数理解析研究所講究録
第593巻 1986年 77-118

知識表現に関する一考察

A Consideration To Knowledge Representation

大須賀 節雄

Setsuo Ohsuga

東京大学工学部境界領域研究施設
153 東京都目黒区駒場4－6－1

Abstract

Knowledge processing is of use if it can undertake a part of jobs man has to do. As well, since conventional style of information processing has already been established, knowledge processing must show its capability in information processing over the conventional style processing. In order to show the usefulness of knowledge processing, therefore, it is quite necessary to enhanse the special features of knowledge processing so that it can apply to the application problems to which the conventional style computers are not well suited.

In this paper, the author focusses attention and discusses on knowledge representation language from this point of view, because knowledge represetation is the key concept of knowledge processing. The requirements for the knowledge representation language is discussed first, taking the way of applying knowledge systems to large scale development problems into consideration.

Then the property of the knowledge representation language is discussed. It is strongly suggested that it must be the purely declarative language. The attention is espetially directed to the data structure in the declarative language. Finally, an expansion of predicate logic is proposed as the knowledge representation language in which structural concept is combined with property description.

１．　はじめに

計算機は人の行なう情報処理の一部を受け持つことにおいて価値が与えられる。人工知能も計算機による情報処理の一分野である以上同様である。それに加え、すでに現行方式の計算機の技術が確立し広く普及しているので、人工知能にはさらに、現行情報処理方式の限界を越えた部分でのみ価値が認められる。従って、人工知能の技術を発展させるうえで努力すべきことは、方式的に現行の計算機による情報処理と異なり、しかもそれが大きな効果を示す人工知能の特徴的な機能を強化し、その利用法を開発することである。

人工知能方式の処理が現行方式と利用上はっきりと異なり、効果を持つと期待されるのは、情報を個々のモジュールとして相互に独立に表現したものを蓄え、必要に応じてそれらを内部で問題解決手順に組み立てるという、新しい情報処理の形態、いわゆる知的な機能を可能にすることによって、環境に動的に適応し、また試行錯誤しながら創造的活動を行なう人のプロセスを支援し得る点にある。

このように知的ということの背景には記憶機能がある。一定の構成規則に従って形式化され、計算機が推論などの手法により自律的に問題解決に用いることの出来るモジュール化された記憶情報を（計算機における）知識と呼ぶ。知識のモジュール化を可能にする基本条件は、情報の表現が非手続き的に行なわれるとともに、その情報を目的に合わせて組み立てる処理機構が存在することである。知識を相互に関連づけて問題解決に利用する機能を推論と呼ぶ。以下では知識に基づいて処理を進めることを知識処理、知識処理を実現する計算機システムを知識処理システムと呼ぶことにする。本稿の目的は知識処理の本質を追求し、知識処理システムのありかたを考察することにある。

知識処理といっても、この際計算機が扱えるのはあくまでも情報の形式であって意味ではない。言葉を換えれば、知識を計算機によって処理するとは、知識を表わす形式の変換規則を通して知識の内容の処理を行なうことである。従って知識処理を行なうには、まず広い範囲の問題に含まれている意味を表現し、かつ、形式的な変換規則によって内容の処理が正しく行なわれることを保証

する形式を求めることが最も基本的な、そして重要な課題である。これは通常知識表現の問題と呼ばれる［11］。これは広義の言語の問題である。

１．１　情報処理方式における言語とその処理

このように情報処理の基本方式を定めるのは、（１）物事を表現するために、その方式に対して定義されている記述言語と、（２）その言語による表現を操作する機構、である。

現行計算機では言語として手続き的に物事を表現する手続き型言語が、操作機構としてはこの言語で表現されたプログラムを忠実に実行する制御．演算装置が与えられ、関数型計算機ではラムダ計算の記法と合成規則からなる関数型言語と、その記号的な変換規則を実行する処理系が作られる［２］。一方、知識処理方式では言語が前記の知識表現言語であり、操作機構として推論機構が用いられる［13］。個々の方式は、定義された言語で記述することが出来、かつその言語に対して定義された操作機構がうまく働く問題に対して最も効果的に利用され

る。

情報処理方式の特徴はこれら言語と操作機構の比較によってより明らかになろう。更に言えば操作機構は言語に依存して定義されるから、言語の中にそれぞれの方式の特徴が集約されているともいえる。従って、知識処理においてもっとも本質的な問題は優れた知識表現言語を開発することである。ただし現状では、知識表現は様々な形式の総称である。前２者と比べて任意性が残っていて、優劣を比較する根拠すら乏しい。知識処理を情報処理の新しい体系的技術として確立するためには正当な理由の下でこれら知識表現を評価．比較する為の根拠を見いださねばならない。

１．２　知識処理に期待される応用

現行情報処理方式の下で解決できない問題あるいは現行方式が適切でない応用とは何であろうか。あるいは逆に現行計算機がうまく適合する種類の問題はどのような性格のものであろうか。論点をはっきりさせるために単純化して示すと、これらの問題は表現され、定式化され、

分析され、最後に、求まった解に対して実行可能な操作手段を適用して結果が求めて終了する、という比較的単純なライフサイクルを持っているものである。計算機が出現するまではこの全ての過程を人が行なっていたが、Turing-von Nuemann型計算機の出現後はこの最後の段階のみは計算機でも実行出来るようになった。しかし計算機の普及に伴って、このスタイルを維持する為の人間側の負担が次第に過大になり、ソフトウエア危機とまで言われる様になってきた。これは明かに現行の処理方式の下では解決の困難な問題である。

他方、現実の世界で発生する問題は複雑かつ多様であって、前記のものとは異なった定式化を必要とするものが多くある。これは上述のような（途中で人の誤りさえなければ）一回の処理プロセスで終了する問題と異なり、計算された結果に基ずいて分析をやり直し、再び計算を行なうという試行錯誤のサイクルを繰り返すことによって、初めて解が得られるという型の問題、いわゆる非決定型の問題である。現実の問題の中には非決定性を部分的に含むものは多いが、問題全体が非決定的な処理を要するものの解を見いだすことは多くの困難を含む。図１

