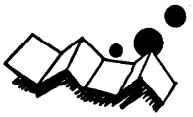
解　説



定　理　の　証　明†

西　村　敏　男††

1. きっかけ

定理の証明を計算機でやってみないか，というなんとなく誘いが，われわれ数学基礎論の研究グループにもたらされたのは，1959年のまだ暑い頃ではなかったかと思う．そのグループは，岩村聯，島内剛一（当時東京教育大，現立教大），前原昭二（当時早稲田大，現東工大），赤摂也（立教大）の各氏と私（当時法政大，現筑波大）を中心にして，毎週水曜日の午後に東京教育大に集まり，雑談に花を咲かせたり，数学基礎論を勉強したりしていた人達である．数学基礎論を勉強していた関係で，Turing 機械という概念，オートマトンという概念は多少知っていた．しかし，私に限れば，電子計算機については，見たこともなかったし，全くの無知であった．新聞には，わが国でも計算機というものが，大学や研究所で試作されだしたこと，証券会社などで外国製の機械が使われていることを報じていた．しかし，私にとってそれらの報道は他人事でしかなく，特別の関心を引く事柄でもなかった．

この誘いをもたらされたのは，黒田成勝先生（当時名古屋大学，故人）であったと思う．京都大学の数理解析研究所の設立準備のために，文部省から，数理科学の総合研究についての科学研究費が出ていて，その研究の一環として前記のテーマを提言されたのだったと思う．

そしてわれわれは，'計算機による定理の証明' にきわめて懐疑（否定）的であったにも抱らず，このなんとなく誘いに，個人個人によって大きな差はあったが，なんとなく乗っかってしまったのであった．もちろん私には，確たる定見があったわけではない．計算機という未知のものへの好奇心もあったろうが，退屈していたのかもしれない．この際計算機の使い方ぐらい，多少勉強しておいてもよかろう，といった軽い気持からだったのではなかったかと思う．

同じ誘いは，高須達氏（当時武蔵野通研，現京都大）にももたらされた．計算機をよく知っていた高須氏は，われわれとは独自に研究された．その成果は[3),7)]にある．後に同氏は，このプログラムを LISP で書き直され，その成果は 14) にある．しかしこの稿では，1959年～1961年あたりのもっぱらわれわれのたどった道のみを述べることとする．

1) には数理科学懇談会の成立，趣旨，呼びかけと共に，「数理科学の総合研究研究計画」があり，その中の　Ⅳ 計算機のプログラミング
を山内二郎先生（当時慶応大）が書かれている．その中の本年度の計画の3. 特別な問題のプログラミングの中に，「特に研究に多くの時間や特別の知識を要求されるやや困難なプログラミングの問題になるものを取り上げて実行する．たとえば，定理の証明を機械にやらせるためのプログラミング，原子物理学上の問題のためのプログラミングなどが提案されている．」と書かれている．また，プログラミングシンポジウムを組織することが書かれている．そして，2) にはニュースとして，UNESCO 主催の情報処理世界大会が，1959年6月15日～20日にパリのユネスコ本部で開かれた（この結果 IFIP が生まれた）ことを後藤英一氏（東大）が書かれている．その A～K のテーマの中の
Ⅰが　Proving of logical propositions
で，その中で，IBM 704 に組まれた「初等幾何の定理を証明する機械」の発表のあったことが記されている．同じ号の第Ⅳ班（代表者山内二郎）第3分科会（特別な問題のプログラミング）第1回会議（1959年6月19日）において，黒田成勝氏が，「数学の証明の電子計算機による実行」について説明し，つぎの2件が決定されたことが記されている．

（a）　数学の機械化の可能性を検討するための小委員会を1カ月以内に開くこと．

† Theorem Prover by Toshio NISHIMURA (Institute of Mathematics, The University of Tsukuba).

†† 筑波大学数学系

（b） 数学の証明のプログラミングを研究するグループを構成するための希望者をつのること．

2. 計算機の勉強

こうしてわれわれは誘いに乗ったわけであるが，計算機についてはまったく素人の集まりであったので計算機の勉強から始めなければならなかった．この時から近藤頌子氏（現慶応大）もわれわれと共に仕事をすることになった．われわれの所には，森口繁一先生（当時東大）のはからいで，赤氏の立教大の同僚である藤川洋一郎氏が，計算機の先生としてきて下さることになった．使用する計算機は，当時東京大学で作られた TAC (TODAI AUTOMATIC COMPUTER) であった．EDSAC をモデルとして，これに若干の機能を付加したものである．約5,900本の真空管を用い，記憶装置には16本のブラウン管が用いられていた．東大の総合試験所に安置されていた．

さて，藤川先生が来て下さるに先立って，われわれには1枚の紙が配布された．それは TAC の命令表であり，

A n (Acc)+(n)→Acc

などという，わけのわからない記号が記入されていた．1週間待てば，藤川先生が，書かれている記号の謎を明快に説明して下さるのであるから，カバンの片隅にしまい，翌週忘れずに持参すればよいのであるが，好奇心に満ちた顔ぶれには，その1週間が待てなかったのだろう．あるいはよほど退屈していたのかも知れない．これはなんだろうか，という謎解きが始まってしまった．手廻しの計算機からの連想で，Acc とは Accumulator のことであろう，A とは Add の頭文字，(Acc) とは Acc の内容であろう．と類推は進んでいった．計算機が2進で表示されていて，有限桁数からなること，そんな程度の知識だけからでも，3人寄れば文珠の智恵で，かなりの程度のことが解読できるものである．全命令中の一命令（どの命令だったか記憶は定かでない）を除いては，一同みな，かなりの自信をもって解読の正しさを信じたようであった．大げさに云えば蘭学事始めのようなものであった．

