

TR-115

演繹・帰納・発想の推論機構化をめざして

國藤 進

June, 1985

©1985, ICOT

ICOT

Mita Kokusai Bldg. 21F
4-28 Mita 1-Chome
Minato-ku Tokyo 108 Japan

(03) 456-3191~5
Telex ICOT J32964

Institute for New Generation Computer Technology

演繹・帰納・発想の 推論機構化をめざして

國藤 進 (ICOT)

1. はじめに

最近、学会・公官庁・産業界の一部に、創造科学あるいは創造学を育成・援助せんとする気運が盛りあがっている。なかでも東京創造性懇話会 [ISK 78] を設立母体とする日本創造学会が1979年10月に発足したこと、科学技術庁において「流動研究システムによる創造科学技術の推進」という制度が1982年に発足したこと、1990年代の知識情報処理システムのプロトタイプ確立をめざす、いわゆる第5世代コンピュータ・プロジェクト [NJK 82] が1982年に発足したことが、記憶に新しい。すなわち産官学各界の識者から、創造性・独創性問題の重要性が指摘されている。このような気運は既存の学会にも影響を与えつつあり、電子通信学会、日本オペレーションズ・リサーチ学会、日本認知科学会といった学会レベルでも創造性の理論・方法・技術の結合と創造性の研究育成発展への堅実な歩みが、漸進的ではあるが進展した。

産業界においては、古くからこの問題の重要性が認知されており、「1950年代は能率の時代、1960年代は生産性の時代、1970年代は創造性の時代」とも言われていた。1970年代のアプローチは主としてKJ法 [Kawakita 67, 70]、NM法 [Nakayama 68] などを中心とする技法的アプローチが中心であった。さらにこれより先1940年代に客観変換理論 [Ichikawa 60] という創造工学的アプローチが生誕していることは特筆に値する。これら諸技法を情報科学的見地から論じたのが川喜田・中山と北川との対話 [Kitagawa 71] である。

著者の所属するICOTは、第5世代コンピュータの開発を目的として、通産省によっ

て設立された財団法人である。ICOTでは1990年代における知識情報処理システムのプロトタイプ確立、すなわち問題解決推論、知識ベース管理、知的インタフェース、や知的プログラミングといった機能を備えたシステムの研究開発を行っている。このような機能をもつシステムのガソリンやエンジンに相当するのが、いわゆる知識ベースと推論機構と呼ばれるものである。このようなシステムが実現すれば、人間の知的問題解決活動、極言すれば人間の創造活動を支援するツールのひとつとして極めて有用であるが、ここではその実現に至る道程として、そのようなシステム実現のための基礎技術となりえる研究の動向を創造科学あるいは創造学の見地から概観することにする。

知識情報処理システム構築の気運の胎動は、歴史的に見ると、学界や産業界における人工知能研究の勃興あるいは見直しの動きと無縁ではない。周知のように人工知能研究の歴史は、1960年代前半はゲームとパズルの時代、1960年代後半は知能ロボットの時代、1970年代は言語と知識の時代 [Fuchi 78]、そして1980年代は知識工学と認知科学の時代といわれている。

知識工学や認知科学の現状をかいま見るに図1や図2に示されているように知識の表現、知識の利用、および知識の獲得 [Kunifuji 81, 83-1, 84-2] といったテーマが人工知能研究の主要研究課題として検討されている。著者らは、このうち最も研究が遅れているが大事な研究課題である知識の獲得に関する研究に関心を持っている。本稿では、知識獲得研究は創造科学の究極のテーマである創造する機械を創造するという研究と密接な関連をもつことを指摘していく。

2. 問題解決や推論のプロセスの本質

2.1 問題解決のプロセス

人間の問題解決や推論プロセスの本質を、

近い将来において機械において実行／模倣できるということを前提に、抽象的に把握してみる。人間の知的問題解決活動を「問題提起、現状把握、本質追及、問題評価、決断、構想計画、具体策、計画評価、手続化、実施、検証、総括」といった一連のプロセスの中でとらえるのが、川喜田らの「問題解決学」である〔Kawakita 70〕。問題解決学は、異質のデータ・情報を統合することによって新しい発想とアイデアを産む方法論であるKJ発想法を中心とする知的生産の技術を駆使して、人間が毎日遭遇する問題を問題として意識して、自らの課題としてとりあげ、解決に導いていく方法論を与えるものである。問題解決学を野外科学・書齋科学・実験科学をつなぐプラグマチックな方法論（図3）とみると、前半のⅤ（A～D部）を発想法のプロセス、真中のⅥ部を演繹法のプロセス、後半のⅤ（E～H部）を帰納法のプロセスとみなすことができることを喚起しておきたい。

2.2 推論のプロセス

人間の問題解決のプロセスを、最も厳密な論理学の体系の中で、初めてふれたのはアリストテレスの分析論〔Aristotle 71〕である。彼はその中で問題解決の論理的方法として、演繹法・帰納法・発想法という3つの推論プロセスを挙げている。

そのうち演繹法の体系は、その後中世に至るまで、学問の明晰な方法論として、順調すぎるほど順調に成長していった。他の2つの方法論は、長い間まどろみの時代を過ごした。近世になってベーコンやミルが帰納法の体系の再発見を行い、20世紀に入って推測統計学〔Kitagawa 68〕という学問の世界で帰納論理の急速な整備が行われた。一人取残されていた発想法の体系化については、19世紀後半から20世紀初頭にかけて生存した独創的な哲学者パースの活躍を待たねばならなかった。

2.3 パースの記号学

19世紀後半から20世紀初頭にかけて数多の著作を残したプラグマティストC.S.パースは、先駆的な記号学者として人間の探求活動の全過程を演繹論理、帰納論理、発想論理〔Yone-mori 81, Nakayama 79〕の立場から体系化した。彼の記号学によれば、人間の探求過程とはある未知の問題（驚くべき事実）に遭遇した人間が、その問題を説明する仮説を発見し（発想）、その仮説から導き出される合理的な帰結を推論し（演繹）、そして導出された帰結を検証する（帰納）過程である。

この探求の3プロセスは、実はそれぞれ前述のW型問題解決学におけるA～D部分、D～E部分、E～H部分にほかならない。

パースは論証プロセスを分析するに当たって、あくまで彼の三分法的カテゴリー体系の完成を意図しており、客観的に操作可能な実体である記号の体系のなかで説明していく。彼の記号学では論証を、解明的（必然的、分析）推論である演繹と、拡張的（蓋然的、総合）推論である帰納と発想に分類している。（図4）。彼はこの3者を分析するツールとして、第1次性（存在様式：性質、可能性）、第2次性（存在様式：個体、事実、現実性）、第3次性（存在様式：一般性、法則性）という現象学的カテゴリーを準備している。彼はさらに、記号それ自体のあり方、記号とその対象との関係におけるあり方、記号とその解釈内容との関係におけるあり方、という記号過程の3側面に注目し、表1のような記号概念の分類を行っている。これらの記号それぞれに対して、(i) いかなる記号も同じ縦線の記号と組合されてはならない、(ii) いかなる記号もその右の欄の下段にある記号とは結合しない、という三分法カテゴリー原理を導入し、10個のクラスの記号を指摘する。ついで記号の退化形式とか変型を細かく分析しつつ、最終的に66クラスの記号の種類を抽出している。この66クラスの記号で説明されるパースの記