にこの状況を示す。この図は技術分野での設計手順の概要を示したもので、多くの技術分野に共通性の高いものである。フィードバック・ループを含む全過程が非決定型の問題を表わす。個々のブロックはさらに細分化され、多くのより小さなブロックが並列にあるいは直列に配置されてプロセス自身が複雑な構造を形作る。それぞれの小ブロック内、あるいはブロック間にもフィード・バックが頻繁に行なわれる。

最近の応用分野からの新しいニーズにはこの非決定型のものが多い。これらのニーヅは例えば機械設計、ＶＬＳＩ設計、高分子材料設計、合金設計など各種技術分野での開発．設計問題、政策や経営の意思決定問題、社会システムや技術システムなどを含めて今後増加すると予想される複雑なシステムの制御問題、技術者教育を含む教育問題、言語理解や図形理解を含む大規摸コミュニケーション問題、などで既に発生し、あるいは近い将来発生すると予想されるものである。

従って人工知能研究の目標は、上記の応用に適した知識表現言語ならびにその処理系の開発およびその特徴を最大限に生かす応用技術の開発にあろう。本稿ではこれ

らの問題を論ずるが、紙数の都合で意を尽くせない部分があり、参照文献、特に今日の多くの知識表現に含まれている問題点を述べた［20］と、知識表現の問題を主としてマン．マシン．コミュニケーションの立場から述べた［19］を御参照頂きたい。

＊＊＊＊＊図１＊＊＊＊＊

２．大規摸応用問題への対応

前節においてソフトウエア開発と一般の非決定型問題を分けて示したが、問題の基本的な性質が著しく異なる訳ではない。というより形式は多少異なるが、ソフトウエア開発も非決定型問題であるという意味では共通性を持つ。それと同時にソフトウエア開発の問題には対象領域が特定されないという特徴がある。後に述べるように、大型の非決定型問題の解決、特に開発型の問題には対象モデルを構築し、それを評価しながら目標を達するという方式がほとんど例外なしに用いられるが、ソフトウエア開発の問題は対象モデルの概念が確立しにくいことが

他の問題と異なる点であり、今後の課題として残されている。しかし開発型の問題に人工知能をどのように適用するかという基本課題を解決しておくことは、これらすべての問題に共通の必要条件である。

ここで重要なことは非決定型問題のうちでも（１）実用規模の複雑さを含む、（２）開発．創造型の問題にいかに対処するかである。非決定型の問題解決の方式自体は人工知能、とりわけ知識工学の本来の目的として研究が進められてきた。　　しかし今日までに開発されてきた知識工学システムは、人工知能研究の結果ようやく機械化のためのアルゴリズムが求まった演えき的推論の機能を、知識ベース用の比較的単純な記憶管理機能ならびに対話型入出力機能と組み合わせてソフトウエア　システムとして実現したものであり［13］、開発より知識の伝達を主目的とし、知識表現自体も非常に単純なものである。従来のプログラム作成方式に比べると、比較的簡単にシステムを実現でき、環境の変化に対応し易く、小回りがきくことなど、それなりの長所を備えているので、この枠組みの範囲で扱える問題を適切に選べば効果が期待されるが、演えきと知識ベースを単独で利用する

システムはいわば単能機に近く、大規摸の開発問題を扱うことは難しい。

開発問題とは与えられた性能等に関する要求を満たす対象を見いだすことであるが、大規摸で複雑な開発問題の処理は人間でも困難な問題である。人はこのための方法を永年掛けて開発してきた。それがモデルを構築し、それをすべての中心に置く体系的処理方式である［15］。この基本的手法は、（１）仮のモデルを作る、（２）そのモデルを評価し、与えられた要求を満たすかどうかを調べる、（３）要求を満たせば終了し、そうでなければモデルを修正して（２）に戻る、というものである。しかし対象の規模が大きく、対象が複雑な開発問題の場合には、このように最初から全対象の詳細モデルを作ることは不可能で、ラフ．モデルから出発して、上記サイクルを繰り返して要求の一部を満たすことを確認し、次いでそのモデルに情報を付加してモデルを詳細化し、再び上記サイクルを実行して未だ満たされていない要求を満たしてゆくという、成長方式をとるのが普通である。このような’逐次的な情報の付加．削除によるモデルの修正と詳細化’を、’成長過程にある（未完成）モデルの

評価’と組み合わせた形式で進めてゆく方式が、複雑な大規摸応用に対する殆ど唯一の攻略法だからである。図１にはこの状況を簡単に示している。概案設計の段階では外形とか全体構造を簡略化した極く粗いモデルが作られ、基本設計、詳細設計と進むにつれ、細部構造や材料、加工法その他の情報を含む詳細モデルが形成されてゆく。

この過程で様々な異なった形態の処理が行なわれ、異なった知識が利用される。さらに効率的に目標に達するような全体のプロセスの制御は一般には複雑である。

このような現実の問題を処理するために人が発揮している機能に比較すると、現在の技術レベルの下での知識処理の機能はあまりにも単純である。上記の作業を遂行する上で人は様々な高度の推論機能を駆使している。これらは今日計算機技術として漸く実現された演えき推論を越えたものである［18］。従って、これらの一連の作業を知識処理システムによって自動化することは困難である。知識処理システムとして今日努力すべきことは、人と協力して上記のように発達的にモデルを構築．成長させてゆくプロセスを支援して、その効率化を可能にす

る支援システムの実現にある。それにはこのプロセスに含まれる多様な機能の中で、知識処理技術で実行可能なものはシステムが引き受け、それ以外は人に任せるという協調的な処理方式を最小のオーバーヘッドで実現することである。