さて翌週，当時の私の手帖では10月14日になって，藤川先生の教えを受けることになった．この時から，この仕事には平本厳（現電力計算センタ）も加わることになった．この時点でも，まさか，われわれ自身がプログラムを組んで，実際に実験をしてみることになろうとは，恐らくはみな考えてもいなかったのではないだろうか．われわれも計算機をよく勉強すると共に，藤川氏にもよく数理論理学を勉強してもらい，あれこれと議論をしているうちに，好奇心に満ちた藤川氏が，プログラムを作ってくれるだろう，と少なくとも私は考えていた．藤川氏の講義は明快でわかりよかったので，われわれは TAC をよく理解できた．しかし，プログラム，プログラミングということを十分に理解できたというわけではなかったが．そして藤川氏も形式的体系，殊に Gentzen の形式的体系をよく理解してくれた．

そして翌週，事態はまさに私の考えた方向に動いているかに見えた．藤川氏は，われわれから習得した論理体系の一部である命題論理の証明方法を flow-diagram化して来てくれたのである．われわれは flow-diagram というものを始めて見た．そしてプログラムへの理解を深めることができた．数理論理学の専門家達は，藤川氏の説明を聞きながら，その中に含まれる1カ所の誤りを訂正すると共に論理体系自身の形式化を変えていった．そして，試行錯誤をまったく含まないアルゴリズムにしてしまった．藤川氏の提案してくれた flow-diagram は，たちまち1/2位に縮小され，命題論理の式を証明する改良の余地の無いまでにほぼ完璧な flow-diagram はその日のうちに完成してしまった．と同時に，私の予想に狂いを生じる大きな芽ができてしまった．

それまで，つまらなそうな顔をして藤川氏の話を聞いていた島内氏が，命題論理の flow-diagram のあたりから除々に，そしてそれが完成しわれわれが解散した後から急に，狂い出してしまったのである．彼はプロジェクトの最も有力な推進者，そして担い手に変ぼうしてしまったのである．その翌週，彼は不完全ながら，述語論理の flow-chart をつくってきてしまった．それはいく度かの推こうの後に，述語論理の体系自体も，Gentzen のもともとの形に変更が加えられ，より完成されたものへと変わっていった．私の手帖によれば，11月7日の土曜日に，われわれは教育大でプログラミングについての特別の集まりをもったようである．11月から12月の初旬にかけては flow-chart の完成と共に，論理式など扱われる対象を計算機内部でどう表現するかの検討，プログラミングの勉強に費やされた．flow-chart の各部分は，黒板で実際の例により試された．

当時われわれは，ポインタとかスタックといった概念を知らなかった．だが，内部表現の検討の過程で今

日の言葉で云えば，岩村氏は string による処理を，島内氏はリスト処理方式を提案され，結局リスト処理方式になった．これは，論理式のみならず，Gentzen 型の体系の証明構成には，まさにうってつけの方法であった．

12月12, 13, 14日の3日間を，箱根木賀温泉で合宿し，プログラムの完成を目指した．島内，近藤組はプログラミング，他は flow-chart の点検とデバッグ用ツールの作成をしたが，この合宿ではプログラムは完成には到らなかった．われわれはこの作業を東京に持ち帰って続けて，12月22日から東大の TAC にかけてみることになった．私は家に帰って flow-chart を眺めなおした．再点検をはじめてやがてつぎのことに気がついた．「私は論理の体系については熟知している．flow-chart の各部分が，なにをしようとしているのかもよくわかっている．とすれば，再点検をするということは，目的に合致していることを証明することである」．私は各部分の目的をほぼ数学的に表現し，逐一証明していった．全体の流れは，述語論理の完全性の証明の変形であり，論理式での変数の置換は，論理式の構成に関する帰納法によるものであった．データをバッファに退避させる順序ひとつに気を配る，そして徒らに場合分けの多い，また，'同様に' と通り過ぎてはいけない，単調な証明にうんざりしながら，まる2日近くをかけて証明をした．証明の途中で，1カ所だけ場合分けのスイッチのつけ誤りを発見したが，その後の実験で，誤りはその1カ所のみであったようである．プログラムの検証に，数学的証明手法が役立つことを実証したようなものであるが，それには，体系自体をよく理解していて，証明すべきことの適当な表現が必要である．そして，なにより大切なことは，プログラム自身がよく整理されていることであろう．島内氏はプログラミングと取り組んだ．そしてその成果は，黒板にプログラムを書いてわれわれに説明するという形で披露された．彼の秘術を尽くしたプログラミング技法に，他の人達はともかくとして，私は当然ついていけなかった．説明の途中で誤りを発見するのは島内氏自身であって，説明を聞いていた人達ではなかった．馬鹿でも聞き手にしておいて説明することは，プログラムのデバッグには良い方法である，ことを実証してみせてくれた．

TAC への入力は紙テープで，機械読取り機によるものである．12月22日の12時30分に教育大に集まったわれわれは，紙テープパンチ部隊を東大に派遣し，プログラムのデバッグと雑談に花を咲かせた後，夕方近くに東大に行った．パンチ機の操作，TAC の操作等の手引きをして下さったのは，森口先生のところに居られた清水留三郎氏（現東大教養学部）であった．エラーを起こしやすい TAC と私達を，夜を徹しておもりして下さったのは，TAC の製作者村田健郎（現日立中研技師長）と中沢喜三郎（現日立神奈川工場長）の両氏とその配下の技官の人達であった．TAC は暖かい計算機であった．それは前記諸氏の暖かい心づかいのおかげでもあったが，また同時に多大の発熱量のせいだった．厳冬のみぎり，窓を開け，上着を脱いでも寒さを感ずることはなかった．

