

知的マン・マシン・システム

大須賀 節 雄

Intelligent Man-Machine Systems

By

Setsuo OHSUGA

Abstract: This paper presents a theoretical basis for the intelligent, interactive information systems with which non-programmer user can communicate in man-machine environments as if it were the human able assistant. A set of requirements for such the system is first listed up and discussed. Then the system named the KAU (Knowledge Acquisition and Utilization) system is presented as the one that satisfies most of them, of which representation of knowledge, the deductive inference algorithm, capabilities for generating both programs and data base access procedures are described somewhat in detail.

摘 要

この論文は、計算機の非専門家である一般のユーザがマン・マシン系という環境のもとで、あたかも有能な人間の秘書のように利用することのできる知的機能を備えた計算機システムについて述べるのが目的である。本論文においては、まずこのようなシステムが満たすべき条件を列挙し、次いでこれらの多くを満たすものとして開発が進められている KAU (Knowledge Acquisition and Utilization) システムと名づけられたシステムについて述べる。特に知識表現、機械的推論処理、プログラム生成およびデータベース・アクセス手順生成機能についてやや詳細に述べる。

1. はじめに

この論文では言語を中心とし通常われわれが用いている情報表現によって直接に計算機とコミュニケーションすること、そのために計算機に要求される情報の理解能力について論ずる。理解という行為は知識の存在を前提として成立するものである。具体的には外界から入力し、処理した情報と、記憶している事実等に関する情報との意味的一致のとれることが、情報の理解の基本である。意味的な一致をとるには意味の同一性を何らかの形式で表現することがまず必要である。また、理解という機能を備えた機械が実際の場で役立つための条件は、理解すべき対象が多数存在するような状況のもとでも上述の一致がとれることであり、それに

は多数の知識の蓄積技術と、その中から意味の一致するものを速やかに検索する技術が必要とされる。このような機能を備えたシステムのもとでは、外部（入力）情報と一致した内部情報（すなわち知識）のみならず、さらにその知識と意味的に一致する別の知識の探索あるいは、その情報が表わす要求を満たす手続きの生成という一連の動作が行われることを意味する。そして最終的には外部入力に対応して、計算機が独自の判断によって最も適切な応答を行うことができる。これが単に入力情報を形態的特徴のみで処理してきたこれまでの計算機利用方式と全く異なる理解機能の特徴である。

理解という機能、あるいはその根底にある知識の表現、蓄積、利用技術に関して言えば言語情報とかパターン情報などという外部表現形式の相違は本質的なものではないが、意味を表現する手段として、通常われわれが言語を用いる比率が多く、言語という形式と、それによって表わされる意味とが時として不可分であるという事実のために、知識の表現や記憶に関して言語情報との結びつきが強い。

言語との関係と同時に計算機内での知識を理解する上で重要なことは、いわゆるデータと知識との関係である。データも一定の約束のもとで何らかの意味を表わす情報であるから、この限りでは知識とデータは同じものといえる。それならデータは知識そのものであるかどうか。それとも知識性の強い情報と薄い情報という区別があるのだろうか。あるとしたらそれはどういう点で異なるのだろうか。知識を論ずる以上、これらは最も基本的な問題であり、明らかにされねばならないものである。

現在、知識に関する諸問題は人工知能の中心的課題として、米国を中心に多くの研究が進められている。この分野はまだ発展途上にあり、知識表現一つをとっても多様なものが提案されている。知識を扱ったものとしては Winograd による、自然言語を理解し、限定された積木の世界内で計算機が入力に正しく反応するシステム [24] が初期のものとして有名である。これは知識を手続きによって表現する方法を用いている。理解ということも前述のように計算機が入力に対して正しく反応するまでのプロセスとするなら、あるいは少なくとも反応から理解の正しさを判定しようとするなら、入力情報と手続きとの対応がどこかでなされなければならないし、手続きによる表現方法というのはそれなりの正当性を含んでいる。しかし一方ではこの方法は拡張性が乏しい点で不十分である。この後に提案された多くの方法は（いずれどこかの段階で手続きに変換されることを前提として）知識の表現自体は非手続き的に行うものが多くなっている。この代表的な手法のひとつに、概念間の関係をグラフ（データ構造）によって表現することにより知識をネットワークで表現する方法がある。この方法はセマンティック・ネットと呼ばれており、Quilliam [19] 以後広く用いられている [4], [11], [21], [23]。これと同様に広く知られている方法が伝統的な論理表現、すなわち一階述語を用いて知識を表わす方法である。論理は言語理解の骨格とも言えるもので、これを形式化しているから、知識表現の方法としてかなり早期から用いられてきた。特に理解の中核にあると考えられる推論方式について、1965年に Resolution principle と呼ばれる機械的推論の理論が J. A. Robinson [9] によって発表されて以来、多く用いられるようになった。

この二つの方法は知識表現の形式としては全く異なっていて、前述の手続き的方法と共に

いわば知識表現の原形のようなものであり、この他の多くの方法に何らかの形で含まれている。たとえば OWL は知識の表現と言語の結びつきを重視し、言語を知識表現の骨格とするという基本構想に基づくシステムであるが、概念間の階層関係を基本構造とするセマンティック・ネットであり、概念の組で知識が表現される。

この他、現在までに発表された知識表現としてはすべての行為を有限の基本動作の組合せで表現する Conceptual Dependency [1], [8]; 前提と結論の組で表わされた有限の production の組を用いて変換を行うことによって意味の処理を行う Production Systems, 手続き的な知識をも非手続き的形式で表現できる統一的体系を作ることがを意図し、“(実体に関する各種の記述を行う為の) descriptor により互いに関連づけられた実体”という形式を有し検索をも考慮してそれらを構造化する KRL (Knowledge Representation Language) その他がある。

以下、本稿ではマン・マシン系のもとで知識を扱うためのシステムの条件、及びそれを具体化する方式について述べる。このような環境を想定するのはコミュニケーションとは情報の“形式”を通して“意味”を伝達するプロセスであり、知識体系に関して最も基本的な問題である意味の定義や、形式と意味の対応および変換などはすべてコミュニケーションの場で生ずること、人の場合、思考プロセスを直接観察することはできず入力情報に対する応答により理解が正しいか否かが判定されるが、計算機の場合も同様であり同一の基準を機械に対しても適用するにはマン・マシンとして考える他ないこと等の理由による。事実、知識というものはコミュニケーションを通して発展してきたものであり、コミュニケーション手段としての言語によって明確に記述される。すでに述べた言語と知識の不可分の関係もここから生じている。したがって以下で述べるマン・マシン系は既成の技術の枠にとらわれない広義のものであり、計算機が人間とほぼ対等に近い立場で同一の表現方法を用いて会話をしながら人間を援助することができると同時に物理的な性能面でも実用に耐えるという条件を設定する。

2. マン・マシン・コミュニケーションの成立条件

マン・マシン系では各サブシステムである人と計算機とがいずれも入力-処理-出力という一貫した処理機能を有することが必要であるが、人間と同じように計算機を使えるという目的から、計算機の論理的な性能としては、

〔A〕 表現形式は人間同志の間の表現に近いものであり、さらに人の要求にこたえる知的能力を有すること

が必要である。一方、計算機の機構は人間のものとは全く異なるから

〔B〕 情報の内部表現形式および処理形式は計算機にとって最も効率のよい形式とし、それによって記憶容量、応答時間、コストなどの点で実現し得るものとする

が必要である。

コミュニケーションとは2つ以上の、それぞれが知的機能を有する実体間で知識の伝達をすることである。マン・マシン・システムという立場において、人は計算機に前もって貯えられた情報を必要に応じて入手できること、および計算機が一定の動作を行うことによって

彼の仕事を援助することを期待している。この際、満足できる結果を得るためには人は自己の欲することを計算機に与えることが必要である。従来のマン・マシン・システムではこの機能が貧弱であり、これがシステムの利用度を著しく制約していた。

人が自分の欲することを完全に記述するには相当量の情報を必要とする。しばしば、これは期待している結果（応答）の情報量より大である。人間同志で会話が成立するのは、実際には両者が共通の知識を有していて、話の受け手が自己の知識により話者の情報を補うこと、それによって記述に必要な努力が大幅に軽減されているからと考えられる。このことはコミュニケーションは知識を有する実体間で始めて成立するものであることを意味する。計算機の場合とて例外ではない。すなわち、マン・マシン・システムを成立させるには、計算機は知識を保有し、それをコミュニケーションに利用できるものであらねばならない。このことはマン・マシン・システムの基本技術として知識の表現・蓄積・利用が含まれることを示す。

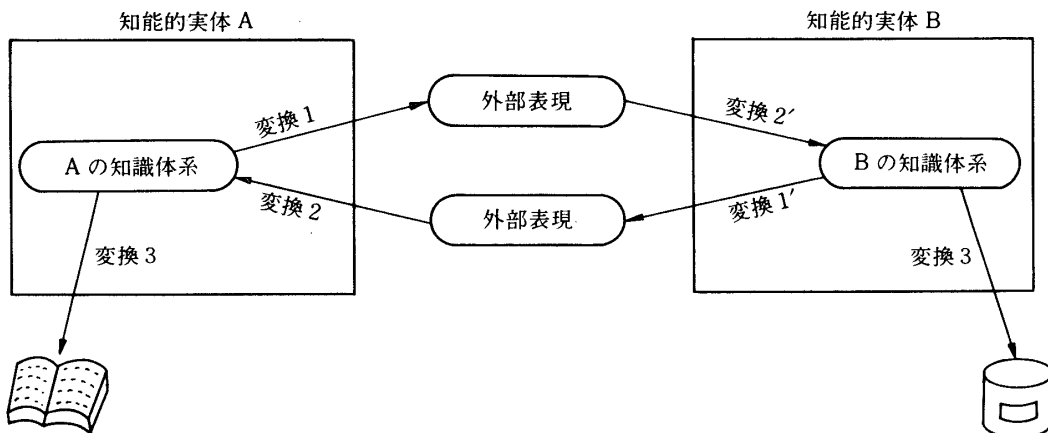
さらに、知識の保有形式は人によりあるいは人と計算機とで一般に異なっている。これをコミュニケーションという共通の場で利用するには第 1 図のような相互の知識間の変換が必要である。

変換 1 (1') は外部表現から知識表現への変換であるが、これは翻訳、認識、推論かぶこととする。この時生成される表現の内容はもとの知識体系によって定まり、相手方の知識内容とは一般に異なっている。