例えばラフ．モデルの段階では詳細構造は無視され、モデル構成要素として扱われていた部分が、モデルの詳細化にさいし、まだ満たされていない残りの条件を満たすように部分構造で置き換えられたり、条件を満たさない構造を他の構造に変更するようなことが繰り返されるが、複雑な開発問題において知識処理が役立つということは、少なくともそのような処理の一部を知識処理システムが受け持つことである。開発とは要求の記述を構造を含む対象モデルの表現に変換することとすると、それを支援する知識自体も要求として与えられる性質等の記述を構造に変換する機能や、構造を性質等の記述に変換する機能、あるいはそれを組み合わせた、ある性質－構造の組を異なる性質－構造の組に変換する機能などを含まねばならない。逐次的に情報を与えながらモデルを評価し詳細化するための絶対的とも言える条件は知識が完

全にモジュール化されていること、そのために知識表現言語は純粋に宣言的な言語であるべきことである。

３．知識表現言語への要求

知識表現への要求はこれだけではない。筆者は［18］や［19］で上記の協調的な開発支援システムを実現するためには、少なくとも（Ａ）計算機内に人と計算機が同時にアクセスし、逐次的に情報を与えながらこれを成長させることのできる対象モデルを表現すること、（Ｂ）従来は人が計算機外でモデルを表現し、処理して来たが、これに比較して特に不便を感じない程度の人間－機械間の対話能力を保証すること、が満たされねばならず、さらに（Ｃ）システム内の諸機能を統合化することによって最大限に利用し、協調的な開発プロセスを効果的に進めるための制御機能を持つこと、が必要である事を述べた。このうち（Ａ）と（Ｂ）はともに知識表現言語に関する要求であり、上記文献においてはこれらの要求事項を分解して重要な項目を挙げた。この議論はここでは繰り返さないが、以下の議論にも関係するので項目を表１

に示す。

なお、[19]においては主として表1（B）のマン．マシン．コミュニケーションの立場から知識表現言語が宣言型言語であるべきことを主張し、宣言型言語の在り方を知識表現としての面から探った。一方、表1（A）のモデル化の面からも同様の結論が導かれることは、前節に述べた通りである。そこで本稿では表1の（A）の部分にも（B）（6）と同じ（1）の条件を挙げた。条件（2）、（7）の推論機能に関しても同様で、モデルの構築にも、会話の理解にも知識を利用するための機能として、推論が不可欠であり、知識表現言語としてそれに適したもので有らねばならない。

これ以外の表1の諸条件について簡単に述べておこう。条件（3）は開発を初めとして多くの実用問題では複雑なモデルの表現が必要であり、言語としてこの記述が可能なこと、条件（4）はこのモデルを処理するという形式で問題解決法が定義されているので、このプロセスを表現出来ることを要求する。条件（5）は逐次的にモデルを成長させる方式は非常に柔軟性に富むため、入力源が時間的にも空間的にも広がり、しかもそれらから入力

が相互に無関係に与えられることもあるので、システムとして矛盾等が生じないことを保証するために極めて重要な条件である。

条件（８）は知識表現言語は自然言語や図形など、人が用いる外部言語を受け入れるべきこと、（９）は宣言型言語として定義された知識表現言語は計算機機構を直接に駆動することは出来ないので、計算機用の内部言語に変換出来ねばならないこと、をそれぞれ要求している。

このように（Ａ），（Ｂ）両条件共に知識表現言語が宣言型であるべきことを要求しているが、この両条件からの要求の動機は異なる。すなわち前者の要求は、人の用いる外部言語が宣言型であり、知識表現言語はこれとの間の言語変換を可能にするための条件として。この要求は記号処理が可能な言語によって満たされる。これに対し、後者の要求は単純に文字列の記号処理を行なうのみでは不十分である。すなわちこの言語は要求記述と対象の具体的な記述、特に構造記述を一体として行なえるものでなくてはならず、更に工夫が要る。これは（Ａ）の条件（３）、（４）に関連する。本稿では６節でこの

議論を通して知識表現言語に関する条件を更に追求する。

４．知識表現言語としての述語言語

表１の諸条件を満たす基本的な言語の枠組みとしては述語論理が適切であることが［８］で論じられた。この理由は、述語言語が宣言型言語であること、述語論理という確個たる理論的背景を持ち、入力の無矛盾性のチェックが可能であること、言語のセマンテイックスが外部言語と同じ型であること、述語言語で表現された論理式の計算機用の手続き的解釈が可能であること[9]、理論的に完全性の保証された推論方式が開発されていること[1]、による。本稿の最初に、知識処理は知識の形式的処理によって内容の処理を行なうことと述べたが、これを保証するには形式的処理と意味的な処理の等価性が証明されていなくてはならない。完全性は理論的にこれを証明している。

これらの特徴は複雑な問題を記述せねばならない基本言語として望ましいものである。しかし述語言語には対

象構造の概念はない。まして、既に述べたように、成長型のモデル構築方式を実現する為には、要求としての属性や性能などの記述と、構造を含む対象そのものの記述を同一の表現に含める形式が望ましいが、そのようにはなっていない。従ってこのままでは必ずしもモデル記述を含む知識表現言語としては適さない。そのため述語言語のシンタックスの拡張が必要である。この拡張は、それによって'要求を満たす構造の表現'を見いだしたり、'与えられた構造を持つ対象の性質の表現'を可能にするものである。

言語の拡張に際しては、述語論理の持つ上述の性質を保存することが必要である。そうでなければ、述語言語を選んだ理由が失われてしまうからである。前述のように、この拡張は言語記述の中にデータ構造を含むものになる。従って、この拡張に際してはデータ構造の体系化が要求される。ここには2つの問題がある。データ構造に述語論理と同様の理論性を見いだせるか否か、およびデータ構造を述語言語とどのように融合させるかである。