島内プログラムはなかなか通らなかった．理由はわからなかった．静止したメモリの中をのぞく（ブラウン管メモリは，外から観察できる）．はじめは読み取ることがむずかしかったが，こんなことも馴れると早くなるものである．赤氏と私は，ビットパターンで論理式を読み取る不思議な能力を身につけた（勿論その能力はすぐになくなったが）．しかしプログラムは正常には動いてくれなかった．われわれは機械を疑った．そして，疑われる程に TAC はよくダウンした．悪戦苦闘の2日目だったかに，島内氏はプログラムが正常に動かない理由を発見した．それは，シフト命令が mod 64 のシフトであることが，マニュアルに書いてなかったことからくるものであった．この修正と共に，プログラムは正常に動きだした．奇蹟的なことに，近藤氏のパンチには1字の打ち誤りもなく，島内プログラムには一命令の誤りもなかったのである．馬鹿な聞き手達の功績は偉大であったのである．こうして主プログラムは12月のうちに完成した．これに入出力プログラムを完成させて，1960年1月21日にいくつかの簡単な定理を証明したのだった．5)の第Ⅰ班の部で，代表者の秋月康夫先生（当時東京教育大）はつぎのように書かれている．「黒田成勝，高須達のコンビ及び岩村聯，赤摂也，西村敏男，島内剛一，藤川洋一郎ら数学基礎論グループは，黒田の提唱により第Ⅳ班と協力して証明のプログラミングの研究に従事した．後者のグループは約3カ月を費し，1960年1月21日，東大工学部 TAC に矛盾律，分配律などの論理式の真偽を判定させることに成功した．小型の計算機にもかかわらずこの成功を収めたのは数学基礎論的思索の大成功と自負している．その後もひきつづき電子計算機による数学の証明のプログラミングの研究が両グループにおいて行われている．」また第Ⅳ班の部では

代表者山内二郎先生は，第3分科会の中でつぎのように書かれている．「数学の証明を電子計算機にやらせる問題が黒田委員（名大）から提出され，通研(M-1)，教育大グループ（TAC），その他のグループがそれぞれ独立にプログラムして，簡単な問題（たとえば二等辺三角形の底角は相等しい）は出来ることがわかった!! これにより基礎数学者の電子計算機利用研究の有効性についての認識を確認した．新らしい発展が期待される．」

3. 証明のアルゴリズム

この節では，当時の記録[4),6)~9),11)~13),17),18)]に基づきながら，設定目標，用いられた論理体系，定理証明のアルゴリズム等を述べることにする．

3.1 設定目標

この研究に着手するに当り，われわれはその目標を次のような2つの段階に分けることにした．

（1） 数学の証明を原理的に機械化すること．すなわち，数学における（正しいか正しくないかわからない）定理を input として機械にあたえたとき，それが正しいものであれば，いつかは必ず output として，その証明が放出されるようなプログラムを書くこと．ただし，その効率は出来るだけ高いことがのぞましく，たとえば，現在容易に使用できる機械によっても，常識的な時間内に，相当数の定理を証明しうるものでなくてはならない．

（2） そのプログラムが完成したならば，それをより効果的にするために，証明構成のための作戦をそれに組み込んで行くこと．それには，もちろん人間の発見的手段 (heuristics) などが参考となるが，これを組込むには，あくまでもそれに証明論的明確さをもった形式化が与えられ，かつそれに対する機械の性能の立場からする相当精密な estimation がなされていなくてはならない．

われわれがまず目標としたのは当然この中の(1)であった．K. Gödel の '完全性の定理' から，「述語論理では，恒等的に正しい論理式は，述語論理の体系で証明可能である」ことはわかっている．命題論理の範囲ならば，それが '恒等的に正しいかどうか' を判定する種々の方法がある．しかし，述語論理の体系では，残念ながら，そのような決定問題が解けないこともわかっている．そこで(1)においても，「もし正しければ，いつかは必ずその証明を output する」という形をとらざるを得ない．さらにまた，完全性の定理は，恒等的に正しい命題に対する証明を与えるアルゴリズムを与えているわけではない．そのような関係で，「命題が正しい」ことが，そのまま「証明できる」ことに直接に結びつくような，述語論理のうまい形式化が必要になってくる．述語論理の形式的体系は，Russell や Hilbert 以来，実に多くの形式化が行われている．ただ多くのものでは，「'A' と 'A ならば B' から 'B' を推論する」という推論法則（三段論法）がきわめて重要な役割りを果たしている，これなしでは証明はつくれない．証明すべき命題Bはわかっていても，Bを得るためのAについてはなんの情報もないわけであり，このAを機械的に見付け出すことは困難である．

幸い，われわれは述語論理の G. Gentzen による形式化を熟知していた．この体系では，公理はきわめて簡単な形をもち，その代り論理演算に関する推理規則は若干多いが，きわめて重要な性質をもっている．その性質とは

「与えられた証明の中から三段論法を取り除くことができる」

という性質である．三段論法がなければ，Bを推論するためのAをみつけ出す必要はないことになる．しかし，「与えられた証明から三段論法のない証明をつくる」ということは，「三段論法のない証明を直接につくる」こととは違う．したがって，Gentzen の形式的体系を採用すればそれで問題が解決するわけではない．しかし，原理的可能性があるということは，展望として一条の光を与えるものである．