号学は、パース自身「人間は記号である」という結論に到達したように、難解な哲学に陥ったため、パース哲学はごく一部の人を除いて注目されずに終ることになる。

3. 計算機による知識処理

3.1 処理対象の拡大

情報処理装置としての計算機システムの近年の発展は、その処理対象をマシンよりのものからヒューマンよりのものに急激に拡大しつつある[Winograd 79]。例えば初期の計算機システムは0、1という二進数の処理中心のものであったが、次第に数値処理やデータ処理向きの計算機システムが作られた。それに伴いユーザの使用する計算機言語も機械語、アセンブラ語、Fortran、Cobol、PL/Iへと発展してきた。1970年代、1980年代になって新たに記号処理や知識処理向きの計算機システムを要求するニーズが高まり、それに伴いユーザもLispやPrologといった記号処理言語や知識処理言語の使い易い高速な処理系を求めつつある。

3.2 ガソリンやエンジンの変遷

このような計算機システムの知識情報処理システム志向への発展の中で、計算機システムのガソリンやエンジンにあたる部分も新たな変容を遂げつつある。ガソリンにあたる部分は知識そのもので、エンジンにあたる部分は知識を処理する機構である。知識とそれを処理する機構の間には密接な関連がある。著者の立場は、知識を容れる容器を、次のような三種のモデルに分類する。第一のものがファクト型個別知識を格納するデータベースと呼ばれるもので、第二のものがルール型一般知識を格納する知識ベースと呼ばれるもので、そして第三のものが個別知識や一般知識の使い方に関するノウハウ型知識を格納するメタ知識ベースと呼ばれるものである。これら三種のモデルに対応し、著者はそれぞ

れ、データベースからの検索機構、知識ベースからの推論機構、メタ知識ベースからのメタ推論機構といった知識処理機構の必要性を喚起してきた。ソフトウェア的にみると、それぞれデータベース管理システム、知識ベース管理システムおよびメタ知識ベース管理システムとして実現できるものである。

3.3 アルゴリズム化の道

2.3で述べたような観点から、計算機による知識処理の本質を分析するに、著者は知的問題解決過程における人間の推論の本質である演繹、帰納、発想[Kunifuji 81,83-1, Arikawa 83,84,85]の計算機システム上での実現方式に注目したい。既存の計算機システム上で演繹的あるいは帰納的推論機構を実現するには、限られた論理の世界を前提とすれば、分解証明法[Robinson 65]あるいはモデル推論法[Shapiro 81,82]というアルゴリズムを用いればよいことが知られている[Kunifuji 83-1, Kitakami 84-1]。発想的推論機構実現のための統一原理は、現時点ではほとんど知られていないが、その中核をなす類推(analogical reasoning)については最近、原口[Haraguchi 84,85]が部分同型に基づく類推の理論を構築中であり、その機械化のための突破点がようやく見えてきたところである。そのアルゴリズムの詳細については、本書第 4章を参照されたい。

4. 演繹的推論の計算機処理

4.1 演繹論理

演繹的推論の計算機システム上での実現方式を述べる前に、まず演繹論理の枠組を明らかにしておく。

演繹論理の体系には一階/高階述語論理、一階/高階直観論理、内包論理、様相論理、多値論理、などの数多くの体系が存在する。

演繹論理には証明論とモデル論という二つのアプローチがあるが、われわれが従来から慣れ親しんできたのは証明論というアプローチである。証明論では演繹的推論を、公理と推論規則という概念を用いて定式化していく。表2に見られるように、公理と推論規則の数はそれぞれの論理で異なるが、十二分に整理された論理ではそのどれかが1である。

最も代表的な演繹論理が一階述語論理であるが、ここではGentzen型の一階述語論理 [Maehara 73] を例にとり説明していく。各 A_1, A_j を formula とするとき、“ A_1 かつ A_2 かつ…かつ A_n ” という仮定のもとで、“ B_1 あるいは B_2 あるいは…あるいは B_n ” が成立する”という概念を、“ $A_1, A_2, \dots, A_n \vdash B_1, B_2, \dots, B_n$ ” という記号の系列 (Sequenz と呼ぶ。) で表記する。Gentzen 型一階述語論理は、1個の公理と21個の推論規則をもつ。1個の公理とは“ $D \vdash D$ ”、すなわち“ D を仮定すれば、 D が成立する”である。21個の推論規則の中で、演繹論理の本質を最も簡明に表現しているのが三段論法 (syllogism) である。 $\Gamma, \Pi, \Delta, \Lambda$ を formula の列、 D を formula とするとき、三段論法は、“ Γ を仮定すれば、 Δ あるいは D が成立する”

と“ D かつ Π を仮定すれば、 Λ が成立する”とを前提とすれば、“ Γ かつ Π を仮定すれば、 Δ あるいは Λ が成立する”という帰結を得る」ことを推論する方法で、つぎのような推論図式で表わされる。

$$\frac{\Gamma \vdash \Delta \quad D, \Pi \vdash \Lambda}{\Gamma, \Pi \vdash \Delta, \Lambda}$$

4.2 分解証明法

4.1で与えた三段論法の形式を、計算機内で明確に実現できる形式で与えたのが、Robi-

nsonの分解証明法 [Robinson 65] である。分解証明法は、所与の一階述語論理命題が充足不能かどうかを定める部分的決定手続きである。すなわちその命題が充足不能なら、いつかはそのことを見出し停止する手続きである。分解証明法は、一階述語論理命題をある標準形式に変換し、導出原理 [Chang 73] と呼ばれる一種の記号演算を用いて推論していくシステムである。導出原理は前述の三段論法の一般化となっている。すなわち、三段論法では formula の引数である変数に定数を代入する操作しか認めていないのに、導出原理では formula の引数である項に他の項を代入する操作を認める。2つの項 t_1 と t_2 が、それぞれの項に含まれる変数の代入操作によって等しくなるとき (このことを統一置換可能というが)、その操作によって置換えられた formula の列のそれぞれに対して、その formula 列の右肩に ' をつけるものとする。すると導出原理は、つぎのような推論図式で表わされる。

$$\frac{\Gamma \vdash \Delta, D(t_1) \quad D(t_2), \Pi \vdash \Lambda}{\Gamma, \Pi \vdash \Delta, \Lambda'}$$