5　集合とデータ構造

5．1　対象の記述

データ構造は記述系の一部として、述語論理と同程度に、知識処理システムの基本的概念の形成に重要である。どのような形式であろうと、記述系は'記述される対象'と'それに関する記述を行なう部分'の２つの要素から構成される。言語ごとのセマンティクスの相違によって表現の具体的な形態は異なるが、記述がこの２つの要素を含むことは本質的なことである。

例えば現行情報処理方式はプログラムによって問題を記述するが、それは処理の対象であるデータ部分と処理を行なう手続きの部分から成っている［10］。知識処理でもこれは同様である。述語論理では基本要素は項と述語である。項は記述される世界の実体に対応するものとして定義され、述語はその属性等についての記述である［１］。述語論理の議論では、これらの概念が、記述しようとする世界内の実体や属性等と対応するという構造が重視されるが、実体によって構成される世界そのもの

の構造については触れていない。述語論理は'記述系'の構造に関する理論であって、'記述される世界の構造'には立ち入らない。世界の構造は必要に応じて述語によって表現することとされている。

これは構造をすべて言語で表現することと同じである。複雑な対象の構造を言語で表わすことは原理的には可能であり、構造の一部分を参照するような時には必要なことでもある。しかし、実用問題にはこれでは不向きである。例えば、設計分野からは既に次世代のＣＡＤ／ＣＡＭとしてビルディング、航空機、原子炉といった設計対象の完全モデルを計算機内に実現する要求があるが[25]、これら設計対象の構造をすべて言語的に表現することは事実上不可能である。これまで人は技術分野で様々な図形的な構造表現法を案出し、それによって技術が進歩してきた。もし構造のすべてを言語で記述しなければならなかったら、扱われる対象は単純なものに限定され、技術の進歩はなかったであろう。

これまで述語論理で扱われてきた問題は、この意味で対象が単純なものになりがちであった。実際の応用問題では処理対象である実体が一般に複雑なものになるし、

知識処理が新しい型の情報処理技術に成り得るとしたら、処理対象も広い範囲にわたるから、その記述が可能なものでなければならない。知識表現言語がめざす宣言的な表現方式では、言語による表現が記述される世界内の概念に、直接対応する。従って、この言語で記述力を保証するためには、記述対象の世界内に実際に存在する様々な構造に対応できるだけの十分な概念が、前もって準備されていること、そのために記述系を構成する上記の2番目の基本要素について検討しておくことが必要である。

宣言型言語では文章の意味が単語の意味と文法規則から構成されていく。しかし単語の意味は理解における意味の基本要素であり、それ以上分解できない。これと同様に、一般の構造が表わす意味も基本の構造要素と構造化の形式で表わされる。このような構造要素は固有の意味を持っていて、それ以上分解することは出来ないものである。このように意味的に基本的な概念を表わす構造は少なくとも何通りか存在する。当面の問題はそのような基本構造はどのようなものかを明かにすることである。

５．２　公理的集合論

このような議論は実は数学の世界で早くに行なわれていたものである［22］。数学においては記述の形式化が重要であるため、論理学とは早くから関わっていたが、ブール、フレーゲなどの論理学者によって形式的演えきの形が与えられて以来、数学において記号主義が重要になった。数学のどの領域も述語記号を適当に選べばこの述語言語に翻訳が可能であると言われるようになった。一方、解析の分野では微分．積分を適切に取り扱うためには無限集合の概念が必要であり、その分析が行なわれた。今日、これらの結果から言えることは、数学＝論理学＋集合論であることが示されたことである。知識処理システムが今後当面する問題の中には、数学という手段を通して解決されるものが多いが、その手段を適切に利用するには問題を表現する記述系に数学的な意味での集合の概念が含まれていなければならない。

本稿で数学の議論をするつもりはない。しかし、上記のように知識処理において対象の記述の方法を確立する

こと、しかもそれに述語論理と同程度に理論的な根拠を与えることを目的としたとき、数学の分野で集合を定義するために行なわれて来た手法が参考になる。本章ではこれを数学の分野での本来の目的とは少し違った面で参考にする。われわれの目的は、知識処理においてデータ構造の基本要素としてどのようなものが必要であるかを知ることである。

集合論の確立のために数学において何が行なわれてきたかを辿ってみよう。問題は数学の体系を定義する基盤としての集合の概念を確立することである。この集合は数学のいかなる領域を対象としても、その定義を制約しないように十分に大きなものであることが望ましい。

問題は記述系として述語論理が形式化されたように、集合を形式的に定義することである。さらに言えば集合も述語言語で表現される幾つかの公理で定義することである。これは公理的集合論と呼ばれる。ではどのような公理によって集合が定義できるだろうか？　集合自体は対象の集まりであるが、この集合を述語で表わされる公理に結びつける直感的な方法は、ある性質に注目したとき、この性質を持つ全ての対象からなる集合が存在する

ことを仮定することである。これは任意の性質について成り立たねばならない。この性質は述語　Ｐ（ｘ）　としよう。すると任意の性質　Ｐ（ｘ）　にたいし

　ｘ∈ｙ　<－>　Ｐ（ｘ）

なる集合ｙが存在する。

　これは正しいように見えるが実は正しくない。Ｐ（ｘ）は任意だからこれを　ｘ∉ｘとしよう。これによって定義される集合をＲとする。すなわち

　ｘ∈Ｒ　<－>　ｘ∉ｘ

である。このように極めて形式的に定義されたＲは、言葉で言えば”それ自身の要素でない集合全体からなる集合”ということになるが、このような集合があったとして、これについて”ＲはＲの要素であるか”という質問をしてみよう。この答えは存在しない。なぜならそれは