3.2 論理体系

証明すべき命題Pが与えられたときに，Pが本来証明されるべきものならば，Pに至る証明を，一歩一歩機械的に与えることができるような形式的体系であることが望ましい．これは，証明をつくり出すアルゴリズムと密接にかかわっていることであるから，採用される論理の形式的体系は，アルゴリズムと関係していることはいうまでもない．しかし，両者を並列的に述べることは困難なので，まず論理体系から述べる．

まずはじめに，数学の '定理' と '証明' の概念から説明する．

3.2.1 論理式

つぎの記号を用いる．1. free variables $a, b, c\cdots$, 2. bound variables $x, y, z, \cdots$, 3. predicates $P, Q, R, \cdots$, 4. 論理記号 $\neg$, $\wedge$, $\vee$, $-$, $\rightleftharpoons$, $\forall$, $\exists$, 5. 補助記号 (,), , , $\rightarrow$, —．これらの記号を組み合わせて論理式を定義する．1. predicate のあとに，いく

つか（0個でもよい）の free variables を並べたものは論理式である．2. A, B が論理式のとき，$\neg(A)$, $(A)\vee(B)$, $(A)\wedge(B)$, $(A)\rightarrow(B)$, $(A)\rightleftharpoons(B)$ は論理式である．これらはそれぞれ，'Aでない'，'AあるいはB'，'AかつB'，'AならばB'，'AとBとは同値' の意味である．( )は，結合の順序を表わすためにつけたのであり，必要がなければ取り去ってもよい．3. $B(a)$ を free variables をいくつか（0個でもよい）含む論理式とする．この a を x でおきかえたものを $B(x)$ と表わす．$B(a)$ が bound variable x を含まないとき，$\forall x(B(x))$ および $\exists x(B(x))$ は論理式である．ここでも，( )は必要がなければとり除いてもよい．これらはそれぞれ，'すべてのxについて$B(x)$'，'あるxについて$B(x)$' の意味である．ここでは constant や function の記号を含めなかったが，これは理論的な困難さのためではない．constant は free variable で代用できるし，function を扱うことは，記憶容量からみて，到底不可能だったからである．

論理記号を含まない論理式を原始命題という．原始命題に含まれる変数の個数は，任意の負でない整数であり得るが，われわれはこれを3以下に制限した．これは記憶容量の制限が主たる原因であったが，論理的にはこれで十分であることも示される．論理式の例としては

$$\forall x\forall y(Pxy\vee\neg Pxy)$$
$$N(a)\rightarrow\exists x(R(a,x)\wedge N(x))$$

などがあげられる．

3.2.2 定　理

数学での '定理' は，$A_1,\cdots,A_m$, $B_1,\cdots,B_n$ を論理式とするとき，

「$A_1,\cdots,A_m$ という仮定から，$B_1,\cdots,B_n$ のうちの少くとも1つが出て来る」

という形をしている．これはつぎのように云いかえることができる．

「$\neg A_1,\cdots,\neg A_m$, $B_1,\cdots,B_n$ のうちの少なくとも1つが成り立つ」

それゆえ，数学の定理は，形式的に，有限個の論理式の列になる．この論理式の有限列のことを式 (sequent) と呼び，Γ, Δ, Θ, … などのギリシヤ大文字で表わすことにする．式は有限個の論理式の列であるから，正しい定理をあらわすこともあるし，そうでないこともあるのは当然である．

3.2.3 推論と証明

$$\Gamma, D, \Delta, \neg D, \Theta$$

という形をした式では，Dか$\neg D$のいずれか一方は必ず正しい論理式になる（排中律）ので，上の式は常に正しい．このような形をした式を公理として考え，始式 (beginning sequent) と呼ぶ．このような有限個の始式から出発して，つぎにあげる6つの推論規則を用いて，下へ下へとつないでいったものが証明である．

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

$$\frac{\Gamma, \neg A, B, \Delta \quad \Gamma, A, \neg B, \Delta}{\Gamma, A\rightleftharpoons B, \Delta}$$

$$\frac{\Gamma, A, B, \Delta \quad \Gamma, \neg A, \neg B, \Delta}{\Gamma, \neg(A\rightleftharpoons B), \Delta}$$

$$\frac{\Gamma, A(a), \Delta}{\Gamma, \forall x A(x), \Delta} \qquad \frac{\Gamma, \neg A(b), \Delta, \neg\forall x A(x)}{\Gamma, \neg\forall x A(x), \Delta}$$

（aは下の式に現われない free variable）　（bは任意の free variable）

最後の推論の free variable b は任意であるが，その証明で，そこから下の式に現われる free variable のどれかであると仮定してもよいことがわかっている．これは，つぎに述べるアルゴリズムとも関係してくる．また，この体系では，$\neg\neg A$ は A と同一のものと見做される．すなわち，$\neg\neg A$ は直ちに A に書き変えられる．

ある証明の一番下の式を証明可能な式という．すなわち，その式を一番下の式としてもつような証明が存在するとき，その式は証明可能というわけである．

さて，こうして論理体形が得られた．この論理体系は，Gentzen によって形式化された述語論理の体系を出発点としているが，公理，推論規則とも同じものではない．しかし，これら2つのものは同値なものであることを証明することができる（これには数理論理学上の証明論の専門知識が必要なので，ここではふれない）．また，三段論法に対応する推論はまったく含まれていないことが重要な点である．

6つの推論の中には，論理記号$\wedge$, $\rightleftharpoons$, $\neg$, $\forall$は含まれているが，$\vee$, $\rightarrow$, , $\exists$という記号は現われていない．しかし，

$$A\vee B\equiv\neg(\neg A\wedge\neg B)$$
$$A\rightarrow B\equiv\neg A\vee B$$
$$\exists x A(x)\equiv\neg\forall x\neg A(x)$$

