意味関数として割り当てられている．意味関数で用いられる line などの記号の意味は公理によって例えば

$$\forall \ell(\mathrm{line}(\ell) \leftrightarrow \exists v_1 \exists v_2 \exists u_1 \exists u_2 \forall P(P \in \ell \leftrightarrow \exists t(P = (v_1 t + u_1, v_2 t + u_2))))$$

のように定義されている.

文中の単語あるいは句の統語範疇と意味関数に対し，組合せ規則を再帰的に用いることで，文の統語・意味解析が行われる．以下は，日本語において最も基本的な組合せ規則である右関数適用規則（$>$）および右関数合成規則（$>B$）の定義である:

$$> \frac{\mathbf{X/Y} : f \quad \mathbf{Y} : x}{\mathbf{X} : fx} \qquad >B \frac{\mathbf{X/Y} : f \quad \mathbf{Y/Z} : g}{\mathbf{X/Z} : \lambda x.f(gx)}$$

組合せ規則では，連接される句どうしの統語的制約がチェックされるとともに，句のもつ意味関数が関数適用や関数合成によって組み合わされる.

ある語（あるいは句）の意味関数の型はその語（あるいは句）の統語範疇から以下の関数 ρ によって定まる：

$$\begin{aligned}
\rho(X/Y) &= \rho(Y) \to \rho(X) \\
\rho(X\backslash Y) &= \rho(Y) \to \rho(X) \\
\rho(NP_c) &= i \\
\rho(S) &= o \\
\rho(N) &= i \to o
\end{aligned}$$

ここで i は様々な「モノ」を表す基底型，o は論理式を表す基底型で，$\alpha \to \beta$ は型 α を定義域，型 β を値域とする関数型である．全ての組合せ規則は，この対応関係を保つように定義されている．この対応関係によって，型 S をもつ句（すなわち文）が得られた場合に，その意味関数は well-formed な論理式であることが保証される.

図 2 に，文「C に接する直線が原点を通る」に対する導出木を示す（一部，導出過程を省略している）．導出木の葉は語彙項目に対応し，各導出ステップは組合せ規則の適用を表している．図の導出木では，例に示した 2 つの組合せ規則のほか，連体修飾節と普通名詞を組み合わせる規則 Adn および普通名詞に対して存在量化を行う規則 $>T_\exists$ が用いられている.

与えられた文に対して，統語範疇 S を根ノードに持つような導出木を探索する処理を構文解析とよぶ．自然言語の文は異なる複数の導出木を持つことがある．それらの導出木は，等価な意味表現を持つ（疑似曖昧性）こともあれば，異なる解釈に対応する異なる意味表現を持つこともある．可能な複数の構文木のうち，もっとも正しそうな意味表現を持つものを選び出すことを曖昧性解消とよぶ．現在の言語データ処理では，曖昧性解消のために，人手で作成した大量の解析済みデータから学習した統計モデルを使うことが一般的である.

$$
\dfrac{
\dfrac{
\dfrac{
\dfrac{
\dfrac{
\begin{array}{c}C\text{に接する}\\ S\setminus NP_{ga}\end{array}
\quad
\begin{array}{c}\text{直線}\\ N\end{array}
}{
Adn\dfrac{
\begin{array}{cc}:\lambda x.tangent(x,C) & :\lambda x.line(x)\end{array}
}{
N:\lambda x.(tangent(x,C)\wedge line(x))
}
}{
S/(S\setminus NP_{nc})
}>T_{\exists}
\quad
\begin{array}{c}\text{が}\\ S\setminus NP_{nc}/(S\setminus NP_{ga})\\ :\lambda x.x\end{array}
}{
:\lambda P.\exists x(tangent(x,C)\wedge line(x)\wedge Px)
}
}{
S/(S\setminus NP_{ga}):\lambda P.\exists x(tangent(x,C)\wedge line(x)\wedge Px)
}>B
\quad
\begin{array}{c}\text{原点を通る}\\ S\setminus NP_{ga}\\ :\lambda x.((0,0)\in x)\end{array}
}{
S:\exists x(tangent(x,C)\wedge line(x)\wedge(0,0)\in x)
}>
$$

図 2: CCG 導出木の例

3.2. 言語処理部における論理的モデリングと統計的モデリング

文法と辞書の開発は，実際のテキストで観察される文とその意味を表す論理式の間の関係をなるべく正確に捉えるように組合せ規則と語彙項目を定義していく作業である．この文とその意味の 2 項関係は同義文の存在および上で述べた曖昧性によって多対多の関係となる．文法と辞書の開発における基本的な方針は，どの文に対しても誤った意味 (ありえない解釈) が出ないようにすることである．この基本方針が貫徹された場合は,「ありうる解釈が出ないことはあっても，ありえない解釈が出ることはない」という条件を維持したまま漸進的に解析可能な文-意味の対応関係が増えていくことになる．これは理想的ではあるが，実際には様々な理由から「ありうる解釈が出ないこともあり，ありえない解釈が出ることもある」という状態を許容しつつ開発を進めざるを得ない場面は多々ある．