変換 2 (2') は外部表現から知識表現への変換であるが、これは翻訳、認識、推論から成る。たとえば、A が知識を $\alpha : (\text{物質 } a \text{ は水に浮かぶ})_A$ の形で有しているものとする。ここで (x) は外部表現で表わすと x であるような知識を表わし、 $()_A$ はこの知識が A に個有の表現形式のものであることを示すものとする。

一方、B は α を含むより広い範囲に適用できる形で、

$B_1 : (\text{物質 } a \text{ の比重は } 1 \text{ より小さい})_B$



第 1 図

B_2 ; (水の比重は1である)_B

B_3 ; (水より比重の小さい物質は水に浮かぶ)_B

なる知識を有しているかもしれない。()_B はB内の知識の表現形式を表わす。

すると、AからBへの話しかけのプロセスで示すと

$(x)_A \rightarrow x \rightarrow (x)_B \rightarrow (y)_B \rightarrow P(y)_B$

生成 翻訳 推論 プログラム生成
A B

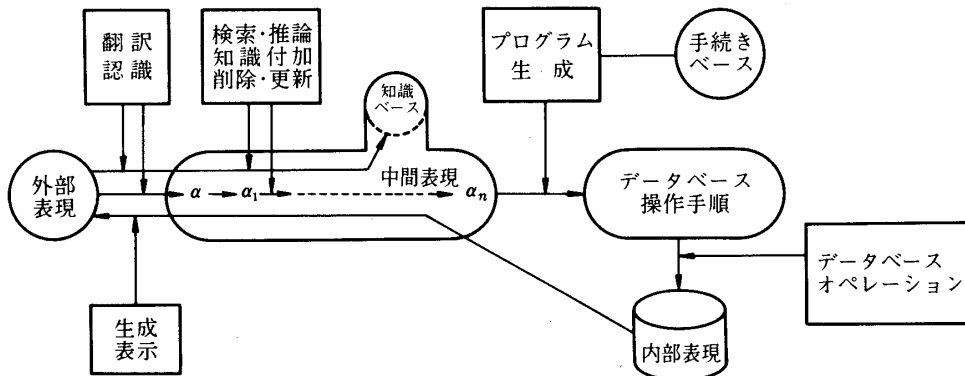
という図式により各変換機能が表わされる。ここで x は外部表現 ($x)_A$ はA内の知識, ($y)_B$ はB内の, ($x)_A$ に等しい意味を表わす知識とする。このような ($y)_B$ が存在する時, $x?$ なるAの質問に対し, Bは解答することができる。

変換3は ($y)_B$ のような形で内蔵された知識の他に, 補助記憶装置に適した記憶形式を用いる場合の ($y)_B$ からの変換である。人の場合でも電話帳, 時刻表など各種の外部記憶を用いるが, 計算機の場合, これがデータ・ベースに相当する。この場合, データ・ベースの処理には一定の手順が必要であり, 変換3は非手続き的に表現された質問や命令を手続きに変換するプログラム生成機能を含む。上の図式で $[y]_B$ は ($y)_B$ に対応する補助記憶内情報表現形式を表わしPはそれへの手続きを表わす。

以上は意味の処理を含むコミュニケーション時の典型的な情報の流れを示すが総合的なシステムとしては情報の検索, 命令の実行の他に随時新しい知識を付与し, それを以後の処理に役立てることが重要である。このような学習の機能を計算機が有することにより, 人は自己の知識, 経験を計算機に与え高度に知的な仕事にも計算機を利用することができる。

計算機にこのような機能を与えると共に, それが現実の場において利用できるだけの物理的性能(処理速度, 要求記憶容量, コスト等)を保証することは極めて重要なことである。

以上のことを考慮し, 以下ではこのようなシステムに課すべき条件を



第2図

- (1) 外部表現 (言語, 図形, 触感等の情報レベル)
(2) 中間表現 (論理レベル) (3) 内部表現 (物理レベル)

の三段階にわけてさらに具体化する。これら各レベルは第1図からも明らかであるが、さらに第2図のマン・マシン・システム用計算機の概略図に示されている。

外部表現は人と計算機とで共通の情報表現であるが、前述のマン・マシン・システムの要求から当然、人間の用いる表現 (言語, 図形など) に近いものが要求される。

中間表現 (知識表現) は外部表現により運ばれてくる意味および知識の表現法である。したがって外部表現形式は必ずこの中間表現の一つに対応すること、逆に中間表現は要求に合う外部表現の組に対応し得るように十分に記述力が確保されねばならない。

内部表現は中間表現形式を通して利用される、計算機の機構に最も適した情報の表現形式である。計算機の主要な利用目的である情報の蓄積と検索のためのデータは一般に大量であり、これを機構に合致した効率的な形式で蓄積することは不可欠となる。反面、これには中間表現との間の変換が必要であり中間表現へ一定の条件を付加することになる。

3. システムの満たすべき条件とシステムの基本構成

本節では知識利用システムに課せられる要求を第1表にまとめ、この各要求の内容、およびそれが課せられる理由を述べる。

3.1 外部表現および言語レベル処理に対する要求

外部表現としては通常多く用いられるものとして言語 (自然言語), 数式, 図形 (画像), 音声などがあるが、これらがすべて意味を表わすための約束された形式である点では相違はない。意味の表現形式が正しく定義されていれば、これらはすべてその形式内の表現に変換され、これ以後の処理は同じである。したがって、これらの相違は外部表現から中間表現への変換方法の相違として現われる。

変換機能の代表的なものは翻訳とパターン認識であり、いずれの場合もこの両機能を含んでいる。

言語では文によって知識が表現され、文は語から構成される。語は特定の性質をもった対象やその間の関係といった概念の識別可能な集合 (もしくは要素) を表わしている。言語による知識の表現あるいは言語の理解能力は、

- ① 識別される概念集合の修得 (語の修得)
- ② 識別される概念集合間の関係の修得 (文の修得)

の2段階で形成される。このうち①には認識能力が、②には論理的関係の表現能力とその処理能力が対応する。

もし十分な概念集合 (したがって語の集合) が前もって定義され、認識能力を用いて獲得した結果と同じ形式で計算機に与えられるなら、以後の言語活動はすべて②の機能のもとで遂行される。文により新しい概念集合を定義する機能により理解能力自体の増加も可能である。これが言語によるコミュニケーションの最大の特徴であり、この範囲の機能のみでも広範囲の問題に役立つことができる。

一方、図形で表現される問題の多くは認識を必要とする。計算機による認識能力は現状で

は困難が多い。図形を構成する図形要素が単純で認識が可能であるかあるいは認識に代わり対話型図形入力技法の利用できる問題以外はコミュニケーション手段としての図形入力や画像入力は現状では制約が大きい。

以下では処理の可能な外部表現を広義の言語と解し、単に言語と呼ぶ。実際にシステムが扱えるのはこのすべてでなく、中間表現、すなわち知識表現に翻訳の可能な表現の集合に限定される。許された外部表現を出来る限り大きく取るための条件は結局、知識表現の表現力と翻訳機能に対する要求となる。

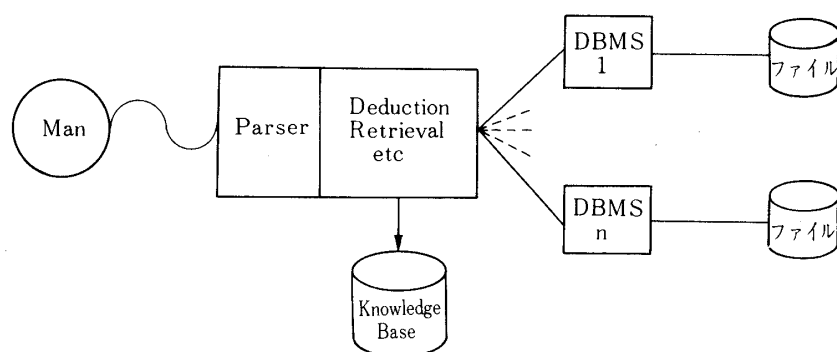
外部表現として、人間の用いる表現形式に制約が加わる場合、マン・マシン・システムとして実現可能となる唯一の条件は“許された外部表現の集合を簡潔にかつ具体的に記述する表現があり、その表現をユーザが容易に理解することができること”である。これを含め、外部表現に関する条件を次のように表わす。

文の表わす意味は語の意味とその関係としての文の構造から定まること、および複文・重文は単文の組に分解されることから、一般の文の意味は (i) 語の定義および意味的処理、(ii) 単文の構造による意味の表現法および (iii) 単文の組合せである複文に対応する意味表現の論理的構造、に展開できる。(ここで単文は“文の意味を変えずに分解し得る最小単位の文”を表わす。このことから外部表現に対する条件が第1表の条件 C_1 と C_2 と表わされる。

なお、語の意味の定義は知識の表現形式を用いて表わされる。この意味で言語内に生ずるあらゆる単文を正しく表現できるということが C_1 を満たす上で最も基本的な条件になる。現実に用いられている言語において用いられる単文の形式は有限個に限定されている。この形式の集合は必要なすべての意味を表現するのに十分大きく、かつ小児でも覚えられる程度に十分小さい。英語の場合、文の構造のみで分類すると、25種類程度である。このような基本文型の数は言語間であまり大きな差はないものと考えられる。

3.2 中間表現および論理レベル処理にたいする要求

論理レベルにおける機能は言語レベルと物理レベルの間であって、この両方を結合し、システムにたいする要求機能を果たすことである。言語レベル、物理レベルの機能はそれぞれ外部仕様および物理的構造から仕様が定められるのにたいし、論理レベルの機能すなわち知識の表現とその処理機能はこの両方にかかわり、この部分の決定がすべてに影響するという意味で、非常に重要な部分である。システムとしては言語レベル、論理レベル、物理レベルが有機的に結合することにより最高の能力を発揮するものである。情報の意味の表現形式という点から考えると、計算機内部ではプログラムとデータベースという二種の異なった形式が存在する。すなわち内部表現はさらにこの二形式に分けられる。しかし、いずれにしろ言語レベルと物理レベルでの表現形式すなわち外部表現と内部表現の表現能力が著しく異なるので、異形式表現間に1対1の対応が成立しない。そこでこれを結合するために一方の形式で表現された情報の意味を保存しつつ、他方の形式内にこれと対応する表現が見出されるまで、同一形式内で表現を変えてゆく操作が必要になる(第4図)。これが、以下で述べる推論処理であるがこの変換を直接外部表現、内部表現間で行わず、このどちらにも変換が可能でありしかも推論アルゴリズムの定義のしやすい形式として中間表現を用いるのである。した