上式は、“ Γ を仮定すれば、 Δ あるいは $D(t_1)$ が成立”し、かつ“ $D(t_2)$ かつ Π を仮定すれば、 Λ が成立”し、かつ t_1 と t_2 が統一置換可能ならば、それぞれの記号列 $\Gamma, \Delta, \Pi, \Lambda$ に対する統一置換を施した記号列 $\Gamma', \Delta', \Pi', \Lambda'$ に対して、“ Γ' かつ Π' を仮定すれば、 Δ' あるいは Λ' が成立する”ことを推論する図式である。分解証明法においては、項同士のパターン照合による統一置換を繰返し行い、空節 (矛盾) を導き出すことを基本戦略としている。すなわち分解証明法は、計算機上にインプリメントする演繹的推論機構の基礎を提供している。

以上を要約すると、演繹法の背後には演繹論理があり、演繹論理の核となるのが、一階

述語論理であり、その形式化は種々の角度からなされているが、分解証明法が見出されて初めて、真に機械でインプリメント可能な演繹という記号処理演算の本質が導かれたということである。換言すると演繹的推論機構の原理が、分解証明法によって確立された。

4.3 論理プログラミングでの演繹

前述の分解証明法の研究を受けて、一階述語論理の定理証明機の研究がインテンシブに行われた。その結果、定理証明機の戦略や定理証明問題のもつ計算量の意味での複雑さについて種々のことが分った。残念ながら定理証明問題のもつ本質的難しさの故に、実用的なスピードで動く使い易い定理証明機は生まれなかった。そのような歴史的経緯を踏まえながらも、Colmerauer [Colmerauer 73] や Kowalski [Kowalski 74] らの努力により、一階述語論理の部分クラスであるホーン論理について、論理式のプログラミング言語としての解釈の発見により、ある程度実用的なスピードで動く使い易いプログラミング言語が生まれた。それが論理プログラミング言語 Prolog である。

Prolog による演繹的推論は、一階述語論理の部分クラスであるホーン論理に対して、上述の分解証明法の一つである Selective Negative Linear Resolution [Loveland 78] を適用したものである。たとえば、つぎのような古典的な三段論法の例題を考える。

$$\frac{\forall X (\text{man}(X) \supset \text{mortal}(X)) \quad \text{man}(\text{socrates})}{\therefore \text{mortal}(\text{socrates})}$$

- * (一般的知識/規則、大前提)
- ** (個別的知識/事実、小前提)
- *** (演繹的知識/事実、結論)

この例題は、「もし X が人間ならば、X は死を免れない」と「ソクラテスは人間である」という知識から、ゆえに「ソクラテスは死を免れない」という知識を推論すること

を意味している。これに対して Prolog では、つぎのような質問応答例が得られる。

```
mortal(X) :- man(X). (rule)
man(socrates). (fact)
| ?-mortal(socrates). (question)
yes (answer)
```

4.4 Prologでの演繹とその知識獲得への応用

Prolog による推論と一階述語論理での推論の根本的相違は、“Negation as Failure” [Clark 78] あるいは “Closed World Assumption” [Reiter 78] として知られている仮説を採用するかどうかである。このあたりの最近のトピックスについては、Shepherdson [Shepherdson 84] や Jaffer [Jaffer 83] を参照されたい。さて DEC-10 Prolog を中心とする Prolog 処理系は、本質的に与えられた知識ベース（公理系）に基づきある質問（定理）が証明可能かどうかを判定する。もしその質問が導かれる（証明可能である）ならば “yes” を返し、そうでなければ “no” を返す。すなわち Prolog 処理系は証明可能かどうかに関する結果のみ返す。

しかしながら定理証明機開発の歴史に見られるように、証明のプロセスを制御したり、プロセスそのものから必要なメタ情報を抽出したりすることが必要な場合がある。そのような場合、著者らは Prolog 処理系そのものの使い方に関する知識をも処理対象とするため、メタ推論という機構を導入した。メタ推論とは対象知識の使い方に関する知識を、プログラムでなくデータとして属に表現するにもかかわらず、必要に応じそのデータをプログラムとして呼出すことにより、ノウハウ型知識の利用を可能ならしめるものである。メタ推論機構実現にあたって、著者らは DEC-10 Prolog が内蔵している幾つかのシステムプログラミング用メタ述語を使用し、論理での証明可能性概念に相当する demo と呼ばれるメタ述

語を実現した。このdemo述語を用いメタ推論機構を実現する方式の検討、およびその各種応用領域での適用については、文献[Kunifuji 84, Kitakami 84-1]に詳しいので、ここでは省略する。ここではこのようなdemo述語が、知識ベースに知識を無矛盾かつ系統的に取組む知識同化の過程において、本質的に必要であることを指摘しておく。このdemo述語を用いれば、演繹的推論の世界で知識ベースの更新をはかる過程を支援する、メタ述語assimilateが実現できる[Bowen 82, Kunifuji 83-2, Miyachi 84-1, -2, Kitakami 83]。assimilate述語は知識ベースの内部矛盾を検出したり、知識ベースの冗長性を除去するのにdemo述語を使用している。

5. 帰納的推論の計算機処理

5.1 学習のモデル

従来人工知能研究の調査文献では、帰納的推論と発見的推論の区別が不明確である。例えば、人工知能ハンドブック第Ⅲ巻[Cohen 84]第XIV章「学習と帰納的推論」によれば、学習システムの議論を系統的に行うために、環境、学習部、知識ベース、および実行部からなる単純な学習モデル(図5)を設定している。

この学習モデルを用い、次のような4種類の学習状況の類別が行われている。

- ①暗記学習(Rote learning) 環境は実行タスクの水準で情報を正確に提供する。したがって、どんな仮説も必要でない。
- ②言われるままの学習(Learning by being told) 環境によって提供された情報はあまりに抽象的あるいは一般的なので、学習部は失われた詳細情報を仮説化しなければならない。
- ③例題からの学習(Learning from examples) 環境によって提供される情報があまりに固くかつ詳細なので、学習はより一般的な規則を仮説形成化しなければならない。

ない。

④類推による学習(Learning by analogy)

環境によって提供される情報は類似の実行タスクに対しても適切である。学習システムは現在の実行タスクに対して、ある類推を発見し、かつ類似の規則を仮説形成化しなければならない。

例題からの学習は、更に単一ステップの学習(単一概念の学習、複数概念の学習)と複数ステップの学習とに分類される。

著者らの見解によれば、このうち、①は演繹的推論のカテゴリー、②、③は帰納的推論のカテゴリーに属する。④は発見的推論のカテゴリーに属する。

5.2 帰納論理

残念なことに現在のところ計算機にインプリメントされた推論機構は、ほとんどフォーマルシステムにもとづく演繹的推論機構である。この種の推論機構(推論エンジンともいわれる)と知識ベースをもつ演繹的質問応答システムの研究は数多くあり、最近では知識情報処理システム研究の一環として行われることが多い。

