佳作論文

次世代計算機システムに関する一考察†

—知識型システムの提案—

大須賀　節　雄††

初期の計算機は社会全体の活動に比べるとごく一部の特殊な分野で用いられることを前提として設計されたものである．しかし計算機の利用範囲はその後急速に拡大し，現代では普遍的な手段として社会生活に欠かせないものとなっている．しかるに計算機の基本設計思想は今日までほとんど変化していない．これまで，応用技術という名で，拡大する新しい分野で生ずる広汎な要求を，特殊分野向けの機能によりいかに満たすかという努力が続けられてきた．しかしこのような努力も限界に達しようとしており，新しい計算機システムの研究が各国で進められている．

本論文はこのような新時代の計算機への要求を分析することにより，今後の計算機が満たすべき機能を求め，それを実現する具体的な方法と，そのための基本機能を備えた実験的なシステムについて述べる．

1. はじめに

近年，計算機の利用形式の多様化とそれに伴う高速度処理・高信頼度要求，マイクロ素子技術の発達，ソフトウェア危機など，計算機を取り巻く環境が急速に変化し，それに伴って従来の計算機すなわち von Neumann 型の計算機の概念をも越えた新しい計算機システムの研究も活発化している．これは，計算機の置かれている環境が初期の計算機設計当時と現在とでは著しく異なり，この変化の程度が同一設計思想のもとで，素子の開発やプログラミング言語の改良のみによって適応できる範囲を越えるまでになってきたからに他ならない．したがって将来の計算機を考察するために，この環境の変化を分析し，それが計算機の設計思想にどのような影響を与えるかを考察することは必要かつ有力な方法である．

筆者らは多年にわたり，計算機の利用様式の変化が計算機構成に及ぼす影響を検討してきた．そこで得た結論は計算機にたいする要求が初期の頃とは質的に変化しており，それに対応するために，将来の計算機は(1)意味を explicit に表現する形式と，それによって表現された知識の集まりである知識ベースを構成する手段を有すること，(2)推論機能，大量の非手続き情報から特定の条件に合致した情報を検索する機能，非手続き的に表現された知識から手続きを生成する機能などを基本機能として含むべきこと，(3)大量のデータを効率的に処理すること，(4)処理は高速に行われることが導かれる．さらには実用化を可能にするには，(5)新しい概念の計算機は長年にわたり多額の投資のもとで蓄積された既存の情報や計算機そのものを否定するものでなく，過渡的にはこれらと共存あるいは漸次的な移行を可能にするものでなくてはならない．重要なことはこれを専用目的の計算機としてでなく，汎用の計算機として実現することである．

本稿ではこれらの諸問題にたいする筆者らの考え方と問題の分析および現在開発中の知識システムの概要について述べたい．本論文においては，2章で初期の計算機設計当時の計算機の環境と現在までのその変化を，3章で将来の計算機に予想される環境を，4章ではこの環境変化が計算機に要求する機能を，5章ではこのような考察に基づいて現在開発中の KAU (Knowledge Acquisition & Utilization) システムと名付けられた知識システムの概要を述べる．なお本論文は今後の計算機のあり方について一つの可能性を示すことが目的であり，紙数の制約のため個々の理論やアルゴリズムの証明等の詳細には立ち入らない．

2. 計算機の環境とその変化

2.1 初期の計算機への要求

応用面から見たとき，初期の計算機の設計時点において計算機に課せられた要求は「数学的に定義された関数の値を数値的に高速に求めること」であったということができる．計算機が扱う問題は，

(A1)；数学的体系という枠組の中で定義され，

† A Consideration to New Computer Systems of Next Generation—A Proposal of Knowledge Based System by SETSUO OHSUGA (Institute of Space and Aeronautical Science, University of Tokyo).

†† 東京大学宇宙航空研究所

(A2)； 解法は得られている
ものであり，計算機はこの解法に従って
(A3)； 結果の数値を高速に求めること
が必要であった．このような要求から来る計算機設計条件の特徴として次の諸点を挙げることができる．

(a1)； 現実の世界に発生する問題を，数学的体系という整った枠組内の表現に変えるための抽象化や意味的な処理はすべて問題の発生源と計算機の間に介在する人間が行う．

(a2)； 抽象化が行われた後の情報のセマンティクスは数学的体系の枠内で定義された変換列すなわちプログラムによって表わされる．計算機はこの変換過程を正しく表現し，実行するという程度に限定された機能を有すれば十分である．

(a3)； データは変換手続きの，いわば付属物であり，したがって手続きと同時に記憶部に存在すべきデータ量は比較的少量と考えられている．

(a4)； 変換は高速に実行されねばならない．

これらの条件は計算機の利用範囲を限定する反面，von Neumann 型計算機が効果的に動作する上で重要な条件となっていることは明らかである．

2.2 環境の変化

しかし，もしこれらの条件が崩れたら von Neumann 型の計算機の正当性もまた再検討されねばならない．事実，初期の設計思想に基づいて実用の計算機が出現し，またその応用技術が進んでくるにつれ，必ずしも上述の条件を満たさない分野にも計算機は進出していった．座席予約や銀行業務などのオンライン・リアルタイム処理や時分割システム，図形・画像処理，CAD，情報検索やデータベース・システム，各種問合せシステム，またこれらをインタラクティブに処理するためのマン・マシンの会話技術など，いずれも数値計算とは性質の異なる利用形式のものである．しかもこのような応用分野が今後計算機利用の主力になろうとすらしている．

これら新しい種類の応用は計算機がおかれている環境を質的に変化させた．一方，個々の問題が大型化し処理量も増大するといった量的な変化も進行してきた．これまでの計算機技術の歴史は von Neumann 型という枠組の中でいかにこのような変化に対応するかの歴史であったと言える．システム的な機能の拡大や応用範囲の拡大にたいしては主としてシステム・プログラムや応用プログラムなどのソフトウェア技術と仮想メモリ方式などのアーキテクチャ技術により，また高速処理の要求にたいしては新素子の開発やパイプライン，並列処理，バッファ・メモリ，連想記憶などのアーキテクチャ技術によって対応してきた．しかしこのような努力は間もなく限界に達すると考えられている．一つには，これ以上の素子による高速化には物理的な特性の限界があること，第二には問題が複雑化するにつれソフトウェア開発が困難になり，プログラムの信頼性が保証できなくなってきたこと，そしてさらに重要なことは，現状でも設計，医療診断，教育，意思決定支援など多くの分野に潜在的なユーザが多数存在しているのに従来の技術ではもはやこれらの応用分野への発展が困難になってきたことなどのためである．