第3図

がって、まず中間表現の形式としては $C_3 \sim C_5$ の形態的な変換の可能性についての条件が不可欠である。

一方、意味的な結合を図る論理レベル処理手続き、すなわち知識利用手続きとしては、推論可能性に関する C_6 の条件が必要であると同時に、この手続きに対して完全性 (C_7) の条件が課せられねばならない。この条件は外部表現による入力に対して、これと同義の手続き表現 (プログラム) が存在すれば必ずそれが実行され、また同義のあるいはこれを含意する情報 (の組) が存在する場合、必ずそれが取り出されるべきことを意味する。論理の完全性はデータ・ファイルの無矛盾性を保証するためにも重要である。

推論に際しては意味の同値 (含意) 関係を示す知識が利用される。このような知識が集められたものを知識ベースと呼ぶ。論理レベルを含むシステムは知識ベースを利用するシステムと呼ぶことができ、論理レベルの機能はシステムを中心部分を形成する。さらに、知識取得のできるシステムとするためには人の知識や経験が、利用の場合と同様に容易に知識ベースに登録できることが必要である。これが C_8 の条件である。知識利用・取得・保守に関するこれらの条件から知識の構造が定められねばならない。

さらにこれに関連して、マン・マシン・システムにおいては、ユーザのシステム利用の習熟度を応答時間に反映できることは実用上重要で C_9 の条件も欠かせない。マン・マシン・システムでは、計算機についてほとんど知識のないユーザから、データベースの内容にまで熟知しているユーザまで広い範囲のユーザが対象となる。システムとしては前者のごときユーザに対しても十分なサービスを保証することが必要であるが、後者に対して過剰なサービスとなることは避けユーザの有する知識を応答時間に転換できる形式としておくことは非常に重要なことである。これには知識の表現および推論アルゴリズムの両方がかかわってくるであろう。

3.3 内部表現および物理レベル処理に対する要求

記憶容量・応答時間・システム効率・システムの複雑さ・コストなどの物理的経済的性能を保証することは実用システムを目的とする上で極めて重要なことである。論理レベルまでの諸条件を満たすことができても物理的な性能の低いシステムは実現し得ない。また、本シ

システムのように高度の機能を要求されかつ厳しい条件の課せられるシステムでは、論理的性能と物理的性能を独立に達成しようとするのは困難であり、設計時に物理的性能達成の条件を考慮した論理アルゴリズムとすることが必要である。この際、物理的性能を考慮することがシステムの柔軟性を失うことのないような注意が必要であり、システムの満たすべき条件を厳しく列挙するのもこのためである。

3.3.1 情報の記憶

システム内には多くの知識要素が記憶される。論理処理の目的には、必要な情報を中間表現の形で知識ベースに記憶することが便利であるが、すべての情報をこの形で記憶することは記憶容量の点で得策でない。知識ベース内の情報は推論に用いられ、検索を早めるために何らかの構造化が行われて知識構造を形成して行くだろう。この構造内には各種アクセスパスが各知識ユニットに与えられる。一方、計算機内に貯えられる情報の内大多数を占めるのは個々の対象に関する情報あるいは対象間の関係を示すレコードとなることが予想される。これらの情報は推論の中間情報の生成に用いられることはなく、したがって推論手順を促進するようなアクセスパスを準備する必要はない。このような情報に対しては記憶効率を向上するような記憶方式を用いることが必要である。C10、C11はこの条件を表わす。C10に示される個別知識ユニットの集まりをデータベースと呼ぶ。

システム内の情報のクラス分けは単に記憶容量の節約およびその結果としての(探索等の)時間短縮のみでなく、一つの質問の答となり得るすべてのデータを取り出すためにも効果的である。質問を満たす個々のデータを探索的に取出す代わりに、質問を推論アルゴリズムにより個別のデータのファイルに対する一定の条件に変換し、集合演算によって、必要なすべてのデータを取り出せるからである。

3.3.2 システム構成

C5の条件のもとで従来の意味でのデータが知識の一形式として利用できる。したがって、実用的なシステムを目指すためには従来のDBMSおよびそれによって管理されているデータベースをできるだけ利用するという立場にたったシステムとすることが必要である。機能的にみると、言語レベルおよび論理レベルは従来のシステムの知的インターフェースともいえるべき部分である。したがって、これを物理的にも既成のDBMSと分離し、条件C5を課すことにより各種DBMSにアクセスできるようにする。この条件は条件C10を前提としてはじめて成立する。これにより、本システムを、既成の、データ・モデルの異なる各種DBMSの共通の知的インテリジェント・フロント・エンド・システムとし、分散型データベースの利用を可能にする。このようなシステムの概略図を第3図に示す。

3.3.3 推論処理の高速化

検索・推論部分が従来のデータ・ベース・システムの知能フロント・エンド・システムとすると、これを高速化することが不可欠である。検索・推論手続きのうち、関連知識ユニットの探索は大量の知識ユニットの集まりの中から一定の条件を満たすものを求めねばならず、最大の時間がかかる。C12の条件はこれを高速化するために、探索範囲を制限するようなアルゴリズムとすべきこと、C13は推論手続きのうち繰り返しの多い部分をできるかぎりハードウェア化できるアルゴリズムとすべきことを示している。

2 に示したマン・マシン・システムを実現するために以上の各種条件以外にも数多くの要求が出されるであろう。上記の諸条件は其中最も基本的であり、互いに関連し合っているため、その全体を考慮した上でシステムの方式を決定すべきものである。この他にマン・マシン・システムとして各種の要求、たとえばインタラクティブに、コミュニケーションに不足している情報を補う方法などは不可欠である。しかし、これら既知でありシステムが備えるべきことが既に暗黙の前提となっている技術は敢えて削除している。この意味で上述の諸条件は最小の条件集合である。

第 1 表 知能的システムに課せられる基本条件

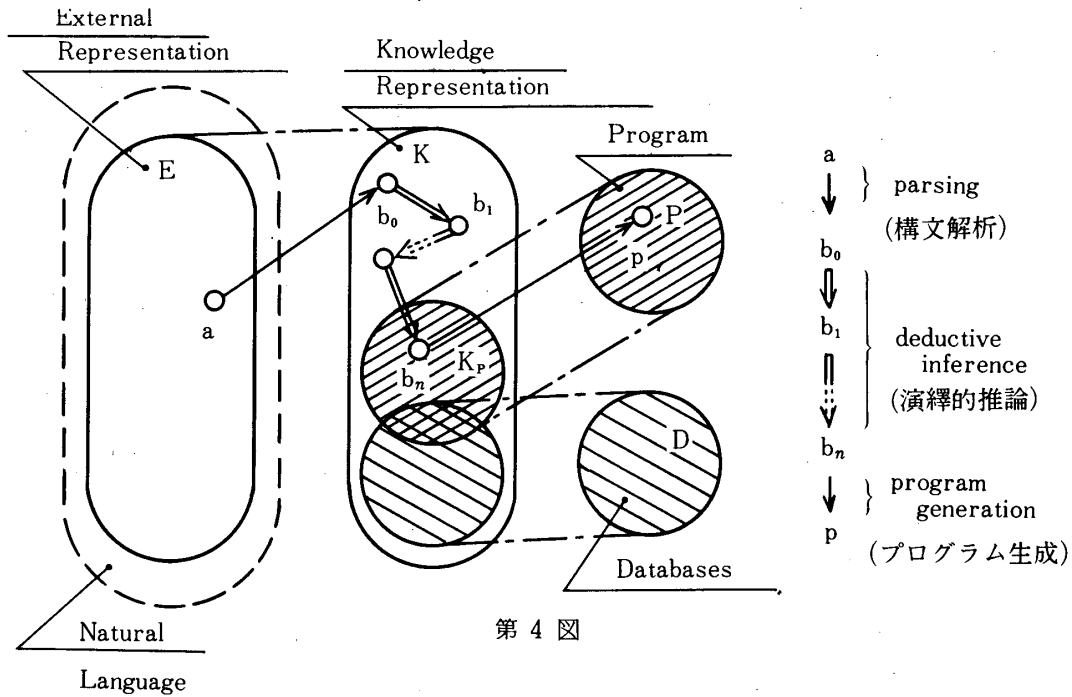
1. 外部表現および言語レベル処理
C 1. すべての語の意味の定義と関係, すべての単文の構造とその意味の関係が表現されること.
C 2. 許された表現を定義する記述が容易であり, かつこの許された表現の翻訳は完全であること.
2. 中間表現および論理レベル処理
C 3. 中間表現は外部表現との相互の変換が可能なること.
C 4. 中間表現はプログラムとの相互の変換が可能なること.
C 5. 中間表現はデータベースとの相互の変換が可能なること.
C 6. 中間表現は推論処理に適したものであること.
C 7. 推論アルゴリズムは完全であること.
C 8. 知識の蓄積・削除・更新・良質化が利用と類似の手順で可能なこと.
C 9. 論理レベル処理はユーザの習熟度を十分に活用し得るものであること.
3. 内部表現および物理レベル処理
C 10. 個別知識のユニットは計算機にとって最も効率的な形で記憶すること (記憶の階層化)
C 11. 知識ベース・データベース共に冗長度をなくす記憶方式とすること.
C 12. 質問自体に含まれる情報をすべて利用し, 効率の向上, 特に探索範囲の縮小に役立つ探索方式とすること.
C 13. 探索, 推論を通じ, 繰り返しの多い部分のハードウェア化を促進すること. 逆に, ハードウェア化のできるアルゴリズムとすること.