このような人工知能学派の演繹志向に対して、渡辺は痛烈な批判[Watanabe 79]を投げかけている。渡辺[Watanabe 75]によると、帰納とは所与の経験的事実の集合から、その底に横たわる一般的規則についての知識を獲得する過程である。そのような過程を、既存計算機で利用可能な形式に変換するには、帰納論理の本質の記号学的形式化が必要である。ここではまず、帰納的一般化(inductive generalization)という帰納の論理図式を紹介する。

帰納とは何かについて諸説があるが、所与の(観測された)経験的知識、すなわち個別知識や演繹的知識、の集合から、その底に横たわる一般的知識を推論するプロセスと定義する。一般的知識を推論するにあたり、与

えられた経験的知識において、ことごとくその事例を列挙し、その知識を整理することから始める。たとえば個体 $x_1, \dots, x_n$ について述語 $p(x_1), \dots, p(x_n)$ が真であるということが経験（観測）されたとする。しかも $x_1, \dots, x_n$ がある有限集合 S ($S \subseteq \{x_1, \dots, x_n\}$) の元として特徴づけられ、このことを述語 $s(x)$ ($x \in S$) を用いて表記するものとする。このとき、つぎのような列挙された事例の整理を考える。

$$\frac{p(x_1), p(x_2), \dots, p(x_n)}{\forall x(x \in S \supset p(x))}$$

これに対して一般的知識を推論するにあたり、前提となる経験的知識において、ことごとくその事例を列挙しつくしていない推論図式を考える。すなわち必ずしも有限でない集合 T ($\supseteq S$) と、それを特徴づける述語 $t(x)$ ($x \in T$) を考えたとき、つぎのような推論図式を帰納的一般化という。

$$\frac{p(x_1), p(x_2), \dots, p(x_n)}{\forall x(x \in T \supset p(x))}$$

5.3 モデル推論アルゴリズム

5.1で述べたように帰納的推論の計算機処理向け研究としては、例題学習が最も良く調べられている。E. Shapiroは、彼の学位論文 [Shapiro 82] の中で、哲学者 K. Popper の“推測と反証”に関する方法論に基づき、事実（経験的知識）から理論（一般的知識）を帰納的に推論するモデル推論アルゴリズムを、つぎのような理論的枠組の中で提案した。

与えられた一階述語言語 L が節形式で表現されているとする。 L の文の 2 つの部分集合として、観測言語 L_0 と仮設言語 L_h を考える。ここに $\square \in L_0 \subseteq L_h \subseteq L$ ($\square$: 任意のモデル中で偽となる空文) と仮定する。また M を L のモデルとする。文集合 T ($\subseteq L_h$) が M の L_0 完全公理化である必要十分条件は、

T が M において真であり、かつモデル M において真となる観測文の集合 L_0^H に対して、 $T \vdash L_0^H$ ($\vdash$: 証明可能) となることである。与えられた観測文 α が、もしそれが M において真であれば “true” を返し、そうでなければ “false” を返すような、 L の未知なモデル M に対する oracle (神託) が存在し、かつ与えられていると仮定する。このときモデル推論問題は、ある有限な L_0 完全公理化を発見することである。

このモデル推論問題に対する彼の提案したモデル推論アルゴリズム [Shapiro 81] は、図6に示されているように、ファクト型知識（経験的知識）からルール型知識（一般的知識）を帰納的に推論する学習アルゴリズムである。その際、ユーザは誤りのないファクトを与えることが要求されるので、彼のシステムは完全教師付きの単一概念の学習システムといえる。

5.4 モデル推論システム

Shapiro は図6で与えられるモデル推論アルゴリズムをホーン論理に適用できるように洗練し、PrologのPrologによるPrologのための知的デバッガを開発した。彼の知的デバッガはモデル推論システムとも呼ばれるが、その中核をなすモデル推論アルゴリズムは矛盾を発見するプログラムと反証を与えた仮説を精密化するプログラムからなる。これら両プログラムは前述のdemo述語を用いてもインプリメントできる [Kitakami 84-1,2]。いずれにせよ、ホーン論理という制限された論理の世界で、論理プログラムのアルゴリズムック・デバッギングやその自動合成といった分野に適用し、彼のシステムは成功をおさめた。図7や図8に論理プログラムの自動合成適用例が示されている。

さて著者らは、前述の知識同化機構とShapiroのモデル推論システムに基づく知識調節機構の両者を中核にし、Prologによる知識獲

得支援システムを構築中である。知識獲得支援システムにおいては、知識の同化と調節の機構を用いて、知識ベースへの知識の獲得の管理を行っている。ここに知識の同化とは、知識ベースへ知識を無矛盾かつ系統的に取得する過程であった。知識調節とは、外から与えられた知識を説明する知識(モデル)を、知識ベース内に自動生成する機構である。この両機構を併用することにより、著者は知識ベースへの知識獲得支援システム [Kitakami 84-2] へと統合中である。知識同化と知識調節という考え方を統合する際、最も注意を払ったのは帰納的推論によって得られた論理プログラム(仮説)が正しいかどうかをユーザによって確認する作業である。現在はこの仮説確認実験をユーザの判断に基づいて行っている。この部分は、概念的には、仮説の検定 [Kitagawa 68, 83] を行っていることを留意しておきたい。北上の知識獲得システムの詳細については、本書第 4 章を参照されたい。

6. 発想的推論の計算機処理

6.1 発想と学習

発想とはある置くべき事実(困難な未知の問題)に遭遇した人間が、その事実を説明する(その問題を解決する)仮説を直観的に得る人間の思考過程である。前章で述べた帰納的推論は、ある関じた知識体系の中で極限としての同定 [Arikawa 83] をめざした仮説の生成を行っている。しかしながら本当に発想的推論と呼ばれるものでは、その知識体系に関する知識(ファクトとルール)が不十分なので、人間は種々の知的技法を用いて、何らかの意味のある仮説を生成している。意味のある仮説生成の仕方により、各種の発想的推論のパターンが分類される。

第一のものは類推と呼ばれ、他の既知の知識体系との間に何らかの翻訳・逆翻訳(創造工学の用語法では等価変換 [Ichikawa 60, 70]

と呼ばれる)を行いつつ、その事実を説明する仮説を生成する。このことを一階述語論理の用語法を用いて直観的に説明すると、発想とは経験的知識である結論と一般的知識である大前提とから、個別的知識にあたる小前提を導く推論図式である。結論を大前提に結び付けるには、一般に種々の等価変換理論 [Ichikawa 60, 70, Kunifuji 76] 的な写像関係の当てはめが必要である。この当てはめとは、一般的に言えば、未知の知識体系の既知の知識体系への写像関係の発見に他ならない。この写像関係を見出すのに、部分同型に基づく類比による思考形式が利用されているとするのが原口の類比の理論 [Haraguchi 84] である。この類比の理論をファクト型知識、ルール型知識の両者に対して適用したのが原口の類推の理論 [Haraguchi 85] である。ルール型知識との類推においては、前向き・後向きの双方向の推論プロセスが存在するのが興味深い。後向き類推プロセスは、アリストテレスによって指摘されていた還元法 (retroduction) という思考形式に相当する。この理論の基礎をなす所与の部分同型そのものを、どのようにして人間が見出しているかについては、現段階では全く未知の問題領域である。おそらく、連想に基づく知的なパターン照合機構等が活用されているのであろう。原口の類推の理論の詳細については、本書第 4 章を参照されたい。