3. 次世代計算機への要求

3.1 将来の計算機の環境

前述の von Neumann 型計算機設計当時の環境と対応させて考えた時，新しい各分野の応用技術に共通しているものとして次のような特徴を挙げることができる．

(B1)； 問題が発生時点で計算機に与えられること．したがって必ずしも計算機にとって都合のよい枠組の中で記述されていないこと．

(B2)； 問題の解法が必ずしも与えられていないこと．

(B3)； 処理の一層の高速性が要求されること．

これらの条件が課せられる一例として，機械設計における CAD (Computer Aided Design) を考えてみよう．対象である“物”の形状や性質の記述を含めた設計過程の記述は従来の数学の枠内では困難で，図面をはじめ数式以外の各種の表現手段が使われ，また設計の過程自身は最終目標に達する道が与えられていることは少なく，試行錯誤の繰り返しである．設計過程のあらゆる個所で中間結果や最終結果が所要の設計条件を満たさないことが判明すると，過去の点に戻って再試行が行われるがこの過程は非決定的であり可能な試行経路を前もって予測することはできない．すなわち解法が前もって与えられていない．また個々の試行において処理が高速であることが実用化の条件となる．

上述の条件のうち (B1), (B2) は環境の質的な変化を表わし，(B3)は量的な変化を表わしている．

3.2 非手続き的な意味の表現

(B1)の条件は 2.1 節の(A1)に対応する．(A1)で

は発生した問題は人間の手を通してプログラムに変換され，問題に含まれる情報の形式と意味の対応や抽象化は人間によってほとんど処理されることを前提としたが，(B1)はこのような前提を除くことを要求する．発生した問題の記述はたとえば回路図で表わされている電気回路の入出力応答を求めるというように発生源の性質に依存し，原則として計算機の機構とは無関係である．したがって(B1)の条件はこれまで人間が行っていた問題の意味的変換を人間に代って行い，解を得るために適した形に問題を表現し直す機能を持った新しい形式の計算機の必要性を示している．

通常，発生する問題は言語，数式，図形などによって記述されるが，これらは意味の特定の表現形式であり，意味と形式の間には一定の関係がつけられている．以下ではこのような，計算機の外部で用いられる情報の表現形式を外部表現と呼ぶ．意味的な変換は問題の発生源によって定まっている表現形式から意味を抽出する過程を経てはじめて可能となる．

さらに前述のように情報の意味がかつては問題の解をいかに得るかを示すプログラムの形で implicit に表わされていたのにたいし，(B2) の条件からそれが困難である．したがって(a1)に対応するものとして将来の計算機には，

(b1)； 現実の世界に発生しかつ表現される問題を人間の手を介さずに直接計算機が受け入れ得ること．そのために計算機は（一部の）情報の意味を非手続き的にかつ explicit に表現できるようにすること，が要求される．

3.3 仮説とその検定

(B2) の条件は問題が与えられた時，解と解法を同時に見い出す問題解決の能力を今後の計算機に要求する．目的を指定し，これに達する解を自律的に見い出す発見的探索法が人工知能の分野で研究されてきたが，自動的に解を見い出せるのは問題の枠組が単純で限定された範囲のものに限られる．

このような要求を実用的な意味で満たすアプローチは人間が仮想的な解（これを以後仮説と呼ぶ）を与え，既存情報を用いて計算機のサポートのもとでその仮説をテストしつつ解に達する方法である．“仮説”は人間を含む問題解決システムでは“サブゴール”であり，CAD の場合は“モデル”と呼ばれる．このように，将来は CAD の例で見たような人間と計算機との対話形式により問題を解決していく方式が増大するものと予想される．この形式を可能にするためには人間と計算機とのインタフェースを強化する必要があり，また計算機は仮説が与えられた後の検定のために，局限された場面で効果的に働く推論能力や問題解決能力等の人工知能的手法を備えることにより，人間を支援することが望ましい．このように今後の計算機に要求される機能を一口で表わせば，“人間が与える仮説を受け入れ，その検定を強力に支援すること”と言うことができる．

種々の形式で記述される仮説を受け入れるには計算機自身が仮説を表現する柔軟な意味の表現形式を備え，その枠組内で表現されたすべての意味を処理する機能が必要になる．

3.4 知識と推論

将来の計算機も従来のものと同様に数学的な体系内で厳密に定義された変換を実行する機構を含むことは必要である．この機構が von Neumann 型のものであるとは限らないが，実行可能な変換（プリミティブ）が前もって定義され，問題がこの定義済みの変換の，部分的に順序付けされた組に展開された時，この問題の解法が得られたと言う．

問題や仮説は計算機と無関係に表現されるから，これが直接プログラムに展開できる形になっていることは期待できない．したがって，この表現を意味的等価関係に基づいて変換し，最終的にはこの定義済み変換の半順序集合を得ることが必要である．このようなプロセスでは問題自身は非手続き的に表現され，これを同義異形式に変換することを意味処理と言い，非手続き情報から関数／手続きの組を得ることをプログラム生成，得られた結果をプログラムと呼ぶ．この過程が問題の解法を求める段階に相当し，その問題の解自身はここで得られた変換の組の実行によって得られる．

このように，意味処理の基本は意味的な等価関係に基づく変換であるが，意味の等価関係を表わす情報も非手続き的に表現され，計算機内に貯えられるようにしておくのが，新しい意味表現の付加が容易になるなどの点で望ましい．このような情報を以後知識と呼ぶ．さらに知識には単に表現間の等価関係のみでなくこれと同じ表現形式で表わされた，事実に関する一般情報も含める．知識を集めたものを知識ベースと呼ぶ．

意味の等価関係を示す知識を用いて，与えられた表現と意味的に等価で，形式の異なる情報を生成したり，事実に関する知識を用いて，それが真か偽かを判定する機能を推論機能と呼ぶ．したがって仮説を意味の表現形式に変換できれば，知識ベースと推論機構と

がそのまま仮説の検定に用いられる．なお今日では上述のような意味の表現形式はむしろ知識表現と呼ばれている．したがって本稿でも以下ではこの呼び方にならう．

以上のことから，(a2) に対応して今後の計算機に課せられる条件が次のように表わされる．

(b2)； 意味処理を通して解に達するまでの手続自体が，解自身と同時に構成されること．この過程において変換手続きは非手続き的情報と共に処理対象と見なされ，このような解法自体を計算機内で構成するために一般の変換手続より高レベルの処理機能を備えるべきこと．

この過程において実際に問題を解くのは人間であり，計算機は所与のデータを用いて与えられた仮説の検定を行ったり，サブゴールへの道を求めるなどによって人間をサポートする．