3.4 マン・マシン・システム用計算機に必要な機能

以上の議論から、マン・マシン・システム用計算機に要求される機能を示したのが第 4 図および第 2 表である。以下ではこれを実現するために実験的に開発されたシステムについて述べる。

4. 知識取得, 利用システム — KAU [12]~[16][18][25]

本節以下では前節までの分析に基づいて設計された知識システム—KAU (Knowledge Acquisition & Utilization) システムについて述べる。



第4図

第2表 システムの備えるべき変換機能

1	$E \rightarrow K$: PARSING, PATTEN CLASSIFICATION
2	$K \rightarrow E$: OUTPUT GENERATION
3	$K \rightarrow P$: PROGRAM GENERATION
4	$(K \rightarrow D)$: DATABASE DESCRIPTION
5	$D \rightarrow K$: GENERALIZATION
6	$K \rightarrow K$: DEDUCTIVE INFERENCE

4.1. 知識の表現

KAUでは知識の表現形式として述語論理を基礎とし、記述力を増すための高階論理の採用、処理効率を増すための集合の概念の導入およびグラフとの組合せなどにより機能を拡張している。この拡張部分は、これまで述べてきた諸条件を満たすために重要な部分であるが以下では、第4図の情報変換の過程を見易くするために先ずその基本となっている一階述語の場合を示す。

4.1.1. 述語表現

知識表現の形式記号は述語記号 f ，集合 X_i を変域とする変数を x_i とした時、 (fx_1, \dots, x_n) の形で表わされるアトム，またはその否定形 $\sim(fx_1, \dots, x_n)$ を，論理結合子， Π (積) \cup (和) \rightarrow (含意)， \leftrightarrow (等価) によって構造化し，変数に関する限量化の条件 \forall ，および \exists を付した論理式で表わす。実際には， $A \rightarrow B$ (A なら B) は， $\sim A \cup B$ ， $A \leftrightarrow B$ (A と B は等価) は $(A \rightarrow B) \cap (B \rightarrow A)$ で置きかえられるが，以下の記述では理解しやすい \rightarrow と \leftrightarrow も用いる。

たとえば，“比重が流体の比重よりも小さければ物体はその流体に浮かぶ”という知識を考えてみよう. すべての物体の集合を P , すべての流体（というものが考えられるとして）を F , 実数の集合を R とすると,

$(\text{spc-grv } x \ r)$; “ x の比重は r である” ($x \in P, r \in R$)

$(\text{flt } x \ y)$; “ x は y に浮く” ($x \in P, y \in F$)

$(\text{lt } x \ y)$; “ x は y より小さい” ($x, y \in R$)

なる アトムを用いて上述の知識が

$(\forall x/P) (\forall y/F) (\forall z/R) (\forall u/R) [(\text{SPC-grv } x \ z)$

$\wedge (\text{spc-grv } y \ u) \wedge (\text{lt } z \ u) \rightarrow (\text{flt } x \ y)]$ (1)

のように表わされる. [] 内は「物体 x , 流体 y についてそれぞれ比重が z, u であり, $z < u$ なら x は y に浮く」であり, 最初の $(\forall x/P)$ 等はこのような x, y, z, u にはそれぞれの変域の中でいかなる値を代入しても成り立つことを示す. 同様にして比重の定義が次のように表わされる.

$(\forall x/P) (\forall y/R) (\forall z/R) (\forall u/R) [(\text{mass } x \ y) \wedge (\text{volume } x \ y)$

$\wedge (\text{div } u \ y \ z) \rightarrow (\text{spc-grv } x \ u)]$ (2)

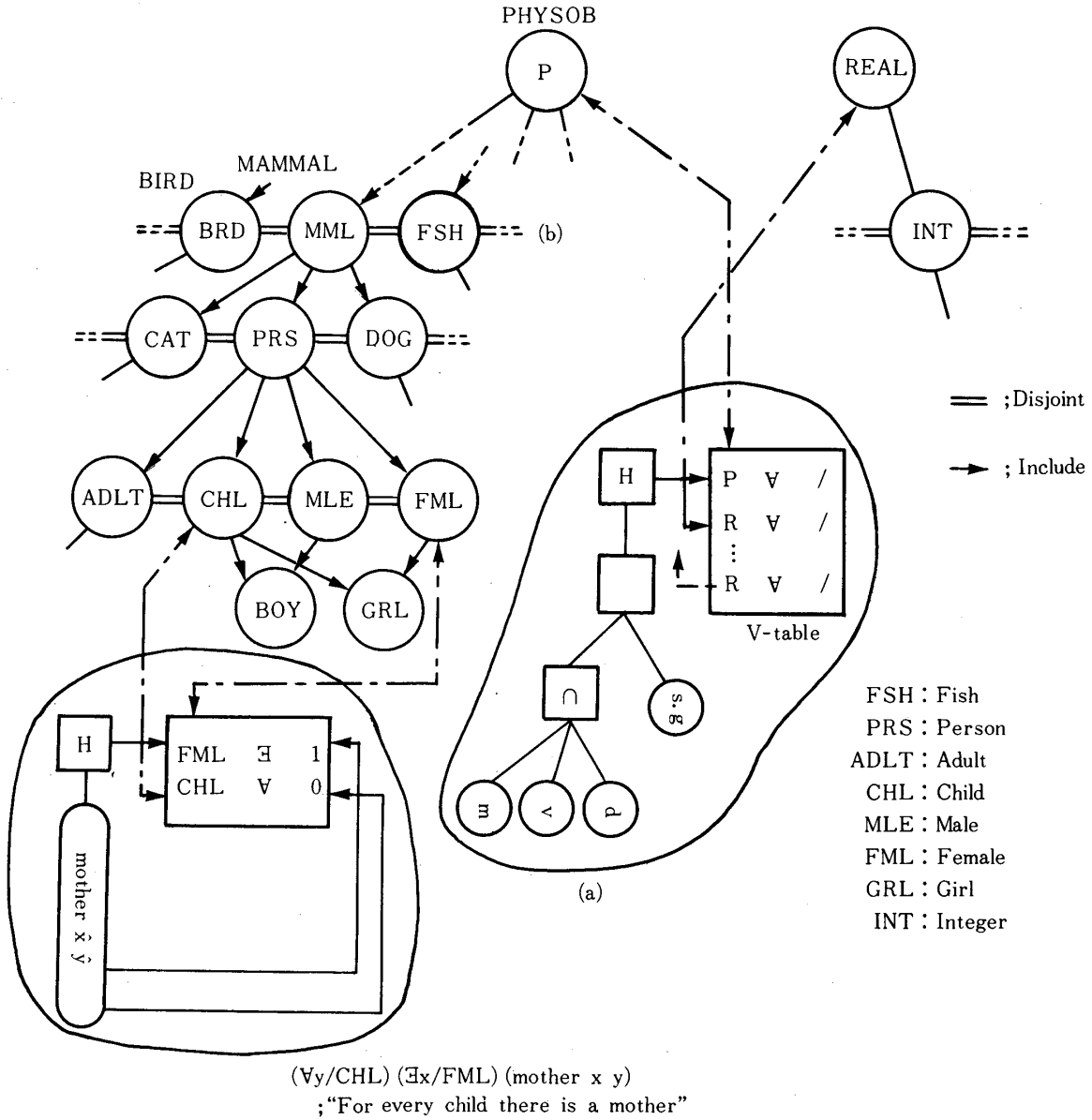
これらは \rightarrow の左右両項の意味的な関係を示す知識であるが $(\text{father } a \ b)$ を “ a は b の父親である” を表わすアトムとすると, $(\forall x/CHILD) (\exists y/MAN) (\text{father } y \ x)$ は “すべての子供には父親である男性が居る” のように単純な宣言を表わすのもある. $(\exists y/MAN)$ は $x \in CHILD$ なる任意の x に対し, father の関係をなすのは MAN の中の特定の y のみであることを示す.

このようにアトムは言語における文, 句, 節に対応し述語記号は動詞, 形容詞, 前置詞のように対象の属性や関係を意味する語に対応する. これらの語にはその意味の定義としてのアトムの原形が与えられ, かつその語の意味が正しく定義されるような最大の変域が各変数に与えられる. これらの語を関係語と呼ぶことにする. これに対し, この変数には対象を指定する語 (名詞) が対応する. これを実体語と呼ぶことにする.

なお, 異種限量記号が同一の表現内にある時, 接頭部内の変数順序により意味が異なる. $(\exists y/MAN) (\forall x/CHILD) (\text{father } x \ y)$ は “ある男 y がおり, すべての子供の父親である” を意味する. このように知識は論理式の構造情報 (アトムの結合関係) と変数情報 (各変数の変域と限量記号, および接頭部内変数順序) が与えられると一意に定まる. KAU ではこれを第5図 (a) のように前者を AND-OR 木で, 後者をこの構造にリンクされた変数表 (V表) で表わしている.

4.1.2 スケルトン構造

実体語, たとえば普通名詞の “男” “少年” などはそれぞれ実体のクラスを表わしこれが変域となる. クラス間には集合的關係がありこれが後述する推論に用いられるので, これをノードと二種のアークを含むグラフで表現する. このグラフで各ノードは一つの集合を表わし, 二つのノード X, Y は, $X \supset Y$ の時 I 型アークで, $X \cap Y = \emptyset$ の時 D 型アークで結合される. たとえば, “男” \supset “少年” “男” \cap “女” = \emptyset である. これにより第5図 (b) の



$$I: \begin{cases} MML \supset PRS, CAT, DOG, \dots \\ PRS \supset ADLT, CH, MLE, FML, \dots \\ CHL \supset BOY, GRL, \dots \\ MLE \supset BOY, \dots \end{cases} \quad ; \quad D: \begin{cases} PRS \cap CAT = \phi \\ PRS \cap DOG = \phi \\ MLE \cap FML = \phi \\ \vdots \end{cases}$$

第 5 图

ような階層的構造が作られる。これをスケルトン構造と呼ぶ。述語の変数 x の変域が X の時、 x を含むアトムは上記階層構造のノード X にリンクされ、これ全体が知識構造を作っている(第5図)。

4.2 推論処理

4.2.1 推論の方式

知識 $A \rightarrow B$ は “A が真なら B も真” を表わし、条件に相当する \rightarrow を含まない知識 C は “C は真” を表わす。仮説 H が C に含意されれば H は真である。また、 $B \rightarrow H$ の関係にあれば、 $A \rightarrow H$ となり、 H の真偽は A の真偽に依存する。すなわち A が新しい仮説となる。

一般に知識 $K \equiv (A_1 \cap \dots \cap A_m \rightarrow B)$ と仮説、 $H \equiv D_1^* \dots D_n^*$ が与えられた時 H の中にある D_i について、 $B \rightarrow D_i$ が示されれば、これらは仮説、 $H' = D_1^* \dots D_{i-1}^* D_{i+1}^* \dots D_n^*$ ($A_1 \cap \dots \cap A_m$) * に等価である。ここで i^* は i 番目の結合子 (\cap または \cup)、 D_j' は (以下で述べる規則により) D_j が変化したものを示す。 H' の AND-OR 木は K の構造から ($\rightarrow B$) の部分を除去したもので H のノード D_i を置きかえることによって得られる。