第二のものはある種のマクロなメジャーを入れて(価値観を導入して)、意味のある仮説を選択する。Lenat の AH [Furukawa 81, Davis 82] がその代表例である。AH は、例えば初等整數論の“面白い”概念を発見する学習システムである。この“面白い”概念の定義は、数学的美観ともある程度一致しており、科学的発見の論理を説明しているところがあり、非常に興味深い。この種の研究は、知識の自己組織化研究として、今後、大いに発展することが期待される。

6.2 信念の翻意と発想

発想的推論機構に関する研究には、Kling [Kling 71] や McDermott [McDermott 70] の類推による学習の研究、Hajek の仮説生成機構の研究 [Hajek 78]、Langley [Langley 79] の規則性発見システム Bacon3 のインプリメンテーション、Morgan [Morgan 71] の仮説生成機構の研究、中島秀之の学習するパーサ Leap の研究 [本書 第 4 章] などが注目される。これら発想に関連する学習研究には、現状では共通の一般理論は存在しない。しかしながら、発想的推論を計算機処理するには、基本的には仮説生成機構を埋藏した計算機システムを構築すればよいことは分っている。

著者らの研究の立場はメタ知識に基づくメタ推論機能を用いて、この種の仮説生成機構を構築していくことが可能なことを示唆している。その際、不要にマージすると互いに矛盾することがありえるが、系統的に管理すれば矛盾しない複数の知識体系のそれぞれを独立に取扱わなければならない。このようなアクセスの仕方によっては互いに矛盾する知識体系を、全体としては無矛盾に管理するということは、人工知能研究の立場からすれば、信念システムを研究開発することに他ならない。信念システムの研究開発は知識ベースの更新を伴い、しかもその依り所となる論理は単調論理ではなく非単調論理となる。非単調論理の世界で信念の翻意をもサポートするシステムを構築するには、その出発点として Reason Maintenance System [Shintani 84, Shirai 83] や Truth Maintenance System [Doyle 78, 79] の研究を行わなければならない。非単調論理に基づく証明システム Truth Maintenance System を開発中の Doyle らの形式化 [Doyle 78, 79] によると、公理と（現在の信念を表わす）仮説の集合から演繹される定理、およびその最小不動点という概念が中心的役割をはたす。非単調論理の本質は、不完

全知識下で仮説を修正していく過程を、これまでの演繹論理の拡張の範囲内でとらえる立場である。これに対して同様の条件下で、公理そのものの修正を許容する論証過程を形式化する研究は、ほとんど行われていない。これら仮説や公理の修正を許容する論理と帰納論理、発想論理との関連を調べてみるのが、論理の立場で知識情報処理システムへ接近する正攻法の一つとなるであろう。

本稿で著者らが述べてきた研究開発の立場からいえば、知識の同化や調節を行う機能のみならず、知識の均衡化機能をもつシステムを構築しなければならない。ここに知識の均衡化とは、知識ベースに対する同化と調節の過程を繰り返す中で、その知識ベースが外部環境に適應していく過程である。このような知識均衡化機構の研究は、現在その端緒についたばかり [Kitakami 84-2] であるが、今後発想的推論機構の計算機処理研究のシーズとして、大いにその成果が期待される分野である。

7. おわりに

本稿では計算機による知識処理研究の現状を、人工知能研究の立場から鳥瞰した。その結果、人間の探求過程の本質を演繹、帰納、発想という三種の推論活動の合理的適用と捉え、その既存計算機システム上での実装方式の検討を行った。演繹的推論の機械化にとっては Robinson の分解証明法が、帰納的推論の機械化にとっては Shapiro のモデル推論アルゴリズムが、現在までに得られた最も美しい成果であることを指摘した。発想的推論の機械化研究は、人工知能研究者にとって最も困難であるが魅惑的な研究領域である。これについては類推に関する若干の研究 [Haraguchi 84, 85, Arikawa 84, 85] が指摘されている以外、みるべき成果を挙げていない。

以上述べてきた各種推論機構を論理型言語 Prolog 上で実現するに当たって、メタ知識に基

づくメタ推論機構実現の重要性を喚起してきた。メタ推論機構を用い、知識の同化、調整、均衡化機構研究 [Kitakami 84-2, Kunifuji 84-2] が着実に前進しつつあることも述べた。しかもこれらが前述の演繹、帰納、発想的推論の機構化研究とも密接に関連していることを強調した。著者らの見解によれば、演繹的推論、帰納的推論に対して大幅に遅れている発想的推論の機構化研究を推進するのに知識均衡化機構の研究が必要であり、またその時機が到来しつつあることを指摘したい。

参 考 文 献

- (Arikawa 83) 有川節夫, 帰納的推論, 先進的コンピュータに関する調査研究報告書 [技術動向編], 日経連 & ICOT (編), pp. 56-69, 1983年 3月.
- (Arikawa 84) 有川節夫, 帰納的推論と類推の応用, 昭和58年度先進的コンピュータに関する調査研究報告書 [技術動向編] - II, 日経連 & ICOT (編), pp. 25-33, 1984年 3月.
- (Arikawa 85) 有川節夫, 将来への展望 - 類推を中心として -, 昭和59年度先進的コンピュータに関する調査研究報告書 [技術動向編] - III, 日経連 & ICOT (編), pp. 34 - 48, 1985年 3月.
- (Aristotle 71) アリストテレス著、井上忠訳: アリストテレス全集 1, 2, 岩波書店 (1971)
- (AI 80) Artificial Intelligence-Special Issue on Non-Monotonic Logic, 13-1, 2, North-Holland (1980)
- (Bowen 82) Bowen, K.A., Kowalski, R. A. Amalgamating Language and Meta-Language in Logic Programming (K.L.Clark and S.-A.Taarnlund eds.), Academic Press, pp. 153- 172, 1982.
- (Chang 73) Chang, C-L. and Lee, R.C-T.: Symbolic Logic and Mechanical Theorem Proving, Academic Press (1973)
- (Clark 78) Clark, K.L., Negation as Failure, in Logic and Data Bases (H. Gallaire and J.Hinker, eds.), Plenum Press, New York London, pp. 293-322, 1978.
- (Cohen 84) Cohen, P.R., Feigenbaum, E.A. (eds.) (田中幸吉, 刈一博監訳), 人工知能ハンドブック第Ⅲ巻, 共立出版, 1984.
- (Colmerauer 73) Colmerauer, A., Kanoui, H., Pasero, R., Raussel, P., Un Systeme de Communication Homme-machin en Francais. Rapport, Groupe Intelligence Artificielle, Universite d'Aix Marseille, Luminy.
- (Davis 82) Davis, R., Lenat, D.B., Knowledge-Based Systems in Artificial Intelligence, McGraw-Hill, Inc., 1982.
- (DTG 80) 電子通信学会教育技術専門委員会 (編): 電子通信学会技術研究報告、教育技術研究会、テーマ創造性 -、機械盛興会館、1980年 8月
- (Doyle 78) Doyle, J., Truth Maintenance System for Problem Solving, MIT Technical Report AI-TR-419 1978.
- (Doyle 79) Doyle, J., Truth Maintenance System, Artificial Intelligence, Vol. 12, No.3 1979.
- (Fuchi 78) 刈一博, 問題解決と推論, 情報処理, 19, 10, 情報処理学会誌, 936-943, Oct. 1978.
- (Furukawa 81) 古川康一, マシンインテリジェンスと知識ベース, 電子通信学会会誌, 64, 9, pp.952-958, 昭和56年 9月.
- (Furukawa 83) Furukawa, K., Takeuchi, A., and Kunifuji, S., Handala: A Knowledge Representation System in Concurrent Prolog, Information Processing Society