　Ｒ∈Ｒ　<－>　Ｒ∉Ｒ

を満たすものであるが、これは明らかに矛盾である。

　これはラッセルの逆理として良く知られたものであるが、これは任意の性質を満たすものの集まりとして集合が定義されるという最初の直感が誤りであることを示し

ている。これは集合の定義としては大き過ぎるのである。そこでこの逆理にかからない範囲で出来るだけ大きい集合を定義する別の方法を考えねばならない。

われわれが興味をもつのは、これに替わる方法として取られている、集合の生成的な定義法である。簡単に言えばこれはまず集合を与え、これを基にして新しい集合を生成し、その結果を集合の定義に加えてゆくことによって集合を拡大してゆくというもので、この生成に何通りかの方法が与えられ、それらが公理化されている。

観点を替えてこれを情報処理の立場でみると、既存集合から新しい集合を生成するこれら生成規則はデータ構造化の要素になる。これらが公理化され、しかもそのように定義された集合が数学の全体系を記述するうえで支障のない程度に十分に大きい集合になっている。従ってこの生成規則をデータ構造化メカニズムとして含むシステムを開発すればわれわれが望むシステム、すなわちデータ構造を含めて理論的に保証された体系を持つ情報処理システムが得られるであろう。事実、これは可能である。先ず、以下にこのような公理を挙げる [4,22] 。

(1) 外延性公理：

$(\forall x)(\forall y)[x=y \Leftrightarrow (\forall z)[z\in x \leftrightarrow z\in y]]$

正確に同じ要素をもつ２つの集合は相等しい。

(2) 内包性公理：

$(\forall u)(\exists v)(\forall x)[x\in v \leftrightarrow x\in u\wedge P(x)]$

集合u と述語論理式P(x)が与えられたとき、u の要素で性質P(x)をもつもののすべてを含む集合v が存在する。

（実際にはこれは(6) に含まれる）。

(3) 対集合公理：

$(\forall x)(\forall y)(\exists z)(\forall t)[t\in z \leftrightarrow t=x \vee t=y]$

２つの集合 x と y が与えられた時、要素が丁度 x, y である集合 {x, y} がある。

(4) 巾集合公理：

$(\forall x)(\exists y)(\forall z)[z\in y \leftrightarrow z\subset x]$

与えられた集合のあらゆる可能な部分集合を要素とする集合がある。

(5) 空集合公理：

$(\forall y)[y\notin \emptyset]$

要素を持たない集合が存在する。

(6) 置換公理（図式）：

$(\forall x)(\forall y)(\forall z)[P(x, y) \wedge P(x, z) \rightarrow y=z]$

$\rightarrow$

$(\forall u)(\exists v)(\forall y)[y \in v \leftrightarrow (\exists x)[x \in u \wedge P(x, y)]]$

述語論理式P(x, y) と集合u が与えられたとき、u に属する各集合（集合の要素は一般には集合である。（４）参照）x にP(x, y) の下で対応する集合y 全体からなる集合v が存在する。

(7) 正則性公理：

$(\forall x)[x \neq \emptyset \leftrightarrow (\exists u)[u \in x \wedge u \cap x= \emptyset]]$

集合－要素関係のいかなる下降列も有限である。（集合の要素が集合であることから、－－ｘ１∈ｘ２∈ｘ３－－のように続く下降列が存在することになる。この下降列が無限に続くと矛盾を生ずるためこれは有限で終わらなくてはならない）

(8) 和集合公理：

$(\forall x)(\exists y)(\forall z)[z \in y \leftrightarrow (\exists u)[u \in x \wedge z \in u]]$

任意の集合の和集合は集合である。

(9) 無限集合公理：

$(\exists u)[\emptyset \in u \wedge (\forall x)[x \in u \leftrightarrow x \cup \{x\} \in u]]$

無限集合が存在する。

(10)選択公理：

$(\forall z)[(\forall x)[x\in z \rightarrow x\neq\emptyset]\wedge$

$(\forall x)(\forall y)[x, y\in z\wedge x\neq y \rightarrow x\cap y=\emptyset]$

$\rightarrow (\exists u)(\forall x)[x\in z \rightarrow (\exists t)[u\cap x=\{t\}]]]$

あらゆる集合は選択関数を持つ。選択関数とは集合x の空でない各要素（一般に集合である）y に対しy の１要素を対応ずける関数である。

５．３　データ構造としての解釈

上記各公理にたいしその述語による表現を付けたが、われわれの目的は数学の研究にある訳ではないから、これは参考までに示したまでである。以下では、これらの公理の情報処理におけるデータ構造の面からの解釈を行なってみる。基本データ型として集合を含むことが前提である。

（１）外延性公理は２集合の相等条件として、集合型データ型に関して既に用いられている。

（２）内包性公理は集合の概念と性質等の概念の等価性を与えるものとして重要な意義を持つ。（４）に示すよ

うにここで集合は階層構造を含む概念であり、上記のことは、例えば与えられた構造を持った対象の性質を求めるとか、逆に要求された性質を持つ構造を求めるといったように、実際の問題がこの関係における一方から他方を求めるという形で現われることから明かである。

（３）対集合公理はこれを繰り返し適用することにより順序付きｎ組を＜ａ０，ａ１，ａ２，－－－，ａｎ－１＞＝＜ａ０，＜ａ１，ａ２，－－－ａＮ－１＞＞によって生成する。これはリスト構造を与える。

（４）巾集合公理はこれを繰り返し適用することにより、－－ｘ１∈ｘ２∈ｘ３∈－－の関係を持つ集合を作る。これは階層構造に他ならない。これは数学的な構造であるが実世界の問題では一般に全体－部分構造として階層構造が現われる。これは集合－要素関係とは同一ではないが、実問題を数学の世界に写像して数学モデルを作ることは問題解決の重要な手法であり、全体－部分構造の中で集合－要素構造として解決される部分も多い。