D_i について $C \rightarrow D_i$ なら、 H の AND-OR 木のノード D_i には “真” を表わすラベル T が、 $C \rightarrow D_i$ なら偽を示すラベル F が、そのどちらでもない (関連知識が存在しない) 時はラベル U が付される。以下この手順を変数を含む場合について示す。

[A] 含意条件テスト (TIC — Test for Implicative Condition)

F ; $(\exists x/\text{GIRL})(\forall y/\text{PERSON})(\text{like } y \ x)$; “ある少女が居てすべての人がその少女を好きである” は

H ; $(\forall y/\text{MAN})(\exists x/\text{PERSON})(\text{like } y \ x)$; “すべての男には好きな誰か(人間)が居る” を含意する ($F \rightarrow H$)。 F と H の対応する変数の変域を入れ替えたり F 、 H 共に接頭部の変数順序を入れ替えると $F \rightarrow H$ は成り立たなくなる。このように同じ述語記号の単一アトム論理式の含意条件は変数の関係で決まる。

F は $x_1 \in X_1, \dots, x_n \in X_n$ を、 H は $y_1 \in Y_1, \dots, y_n \in Y_n$ を変数として含む時 $F \rightarrow H$ の条件は単純で

(1) 各変数ごとに第3表の関係が成立する。

(2) 接頭部の変数順序について F では $\dots (\forall x_i) \dots (\exists x_j) \dots$ であり、 H では $\dots (\exists y_j) \dots (\forall y_i) \dots$ になるような添字の対 (i, j) が存在しない。

の両方を満たすことである ([13] 参照)。

第3表 含意規則 (q_i)

Q_{pi}	$Q_{qi}(\bar{Q}_{qi})$	CONDITON on DOMAINS
V	V(E)	$X_i \supset Y_i$
V	E(A)	$X_i \wedge Y_i \neq \phi$
E	V(E)	NON IMPLICATIVE
E	E(A)	$X_i \subset Y_i$

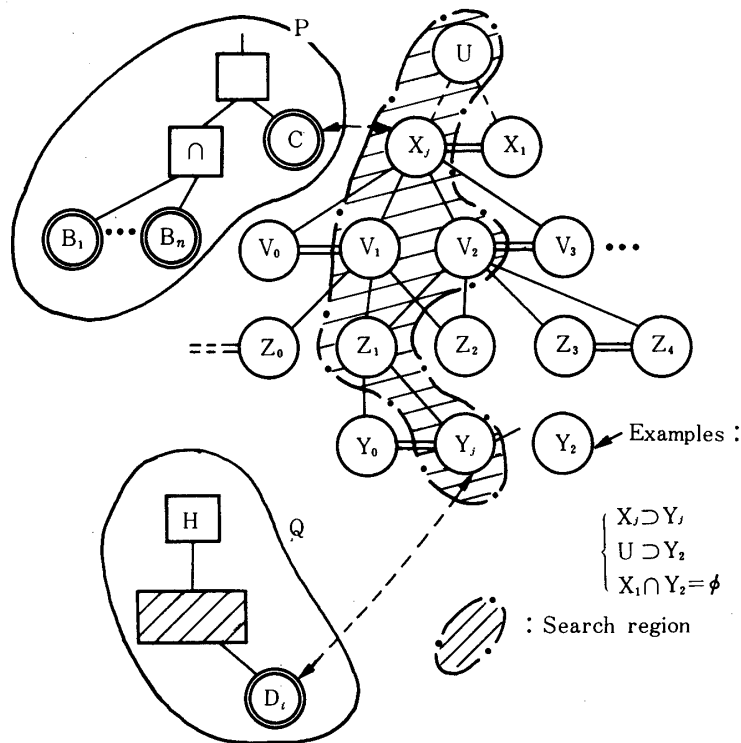
(実際には仮説Hの否定形 $\sim H$ を作って処理を行うが議論の本質にはかかわらないので、Hのまま処理するものとして述べる)

KAUではこの含意テストをTIC (Test for Implicative Condition)と名づけられたプログラムが受持っている。

TICは与えられた D_i に対して $F \rightarrow D_i$ を満たすFを含む知識を知識ベースの中から探し出す。この際スケルトン構造を利用する。

D_i の中で限量記号 V 、変域 Y_j を持つ変数(もしくは定数)を y_j とする。もし、 $F \rightarrow D_i$ を満たすようなFを含む知識が存在し、 y_j に対応するFの変数を x_j とすると、 x_j は限量記号 V をもち、変域 X_j は $X_j \supset Y_j$ でなくてはならない(第1表)。

このような知識はスケルトン構造内でノード Y_j の上方にI型アークをたどって達することができる範囲のノードにリンクされている(第6図)。TICは D_i の内のこのような変数 y_j を選びこの範囲のうち述語記号が同じ知識のみを $F \rightarrow D_i$ のテスト候補とする。もし V をもつ変数が二つ以上あれば、この範囲はさらにしぼられる。TICは各候補について、残りの変数についても第1表を満たすかどうかを調べる。この時はテストすべき両方の変域 X_k, Y_k がわかっているのでこの関係をスケルトン構造の対応するノード間の位置関係によって調べることができる(詳細略)。このアルゴリズムは極めて単純で専用ハード化が可能であり、また変数ごとに独立して実行可能なので並列化が可能である。



第6図

〔B〕 置換規則 (RR — Replacement Rule)

K: $A_1 \cap \dots \cap A_m \rightarrow B$ の B と $H = D_1^* \dots D_n^*$ の D_i について $B \rightarrow D_i$ が判定されたとする. K に含まれる変数 X のうち B に含まれるものを X_0 , 残りを X_1 とする. また, H に含まれる変数 Y のうち D_i に含まれるものを Y_0 , 残りを Y_1 とする. すると H' の各変数の変域と限量記号は第4表のように K, H の変数の関係で定まり接頭部変数順序も K, H の関係で定まる (詳細略 [13] 参照). 変域が変わることは変数に代入が行なわれることに相当する.

第4表 置換規則

	Q_{bi}	$Q_{qi}(\bar{Q}_{qi})$	$Q_{ri}(\bar{Q}_{ri})$	Z_{ri}/W_{ri}
$x_i \in X_0 = Y_j$	V	V (E)	V (E)	Y_i
	V	E (V)	E (V)	$X_i \wedge Y_i$
	E	E (V)	V (E)	X_i
$x_i \in Y - Y_j$	—	V (E)	V (E)	Y_i
	—	E (V)	E (V)	Y_i
$x_i \in X - X_0$	E	—	V (E)	X_i
	V	—	V (E)	X_i

KAU では RR と呼ばれるアルゴリズムにより K, H の木から H' の構造を生成し, かつ V 表を作成する.

〔C〕 選択規則 (SR — Selection Rule)

SR は仮説の中から未評価アトムを選び出し, TIC に渡す. この際, V 限量記号をもつ変数を多数含むアトムを選ぶなど探索の高速化を考慮する (詳細略).

〔D〕 終了判定 (TT — Test for Termination)

論理式 (1) の左辺の $(It z n)$ は z, n に値が代入された時点で評価可能であり条件が成り立てば真, さもなければ偽である. これは一般のアトム D_i が自分以外の知識 C により $C \rightarrow D_i$ が満たされる時にのみ評価されるのと異なる. 前者のようなアトムには, これを評価する関数/手続きルーティンを準備し, プログラム・ベースに格納し評価時に取り出せるようにしておく. すなわちこの種のアトムは非手続き的側面と手続き的側面を併せもつ. これを“手続き型アトム (PTA)”と呼びこれ以外の一般のアトムは“非手続き型アトム (NTA)”と呼ぶ. PTA としてはシステム側で準備する基本オペレータの他, ユーザが任意のサブルーテン・パッケージを登録して用いることができる [22].

推論に際して仮説の中に現われる PTA は評価可能になると対応するルーティンが呼ばれ評価されるがそれ以外は未評価のまま残される. これは TT と呼ばれるアルゴリズムの前半で行われる.

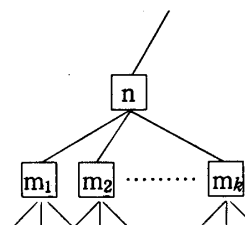
仮説の AND-OR 木の末端のアトムが評価されると上方のノードが順次第5表の規則に従って評価され, ルート・ノードの評価が仮説の真偽を表わす. 第5表で評価されるノー

第5表 AND-OR 木内のノードの評価

n : node to be evaluated (U or Ω)

m_i : node below n ($i = 1, 2, \dots, k$)

	(n : Ω / n : U)		(n : Ω / n : U)
a)	if for $\forall i$	($m_i = T / m_i = F$)	then (n = T / n = F)
b)	if for $\exists i$, s. t.	($m_i = F / m_i = T$)	then (n = F / n = T)
c)	if for $\forall i$,	($m_i \neq F / m_i \neq T$)	and
	for $\exists j$, s. t.	($m_j = U / m_j = U$)	then (n = U / n = U)



T : True, F : False, U : Unknown

Q : (Ex/I)(fct x z) ? : "What is z's factorial ?"