of Japan, Preprints of WG on Knowledge Engineering and Artificial Intelligence, No v. 1983.

(Furukawa 84-1) 古川康一, 竹内彰一, 國藤 進, Mandala, A unified system for modular programming and knowledge representation on Concurrent Prolog, 京都大学数理解析研究所, 「情報の構造化と意味に関する研究」研究集会, 1984年 2月.

(Furukawa 84-2) 古川康一, 國藤 進, 竹内彰一, 上田和記, 核言語第1版概念仕様書, 新世代コンピュータ技術開発機構, ICOT TR-054, March 1984.

(Furukawa 84-3) Furukawa, K., Takeuchi, A., Kunifuji, S., Yasukawa, H., Ohki, M., and Ueda, K.: Mandala: A Logic Based Knowledge Programming System, Proc. of FGC S'84, pp. 613-622, Oct. 1984

(Gallaire 78) Gallaire, H. & Hinker, J.: Logic and Data Bases, Plenum Press, New York - London, 1978

(Hajek 78) Hajek, P. & Havranek, T.: Mechanizing Hypothesis Formation, Springer-Verlag, Berlin - Heidelberg - New York, 1978

(Haraguchi 84) Haraguchi, M., An Analogy as a Partial Identity, Proc. of the LPC'84, ICOT, March 1984.

(Haraguchi 85) 原口誠, ルールの交換による類推の逆向き推論について, Proc. of the LPC'85, ICOT, July 1984.

(Hatano 74) 波多野編: ピアジェの発達

心理学, 国土社, 1974

(Ichikawa 60) 市川亀久彌, 独創的研究の方法論, 三和書房, 1960.

(Ichikawa 70) 市川亀久彌, 創造性の科学, 日本放送出版協会, 1970.

(Jaffer 83) Jaffer, J., Lassez, J. L., Completeness of the Negation as Failure Rule, TR 83/1, Dep. of Computer Science, The Univ. of Melbourne, 1983.

(Kawakita 67) 川喜田二郎, 発想法, 中公新書(1967)

(Kawakita 70) 川喜田二郎, 統発想法, 中公新書(1970)

(Kawakita 70) 川喜田二郎, 牧島信一編著: 問題解決学-KJ法ワークブック, 講談社(1970)

(Kitagawa 68) 北川敏男, 統計学の認識, 白揚社, 1968.

(Kitagawa 69) 北川敏男: 情報学の論理, 講談社現代新書, 1969.

(Kitagawa 71) 北川敏男編: 創造工学, 中公新書, 1971.

(Kitagawa 79) 北川敏男: 情報と認識 (NHK 大学講座, 1979年10月~1980年 3月), 日本放送出版協会, 1979.

(Kitagawa 81) 北川敏男: 情報関係圖の理論, 富士通・国際情報社会科学研究所研究報告第1号, Jan. 1981.

(Kitagawa 83) 北川敏男, 知識情報処理システム(KIPS)への情報学的接近, 富士通(株)国際情報社会科学研究報告第10号, 昭和58年 3月.

(Kitakami 83) 北上 始, 麻生盛敏, 國藤進, 宮地泰造, 古川康一, 知識獲得機構の一実現法, 情報処理学会知識工学と人工知能研究会資料30-2, 1983年 6月.

(Kitakami 84-1) Kitakami, H., Kunifuji, S., Miyachi, T., and Furukawa, K., A Methodology for Implementation of A Knowledge Acquisition System, Proc. of the 1984 International Symposium on Logic Programming, Atlantic City, U.S.A., Feb. 6-9, 1984, also available from ICOT as ICOT Technical Report TR-037 1984.

(Kitakami 84-2) 北上 始, 國藤進, 宮地泰造, 古川康一, 大規模な知識ベース管理システムのアーキテクチャ, 知識理解システム・シンポジウム, 富士通(株)国際情報社会科学研究報告, 昭和58年7月.

(Kling 71) Kling, R.E.: A Paradigm for Reasoning by Analogy, Artificial Intelligence, 2 (1971)

(Kowalski 74) Kowalski, R.A.: Predicate Logic as Programming Language, Proc. of IFIP 74, pp. 569-574, North-Holland, 1974.

(Kowalski 79) Kowalski, R.A.: Logic for Problem Solving, North Holland (1979)

(Kunifuji 76) 國藤進: 等価変換論の立場からのブラバズルNo.5のコンピュータによる解法と人間による解法の比較, 創造性研

究, 東京創造性懇話会誌, Oct. 1976.

(Kunifuji 81) 國藤進, 知識情報処理システムから創造科学へ, オペレーションズ・リサーチ, pp. 261-268, 1981年 5月号.

(Kunifuji 82) Kunifuji, S. and Yokota, H., PROLOG and Relational Data Bases for Fifth Generation Computer Systems, Proc. of CERT Workshop on "Logic Base for Databases", DEC. 1982(TR-002).

(Kunifuji 83 -1) 國藤進, 問題解決と推論, 計測と制御, Vol. 22, No 1, 昭和58年 1月, pp. 153-159.

(Kunifuji 83 -2) 國藤進, 麻生盛敏, 竹内彰一, 宮地泰造, 北上 始, 横田治夫, 安川秀樹, 古川康一, Prologによる対象知識とメタ知識の融合とその応用, 情報処理学会知識工学と人工知能研究会資料30-1, 1983年 6月(TR-009).

(Kunifuji 84 -1) 國藤進, 北上 始, 宮地泰造, 竹内彰一, 横田治夫, 古川康一, 論理型言語によるメタ推論とその応用, 京都大学数理解析研究所, 「情報の構造化と意味に関する研究」研究集会, 1984年 2月.

(Kunifuji 84 -2) 國藤進, 北上 始, 宮地泰造, 古川康一: 論理型言語Prologによる知識ベースの管理, Proc. of the LPC'85, ICOT, 1985.