（５）空集合公理は集合に関する基本概念として既に一般的になっている。

（６）置換公理（図式）は数学的な概念で、直感的に理

解しにくい点があるが、与えられた集合の各要素に与えられた関数関係で対応ずけられた集合の全体はやはり集合であること、言い換えると集合を対応ずけられた集合の分だけ拡大したことになる。このような公理が必要なのは、最初に述べたように、矛盾を生じない最大の集合を定義することが必要だからである。この公理の実際的な効果は集合の和集合、積集合および直積などの概念がこれから導かれることである。和集合、積集合は集合に関する基本操作としてすべての集合型データに定義されている。直積は情報処理において非常に重要なデータ構造である。これは関係型データベースの基本構造が直積で表わされることから実用性が明かであろう。

（７）正則性公理は階層構造が下方に有限であること、すなわち構造に末端要素があることに対応ずけられる。実体にはアトムがあると言ってもよい。計算機内の実際の処理に際しては必然的な条件である。

（８）和集合公理は（６）の置換公理に含まれている。情報処理的には重要な操作の一つである。

（９）無限公理はこれ自体が情報処理に直接係わるものではない。

（１０）選択公理は数学的には問題の残されているもので、この公理に疑問を持つ数学者もいる。情報処理技術としてみると公理の正当性そのものよりも、その実現法が問題になるが、集合の要素（集合）から一つの要素を取り出す関数を作ることは容易であり、既に必要に応じてこのような操作は実施されている。

このように公理的集合論の各公理は集合の性質を与えるもの、集合（データ構造）を定義付けるものおよび集合に対する操作を定義するものとして情報処理システムに対応づけることができる。重要なことはここには有限個の基本的な機能があること、特に集合の生成に関する機能で、他のもので置き換えられないものがあることである。もし情報処理システムでこれらのいくつかが欠落していたらそれを必要とするデータ構造を用いる応用は扱えない。

基本的な生成機能は、和、差、積、直積、対、巾である。これらをオペレータとして記号 $\cup$、$-$、$\cap$、$\times$、$<>$、$*$　で表わすとしよう。既定義の集合からこれらを用いて新しい集合を定義出来る。例えば　$*(A \cup B)$

は集合　A　と　B　の和集合のすべての部分集合から作られる集合である。ある既定義の集合Dをもちいて用いて新しく集合Aが構成的に定義されたとき、DをAの基底集合と呼ぶことにしよう。巾集合オペレータによって作られる階層構造の概念は重要である。これは　＊n = ＊(＊n-1) という再帰形で定義される。

はこれらの構造の基本要素をすべて含むものでなければならない。宣言型言語におけるデータ構造の表現法は手続き的方式における表現の方法とはまったく事情が異なり、対象記述の方法が重要なものになる。宣言的表現では記号の意味が、記述される世界内の概念との直接の対応として定義されるからである［ 17 ］。

これにたいし、手続き型言語では計算機内の '記号' に一定の '操作' が対応する。それに与える入力も、操作結果として得られる出力も、直接には外界の事物には対応せず、人が解釈としてデータと実世界との対応をつける。構造の表現についても、配列やポインタなど、計算機内でデータ構造を表現するために必要な '表現上' の基本形式（ '意味' の基本ではなく）が準備されていれば、それを用いて計算機内に任意のデータ構造を定義

し、それが対象の構造を表わすように人が意味付けすることによって、様々な構造を生成することができる。これは抽象データ型を含む、データ構造論として多くの議論がなされてきた [12,26] 。すなわち、手続き型言語では、構造表現の基本要素は計算機内表現としてデータ構造を構成するために必要なものであればよい。上記の基本構造要素も、その幾つかは、手続き型言語では必要に応じて作ることができるから、意味的にどれだけの構造要素を準備すべきかという問題もなかったといえる。

従って、もし上記の基本的な構造要素のどれかが欠けている言語はどうしても手続き的に定義する手段を持たねばならず、純粋に宣言型ではなくなってゆくことになる。

6．述語言語へのデータ構造の導入

次の問題はこのように定義される集合をいかに述語論理と結びつけるかである。述語を　F(t1,t2,---,tn) としよう。ti、i=1,2,--,n, は項（定数または変数）であり、記述しようとしている世界内の実体を表わす。述

語自体は実体の性質（n = 1 ）あるいは実体間の関係（n > 1 ）を表わす。ti が定数の場合は特定の実体に関する記述を与える命題として真または偽の値を持ち、変数の場合には定数が代入されて命題となるが、この変数の変域Dは記述される世界の中のすべての実体の集合である。通常の述語論理では、すべての変数の変域はDで共通なので、表現上は省略されるが、厳密には述語と共に xi∈D （xi はtiに対応する変数）と書くべきものである。

述語論理では変数は限量記号 ∀ または ∃ で限量化される。限量化された変数は束縛変数と呼ばれる。述語論理式でそこに含まれているすべての変数が限量化されているものは文と呼ばれる。以下では文のみを考える。また記述を簡単にするため1変数述語で考える。

変数の変域がDの部分集合Di として定義され、変数ごとに異なるものとしよう。すなわちこの範囲内でのみ述語で記述される関係が成立ものとする。内包性公理によって部分集合Di を定義するある性質がQi があって、

Qi(x) <-> x∈Di であるとする。この時の通常の論理式の表現は限量記号に依存して、

$(\forall x)[\ Qi(x) \rightarrow P(x)\ ]$　あるいは、

$(\exists x)[\ Qi(x) \wedge P(x)\ ]$

である。これは集合Di を性質によって定義しているが、述語論理の一形式の多類論理（Many Sorted Logic)では変数の型の概念を導入して、$x \in Di$ の関係を用いる[　]。たとえば整数型の変数では変域はDの部分集合である整数集合である。この時は上記の表現は

$(\forall x\ /Di)P(x)$ および $(\exists x\ /Di)P(x)$

と表わされる。x ／Di は　$x \in Di$ を表わす。そこで、このメカニズムをすべての変数について適用するように拡張した形式を導入する。Di としては前記のように構成的に定義される集合を用いる。