3.5 データベース

上述の (b1) の条件は，与えられた問題の表現を意味的に処理して解法に達するまでに利用される非手続き情報に関するものであり，すべての非手続き情報がここに述べられた条件を満たすことを要求するものではない．プログラムが得られた後はこの実行により問題の解が得られるが，実行段階でのみ用いられる情報には (b1) に示されたような表現の柔軟性は要求されない．以下このような情報をデータと呼ぶことにする．たとえば仮説は測定値など観測可能なデータに基づいて検定される．

知識とデータを分離する理由は次の二つである．

(1) 今後，計算機の大量記憶能力を利用するという傾向は一層増大することが予想されるが，大量のデータをすべて (b1) を満たすような柔軟な形式で貯えるのは記憶効率の点で好ましくない．したがって柔軟性は乏しいが効率的な記憶方式を用いた方が良い．

(2) 現在までに多大の投資のもとで多くのデータベースが構築されている．計算機の方式の変更により，これらが使用不可能になったり，引き続き利用するためには複雑な変換を要することは好ましくない．したがって，従来のデータベース（と類似の）形式のデータを利用する機能を備えることが望ましい．

特定の情報が意味処理にもプログラム実行にも用いられる可能性があるため，意味処理の機能はこの両形式間の変換機能を含まねばならない．この前提のもとで，情報をどちらの形式で貯えるかの決定はユーザに任せられる．

データベースを含めた非手続き情報の量に関しては，(a3)に対応して，

(b3)； 計算機利用が従来の高速変換機能の利用から大量記憶機能に移行しており，計算機内には少なくとも主メモリに入り切らない大量の非手続き情報が存在すること，が指摘される．

3.6 処理の高速化

高速演算処理の能力は計算機にたいする当初からの要求であるが，計算機で処理される問題の規模が大型化している．このような条件の厳しさを加味した上で (a4)に対応して，

(b4)； 変換は高速に実行されるべきこと，

が得られる．

4. 計算機に要求される機能

4.1 三つのボトルネック

現在の計算機を用いた人間――機械系に前記の諸要求をあてはめてみると，情報の流れに関して少なくとも三つの異なった型のボトルネックが生ずる可能性がある．第一は人間と計算機間インタフェースに関する部分で，前述(b1), (b2)に示された条件に対応する．第二は主として (b3) に関するもので，非手続き情報が増大した場合の主メモリ――補助メモリ間の情報転送に関し，アクセス時間の差に起因するもの，第三は主として(b4)に関し，主メモリと CPU 間の情報転送と CPU 内での変換速度に関わる部分である．

これらのボトルネックにたいする対策はそれぞれに異なる．本稿では第一のものを解決するための方式について述べるのが主目的であり，(b1), (b2) の要求をさらに具体化するが，他のものにたいする対策との相互の関連について簡単に触れる．

4.2 人間―計算機間インタフェース

人間――計算機間のボトルネックは他の二つと異なって論理的性能に関わり，これを改善するには外部表現で記述された問題（仮説）を受け取り，その解法に相当するプログラムを生成したり，データベースを検索する機能が必要であることが3章の結論であった．求まった解を外部表現の形で出力する機能をこれに加えたものが今後の人間――機械間のインタフェースの主要部分を構成する．ここには意味を表わす形式として少なくとも三種類の情報表現形式が存在する．外部表現，プログラム，データベースである．入力情報はこれら形式間であるいは同一形式内で意味を保存しつつ変換されることにより処理される．上記形式間の変

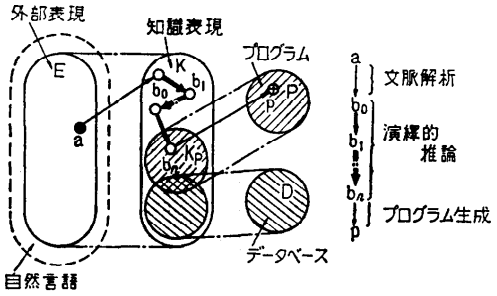


図 1 各種表現形式間の関係
Fig. 1 Relations between various representations.

換を直接行わず，知識表現形式を定めてこれを仲介して変換を行うのが知識システムである．こうする理由は第一に表現形式間に両方に共通の面をもった中間形式を導入することにより変換アルゴリズムが明確になること，第二に各表現形式の記述力や冗長性が相違し，各形式内の表現間で一対一の対応が取れないためである．後者の場合，与えられた表現と同義な表現を，他方の形式内に対応する表現が見い出されるまで繰り返し求める変換すなわち形式的推論機能が必要になるが，推論処理を知識表現で行うことにすれば推論アルゴリズムを一つだけ準備すればよい（**図 1**）．

以上のことから，知識表現が満たすべき条件は次のようになる[2),3)]．

（c1）；外部表現，プログラム，データベースの各各との間で表現の変換が可能な形式であり，かつその変換アルゴリズムが求まること．

（c2）；推論アルゴリズムが定義できること．

知識システム内で実際に必要な形式間および形式内変換のリストを**表 1** に示す．E, P, D, K はそれぞれ外部表現，プログラム，データベース，知識表現を表わす．

4.2.1 外部表現，知識表現間変換

外部表現として自然言語を考えてみる．自然言語を完全に受け入れるには未だ多くの困難があり，実用化の目的には計算機の受け入れられる範囲の外部表現の形式を定め，その範囲内で厳密に計算機の仕様を定めることが必要である．自然言語で表現される知識には曖昧性が含まれるが知識型計算機が曖昧さを含む情報を利用できるようにすることは重要である．個々の情報は曖昧でも他と組合せてより正確な情報を引出すことが可能であり，現実の世界では曖昧な情報は非常に多いからである．このためには曖昧さの評価を各変換過程に含めることが必要となる．

表 1 システムの備えるべき変換機能
Table 1 Necessary conversions.