{0th} TT : Is there any atom that can be evaluated ? - No. Next

{1st} SR : D = Q (Select Q as D)

TIC : (A) \rightarrow D

(RIGHT HAND FORMULA of (B) (= ($\forall x/I$) ($\forall n/I$) (fct x n)) \rightarrow D
 $n \leftarrow 2$

RR : \sim (eql 2 0) \cap (sub m ; 2 1) \cap (mult x ; y 2) \cap (fct y m)

TT : \sim (eql 2 0) @ T, (sub m ; 2 1) @ T, $m \leftarrow 1$

then Q \rightarrow (mult x ; y 2) \cup (fct y 1)

(eliminate atoms that have been evaluated)

{2nd} SR : D' = ($\exists y/I$) (fct y 1)

(Repeat similar procedure to {1st}'s and obtain)

then Q \rightarrow (mult x ; y 2) \cup (mult y ; z 1) \cup (fct z 0)

{3rd} SR : D'' = ($\exists Z/I$) (fct z 0)

TIC : (A) \rightarrow D'', Z \leftarrow 1, (fct 1 0) @ T

TT : Q \rightarrow (mult x ; y z) \cap (mult y ; 1 1)

END

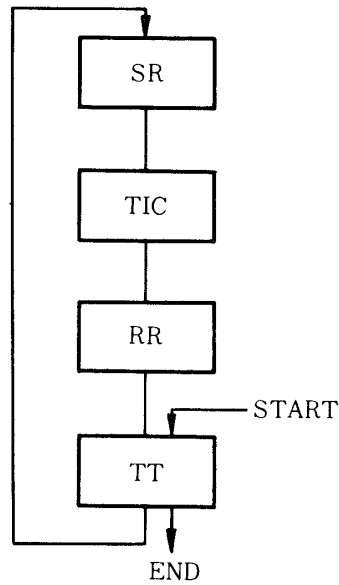
where (\dots) @ T ; (\dots) is labelled T

$x \leftarrow \alpha$; α is substituted to x

Q \rightarrow (Formula) ; Q is changed to (Formula)

(A) (fct 1 0)
 (B) ($\forall x/I$) ($\forall y/I$) ($\forall m/I$) ($\forall n/I$) { \sim (eql n 0) \cup (sub m ; n 1)
 \cap (mult x ; y n) \cap (fct y m) \rightarrow (fct x n)}
 where (fct x n) ; "n! = x"

第7図 再帰的に定義された関数評価の例 - 階乗



第 8 図

ドが n , その直下ノードの集合を $\{m_1, \dots, m_n\}$ とする. ノード n が \cap ノードか \cup ノードかにより規則が異なり ($\dots | \dots$) の左半は n が \cap の場合, 右半は \cup の場合を示す. この評価は, TT の後半が行う.

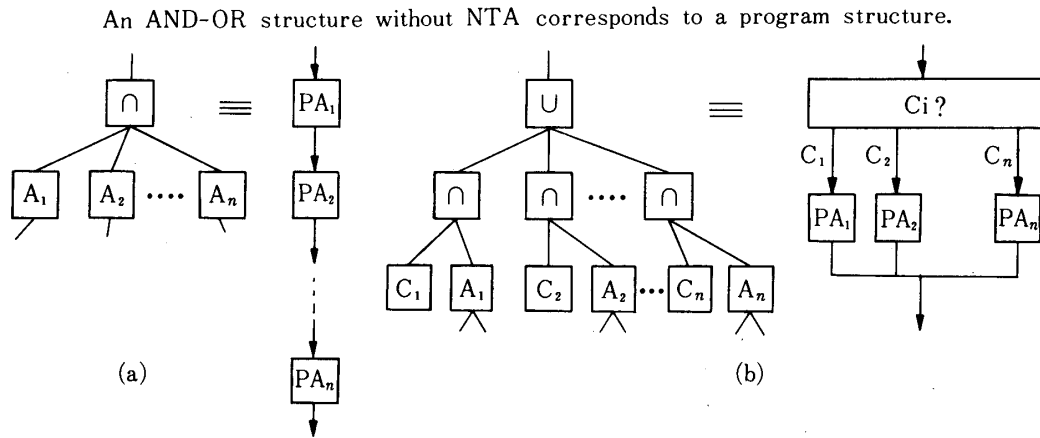
推論過程で新しい部分木構造により置き換えられるのは NTA のみであり未評価 NTA がなくなった時, 推論を終了する. ここで推論が有限回の繰り返しで終了するか否かが問題となる. 以下知識ベース内の知識は有限とし, これを調べてみる.

ある仮説の推論過程に用いられる知識の集合を考えよう. この中の一つの知識が推論に用いられ, この含意記号 \rightarrow の右辺のアトム述語記号を f とする. もし, この知識集合がその中の他のすべての知識について \rightarrow の左辺に述語記号 f のアトムを含まないという条件を満たしている場合, 知識の有限性から推論過程は有限回で終了する. この条件が成立しない場合は再帰的に定義された知識が推論に用いられる場合である. この場合も再帰表現が有限時間内で結果を得られる形に表わされていれば有限回で終了する. 例として $(fct\ z, n)$: " $n! = z$ " で表わされる階乗の定義を次のように表現する.

$$(A) (fct\ 1, 0) \quad (3)$$

$$(B) (\forall x/I)(\forall y/I)(\forall m/I)(\forall n/I)[\sim(eql\ n\ 0) \cap (sub\ m : n\ 1) \cap (mult\ x\ y\ n) \cap (fct\ y\ m) \rightarrow (fct\ x\ n)] \quad (4)$$

ここで $(eql\ a\ b)$; " $a=b$ ", $(sub\ c ; a\ b)$; " $a-b=c$ ", $(mult\ c : a\ b)$; " $a \times b = c$ " などは PTA でいずれも a, b の値が定めれば評価可能となる. 後二者については演算の結果が c に代入される. I は整数である.



第 9 図

この関数は有限の回数で求まる. 2の階乗を求める例が第7図に示されている. 推論が終了した時点で実行可能な乗算の列が求まる. 第8図は推論処理のプロセスを示すこれを知識ベースの構造を利用するという意味で SBDS (Structure Based Deduction System) と呼ぶ.

4.3 プログラムへの変換

AND-OR 木内に未評価 NTA が無くなった時点で推論を終了したとしよう. この時の仮説は評価済みアトムと未評価 PTA から成る AND-OR 木である. この木構造内の評価済みのノードをすべて消すことによって得られる部分木は構造内で一定の順序 (たとえば左から右へ) を定めておくと, プログラム構造に対応する. 実行計算機がデータフロー型のものならこのような人為的な順序の導入は必要ない.

AND-OR 木の基本形式は

$$A_1 \cap \dots \cap A_n \tag{5}$$

$$(C_1 \cap A_1) \cup \dots \cup (C_m \cap A_m) \tag{6}$$

であるがこれらはそれぞれ第9図のプログラム構造に対応する. (5), (6) において A_i 自体も部分木であるとする, これらがさらに詳細なサブプログラム構造に展開される. 任意の AND-OR 木は (5) または (6) の形式で表現できるから適当な翻訳プログラムにより目的プログラムに変換される. 繰り返し構造は前述のように推論の段階で上記2形式に展開されているが, このようにせず WHILE や DO に相当するプログラム・ルーチンを生成するには推論処理中に再帰型パターンを識別し, これを繰り返し構造を持ったプログラムに置きかえる. また, 知識の誤った定義のために無限ループに入るのを防ぐことも必要であるが, これらの識別のアルゴリズムは類似しているので同時に行われる. 仮説内でアトム

D_i が選ばれ、知識の部分木構造 P' で置きかえられたとする。後に P' 内のアトム D_j' が選ばれた時、もし $D_i \rightarrow D_j'$ なら上述のいずれかの場合である [15]。

4.4 データベースとの関係

データベースには多種のデータモデルがあり、意味と形式の関係が各々異なる。ここでは関係データベースを例にとって知識との関係を示す [2][3]。関係型データベースではある一定の関係 f にある n 組の集まりである関係ファイル F が $F = \{(x_1^j, \dots, x_n^j) / x_1^j \in X_1, \dots, x_n^j \in X_n, j = 1, 2, \dots, N\}$ で表わされる。これは幾何学的に見ると $X^1 \times \dots \times X^n$ なる n 次元直積空間の点 (x_1^j, \dots, x_n^j) の集まりである。もし $F = X_1 \times \dots \times X_n = \mathbf{X}$ なら、このファイルは n 次元超立方体を形成し、この中のすべての点が関係 $(f x_1^j, \dots, x_n^j)$ を満たすから、これは論理表現 $(\forall x_1 / X_1) \dots (\forall x_n / X_n) (f x_1 \dots x_n)$ と同義である。この時はファイルを作る必要はなく、論理表現を知識ベースに加えておけば十分である。これは $D \rightarrow K$ の一例でもある。

F が \mathbf{X} の真部分集合である一般の場合はこのような単純な同義関係は成立しないが、 $(FGET F ; x_1 \dots x_n)$; “ n 組 $\mathbf{x} = (x_1 \dots x_n)$ は F に含まれる。” なる特別のアトムを定義することにより、

$$(\forall x_1 / X_1) \dots (\forall x_n / X_n) [(FGET F x_1 \dots x_n) \rightarrow (f x_1 \dots x_n)] \quad (7)$$

と表わすことができる。さらにアトム $FGET$ は n 組 \mathbf{x} をファイル F から取り出す役割をもった手続き型アトムとすると、上式はファイルのアクセスを含めた、ファイルの型式と意味間関係を表わす記述となっている。各ファイルについてこの形式のファイル記述を知識ベースに入れておけばファイル操作手順が自動的に生成される。

このようにファイルは知識表現におけるアトムに対応する表現である。関係データベースの場合、join, projection, selection, restriction, division などの基本オペレータを組合わせたファイル操作を人間が指定することにより要求を表現している。これはファイルの処理操作が意味的に知識表現に等価であること、したがって知識における論理構造部分すなわち、限量化と AND-OR 木構造情報がファイル操作と等価であるべきことを示す。事実、これらの間には、ほぼ一対一の対応関係がある。以下では記述の便宜上、オペレータを第 6 表のように定義する。また、推論処理終了時にファイル検索型アトム $FGET$ を含む PTA から成る AND-OR 木が作られたものとし、論理式内の全変数の変域の集合を $\mathbf{X} = \{X_1, \dots, X_n\}$ とする。

(1) ファイル F_1, F_2 がそれぞれ属性の集合 $\mathbf{X}_1 = \{X_{i_1}, \dots, X_{i_r}\}$, $\mathbf{X}_2 = \{X_{j_1}, \dots, X_{j_s}\}$ を有し、 $\mathbf{X}' = \mathbf{X}_1 \vee \mathbf{X}_2$ とする。また $F_1^* = F_1 \times (\mathbf{X}' - \mathbf{X}_1)$, $F_2^* = F_2 \times (\mathbf{X}' - \mathbf{X}_2)$ とすると $(FGET F_1 ; x_1 \dots x_{i_r}) \circ (FGET F_2 ; x_{j_1} \dots x_{j_s}) = (FGET F_1^* ; x_1 \dots x_n) \circ (FGET F_2^* ; x_1 \dots x_n) = (FGET F_1^* \hat{\circ} F_2^* ; x_1 \dots x_n)$ が成り立つ (証明略)。

ここで \circ は論理的結合子 \cap または \cup を、 $\hat{\circ}$ は \circ に対応する集合演算子で $\circ : \cap / \cup$ に応じ $\hat{\circ} : \cup / \cap$ である。上式はアトムの論理結合をファイルの集合演算に変換する規則であり、このまま 2 個以上の $FGET$ アトムが存在する場合に拡張される。各ファイル F_i 内