であるから，それに基づいて，つぎの推論規則を入れればよい．

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

$$\frac{\Gamma, \neg A, B, \Delta}{\Gamma, A \rightarrow B, \Delta} \qquad \frac{\Gamma, A, \Delta \quad \Gamma, \neg B, \Delta}{\Gamma, \neg(A \rightarrow B), \Delta}$$

$$\frac{\Gamma, A(b), \Delta, \exists x A(x)}{\Gamma, \exists x A(x), \Delta} \qquad \frac{\Gamma, \neg A(a), \Delta}{\Gamma, \neg \exists x A(x), \Delta}$$

(bは任意の free variable) $\left(\begin{array}{l} a\text{は下の式に現われ} \\ \text{ない free variable} \end{array}\right)$

論理記号は，すでにあげたすべてを含んでいる方が，命題を論理式で表現しやすいし，また，理論的にも，プログラムの作成上にも，原理的にはなんの障害もない．しかし，われわれが推論規則として前の6個に制限したのは，もっぱら記憶容量からくる制限であった．

3.3 アルゴリズム

われわれの目標は，任意の式が与えられたとき，その式が正しいものであれば，それに到る証明をつくり出すことである．そのためにわれわれは，「推論図を逆にたどる」という方法を取ることにした．ここで注目してほしいのは，6つの推論の中の，$\neg\forall$に関する最後の推論を除けば，下の式に比べて上の式は，論理記号が少ないという意味で簡単になっていることである．その意味で，推論図を逆にたどる操作というのは，下の式を推論図に従って分解していく操作にもなるわけである．以下にアルゴリズムを与える．このアルゴリズムをたどれば，'本来証明されるべき式については必ず証明が得られる'ということの証明は，この論理体系の'完全性'の1つの証明を与えるものであって，それには数理論理学の専門知識が必要である．

さて，free variables を1列に並べ

$$a_0, a_1, a_2, \cdots$$

とする．われわれは補助的な variable を導入し，これをalah variablesと呼び，ギリシャ文字を用い，これも1列に並べて

$$\alpha_0, \alpha_1, \alpha_2, \cdots$$

とする．これらの変数を使うときには，前のものから順々に使っていくことにする．以下の手続きの途中では式の列

$$S_0\,;\ S_1\,;\ S_2\,;\ \cdots\,;\ S_n$$

が操作の対象になる．これらの式のうちのいくつかのものには適当な印がつけられていて，'消されている'と呼ばれる．また使われた alah variables に対しては，それぞれいくつかの free variables が対応つけられて，次のような表が作られてゆく：

α_0 : $a_0^0, a_1^0, \cdots, a_{i_0}^0$

α_1 : $a_0^1, a_1^1, \cdots, a_{i_1}^1$

⋮

α_q : $a_0^q, a_1^q, \cdots, a_{i_q}^q$

与えられた式を S_0 とする．このとき手続きはつぎのように進行する．はじめ S_0 は消されていない．

(1) 式の列の中の消されていない式の中に，$\neg$以外の論理記号を含む論理式で，$\neg\forall x F(x)$ の形をしていないものがあるとき，そのような論理式のもっとも左にあるものを取り上げる．この論理式を含む式を消す．ここから，注目している論理式の形に従って場合を3つに分ける．

(1.1) 注目している論理式が $A \wedge B$ の形をしているとき．いま消した式は $\Gamma, A \wedge B, \Delta$ の形である．式の列の最後に2つの式，Γ, A, Δ と Γ, B, Δ を加える．

(1.2) 注目している論理式が $\neg(A \wedge B)$ で消した式が $\Gamma, \neg(A \wedge B), \Delta$ のとき，式の列に $\Gamma, \neg A, \neg B, \Delta$ を加える．

(1.3) 注目している論理式が $\forall x F(x)$ で，消した式が $\Gamma, \forall x F(x), \Delta$ のとき，式の列に $\Gamma, F(a), \Delta$ を加える．aはまだ使われていない free variable の最初のものとする．

(2) 式の列の中の消されていない式の中に，$\neg$以外の論理記号を含む論理式で，$\neg\forall x F(x)$ の形でないものがないとき，alah variables の対照表に従って free variable を代入して，消されていない式がすべて，同一の代入で

$$\Lambda, A, \Delta, \neg A, \Pi$$

の形になるかどうかを調べる．代入は同一の alah variable には対照表のその alah の右に書いてある free variables のうち1つをすべての箇所に入れる．また，このような代入は alah variables と free variables の考えうるすべての組み合わせについてやってみる．もし，ある代入で，消されていないすべての式が $\Lambda, A, \Delta, \neg A, \Pi$ の形になれば，はじめ与えられた式 S_0 は証明可能で，得られた式の列にその代入を施したものは，S_0 に到る証明である．また，どんな代入でもうまくゆかなかったときは，

(2.1) 式の列の中の消されていない式の中に $\neg\forall x F(x)$ の形の論理式がある場合，そのような論理式の中で最も左にあるものを取り上げ，それを含む式を消す．いまこれが，$\Sigma, \neg\forall x F(x), \Gamma$ だったとしよう．このとき，式の列に $\Sigma, \neg F(\alpha), \Gamma, \neg\forall x F(x)$ を加える．ここで α はそれまで使われていない alah variables の中で最初のものである．また，$\Sigma, \neg\forall$