(Langley 79) Langley, P.W.: Rediscovering Physics with BACON3, 6IJCAI, Tokyo (1979)

(Loveland 78) Loveland, D.W., Automated Theorem Proving: A Logical Basis, Nort

h-Holland, Amsterdam New York Oxford, 1978.

(Maehara 73) 前原昭二：数理論理学－数学的理論の論理的構造，培風館(1973)

(McDermott 79) McDermott, J.: Learning to Use Analogies, 61JCAI Tokyo (1979)

(McDermott 79) McDermott, D. & Doyle, J.: An Introduction to Non-Monotonic Logic, Proceedings of the 6th International Joint Conference on Artificial Intelligence, Tokyo, 1979.

(Miyachi 84-1) Miyachi, T., Kunifuji, S., Kitakami, H., Furukawa, K., Takeuchi, A., and Yokota, H., A Knowledge Assimilation Method for Logic Databases, Proc. of the 1984 International Symposium on Logic Programming, Atlantic City, U.S.A., Feb 6-9, 1984.

(Miyachi 84-2) Miyachi, T., Kunifuji, S., Kitakami, H., and Furukawa, K., A Constraint-based Dynamic Semantic Model for Logic Databases, Institute for New Generation Computer Technology, ICOT TR-0056, April 1984.

(Morgan 71) C.G.Morgan: Hypothesis Generation by Machine, Artificial Intelligence, 2 (1971)

(Nakayama 68) 中山正和：カンの構造，中公新書，1968.

(Nakayama 79) 中山正和：演繹・帰納仮説設定－新しい科学の構造－，産業能率大学出版部，1979.

(NJK 82) 日本情報処理開発協会編：第5世代コンピューター研究開発計画，5G-R011，(財)日本情報処理開発協会(1982)

(Ohsuga 80) Ohsuga, S.: Perspective on New Computer Systems of the Next Generation-A Proposal for Knowledge-Based Systems, Journal of Information Processing, Vol.3, No.3, 1980.

(Popper 68) K.R.Popper: Conjectures and Refutations-The Growth of Scientific Knowledge, Harper Torch Books (1968)

(Reiter 78) Reiter, R., On Closed World Databases, In Logic and Data Bases (H.Gallaire and J.Minker, eds.), Plenum Press, New York London, pp.55-76, 1978.

(Robinson 65) Robinson, J., A Machine-Oriented Logic Based on the Resolution Principle, JACH, 12, 1965.

(Shapiro 81) Shapiro, E.Y., Inductive Inference of Theories From Facts, Yale University Research Report 192, 1981.

(Shapiro 82) Shapiro, E.Y., Algorithmic Program Debugging, ACM Distinguished Dissertations, The MIT Press, 1982.

(Shepherdson 84) Shepherdson, J.C., Negation as Failure: A Comparison of Clark's Completed Data Base and Reiter's Closed World Assumption, J. Logic Programming 1, 1, pp. 51-79, 1984.

(Shimura 80) 志村正高：知識の獲得と学習，電子通信学会技術研究報告，オートマン

と言語研究会, AL80-46, Dec. 1980.

A Twenty-Year View, Edited by M.L. Dertouzos and J. Moses, MIT Press, 1980.

(Shintani 84) 新谷虎松, 知識情報処理における知識ベースシステムの設計, 富士通(株)国際情報社会科学研究所研究報告第13号, 昭和59年 1月.

(Shirai 83) 白井英俊, RMS(理由保持機構)を用いた知識表現システム, 情報処理学会第27回全国大会 1983.

(Sloman 71) Sloman, A.: Interactions between Philosophy and Artificial Intelligence, Artificial Intelligence, Vol.2, 1971.

(TSK 78) 東京創造性懇話会(編): 東京創造性懇話会第4回年次大会予稿集, 東京工業大学, 1978年10月.

(Yokota 84) Yokota, H., Kunifuji, S., Kakuta, T., Miyazaki, N., Shibayama, S., and Murakami, K., An Enhanced Inference Mechanism for Generating Relational Algebra Queries, Proc. of Third ACM SIGACT-SIGMOD Symposium on Principles of Database Systems, Waterloo, Canada, April 2-4, 1984.

(Yonemori 81) 米盛裕二, パースの記号学, 勁草書房, 1981.

(Watanabe 75) 渡辺 慧: 知識と推測, 2. 演繹と帰納の数理, 東京図書, 1975.

(Watanabe 79) 渡辺 慧: "人工知能"の挫折と救済, 自然, Oct. 1979.

(Winograd 79) Winograd, T., Toward Convivial Computing, in The Computer Age:

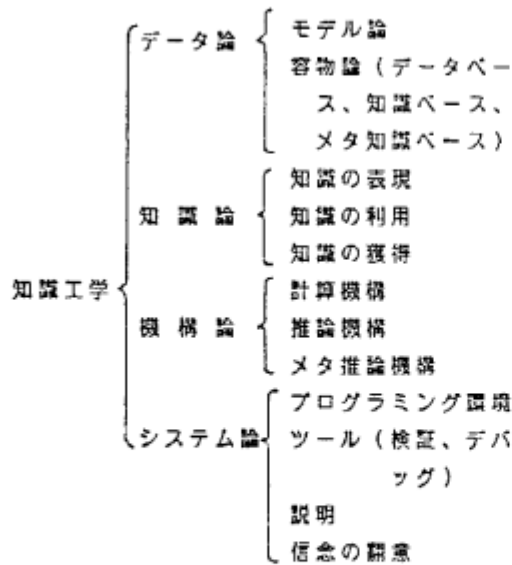


図1 知識工学の諸領域

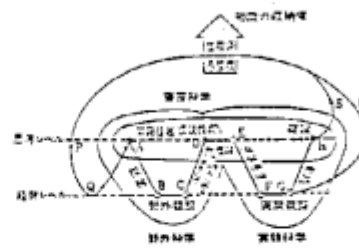


図3 W型問題解決学 [Kawakita 70]

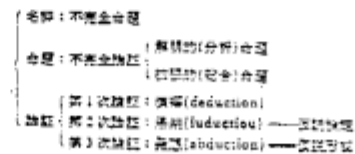


図4 パース記号学における名辞・命題・論証 [Yonemori 81]

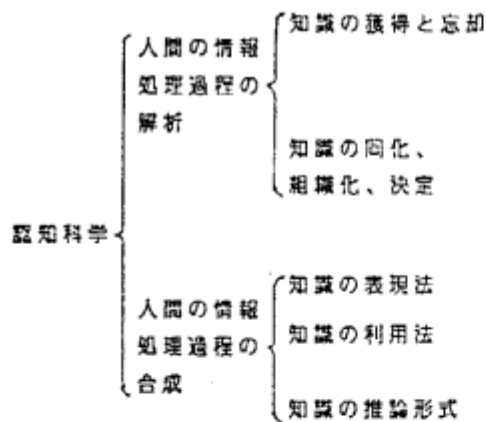


図2 認知科学の諸領域

	記号の第一性	記号の第二性	記号の第三性
第1次性	性質記号	関係記号	名 辞
Firstness	quali-sign	icon	rheme/term
第2次性	識別記号	指示記号	命 題
Secondness	sign-sign	index	rheme-sign (proposition)
第3次性	論理記号	論理記号	論 証
Thirdness	legi-sign	symbol	argument