	変　　換
1	E→K：文脈解析　パターン分類
2	K→E：出力生成表示
3	K→P：プログラム合成
4	(K→D)：データベース記述・アクセス
5	D→K：データ一般化
6	K→K：演繹的推論

このような外部表現を定義する上で考慮すべき条件は，（d1）定義された範囲内ですべての問題を記述することができるだけの記述力を有すること，（d2）受け入れられる範囲をユーザに容易にかつ確実に指示できること，である．

4.2.2 知識表現からプログラムへの変換

知識表現は非手続き的であることによって（e1）外部表現に対応できるだけの記述力を有し，（e2）プログラム化を意識せずに知識を付加すること，が可能になるが手続きへの変換を行うには手続き的な面を併せもつ部分（図1の K_P）を含むことが必要である．これには K-K_P 内の表現に対応する外部表現を K_P 内の表現に変換しプログラム化を可能にする推論アルゴリズムの存在を前提とする．

4.2.3 知識ベースとデータベース

データベースはアクセス手段があって初めて利用可能であり，利用に際して必要なのは K→D の変換でなく，データベースへのアクセス手順を生成することである．アクセス手順はデータベースの構造[4)]に依存するから，この構造を記述する知識が必要になる．もし知識表現が十分強力で任意のデータモデルを記述できるなら，これを用いて推論アルゴリズムのもとでアクセス手順が自動生成されることになる．

データと知識表現の変換が実際に行われるのは，D→K の型の場合である．個々のデータ・レコードを知識の形で表現するのは非効率でも，一群のレコードの集まりであるファイルを単一の知識で表わせるなら，利用度も記憶効率も向上する．D→K の変換は計算機にとって不可欠ではないが，実用的な学習機能の一種として極めて有用である．

4.2.4 推論機能

推論機能が知識システムの中核となることは上述のことから明らかである．またそれ故に，推論機能は論理的に完全であり，効率的に働くものでなくてはならない．推論機能に関しては5章で実例に即して述べる．

4.3 主メモリ—二次メモリ間インタフェース

データが大量になるにつれ，主メモリ，二次メモリのアクセスの時間の差の影響が増大する．従来の計算機方式のもとではプログラムは計算機の制御の流れを表わしており，メモリ上の静的な配列順序と動的な参照順序間の相関（局所参照性）が強く[5),6)]，適切に選ばれた一部が主メモリに置かれた時，実際に参照される部分の割合が大きい．これにたいしデータについては局所参照性の小さいものがあり，データ量が増えると主メモリの実効的な利用度が低下し，処理効率への影響も OS の制御の範囲を越えて増大する可能性がある[7)]．

データベース技術の発達に伴い，データの処理をホスト計算機外で行うデータベース・マシンの研究[8)]が進められており，上記問題もこの中に含まれていくであろう．ただし将来，CPU も変化する可能性があり，これを一体として考えることが必要であろう．

一方，同じ非手続き情報でも知識ベース利用はデータベース利用と異なるのでこれらは構造的に分離して扱うことができる．

4.4 CPU—主メモリ間インタフェース

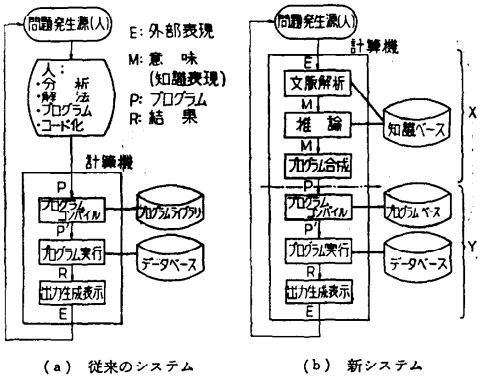
従来の計算機の一つの欠点は CPU と主メモリの間の情報転送路が細く，しかもアドレス情報のような付随的な情報の移動のために占有される割合が大であることが指摘されている[9)]．この部分の改善は計算機の高速化，プログラミングの容易化などを目的にこれまで各種のアーキテクチャが研究されてきた．特に近年，関数型言語の開発[10)]と，それに適した新しい革新的なアーキテクチャの開発[11)]など多くの研究論文が発表されている．

しかし，人間——計算機間の新しい環境下では CPU によるプログラム処理は推論機能やプログラム生成機能によって生成されたプログラムの実行部分に相当し，これを行う計算機が von Neumann 型であっても，関数型であっても論理的な面では変わりがない．

4.5 現実的なアプローチ

計算機をシステムとして統一のとれたものとするためには上述の諸技術を関連させ最適に機能するものを求めるのが理想であるが現実的なアプローチとしては上述のように（b1），（b2）の条件が他と比較的独立しているのでこれを分離して扱う．

図2は第一のタイプの問題を解決するのに必要とされる機能を配置した計算機を，従来のものに対応させて示したものである．この図の（b）でXの部分は新しい機能を果たす部分，Yは従来の計算機に相当する部分とする．このように従来の計算機に今後要求される機能を付加する形式にしておくと，



（a） 従来のシステム （b） 新システム

図 2 従来型計算機と今後の計算機
Fig. 2 Conventional and new computer system.

（f1）； 人間——計算機間の問題を他の要求と独立に解決を図ることができ実現を早めることができる．

（f2）； 既存の装置や情報をそのままいかすことができる．

などの利点がある．XとYを物理的に分離し，Xを従来型計算機のインテリジェントのフロント・エンドとすることも可能である．

4.6 知識表現の各種の方式

知識システムの問題は結局知識表現の形式と表1に挙げた各種変換を定義することであるが，変換は表現形式に依存して定まるので，知識表現をどのように定めるかが最も基本的な問題となってくる．これまでに多種類の知識表現の方式が研究されているが[12)]表現形式には現在いくつかの基本的な方式があり，これらは大きく（1）手続き的方法と，（2）非手続き的方法，に分けられる．後者はさらに（2a）グラフ表現を用いるもの（2b）生成規則によるもの，（2c）述語論理を用いるもの，などがある．これらの相違は表1の変換のうちのどれに重点を置くかそしてそのためにどの方法を用いるかにあるということもできる．

手続き的方法[13)]は個々の知識を手続きに直接に対応するように表わすもので，K→P の変換に特に適しているが記述力は限定的であり，新しい知識の付加や知識の変更が困難である．

セマンティック・ネットと呼ばれるグラフ表現はグラフのノードにより概念を，アークにより概念間の関係を表わすもので[14)~16)]，記述力が大きく，グラフが

単純なら処理効率も良い．しかし推論における完全性の保障が得にくく既存データベースの利用に難点がある．

生成規則を用いる方式[17]は知識表現が，“条件→動作”の形式で表わされるので手続きへの変換は容易であるし，グラフ表現や後述する述語論理の場合と同様に新しい知識の付加は容易である．しかし知識全体の整合性をとるのが難しく，知識ベースが大きくなった時の管理が困難になることが予想される．

述語論理[18],[19]は一階の述語について推論の完全性が保証されていること，プログラムへの変換が可能なこと[20]，データベースとの意味的対応関係があること，元来，言語の意味的関係を論ずる論理学から発達したもので言語表現との関係が深い[21]などの利点があるが，一階述語の範囲では記述力が不十分であること，推論の効率が良くないなどの欠点がある．

このようにいずれも利点・欠点を有し，前述の要求を満たすには上記の方式を拡張したり，組合わせたりして改善する必要がある．

5. 知識取得・利用システム—KAU[2),22)~26)]