$xF(x), \Gamma$ の中に含まれる free variables を $b_0, b_1, \cdots, b_m$ とすれば，alah variables と free variables の対照表に

$$\alpha : b_0, b_1, \cdots, b_m$$

をつけ加える．ただし，$\Sigma, \nearrow \forall xF(x), \Gamma$ に free variables が1つも入っていないときには

$$\alpha : a_0$$

をつけ加えることにする．

(2.2) 式の列の中の消されていない式の中には，$\nearrow$ 以外の論理記号をもつ論理式がない場合には，与えられた式 S_0 は証明不可能である．

3.4 TAC，プログラム，解かれた問題

すでに述べたように，計算機としては TAC を使用した．TAC は2進補数表示の計算機で，1語 35 bits (18+17)，512語 (1024語) からなる．Accumulator は 70 bits, MD-register は 35 bits, B-register が 10 bits である．プログラムは前述のアルゴリズムを coding したもので，その長さはつぎの通りである．

input routine	約210語
master routine	約670語
output routine	約210語

これからわかるように，入力，証明，出力はそれぞれ別々に行わなくてはならない．しかし，入出力は，機械の記号上の制約はあるが，日常われわれが使いなれている論理式と同じ形で行われる．その変換は，今日コンパイラの算術式の変換と全く同じように行われていると考えてよい．

論理式が機械の中でどう表わされているかは**付録1**に，解かれた問題の例は付録2にある．

4. 定義，補助定理の導入[10),11),15),16),19)]

4.1 評価と展望

設定目標の1の項，すなわち「数学の証明を原理的に機械化すること」を現実の計算機の上に実現したので，最初の目的を達成した．しかし，実際の数学の証明ということになると，これはあまりにも現実離れのしたものであり，数学というよりは論理の機械化といった方がよいものであった．簡単な欠点をひろい上げるとつぎのようなものがある．

(1) 関数を表わす記号をもたないこと．これは，原理的な困難さではないが，関数の諸性質をpredicate で表現しようとすれば，多くの公理を含む長い長い論理式が必要になる．

(2) この体系は equality axiom をもっていない．

(3) alah variables が増加すると，代入する free variables の組合わせが天文学的数字にまで増大する．

(4) 論理の公理は簡単な形をしているが，数学の証明では多くの定理を仮定する．このような補助定理が利用できない．

(5) 数学の定理は複雑な概念構成からできている．いままでの体系では，簡略記号の定義をすれば，徒らに alah variable が増えて，実際上は使いものにならない．

(6) 証明の手法が1つに固定している．

(7) 発見がない．

(8) 過去の経験を証明の上で生かせない．

問題点を指摘することは簡単であるが，解決は容易ではない．

幸いにして，TAC 以外にもわれわれが使える機械が世の中に出てきた．HIPAC 101 という，日立製作所で作った初期の計算機が，日本科学技術研修所に入り，これを利用できる環境を得ることができた．これは，森口先生，藤川氏そして矢島敬二氏（日科技研）のおかげであった．この機械は，主記憶は磁気ドラムで，2kW 程度のものであるから，雄大な夢を託せるものではなかったが，原理的な可能性の追究には，それなりに役に立つものであった．もちろんここでも，input routine, master routine, output routine はそれぞれ別々に入れて働かせるより他に方法はなかった．1回の実験では，約2700ステップのプログラムの入力だけで45分を要し，機械にかけてのデバッグは最小限におさえなければならなかった．

4.2 新らしい体系

私は新らしい環境のもとで，1960年の夏から検討に入り，1961年のはじめまでに，つぎの事柄を計画し，無事に目的を果たした．

問題点として(1)～(8)をあげたが，(1)は記憶容量の関係でやめることにした．(6)，(7)，(8)は，問題の概念構成が容易でないので，これもさし当っては取り上げないことにした．(2)，(3)，(4)，(5)の4点に問題をしぼった．この中で(2)は(4)に含ませることにした．また，(3)は，全面的な解決ではないが，(4)と(5)によってかなり前進するので，(4)と(5)を中心課題とした．

まず(4)から述べよう．

「2等辺3角形の両底角は相等しい」

という命題を証明するには，3角形の合同の定理などいくつかの仮定を用いるのが普通である．そして，例

えばつぎのようにできる.

仮定

① $\nearrow xy=uv, \nearrow xz=uw, \nearrow\angle yxz=\angle vuw,$
$\angle xyz=\angle uvw$

② $\nearrow xy=uv, uv=xy$

③ $\angle yxz=\angle zxy$

定理

$\nearrow ab=ac, \angle abc=\angle acb$

①で，uをx，vをz，wをyと同一視すれば①は

$\nearrow xy=xz, \nearrow xz=xy, \nearrow\angle yxz=\angle zxy,$
$\angle xyz=\angle xzy$

となり，③との三段論法で

④ $\nearrow xy=xz, \nearrow xz=xy, \angle xyz=\angle xzy$

が得られる．②の，uをx，vをzと同一視すれば，④との三段論法で

⑤ $\nearrow xy=xz, \angle xyz=\angle xzy$

が得られる．⑤は定理として与えられたものの一般の形である．

証明に背理法を用いるならば，仮定として①，②，③の他に

④′ $ab=ac$

⑤′ $\nearrow\angle abc=\angle acb$

を加える．背理法は仮定から矛盾を導くのであるから，定理は空列（矛盾を表わす）ということになる．①〜⑤に三段論法を適用して空列を導くことは困難ではない．

後に Robinson は，これらの仮定を clause と呼び，三段論法を resolution principle と名付け，同一視を unification と呼び，背理法により空列を導いて証明を出す方法を与えている．以下の方法は Robinson の方法を含むものであることを注意しておく．

①は

$$\overset{1\,2}{\nearrow}\overset{3\,4}{xy=uv}, \overset{5\,6}{\nearrow}\overset{7\,8}{xz=uw}, \overset{9\,10\,11}{\nearrow\angle}\overset{12\,13\,14}{yxz=\angle vuw},$$
$$\overset{15\,16\,17}{\angle}\overset{18\,19\,20}{xyz=uvw}$$