集合Cを基底集合として、巾集合オペレータをn回繰り返し適用して得られる集合はCを要素とするn段の階層構造をすべて含む集合である。この1要素Hi は階層構造を持った特定の実体を表わすから、Hi に関する述語　P（Hi）　は通常の命題と同様である。次ぎにこの構造の中のある要素について記述する場合を考えよう。n＝2としておく。述語による記述は基本的には対象である実体か、変数の変域が同定されることを前提として

いる。通常の述語論理では前者の場合は名前によって、後者の場合は記述世界として同定される。階層構造内の中間構造の場合、それに前もって名前が与えられていれば特に中間構造として意識する必要はない。しかし、'構造Hi の中に、全部品がA社から供給されている部分構造がある'のように、対象である中間構造が特定されていないで、この部分に変数を必要とする表現をどうしたら良いだろうか？　述語 SUPPLY(x, y) ; 'x が y を供給する'を用いるとしよう。 x はこの例ではAであるが、y は変数であって、その変域自体も特定されていない。すなわち、変域もまた変数（z）である。zの変域はHi によって特定されている。、したがって、上記の記法をそのままここにも適用すると、

（∃z ／Hi)(∀y ／z)SUPPLY(A, y)

のような形式が必要になる。この表現の形式は接頭部で構造を参照し、本体部で記述を行なっている。これは開発問題について挙げた知識表現への要求に良く適合している。　このように拡張されたシンタックスをもつ述語論理を多層論理と呼んでいる。この拡張に伴って、推論アルゴリズムも拡張が必要になる。一般のデータ構造に

対して、従来の方式では全体構造を個々の構造関係に展開した後に推論を適用して処理することになるが、この拡張した方式では、推論に際して生じる構造間の関係のテストや構造の変更は、推論機構の中で手続きとしてまとめて処理することになる。これはデータ構造の形式と推論の形式が定まっているので、推論時の構造処理の手順が定まるからである。

紙数の都合で多層論理の詳細をここで示すことが出来ないが、[5, 8]を参照されたい。また多層論理の一例としては本誌の論文[2]を参照されたい。

7. むすび

知識処理において最も基本的な問題は優れた知識表現言語を開発することである。現状では、知識表現は様々な形式の総称であり、任意性が残っていて、優劣を比較する根拠すら乏しい。知識処理を情報処理の新しい体系的技術として確立するためには正当な理由の下でこれら知識表現を評価．比較する為の根拠を見いだし、望ましい知識表現言語を求める事が必要である。

本稿では大規模な開発型問題がこれからの知識処理の重要な分野であることを指摘し、それを実現するための条件として知識処理への要求項目を挙げた。今日、ここに挙げたすべての要求項目を満足する知識表現言語は存在しない。従って、最も多くの項目を満たすものとして述語論理を基本言語に選び、それを他の項目も満たすように拡張した。

知識表現言語は宣言型言語であることは基本条件の一つであるが、宣言型の言語設計においては基本データ構造の選定が極めて重要であることを指摘し、その選定を公理的集合論に求め、それによって対象構造を表わすためのデータ構造が、対象の性質等を表現するための述語と同様に根拠のある体系の上に定義出来ることを示した。最後に述語とデータ構造を一体化する方式として多層論理となづけた拡張論理表現が可能であることを示した。

これらによって知識表現の部分についてはかなりの程度見通しがついたが、知識処理のためにはシステムの制御問題が残されており、本稿では紙数の都合で触れなかった。この部分は未だ未解明の部分が多く、これからの重要課題である。

[1] H.B. Enderton : Mathematical Introduction to Logic, Academic Press (1972).
[2] H. Glaser, C. Hankin, D. Till: Principle of Functional Programming, Prentice-Hall, 1984.
[3] F. Hayes-Roth, D.A. Waterman and D.B. Lenat: Building Expert Systems, Addison-Wesley, 1982.
[4] J.L. Krivine: Introduction to Axiomatic Set theory, D. Reidel Pub. Co./Dordrecht-Holland (1971).
[5] S. Ohsuga : A New Method of Model Description -- Use of Knowledge Base and Inference, in CAD System Framework (ed. K. Bo and F.M. Lillehagen), North-Holland (1983), pp.285-312.
[6] S. Ohsuga : Predicate Logic Involving Data Structure as a Knowledge Representation Language, Proc. '83 Eigth Int. Joint Conf. on Artificial Intelligence, (1983), pp.391-394.
[7] S.Ohsuga : An Integrated, Intelligent, Interactive and Incremental Model-Based Problem Solving System, Proc. Artificial Intelligence and Advanced Computer Technology, Wiesbaden, Sept. 1985
[8] S. Ohsuga : Multi-Layer Logic-- A Predicate Logic Including Data Structure as Knowledge Representation Language, New Generation Computing, 3, (1985), pp.403-439.
[9] M. Van Emden& R.A. Kowalski : The Semantics of Predicate Logic as a Programming Language, J. ACM, Vol.23, No.4, (1976), pp. 733-742.
[10] N. Wirth: Algorithms + Data Structure = Programs, Prentice-Hall, 1976.