本章以下では前章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．

5.1 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わ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ものであるが，以下では図1の情報変換の過程を見やすくするためにまずその基本になっている一階述語の場合を示す．

5.1.1 述語表現

知識表現の基本形式は述語記号 f，集合 X_i を変域とする変数を x_i とした時，$(f x_1, \cdots, x_n)$ の形で表わされるアトムまたはその否定形 $\sim(f x_1, \cdots, x_n)$ を，論理結合子，∩(積)，∪(和)，⇒(含意)，⇔(等価)によって構造化し，変数に関する限量化の条件∀および∃を付した論理式で表わす．実際には A⇒B (A ならB) は ∼A∪B，A⇔B (AとBは等価) は (A⇒B)∩(B⇒A) で置きかえられるが，以下の記述では理解しやすい ⇒ と ⇔ 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，すべての流体の集合（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，

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

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

(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

$$(\forall x/P)(\forall y/F)(\forall z/R)(\forall u/R)[(\text{spc-grv } x\ z)\cap(\text{spc-grv } y\ u)\cap(\text{lt } z, u)\Rightarrow(\text{flt } x\ y)]. \quad (1)$$

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．

$$(\forall x/P)(\forall y/R)(\forall z/R)(\forall u/R)[(\text{mass } x\ y)\cap(\text{volume } x\ z)\cap(\text{div } u\,;\, y\ z)\Rightarrow(\text{spc-grv } x\ u)] \quad (2)$$

これらは ⇒ の左右両項の意味的な関係を示す知識であるが，$(father\ a\ b)$ を“a は b の父親である”を表わすアトムとすると，$(\forall x/CHILD)(\exists y/MAN)(father\ y\ x)$ 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ものもある．$(\exists y/MAN)$ は $x\in CHILD$ なる任意の x にたいし，$father$ 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．これにたいし，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．$(\exists y/MAN)(\forall x/CHILD)(father\ y\ x)$ は“ある男 y が居り，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 ではこれを図3(a) のように，前者を AND-

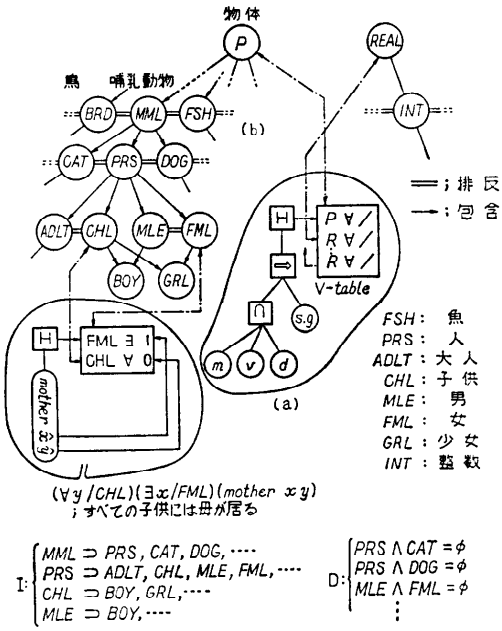


図 3 概念および知識の構造
Fig. 3 Structure of concepts and knowledge.

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．

5.1.2 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関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，$X \wedge Y = \phi$ の時D型アークで結合される．例えば“男”⊃“少年”，“男”∧“女”$=\phi$ である．これにより図 3(b)の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（図 3）．

5.2 推 論 処 理

5.2.1 推論の方式

知識 A⇒B は“Aが真ならBも真”を表わし，条件に相当する ⇒ 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⇒H の関係にあれば A⇒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 $K \equiv (A_1 \cap \cdots \cap A_m \Rightarrow B)$ と，仮説 $H \equiv D_{11}{}^*, \cdots, {}_{n-1}{}^*D_n$ が与えられた時Hの中のある D_i について $B \Rightarrow D_i$ が示されれば，これらは仮説 $H' = D'_{11}{}^*, \cdots, {}_{i-2}{}^*D'_{i-1\,i-1}{}^*(A_1 \cap \cdots \cap A_m)_i{}^*D'_{i+1\,i+1}{}^*, \cdots, {}_{n-1}{}^*D'_n$ に等価である．ここで i^* は i 番目の結合子（∩または∪），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(⇒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，$C \Rightarrow \sim D_i$ なら偽を示すラベル**F**が，以上のどれでもない（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).

$F = (\exists x/GIRL)(\forall y/PERSON)(like\ y\ x)$；“ある少女が居てすべての人がその少女が好きである”は $H = (\forall y/MAN)(\exists x/PERSON)(like\ y\ x)$；“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 $F \Rightarrow H$ は成りたたなくなる．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きまる．Fは $x_1 \in X_1, \cdots, x_n \in X_n$ を，Hは $y_1 \in Y_1, \cdots, y_n \in Y_n$ を変数として含む時，$F \Rightarrow H$ の条件は単純で，

(g1)； 各変数ごとに**表 2** の関係が成立する．

(g2)； 接頭部の変数順序について，Fでは …$(\forall x_i)\cdots(\exists x_j)$ であり，Hでは …$(\exists y_j)\cdots(\forall y_i)\cdots$ 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（文献23）参照）．（実際には仮説Hの否定形 ～H を作って処理を行うが，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**

表 2 含意規則 (g1)
Table 2 Implicative rule (g1).

Q_{pi}	$Q_{qi}(\overline{Q}_{qi})$	CONDITION on DOMAINS
∀	∀(∃)	$X_i \supset Y_i$
∀	∃(∀)	$X_i \wedge Y_i \neq \phi$
∃	∀(∃)	NON IMPLICATIVE
∃	∃(∀)	$X_i \subset Y_i$

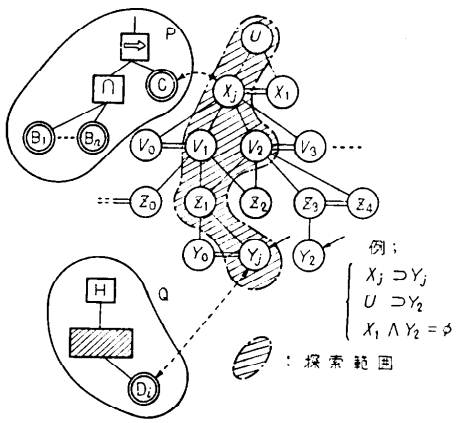


図 4 候補論理式の探索とテスト
Fig. 4 Searching a candidate and its test.