の形をしているが，この x, y, z, u, v, w は‘任意の $x, y, \cdots$’の意味であり，だからこそ前の同一視ができたのである．とすると，①は，述語の並びと，各変数が左から数えて‘なん番目’に現われるかだけで特徴づけられる．すなわち

$\nearrow=, \nearrow=, \nearrow\angle=\angle, \angle=\angle$

x: 1, 5, 10, 15; y: 2, 9, 16; z: 6, 11, 17

u: 3, 7, 13, 18; v: 4, 12, 19; w: 8, 14, 20

によって特徴づけられる．したがって

$$\overset{1\,2}{\nearrow}\overset{3\,4}{ab=ac}, \overset{5\,6\,7}{\angle}\overset{8\,9\,10}{abc=\angle acb}$$

では，a: 1, 3, 5, 8; b: 2, 6, 10; c: 4, 7, 9

であるが，⑤は

$\nearrow=, \angle=\angle$

x: 1, 3, 5, 8; y: 2, 6, 10; z: 4, 7, 9

で一致するから，⑤の特別の場合に当たるわけである．

一般に，注目している式が，仮定から直接導かれるものかどうかは，注目している式が，仮定よりも弱い形をしていればよい．したがって，注目している式の任意の部分集合について，その各項の任意の順列について，つぎのことを調べることになる．

（1） 述語の列が一致しているか．

（2） 変数の一致条件が仮定より強い形をしているか．

（1）と（2）が共に Yes ならば，注目している式は仮定に含まれることになる．

つぎに定義の導入について述べる．集合の‘$=$’の推移性という簡単な問題を考える．これは

$\nearrow\forall x\forall y(x=y\rightleftharpoons\forall z(z\varepsilon x\rightleftharpoons z\varepsilon y)), \nearrow a=b,$
$\nearrow b=c, a=c$

という定理になる．このとき，このxとyにはそれぞれaとbが，そしてまた，それぞれbとcが代入されてはじめて証明ができる．すなわち

$\nearrow\forall x\forall y(x=y\rightleftharpoons\forall z(z\varepsilon x\rightleftharpoons z\varepsilon y))$

が2回使われることになり，使われる alah variables の数は徒らに多くなる．そこでつぎの定義と推論を導入した．

Definition　$P(x_1, \cdots, x_n)\equiv A(x_1, \cdots, x_n)$

推論D　$$\frac{\Gamma, \varepsilon A(a_1, \cdots, a_n), \Delta, \varepsilon P(a_1, \cdots, a_n)}{\Gamma, \varepsilon P(a_1, \cdots, a_n), \Delta}$$

$\left(\begin{array}{l}a_1, \cdots, a_n \text{ は任意の free variables.}\\ \varepsilon \text{は}\nearrow\text{か空}\end{array}\right)$

Definition は1つとは限らない．一般にn個の Definition が与えられるが，それは自動的にn個の推論 $D_1, \cdots, D_n$ が導入されたのと同じことになる．

そして，つぎのような方法によって証明をつくる．

（1） 証明すべき定理，証明に必要な仮定，定義を入力する．

（2） 目的の定理を3節で述べた方法によって分解していく．

（3） 得られた消されていない各式に，$\nearrow$以外の論理記号をもつ論理式の一番外側の論理記号が$\nearrow\forall$のみになったとする．このとき，公理が与えられた仮定にマッチするような alah variable への free variables

の代入を探す．すべての式にマッチするような代入が見つかれば，それで証明を終る．

(4) マッチしないような式があれば，つぎのどれかを行う．

(4.1) ∀を一番外側にもつ論理式を分解して新らしい式をつくり，(2)にもどる．

(4.2) 与えられた仮定から，可能な三段論法を適用して，いわば'補助定理'をいくつかつくり出し，これを新らしい仮定とに加え(3)にもどる．

(4.3) 定義が適用できれば，定義による書きかえの推論を適用して(2)にもどる．

(4.1)，(4.2)，(4.3)の適用の順序は，証明の戦術であり，問題によって，その順序はいろいろに変えられる．

この方法によって，群，環，体の諸式などが証明された．また，初等幾何では，「二等辺三角形の両底角は等しい」，「三角形の内角の和は二直角」等の諸定理，集合論では「$\langle xy\rangle$ を $\{\{x\},\{xy\}\}$ によって定義するとき，$\langle ab\rangle=\langle cd\rangle$ ならば，$a=c$ かつ $b=d$」などが証明された．

証明すべき定理が'空式'であれば，(4.2) のみを適用することになり，これが Robinson の resolution principle による prover になる．

数年をおいて1965年に，ほぼ似たアルゴリズムで，記憶容量の少し大きい，そしてスピードも早いHIPAC 103 によって，私はさらに実験を重ねた．その結果のごく一部は 19) に解説した．

このアルゴリズムでは，3節で述べたものよりはるかに証明力は増えた．しかし，(4.2) によりつくり出される補助定理は，きわめてぼう大なものになることは，私自身の実験からも確かめられた．

また，関数記号を含まないので，unification は変数と変数の間だけになるので，この部分は技術的には殆んどなにものも含まれていない．

5. おわりに

1959年～1961年にかけての Theorem Prover のわれわれの歩みを述べてきた．その当時，われわれは数学を専業としていたので，こうした事を欧文の論文として書こうという者がいなかったのは惜しまれる．その後に，米国あたりで prover についての論文も見るが，当時のわれわれの考え，あるいはアルゴリズムより優れているとも思えない．私自身その後，関数を対象とするプログラムの検証系の prover，並列処理系の prover など作り発表した．しかし，そこでは，形式的体系の構成方法に苦心したが，それは，前述の諸体系に盛られた考え方をいかにうまく生かせるか，であったように思う．そして，これまで述べてきたことはすべて，数理論理学上の証明論の基本考察から出ていることばかりなのである．