第6表 基本データベース操作関数

-
- i) $\text{AND}(F_1, F_2) = \{(x_1^i \cdots x_n^i) / (x_1^i \cdots x_n^i) \in F_1 \vee F_2, i = 1, 2, \dots\}$
- ii) $\text{OR}(F_1, F_2) = \{(x_1^i \cdots x_n^i) / (x_1^i \cdots x_n^i) \in F_1 \vee F_2, i = 1, 2, \dots\}$
- iii) $\text{MULT}(F_1, Y) = \{(x_1^i \cdots x_n^i, y^j) / (x_1^i \cdots x_n^i) \in F_1, y^j \in Y, i = 1, 2, \dots, j = 1, 2, \dots\}$
- iv) $\text{DIV}(F_1, X_k) = \{(x_1^i \cdots x_{k-1}^i, x_{k+1}^i \cdots x_n^i) / (x_1^i \cdots x_n^i) \times X_k \in F_1, i = 1, 2, \dots\}$
 $\text{DIV}(\hat{X}_k, X_k) = 1(\text{if } \hat{X}^k > X^k) \text{ or } 0(\text{if } \hat{X}_k > X_k)$
- v) $\text{PRJ}(F_1, X_k) = \{(x_1^i \cdots x_{k-1}^i, x_{k+1}^i \cdots x_n^i) / (x_1^i \cdots x_n^i) \in F_1, i = 1, 2, \dots\}$
 $\text{PRJ}(\hat{X}_k, X_k) = 1(\text{if } \hat{X}_k \vee X_k = \phi) \text{ or } 0(\text{if } \hat{X}_k \vee X_k = \phi)$
- vi) $\text{CLP}(F_1, X_k) = \{(x_1^i \cdots x_k^i \cdots x_n^i) / (x_1^i \cdots x_k^i \cdots x_n^i) \in F_1, x_k^i \in X_k, i = 1, 2, \dots\}$
- vii) $\text{EXPD}(F_1, X) = \{(x_1^i \cdots x_n^i x^i) / (x_1^i \cdots x_n^i) \in F_1, x^i \in X, i = 1, 2, \dots\}$

where

$$F_1; \{(x_1^i \cdots x_n^i) / x_r^i \in X_r, 1 \leq r \leq n, i = 1, 2, \dots, N_1\}$$

$$F_2; \{(x_1^{i'} \cdots x_n^{i'}) / x_r^{i'} \in X_r, 1 \leq r \leq n, i = 1, 2, \dots, N_2\}$$

属性集合を \mathbf{X}_i ; $F_i^* = F_i \times (\mathbf{X} - \mathbf{X}_i)$, AND-OR 構造の各ノードの Π/U を \wedge/\vee でおきかえ末端ノードの FGET アトムをファイル F_i^* でおきかえた集合演算の構造から求める統合ファイルを F^* とするととの論理式が,

$$(Q_1 x_1 / X_1) \cdots (Q_n x_n / X_n) (\text{FGET } F^*; x_1 \cdots x_n) \quad (8)$$

に帰する.

(2) この結果得られた単一アトム論理式は次の規則で第6表の基本オペレータによる操作手順に変換される.

$$\begin{aligned} & (Q_1 x_1 / X_1) \cdots (Q_{m-1} x_{m-1} / X_{m-1}) (\forall x_m / X_m) (\text{FGET } F^*; x_1 \cdots x_m) \\ & = (Q_1 x_1 / X_1) \cdots (Q_{m-1} x_{m-1} / X_{m-1}) (\text{FGET } \text{DIV}(F^* X_m); x_1, \dots, x_{m-1}) \end{aligned} \quad (9)$$

$$\begin{aligned} & (Q_1 x_1 / X_1) \cdots (Q_{m-1} x_{m-1} / X_{m-1}) (\exists x_m / X_m) (\text{FGET } F^*; x_1 \cdots x_m) \\ & = (Q_1 x_1 / X_1) \cdots (Q_{m-1} x_{m-1} / X_{m-1}) (\text{FGET } \text{PRJ}(F^* X_m); \end{aligned}$$

$$x_1, \dots, x_{m-1}) \quad (10)$$

ただし、 Q_i は V または \exists である。この規則を繰り返し適応すると最終的に (FGET \hat{F} : ϕ) が得られる。 \hat{F} は F^* に対する操作手順を示す。 \hat{F} はさらに第7表の最適化規則が適用される。第7表の規則は、関係データベースにおける最適化規則 [10] と一部の表現法を除き同じである。相違は関係データベースでは join 演算が基本演算になっているが、本方式ではこれは直積と集合積で表わされる複合操作となっているなどのためである。

第7表 最適化規則

Database Accesses - 3

- (1) $\text{DIV}(\text{MULT}(X, F), Y) = \begin{cases} \text{MULT}(X, \text{DIV}(F, Y)), & \text{if } X \neq Y \\ F, & \text{if } X = Y \end{cases}$
- (2) $\text{DIV}(\text{AND}(F_1, F_2), X) = \text{AND}(\text{DIV}(F_1, X), \text{DIV}(F_2, X))$
- (3) $\text{PRJ}(\text{MULT}(X, F), Y) = \begin{cases} \text{MULT}(X, \text{PRJ}(F, Y)), & \text{if } X \neq Y \\ F, & \text{if } X = Y \end{cases}$
- (4) $\text{PRJ}(\text{OR}(F_1, F_2), X) = \text{OR}(\text{PRJ}(F_1, X), \text{PRJ}(F_2, X))$
- (5) $\text{DIV}(\text{OR}(F_1, \text{MULT}(X, F_2)), X) = \text{OR}(\text{DIV}(F_1, X), F_2)$
- (6) $\text{PRJ}(\text{AND}(F_1, \text{MULT}(X, F_2)), X) = \text{AND}(\text{PRJ}(F_1, X), F_2)$
- (7) $\text{AND}(\text{MULT}(X, F_1), \text{MULT}(X, F_2)) = \text{MULT}(X, \text{AND}(F_1, F_2))$
- (8) $\text{OR}(\text{MULT}(X, F_1), \text{MULT}(X, F_2)) = \text{MULT}(X, \text{OR}(F_1, F_2))$

$$\begin{aligned} \text{例, } H &= (\forall x/X)(\exists y/Y)(\forall z/Z)[(\text{FGET } F_1 : x y) \cap (\text{FGET } F_2 : y z)] \\ &= (\forall x/X)(\exists y/Y)(\forall z/Z)[(\text{FGET } F_1 \times Z \cap F_2 \times X : x y z)] \\ &= (\text{FGET } \text{DIV}(\text{PRJ}(\text{DIV}(\text{AND}(\text{MULT}(F_1, Z), \text{MULT}(F_2, X)) \\ &\quad Z)Y)X) ; \phi) \\ &= (\text{FGET } , \text{DIV}(\text{PRJ}(\text{AND}(F_1, \text{MULT}(X, \text{DIV}(F_2, Z))))Y)X) ; \phi) \end{aligned}$$

$\text{PRJ}(F, X)$, $\text{DIV}(F, X)$ の演算は (1) 条件を満たす組以外の組はすべて消去すると同時に (2) 属性 X も消去するが、答を生成するため (1) のみを行なう VPRJ , VDIV が準備されている。

KAU ではここまでのシステムのインプリメントが済んでいる。以下にデータベース、アクセスを含む一例を示す。

例. 実験者が、与えられた数種類の液体資料のすべてに浮く新材料を探している。彼は候補材料を数種類入手し、どれが条件に合うかを調べる。実験の過程で新材料の質量と体積が別個に測定され、これが液体資料の比重と共にファイル化されている。P ; 候補材料, L ; 液体資料, R ; 実数とし、ファイルをそれぞれ $F_m(M, R_1)$, $F_v(M, R_2)$, $F_{sg}(L, R)$ と

1602 G

```

[E</PU][AY/LIQ](FLT, SX, SY)?
[1] (# FGET # WT PU1 RNUM9 *WRK1)
[2] (VPRJ *WRK1 RNUM9 *WRK1)
[3] (# FGET # VUL PU1 RNUM10 *WRK2)
[4] (VPRJ *WRK2 RNUM10 *WRK2)
[5] (AND *WRK1 *WRK2 *WRK3)
[6] (E<PD *WRK3 RNUM4 *WRK3)
[7] (# DIV RNUM4 RNUM9 RNUM10 *WRK3)
[8] (PRJ *WRK3 RNUM9 *WRK3)
[9] (PRJ *WRK3 RNUM10 *WRK3)
[10] (# FGET # SW LIQ2 RNUM5 *WRK1)
[11] (VPRJ *WRK1 RNUM5 *WRK1)
[12] (VPRJ *WRK3 RNUM4 *WRK3)
[13] (MULT *WRK1 PU1 *WRK1)
[14] (MULT *WRK3 LIQ2 *WRK3)
[15] (AND *WRK1 *WRK3 *WRK2)
[16] (# GT RNUMS RNUM4 *WRK2)
[17] (PRJ *WRK2 RNUM4 *WRK2)
[18] (PRJ *WRK2 RNUM5 *WRK2)
[19] (DIV *WRK2 LIQ2 *WRK2)
[20] (VPRJ *WRK2 PO1 *WRK2)
[21] (UUT *WRK2 *TTY)

```

第10図 実 例

する。これらの入力時に同時にファイル記述情報として

$$(\forall x/P)(\forall y/R)[(FGET; F_m, x, y) \rightarrow (mass \ x \ y)] \quad (11)$$

$$(\forall x/P)(\forall y/R)[(FGET; F_v, x, y) \rightarrow (vol \ x \ y)] \quad (12)$$

$$(\forall x/L)(\forall y/R)[(FGET; F_{sg}, x, y) \rightarrow (spc - grv \ x \ y)] \quad (13)$$

が知識ベースに入れられる。知識ベースには、この他に一般的な知識(1)や(2)が入っているとする。

この時、ユーザは計算機に関する知識は要求されない。たとえば $(\exists x/P)(\forall y/L)(flt \ x \ y)$ ：“すべての液体に浮く物質は？”のように聞くだけで良い。計算機は液体資料に浮く物体を示すデータが与えられていないので、知識ベースを探索し(1)の知識を用いて推論を行い、“浮く”ことは“比重が小”なることであることを知る。しかし材料の比重データも与えられていないため、さらに知識ベースを探し、比重の定義(2)を見出す。定義中の質量、体積は(11)、(12)からファイル化されていることを知り、以後その操作手順を自