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 ∀，変域 Y_j を持つ変数（もしくは定数）を y_j とする．もし $F \Rightarrow D_i$ をみたすような 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 ∀ をもち，変域 X_j は $X_j \supset Y_j$ でなくてはならない（表 2）．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 型アークを辿って達することのできる範囲のノードにリンクされている（図 4）．TIC は D_i 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 ∀ をもつ変数が 2 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表 2 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 が判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）[22]．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．

〔B〕 置 換 規 則 (RR-Replacement Rule)

K; $A_1 \cap \cdots \cap A_m \Rightarrow B$ の B と $H = D_{11}{}^*, \cdots, {}_{n-1}{}^*D_n$ の D_i について $B \Rightarrow D_i$ が判定されたとする．K に含まれる変数のうち B に含まれるものを $\mathbf{X}_0$，残りを $\mathbf{X}_1$ とする．また，H に含まれる変数のうち D_i に含まれるものを $\mathbf{Y}_0$，残りを $\mathbf{Y}_1$ とする．すると H′ の各変数の変域と限量記号は表 3 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（文献 23）参照）．変域が変わることは変数に代入が行われることに相当する．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．

表 3 置 換 規 則
Table 3 Replacement rule.

	Q_{pi}	$Q_{qi}(\overline{Q}_{qi})$	$Q_{ri}(\overline{Q}_{ri})$	Z_{ri}/W_{ri}
$x_i \in \mathbf{X}_0 = \mathbf{Y}_j$	∀	∀(∃)	∀(∃)	Y_i
	∀	∃(∀)	∃(∀)	$X_i \wedge Y_i$
	∃	∃(∀)	∀(∃)	X_i
$x_i \in \mathbf{Y} - \mathbf{Y}_j$	—	∀(∃)	∀(∃)	Y_i
	—	∃(∀)	∃(∀)	Y_i
$x_i \in \mathbf{X} - \mathbf{X}_0$	∃	— —	∀(∃)	X_i
	∀	— —	∃(∀)	X_i

〔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，∀ 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 (lt x, n) は x, n に値が代入された時点で評価可能であり，条件が成立てば真，さもなくば偽である．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／手続きルーチンを準備し，図 2 のプログラム・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チン・パッケージを登録して用いることができる[26]．

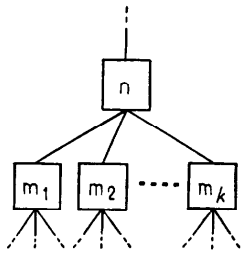
推論に際して仮説の中に現われる PTA は評価可能になると対応するルーチンが呼ばれ評価されるが，それ以外は未評価のまま残される．これは TT と呼ばれるアルゴリズムの前半で行われる．

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表 4 の規則にしたがって評価され，ルート・ノードの評価が仮説の真偽を表わす．表 4 で評価されるノードが n，その直下ノードの集合を $\{m_1, \cdots, m_n\}$ とする．ノード n が ∩ ノードか ∪ ノードかにより規則が異なり，(…1…) の左半は n が ∪ の場合，右半は ∪ の場合を示す．この評価は TT の後半が行う．

表 4 AND-OR 木内のノードの評価
Table 4 Evaluation of nodes in AND-OR tree.

	(n: ∩/n: ∪)	(n: ∩/n: ∪)
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
n: 評価されるべきノード
(∪ or ∩)
m_i: n の下のノード
($i=1,2,\cdots,k$)



Q: $(\exists x/I)$(fct xz)?: "2の階乗は?"

[0th] TT: 評価できるアトムあり?——否. 次へ
[1st] SR: D=Q (DとしてQそのものを用いる)
TIC: (A) ⇏ D
(B)の右辺 ($=(\forall x/I)(\forall n/I)$(fct xn)) ⇒ D
$n \leftarrow 2$
RR: ∼(eql 2 0)∩(sub m; 2 1)∩(mult x; y 2)∩(fct y m)
TT: ∼(eql 2 0)@T, (sub m; 2 1)@T, $m \leftarrow 1$
Q → (mult x; y 2)∩(fct y 1)
(評価済みアトムは除去)
[2nd] SR: D′=$(\exists y/I)$(fct y 1)
([1st] と同じ手順を繰りかえす. 次のようになる)
Q → (mult x; y 2)∩(mult y; z 1)∩(fct z 0)
[3rd] SR: D″=$(\exists z/I)$(fct z 0)
TIC: (A) ⇒ D″, $z \leftarrow 1$, (fct 1 0)@T
TT: Q → (mult x; y 2)∩(mult y; 1 1)
END
ここで (…)@T; (…) にラベル T を付す
$x \leftarrow a$; a が x に代入される
Q → (Formula); Q は (Formula) に変えられる

(A) (fct 1 0)
(B) $(\forall x/I)(\forall y/I)(\forall m/I)(\forall n/I)$[∼(eql n 0)∩(sub m; n 1)
∩(mult x; y n)∩(fct y m) ⇒ (fct x n)]
where (fct x n); "$n!=x$"

図 5 再帰的に定義された関数評価の例—階乗
Fig. 5 Definition of factorial function and its evaluation through deductive operation.

推論過程で新しい部分木構造により置きかえられるのは 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 ⇒ の右辺のアトムの述語記号を 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, ⇒ 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が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) (3)

(B) $(\forall x/I)(\forall y/I)(\forall m/I)(\forall n/I)$[∼(eql n 0)
∩(sub m : n 1)∩(mult x y n)
∩(fct y m) ⇒ (fct x n)] (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 の値が定ま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.

この関数は有限の回数で求まる. 2 の階乗を求める例が**図 5** に示されている. 推論が終った時点で実行可能な乗算の列が求まる. **図 6** 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.

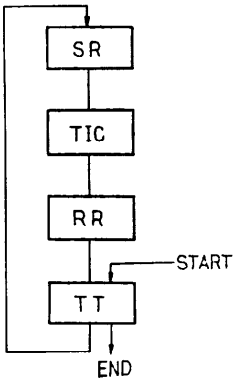


図 6 推論手続き
Fig. 6 Duductive procedure.

5.3 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

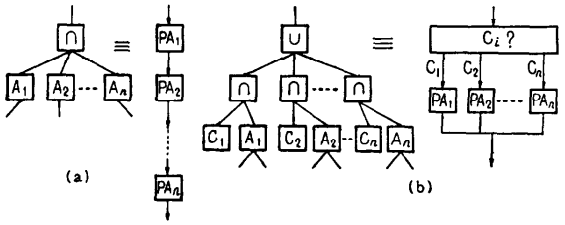


図 7 AND-OR 構造とプログラム構造
Fig. 7 AND-OR structure and program structure.