[11]石塚　満；知識の表現と利用、知識ベース入門（大須賀編著）第１章、オーム社、1986

[12]稲垣康善；抽象データ型の概念と仕様記述法、情報処理、第27巻第2 号、昭和61年2 月、pp.120-128

[13]上野晴樹；知識工学入門、オーム社、1985

[14]大須賀節雄；”コンピュータによるモデリング”、精密機械、第51巻第6 号、昭和60年6 月、 pp. 53-59

[15]大須賀節雄；”次世代ＣＡＤ／ＣＡＭのための知識処理の応用”,マグロウヒル　ブック、昭和60年11月

[16]大須賀節雄；”知識処理システムへの展望－新コンピュータ．システムへの道”、情報処理学会［コンピュータ．システム］シンポジューム講演、昭和60年12月

[17]大須賀節雄：”データ構造とモデリング技術”、情報処理、第27巻第2 号、昭和61年2 月、pp.178-187

[18]大須賀節雄；”知的情報処理の現状と課題、計測と制御、第25巻第4 号、昭和61年4 月

[19]大須賀節雄；知的モデル構築支援方式による問題解決システム、電子通信学会論文誌（Ｄ）、Ｖｏｌ．Ｊ69－Ｄ，Ｎｏ.7（予定）、昭和61年 7月

[20]大須賀節雄；人工知能の現状と今後の動向、日本機械学会誌、第89巻815 号（予定）昭和61年10月

[21]韓、大須賀、山内；知識工学の機械設計ＣＡＤへの応用、人工知能学会誌、第１巻、第１号、1986

[22]Ｊ．クロスリー他、田中尚夫訳；現代数理論理学入門、共立全書、553 昭和52年

[23]情報処理学会誌、＜知識工学特集号＞、第26巻第12号、昭和60年12月

[24]日本航空宇宙工業会編；航空宇宙工業におけるコンピューター利用高度化調査検討報告書、昭和57年3月

[25]日本航空宇宙工業会編；航空宇宙工業におけるコンピューター利用高度化調査検討報告書（その２）、昭和58年8 月

[26]野下浩平；基本データ構造とアルゴリズム、情報処理、第27巻第2 号、昭和61年2 月、pp.111-119

（Ａ）モデル構築支援の条件

（１）宣言的表現

（２）推論機能

（３）モデル構造．属性等の統合的表現

（４）モデルの動的処理

（５）無矛盾性のチェック

（Ｂ）マン．マシン．コミュニケーションの条件

（６）宣言的表現

（７）推論機能

（８）外部言語（自然言語、図形等）から／への変換

（９）内部言語（手続き言語、データベース言語等）への変換

表１　知識表現言語の条件

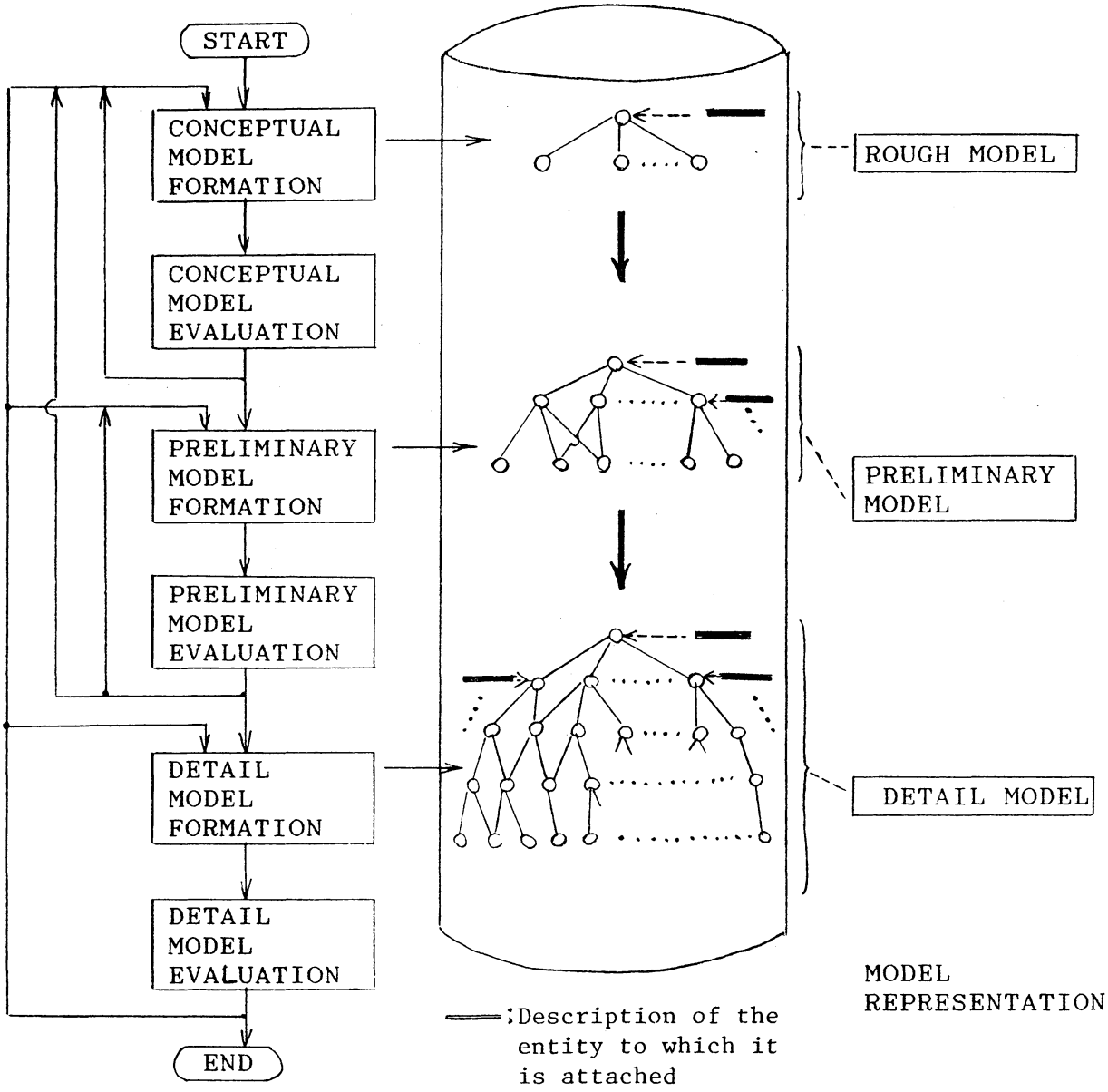

Fig. 1 Typical design process