参 考 文 献

1) 数理科学ニュース No. 1, 1959年4月.
2) 数理科学ニュース No. 3, 1959年10月.
3) 高須 達：数学の証明の計算機による実施のためのプログラミング，第1回プログラミング・シンポジウム報告集 pp. 235-242 (1960年1月).
4) 島内剛一：LK の証明のプログラミング，同上，pp. 251-253.
5) 数理科学総合研究 1959年度報告，1960年2月.
6) 森口繁一：電子計算機と数理科学，第3章，数理科学総合研究研究報告 1959年度，pp. 6-8 (1960年2月).
7) 黒田成勝：証明機械としての電子計算機，
   1. 黒田成勝：機械による数学の証明
   2. 高須 達：M-1 による数学の証明
   3. 島内剛一：TAC による数学の証明のプログラム，

   同上，pp. 11-44.
8) 赤 摂他：数学の証明の機械化について，数理科学ニュース，No. 5, pp. 14-21 (1960年4月).
9) 赤 摂他：機械による数学の証明のプログラム—推理解析学の現状—数学，Vol. 12, No. 2, pp. 114-118 (1960年10月).
10) 西村敏男：証明のプログラミング，数理科学総合研究研究報告 1960年度，pp. 35-51 (1961年2月).
11) 西村敏男：証明と計算機，科学基礎論研究 18号，Vol. 5, No. 3, pp. 15-20 (1961).
12) 島内剛一：人工頭脳への一歩，同上，pp. 20-24.
13) 総合試験所年報第20年別冊，昭和37年3月，東大自動電子計算機報告，第2章 TAC を利用した計算，例 7) LK における証明，pp. 144-145.
14) 高須 達：記号操作と定理の機械的証明，第3回プログラミングシンポジウム報告集，E. 自由課題 pp. E43-62 (1962年1月).
15) 西村敏男：証明のプログラミング，第4回プログラミング—シンポジウム報告集，pp. E44-45 (1963年1月).
16) 西村敏男：推理と計算機，数理科学総合研究研究報告，pp. 36-38 (1963年2月).
17) 島内剛一：証明のプログラミング，数学，Vol. 15, No. 1, pp. 48-55 (1963年7月).
18) 島内剛一：数学の機械化，電気通信学会雑誌，第46巻11号，pp. 45-50 (昭和38年11月).
19) 西村敏男：定理の証明，情報処理，Vol. 11, No 11, pp. 646-651 (1970年).

(昭和57年12月23日受付)

付　録 1　論理式，式の記憶される形

1. 記号

論理記号　2 bits

∧　01　；　⇌　10；　∀　11

符号　1 bit　↗なし：0；　↗あり：1

Predicate　6bits

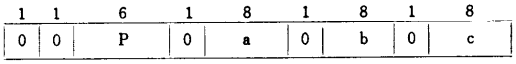
Free variable　7 bits＋0,

Bound variable　0＋6 bits＋1；

Alah variable　1＋6 bits＋1

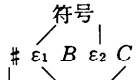
2. 論理式　構成の一段階ごとに 1 語 (35 bits) を使う.

2.1　原始論理式

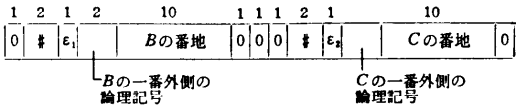


2.2　論理記号を含む論理式

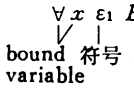
2.2.1　∧(0.1) または ⇌(10) を一番外側の論理記号とする論理式

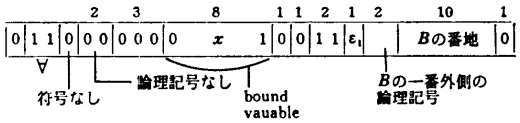


論理記号　構成要素(すでに構成された論理式)



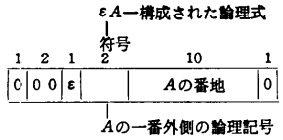
2.2.2　∀ (11) を一番外側の論理記号とする論理式





3. 式の中での論理式

1 つの論理式に 17 bits (半語) を用いる.



付　録 2　解かれた問題の例

1. Principia Mathematica の約 200 題の命題論理の論理式　(3 分 15 秒)

2. $\forall x(A(x)\rightarrow B(x)), \forall x(B(x)\rightarrow C(x))\rightarrow \forall x(A(x)\rightarrow C(x))$　(4.7 秒)

3. $\forall x(A(x)\wedge B(x))\rightleftarrows \forall x A(x)\wedge \forall x B(x)$　(15 秒)

4. $\forall x(A\vee B(x))\rightleftarrows A\vee \forall x B(x)$　(7 秒)

5. $\exists x \forall y(A(y)\rightarrow A(x))$　(5 秒)

6. $\forall p \forall q \forall l \forall m(\nearrow p=q\wedge Tpql\wedge Tpqm \rightarrow l=m)\rightarrow \nearrow a=b\wedge Tcda\wedge Tcdb \rightarrow c=d$　(3 分 16 秒)

7. $\forall x xe=x, \forall x e'x=x, \forall x\forall y\forall z\forall w (xy=z\wedge xy=w\rightarrow z=w)\rightarrow e=e'$　(61 秒)

括孤内は所要時間