動的に生成する。この実験結果が第10図である。第[0]行の質問を与えると、[1]~[21]まで、ファイル処理を行うプログラムが生成されている[22]。この手順は冗長なファイル・アクセスや処理を排除しているという意味で最適なものである。この結果を得るために要した時間は厳密に測定していないが、この程度の問題の答は遅れを感じることもなしに得られる。ただし、処理時間は知識ベース内の知識の数に依存するため、実用上の有意義な程度に複雑な問題について調べる必要がある。これは現在計画中である。

4.5 外部表現との関係および表現の拡張

述語形式ではアトムが言語における単文、句、節等に対応していること、論理式はこれから構成される複文や重文に対応することはこれまでの例から明らかであろう。この意味で述語形式と言語とは近い関係にある。しかし、一階述語で表わせるのは自然言語文のごく一部であり、第1表の(C1)の条件を満たしていない。この問題を解決するには高階述語の採用が不可欠である。一方、高階論理のもとでの推論は極めて能率が悪い[6]。この拡張はできるだけ限定したい。そこで、どの程度の拡張が必要かを言語の中に表われる基本構文パターンの種類から求める。それは構文パターンが語の意味と共に文の意味を定めるからである。構文パターンのうち“太郎は……することを考える”や“花子は……を見に行く”など、英文の that-clause や to-infinitive に相当する文を一階述語で表わすことは難かしい。これらには主動詞を含む文、“太郎は*を考える”の*の部分に、対象の代わりに、それ自体が論理式に対応する文が入っている。そこで知識の表現が扱う対象の集まりである世界を拡大し、“論理式”というタイプの集合を含むようにする。これにより知識表現は実質的に

(述語記号 変数/論理式, …… , 変数/論理式)

なるアトムを含むネスト形式に拡大する。英文の場合、基本的な構文パターンは25種程度であるが[5]、この拡張によりこれらのほぼすべてを含むことができる。例えば次の様に用いられる。

($\exists x$ /PERSON)(see Taro (fall x roof)); “太郎は誰か(人)が屋根から落ちるのを見る”

“論理式”タイプの変数が許されるのは関係語のうち、see, tell 等の語に対応するアトムのみであり、add, in 等には許されていない。これは語の定義(5・1・1参照)で指定される。

もう一つの拡張は述語記号を他の変数と同じ様に代入可能な変数として扱う。これは

($\exists x$ /RELATION)(x Taro Jiro) “太郎と次郎はどんな関係か?”のような質問に対応する。

4.6 拡張に伴う各種変換規則への影響

表現の拡張は任意に許されるものでなく、対応する変換規則の拡張の可能な範囲に限定される。

特に推論規則の拡張が可能でなくてはならない。表現の拡張に伴う推論規則の拡張は全体象の集合の構造の分析に基づき[17]、TICにおいてアトム内の通常変数に対しては第1表を適用し、内部論理式に対しては知識と仮説内の対応する内部論理式を(\dots)_x, (\dots)_yと

すると第1表の関係をこの含意関係 $(\dots\dots)_X \rightarrow (\dots\dots)_Y$ でおきかえる。このために推論規則を再帰的に適用するように制御機構を導入する。

4.7 情報の信ぴょう性

実世界で用いられる情報の多くは曖昧さを含む。従って、このような情報を正しく利用する方法が必要である。このため、情報の論理的処理と並行して曖昧さを含めた情報の信ぴょう性評価の体系を準備している。

信ぴょう性評価の扱いには情報の論理的関係と、信ぴょう性の程度を分離して扱う方法(確率)、この両者を一体に処理する方法(様相論理, ファジー論理)[7][22]などがある。後者の場合、計算機上での実現に際して知識表現に要するメモリ量や推論処理に時間を要することが予想される。これに対し、確率的方法は手法が単純であるため、KAUではこの方法を採用している[14]。

確率的評価は論理的表現を確率事象としてこれに $[0, 1]$ 内の値を割当て、論理的処理による事象の変化に応じて確率(信ぴょう性)の変化を計算する方法を与える。

情報の信ぴょう性に関わる要因には少なくとも以下の四種が含まれる。

- (1) 知識そのものの不確実性
- (2) 知識の内容に含まれる不確実性 ($A \rightarrow B$ においてAは正しくてもBが確実でないという種類の知識)
- (3) 集合の概念における $x \in X$ の曖昧性 (Fuzzy 集合)
- (4) データベース内のデータの不確実性 (実験値の誤差など)

ここで(2)の $A \rightarrow B$ は条件付確率の事象そのものであり、“Aを正しいとした時のBの信ぴょう性”を $P(B/A)$ と表わす。

もし、A自身の信ぴょう性が $P(A)$ なら、Bのものは $P(B) = P(A) \cdot P(B/A)$ と条件付確率の定義で与えられる。また(3)はFuzzy論理におけるメンバ関数 $\mu(x)$ を確率と見なしこれをTICにおける含意条件の成立する確率と解する。推論処理は含意関係を辿るプロセスであるから、結論の信ぴょう性は用いた知識の信ぴょう性と含意条件に関する信ぴょう性の積として求まる。このように信ぴょう性評価は、SBDSのもとで極めて単純であり、第8図の1サイクルごとに用いた知識の上記4種類のすべて或いは関連する信ぴょう性を乗ずればよい。もし、推論途中でこの値が前もって定めたいき値より小になったら、その推論の筋道は捨てる。これにより“風が吹けば桶屋が儲かる”式のナンセンスな結論を防ぐことができる。

5. 結 論

本稿では、前半に将来の計算機が環境の変化に応じてどのように変化せねばならないかを分析しそれに対応するために必要な条件を求め、さらにそれを満たすための機能を求めた。次いで、この分析に沿って現在開発が進んでいる知識型システム(KAU)の概要を述べた。KAUは一階述語を用いる基本部分は推論アルゴリズム、プログラム生成、データベース・アクセスを含めてインプリメントされ、現在表現の拡張、外部表現からの変換、情報の信ぴょう性評価の部分が進行中あるいは検討中である。これと同時に実用問題への適用の具体的

な計画が進行中である。

本論文は、新しい計算機のあり方に対する一つの考え方を述べたものでありそのためには細部の詳述より環境への適応性やそのための全体の構成に重点をおいた。それでも紙数の都合で省略した部分も多い。これらの点や必要な細部については別の機会に報告したい。

なお、KAU システムは東大宇宙航空研究所、山内平行、大学院生宇田川佳久、両君の努力によりインプリメントされたものである。

1979年12月21日 航空力学部

参 考 文 献

- [1] D.G. Bobrow (chaired); A Panel on Knowledge Representation Proc. 5th IJCAI pp. 983-992 (1977).
- [2] E.F. Codd; A Relational Model of Data for Large Shared Data Banks, Comm, ACM, Vol. B, No. 6, pp. 377-397 (1970).
- [3] E.F. Codd; Recent Investigation in Relational Data Base Systems, Information Processing pp. 1014-1021 (1974).
- [4] G.G. Hendrix, Expanding the Utility of Semantic Networks Through Partitioning, 4th Int'l Joint Conf. on AI (1975).
- [5] A.S. Hornby; Oxford Advanced Learners Dictionary, Oxford University Press, Third ed. (1975).
- [6] G.P. Huet; A Mechanization of Type Theory, 3rd IJCAI, pp. 139-146 (1973).
- [7] R.C.T. Lee; Fuzzy Logic and the Resolution Principle JACM Vol. 19, No. 1, pp. 109-119 (1972).
- [8] R. Schank; "Finding the Conceptual Content and Intention in an Utterance in Natural Language Conversation", *Second Int'l Joint Conf. on AI*.
- [9] J.A. Robinson "A Machine Oriented Logic Based on the Resolution Principle", *JACM* Vol. 12, Jan. 1965.
- [10] J.M. Smith & P.Y.T. Chang; Optimizing the Performance of a Relational Algebra Databases Interface, *CACM* Vol. 18, No. 18, pp. 568-579.
- [11] J. Mylopoulos, P. Cohen, A. Borgida & L. Sugari; Semantic Networks and the Generation of Context, 4th Int'l Joint Conf. on AI (1975).
- [12] 大須賀節雄; 知識構造に基づく機械的推論規則について, 情報処理, vol 17, No 12, (1976).
- [13] 大須賀節雄, 山内平行; 推論能力を備えた情報検索方式について, 情報処理, vol 18, No 8 pp 789 ~ 798 (1977).
- [14] S. Ohsuga; Semantic Information Processing in Man-Machine Systems, Proc. 1977 IEEE Conf. on Decision & Control, pp. 1351-1358 Dec. 1977.
- [15] 大須賀節雄; 意味処理と知識利用のシステムについて, 日本語処理情報シンポジウム前刷(1978).
- [16] 大須賀節雄; 知識の表現と利用—知識システムの満たすべき条件, 情報処理, vol. 19, No. 10, (1978).
- [17] S. Ohsuga; Theoretical Basis for a Knowledge Representation System, Proc. Int'l Joint Conf. on AI (1979).

- [18] S. Ohsuga; Toward Intelligent Interactive Systems. Proc. The IFIP W.G. 5.2 Workshop Seillac II on Methodology of Interaction, Noth-Hooland, (1979).
- [19] M.R. Quillian; The Teachable Language Comprehender: A Simulation Program and Theory of Language, Comm. ACM, Vol. 12, No. 8, (1969).
- [20] J.A. Robinson; A Machine Oriented Logic Based on the Resolution Principle, JACM Vol. 12, Jan. (1965).
- [21] N. Roussopoulos & J. Mylopoulos; Using Semantic Networks for Data Base Management *First Int'l Conf. on VLDB*, 1975.
- [22] 深海悟, 水本雅晴, 田中幸吉; ファジィ推論について, 電子通信学会論文, Trans. TECS' 78/8 vol. 61-D No. 8 (1978).
- [23] S.C. Shpiro; "A Net Structure for Semantic Information Storage, Deduction and Logic, *Second Int'l Joint Conf. on AI*.
- [24] T. Winograd; "Understanding Natural Language", *Academic Press*, 1972.
- [25] 宇田川佳久, 大須賀節雄; 知識システムの設計問題への応用について, 情報処理学会第20回全国大会予稿集.
- [26] 山内平行, 大須賀節雄; 知識システムにおけるデータベース・アクセスについて, 情報処理学会第20回全国大会予稿集.