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，

$$A_1 \cap \cdots \cap A_n \qquad (5)$$

$$(C_1 \cap A_1) \cup \cdots \cup (C_m \cap A_m) \qquad (6)$$

であるがこれらはそれぞれ**図 7** 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と，これ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二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も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[22]．

5.4 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[27]．関係型データベースはある一定の関係 f にある n 組の集まりである関係ファイル F が $F=\{(x_1^j, \cdots, x_n^j)/x_1^j \in X_1, \cdots, x_n^j \in X_n,\ j=1,2,\cdots,N\}$ で表わされる．これは幾何学的に見ると $X^1 \times \cdots \times X^n$ なる n 次元直積空間の点 $(x_1^j, \cdots, x_n^j)$ の集まりである．もし $F=X_1 \times \cdots \times X_n = \boldsymbol{X}$ ならこのファイルは n 次元超立体を形成し，この中のすべての点が関係 $(f x_1^j, \cdots, x_n^j)$ を満たすから，これは論理表現 $(\forall x_1/X_1), \cdots, (\forall x_n/X_n)(f x_1, \cdots, x_n)$ 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→K の一例でもある．

F が $\boldsymbol{X}$ 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: $x_1 \cdots x_n$)；“n 組 $\boldsymbol{x}=(x_1, \cdots, x_n)$ 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，

$$(\forall x_1/X_1)\cdots(\forall x_n/X_n)[(\text{FGET F}: x_1 \cdots x_n) \Rightarrow (f x_1 \cdots x_n)] \qquad (7)$$

と表わすことができる．さらにアトム FGET は n 組 $\boldsymbol{x}$ をファイル F から取り出す役割りを持った手続き型アトムとすると，上式はファイルへのアクセスを含めた，ファイルの形式と意味間の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＋処理操作が意味的に知識表現と等価であること，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**表 5** のように定義する．また推論処理終了時にファイ

表 5 基本データベース操作関数
Table 5 Primitive database operation.

i) AND (F_1F_2)
$=\{(x_1^i, \cdots, x_n^i)/(x_1^i, \cdots, x_n^i) \in F_1 \wedge F_2,\ i=1,2,\cdots\}$

ii) OR (F_1F_2)
$=\{(x_1^i, \cdots, x_n^i)/(x_1^i, \cdots, x_n^i) \in F_1 \vee F_2,\ i=1,2,\cdots\}$

iii) MULT $(FY)=\{(x_1^i, \cdots, x_n^i, y^j)/(x_1^i, \cdots, x_n^i) \in F_1, y^j \in Y, i=1,2,\cdots, j=1,2,\cdots\}$

iv) DIV $(F_1x_k)=\{(x_1^i, \cdots, x_{k-1}^i, x_{k+1}^i, \cdots, x_n^i)/(x_1^i, \cdots, x_n^i) \times X_k \in F_1,\ i=1,2,\cdots\}$
DIV $(\hat{X}^k X^k)=1$(if $\hat{X}^k \supset X^k$) or 0(if $\hat{X}^k \not\supset X^k$)

v) PRJ $(F_1X_k)=\{(x_1^i, \cdots, x_{k-1}^i, x_{k+1}^i, \cdots, x_n^i)/(x_1^i, \cdots, x_n^i) \in F_1,\ i=1,2,\cdots\}$
PRJ $(\hat{X}^k X^k)=1$(if $\hat{X}^k \wedge X^k \neq \phi$) or 0(if $\hat{X}^k \wedge X^k = \phi$)

vi) CLP $(F_1X_k)=\{(x_1^i, \cdots, x_k^i, \cdots, x_n^i)/(x_1^i, \cdots, x_k^i, \cdots, x_n^i) \in F_1,\ x_k^i \in X_k,\ i=1,2,\cdots\}$

vii) EXPD $(F_1X)=\{(x_1^i, \cdots, x_n^i x^i)/(x_1^i, \cdots, x_n^i) \in F_1,\ x^i \in X,\ i=1,2,\cdots\}$

where
F_1; $\{(x_1^i, \cdots, x_n^i)/x_r^{i'} \in X_r,\ 1 \leq r \leq n,\ i=1,2,\cdots,N_1\}$
F_2; $\{x_1^{i'}, \cdots, x_n^{i'})/x_r^{i'} \in X_r,\ 1 \leq r \leq n,\ i=1,2,\cdots,N_2\}$

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 $\boldsymbol{X}=\{X_1,\cdots,X_n\}$ とする．

(h1)； ファイル F_1, F_2 がそれぞれ属性の集合 $\boldsymbol{X}_1=\{X_{i_1},\cdots,X_{i_r}\}$, $\boldsymbol{X}_2=\{X_{j_1},\cdots,X_{j_s}\}$ を有し，$\boldsymbol{X}'=\boldsymbol{X}_1\vee\boldsymbol{X}_2$ とする．また $F_1{}^*=F_1\times(\boldsymbol{X}'-\boldsymbol{X}_1)$, $F_2{}^*=F_2\times(\boldsymbol{X}'-\boldsymbol{X}_2)$ とすると，

$$\begin{aligned}&(\text{FGET } F_1\colon x_{i_1}\cdots x_{i_r})\\&\quad\circ(\text{FGET } F_2\colon x_{j_1}\cdots x_{j_s})\\&=(\text{FGET } F_1{}^*\colon x_1\cdots x_n)\\&\quad\circ(\text{FGET } F_2{}^*\colon x_1\cdots x_n)\\&=(\text{FGET } F_1{}^*\hat{\circ}F_2{}^*\colon x_1\cdots x_n)\end{aligned}$$

が成り立つ（証明略）．ここで$\circ$は論理結合子 $\cap$ 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 $\hat{\circ}$：$\wedge/\vee$ 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 内属性集合を $\boldsymbol{X}_i$, $F_i{}^*=F_i\times(\boldsymbol{X}-\boldsymbol{X}_i)$, AND-OR 構造の各ノードの $\cap/\cup$ 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まる統合ファイルを F^* とすると，もとの論理式が

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

に帰する．

(h2)； この結果得られた単一アトム論理式は次の規則で表5の基本オペレータによる操作手順に変換される．

$$\begin{aligned}&(Q_1\ x_1/X_1)\cdots(Q_{m-1}x_{m-1}/X_{m-1})(\forall\, x_m/X_m)\\&\quad(\text{FGET } F^*\colon x_1\cdots x_m)\\&=(Q_1\ x_1/X_1)\cdots(Q_{m-1}\ x_{m-1}/X_{m-1})\\&\quad(\text{FGET DIV } (F^*\ X_m)\colon x_1\cdots x_{m-1}))\qquad(9)\\&(Q_1\ x_1/X_1)\cdots(Q_{m-1}\ x_{m-1}/X_{m-1})(\exists\, x_m/X_m)\\&\quad(\text{FGET } F^*\colon x_1\cdots x_m)\\&=(Q_1\ x_1/X_1)\cdots(Q_{m-1}\ x_{m-1}/X_{m-1})\\&\quad(\text{FGET PRJ } (F^*\ X_m)\colon x_1\cdots x_{m-1}))\qquad(10)\end{aligned}$$

ただし，Q_i は $\forall$ または $\exists$ である．この規則を繰り返し適用すると最終的に (FGET $\hat{F}\colon\phi$) が得られる．$\hat{F}$ は F^* にたいする操作手順を示す．$\hat{F}$ はさらに**表6**の最適化規則が適用される．表6の規則は，関係データベースにおける最適化規則[28]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．

表 6 最適化規則
Table 6 Optimizing rules.