表1 パース記号学における記号の分類 [Yonemori 81]

	公理	推論規則
1. 等しい性質 (Gentzen 型)	1	21
1. 等しい性質 (Muehle 型)	5	1
1. 等しい性質 (Gentzen 型)	1	12
内 容 数	8	1

表 2 公理と推論規則の個数の実例

1	1. 等しい性質
2	2. 等しい性質
3	3. 等しい性質
4	4. 等しい性質
5	5. 等しい性質
6	6. 等しい性質
7	7. 等しい性質
8	8. 等しい性質
9	9. 等しい性質
10	10. 等しい性質
11	11. 等しい性質
12	12. 等しい性質
13	13. 等しい性質
14	14. 等しい性質
15	15. 等しい性質
16	16. 等しい性質
17	17. 等しい性質
18	18. 等しい性質
19	19. 等しい性質
20	20. 等しい性質
21	21. 等しい性質
22	22. 等しい性質
23	23. 等しい性質
24	24. 等しい性質
25	25. 等しい性質
26	26. 等しい性質
27	27. 等しい性質
28	28. 等しい性質
29	29. 等しい性質
30	30. 等しい性質
31	31. 等しい性質
32	32. 等しい性質
33	33. 等しい性質
34	34. 等しい性質
35	35. 等しい性質
36	36. 等しい性質
37	37. 等しい性質
38	38. 等しい性質
39	39. 等しい性質
40	40. 等しい性質
41	41. 等しい性質
42	42. 等しい性質
43	43. 等しい性質
44	44. 等しい性質
45	45. 等しい性質
46	46. 等しい性質
47	47. 等しい性質
48	48. 等しい性質
49	49. 等しい性質
50	50. 等しい性質
51	51. 等しい性質
52	52. 等しい性質
53	53. 等しい性質
54	54. 等しい性質
55	55. 等しい性質
56	56. 等しい性質
57	57. 等しい性質
58	58. 等しい性質
59	59. 等しい性質
60	60. 等しい性質
61	61. 等しい性質
62	62. 等しい性質
63	63. 等しい性質
64	64. 等しい性質
65	65. 等しい性質
66	66. 等しい性質
67	67. 等しい性質
68	68. 等しい性質
69	69. 等しい性質
70	70. 等しい性質
71	71. 等しい性質
72	72. 等しい性質
73	73. 等しい性質
74	74. 等しい性質
75	75. 等しい性質
76	76. 等しい性質
77	77. 等しい性質
78	78. 等しい性質
79	79. 等しい性質
80	80. 等しい性質
81	81. 等しい性質
82	82. 等しい性質
83	83. 等しい性質
84	84. 等しい性質
85	85. 等しい性質
86	86. 等しい性質
87	87. 等しい性質
88	88. 等しい性質
89	89. 等しい性質
90	90. 等しい性質
91	91. 等しい性質
92	92. 等しい性質
93	93. 等しい性質
94	94. 等しい性質
95	95. 等しい性質
96	96. 等しい性質
97	97. 等しい性質
98	98. 等しい性質
99	99. 等しい性質
100	100. 等しい性質

図 7 算術式の自動合成例

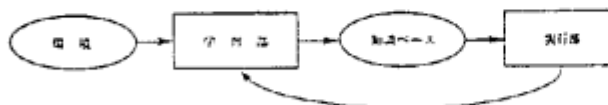


図 5 学習モデル [Cohen 84]

1	1. 等しい性質
2	2. 等しい性質
3	3. 等しい性質
4	4. 等しい性質
5	5. 等しい性質
6	6. 等しい性質
7	7. 等しい性質
8	8. 等しい性質
9	9. 等しい性質
10	10. 等しい性質
11	11. 等しい性質
12	12. 等しい性質
13	13. 等しい性質
14	14. 等しい性質
15	15. 等しい性質
16	16. 等しい性質
17	17. 等しい性質
18	18. 等しい性質
19	19. 等しい性質
20	20. 等しい性質
21	21. 等しい性質
22	22. 等しい性質
23	23. 等しい性質
24	24. 等しい性質
25	25. 等しい性質
26	26. 等しい性質
27	27. 等しい性質
28	28. 等しい性質
29	29. 等しい性質
30	30. 等しい性質
31	31. 等しい性質
32	32. 等しい性質
33	33. 等しい性質
34	34. 等しい性質
35	35. 等しい性質
36	36. 等しい性質
37	37. 等しい性質
38	38. 等しい性質
39	39. 等しい性質
40	40. 等しい性質
41	41. 等しい性質
42	42. 等しい性質
43	43. 等しい性質
44	44. 等しい性質
45	45. 等しい性質
46	46. 等しい性質
47	47. 等しい性質
48	48. 等しい性質
49	49. 等しい性質
50	50. 等しい性質
51	51. 等しい性質
52	52. 等しい性質
53	53. 等しい性質
54	54. 等しい性質
55	55. 等しい性質
56	56. 等しい性質
57	57. 等しい性質
58	58. 等しい性質
59	59. 等しい性質
60	60. 等しい性質
61	61. 等しい性質
62	62. 等しい性質
63	63. 等しい性質
64	64. 等しい性質
65	65. 等しい性質
66	66. 等しい性質
67	67. 等しい性質
68	68. 等しい性質
69	69. 等しい性質
70	70. 等しい性質
71	71. 等しい性質
72	72. 等しい性質
73	73. 等しい性質
74	74. 等しい性質
75	75. 等しい性質
76	76. 等しい性質
77	77. 等しい性質
78	78. 等しい性質
79	79. 等しい性質
80	80. 等しい性質
81	81. 等しい性質
82	82. 等しい性質
83	83. 等しい性質
84	84. 等しい性質
85	85. 等しい性質
86	86. 等しい性質
87	87. 等しい性質
88	88. 等しい性質
89	89. 等しい性質
90	90. 等しい性質
91	91. 等しい性質
92	92. 等しい性質
93	93. 等しい性質
94	94. 等しい性質
95	95. 等しい性質
96	96. 等しい性質
97	97. 等しい性質
98	98. 等しい性質
99	99. 等しい性質
100	100. 等しい性質

図 8 論理プログラムの自動合成例

それを (二) と見なす。

Repeat

その結果を試す。

Repeat

While 条件が真である **do**

条件が真アルゴリズムを実行し、

その結果を試す。

While 条件が真である **do**

条件が真アルゴリズムを実行し、

その結果を試す。

Until 条件が真であることまで

繰り返す。

(繰り返すことが必要になるまで)

forever 条件が真であることまで

図 6. モデル推論アルゴリズム
[Shapiro 81]