その根本的な理由は，辞書と文法による文-意味関係のモデリングは不良設定問題であることによる．例えば統語範疇を細分することによって，組合せ規則や語彙項目を特定の文の (正しい) 解析のみに使えるように定義することは可能ではある．極端な例としては，一つの文全体を「単語」とみなし，その意味を語彙項目として登録するということができる．しかし当然これでは新しい（未見の）文の解析はできない．そのため，ある文法現象に対応するよう文法・辞書を拡張する際は，そのターゲットとなる現象とそれ以外の現象が様々な形で組み合わさった場合にも，なるべく一般的に通用するように規則や語彙項目を定義することになる．しかし，この場合に予見しなかった組み合わせにおいて誤った解釈が出ることがあり，場合によっては，特定のケースで誤った解釈がでることが分かっていても，解析可能な文を増やすことを優先し，一部の誤りを許容して規則や語彙項目を増やすこともある．

文法と辞書による論理的モデリングのこのような状況においては，統計モデルによる曖昧性解消はその教科書的な定義とはやや異なる役目を負わされることになる．すなわち，統計的な曖昧性解消とは，第一義的には，与えられた文 s に対する可能な解釈の集合 $G(s)$ の上に確率分布や実数値関数 (スコア) を定義し，その確率ないしスコアを最大化する解釈 $\hat{t}\in G(s)$ を選ぶことによって一番もっともらしい（学習データと整合する）ものを選択するという手段であった．しかし，上で述べたように「そもそもありえない解釈」が文法と辞書によって決まる $G(s)$ に含まれる場合には,「蓋然性の高い解釈を選ぶ」のみでなく「ありえない解釈を除外する」という役割も一部「曖昧性解消モデル」によって担われることになる．純粋に統計的モデリングの立場から見れば,このことは単にそもそも起こりえない事象に対して確率 0 を割り当てているだけとも

言えるが,「記号列の形と意味の間には規則が存在する」ということを作業仮説（ないしドグマ）とする論理ベースの言語の理論の立場からは逸脱的な工学的妥協とも見られる．

統計モデルが与えるスコアを最大化する解釈を実際に求める処理を考えると，さらに状況は複雑となる．文 s に対する解釈 t のスコアを $f(s,t)$ と書くと，これは $\hat{t} = \mathrm{argmax}_{t \in G(s)} f(s,t)$ を求める最適化問題となる．この最適化問題の難しさは，主として文法と辞書によって決まる $G(s)$ の構造の複雑さと，スコア関数 $f(s,t)$ の種類によって決まるが，現在の解答システムで行っているレベルの詳細な意味解析のための文法に対しては現実的な時間で最適解を求めることは困難になる．このような場合の常套手段はふたつあり，ひとつは何らかの方法で解釈候補の領域を狭めた $G'(s) \subset G(s)$ を考え，$G'(s)$ の要素の中でスコアを最大化するものを選ぶことである．この場合，やはり計算量の問題から，厳密な最適解が得られること，すなわち $\mathrm{argmax}_{t \in G(s)} f(s,t) = \mathrm{argmax}_{t \in G'(s)} f(s,t)$ が保証されないような方法で $G'(s)$ を「決め打ち」せざるを得ないことが多い．もうひとつの常套手段は与えられた文に対して解釈を選ぶ処理プロセスを確率過程とみなし，各時刻（処理ステップ）において確率的に定まる処理アクション (例えば語彙項目や組合せ規則の選択）の列によって出力となる解釈が決定されると考える方法である．この場合も，厳密に確率が最大となるアクション列を求めることは計算量の問題から難しいことが多い．また,「解釈のもっともらしさの数量化」という見方からはやや不自然なモデリングであるとも言える．

このように，数学解答システムの言語処理部は文法と辞書による論理的な言語のモデルを基礎としながらも，形-意味の対応関係のモデリング・曖昧性解消のためのスコアリング・曖昧性解消アルゴリズムのいずれの局面においても統計的な手段との折衷によって動いている．この折衷性は論理によって言語の意味を捉えるという伝統的な目標からは妥協・逸脱とも見られようが，部分的にではあれ現実のテキストに対して実際に動くプログラムを基にすることで言葉と数学的思考にまたがる知的活動についての新しい知見が得られることを期待し，このような手段を採っている．

4. おわりに

本稿では入試数学問題を対象とした自動解答プログラムの概要について述べ，演繹処理部とともにプログラムの主要部をなす言語処理の部分についてその一端を紹介した．特に，言語データの処理における論理的なモデリングと統計的なモデリングの相補的ないし折衷的な関係について，その必要性と意義を説明した．

参考文献

[1] Takuya Matsuzaki, Munehiro Kobayashi, and Noriko H. Arai. An information-processing account of representation change: International mathematical olympiad problems are hard not only for humans. In *Proceedings of the 38th Annual Cognitive Science Society Meeting*, pages 2297–2302, 2016.

[2] Hidenao Iwane, Hitoshi Yanami, and Hirokazu Anai. Synrac: A toolbox for solving real algebraic constraints. In *Mathematical Software–ICMS 2014*, pages 518–522. Springer, 2014.

[3] Mark Steedman. *The Syntactic Process.* Bradford Books. MIT Press, 2001.

[4] Mark Steedman. *Taking Scope - The Natural Semantics of Quantifiers.* MIT Press, 2012.