(1) DIV (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) DIV (AND (F_1 F_2) X) $=$ AND (DIV (F_1 X), DIV (F_2 X))

(3) PRJ (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) PRJ (OR (F_1 F_2) X) $=$ OR (PRJ (F_1 X) PRJ (F_2 X)).

(5) DIV (OR (F_1 MULT (X F_2)) X) $=$ OR (DIV (F_1 X) F_2)

(6) PRJ (AND (F_1 MULT (X F_2)) X) $=$ AND (PRJ (F_1 X) F_2)

(7) AND (MULT (X F_1) MULT (X F_2)
$=$ MULT (X AND (F_1 F_2))

(8) OR (MULT (X F_1), MULT (X F_2)
$=$ MULT (X OR (F_1 F_2))

ある．

例．

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

PRJ (F, X), DIV (F X) の演算は，(i 1) 条件を満たす組以外の組はすべて消去すると同時に，(i 2) 属性Xも消去するが，答を生成するため (i 1) のみを行う 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．以下に極めて単純ではあるが上述の諸機能を利用する一例を示す．

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 $F_m(M, R_1)$, $F_v(M, R_2)$, $F_{sg}(L, R)$ とする．これらファイルの入力時に同時にファイル記述情報として

$$(\forall\, x/P)(\forall\, y/R)[(\text{FGET}\colon F_m\ x\ y)\Rightarrow(\text{mass } x\ y)]\qquad(11)$$

$$(\forall\, x/P)(\forall\, y/R)[(\text{FGET}\colon F_v\ x\ y)\Rightarrow(\text{vol } x\ y)]\qquad(12)$$

$$(\forall\, x/P)(\forall\, y/R)[(\text{FGET}\colon F_{sg}\ x\ y)\Rightarrow(\text{spc-grv } x\ y)]\qquad(13)$$

が知識ベースに入れられる．知識ベースにはこの他に

```
# 1602 G

[EX/PO] [AY/LIQ] (FLT, SX, SY)?
[1] (#FGET #WT PO1 RNUM9 *WRK1)
[2] (VPRJ *WRK1 RNUM9 *WRK1)
[3] (#FGET #VOL PO1 RNUM10 *WRK2)
[4] (VPRJ *WRK2 RNUM10 *WRK2)
[5] (AND *WRK1 *WRK2 *WRK3)
[6] (EXPD *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 PO1 *WRK1)
[14] (MULT *WRK3 LIQ2 *WRK3)
[15] (AND *WRK1 *WRK3 *WRK2)
[16] (#GT RNUM5 RNUM4 *WRK2)
[17] (PRJ *WRK2 RNUM4 *WRK2)
[18] (PRJ *WRK2 RNUM5 *WRK2)
[19] (DIV *WRK2 LIQ2 *WRK2)
[20] (VPRJ *WRK2 PO1 *WRK2)
[21] (OUT *WRK2 *TTY)
```

図 8 実 例
Fig. 8 An example.

一般的な知識(1)や(2)が入っているとする.

ユーザは計算機に関する知識は要求されない. たとえば $(\exists x/P)(\forall y/L)(\mathrm{ft}\ x\ y)$; "すべての液体に浮く物質は?" のように聞くだけでよい. 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)を見い出す. 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. この実験結果が図8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[29]. この手順は冗長なファイル・アクセスや処理を排除す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.

5.5 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, 4.2.1 項の (d1) 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ため[30]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しい. これからは主動詞を含む文, "太郎は * を考える" の * の部分に, 対象の代りに, 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, 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. たとえば次のように用いられる.

$(\exists x/PERSON)$ (see Taro (fall x roof));

"太郎は誰か (人) が屋根から落ちるのを見る"

関係語のうち see, tell などの語に対応するアトムは "論理式" タイプの変数が許されるが, add, in などは許されていない. 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よう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.

5.6 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対象の集合の構造の分析に基づき[31], TIC においてアトム内の通常変数に対しては表2を適用し, 内部論理式にたいしては知識と仮説内の対応する内部論理式を $(\cdots)x, (\cdots)y$ とすると表2の関係をこの含意関係 $(\cdots)x \Rightarrow (\cdots)y$ 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．これらの根拠や具体的な方式については紙数の都合で省略する．

5.7 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．したが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，KAU では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基本サブシステムとして組み込めるように考慮している．しかし紙数の都合でこの部分は省略する．

6. 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を含む次の版の計画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たい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置いた．それでも紙数の都合で多くの部分を省略せねばならなかった．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．

参 考 文 献

1) Ernst, G. W. and Newell A.: GPS; A Case Study in Generality and Problem Solving, Academic Press (1969), 他.
2) Ohsuga, S.: Toward Intelligent Interactive Systems, Proc. The IFIP W. G. 5.2 Workshop Seillac II on Methodology of Interaction, North-Holland (1979).
3) 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, pp. 944-951 (1978).
4) Date, C. J.: An Introduction to Database Systems, Addison-Wesley (1975).
5) 益田隆司： 仮想メモリ方式による計算機システムの解析と評価に関する研究，東京大学工学系学位論文（2月 1977).
6) Denning, P. J. and Schwartz, S. C.: Properties of the Working Set Model, Comm. ACM, Vol. 15, No. 3, pp. 191-198, 他.
7) 大須賀節雄： プログラムおよびデータの特性指標とその仮想記憶システムの特性への影響，情報処理，Vol. 20, No. 3, pp. 256-264 (1979).
8) Babb, E.: Implementing a relational databases by means of specialized hardware, ACM Trans. on Database Systems, Vol. 4, No. 1, pp. 1-29 (Mar. 1979).
9) Buchus, J.: Can programming be liberated from the von Neumann style? A functional style and its algebra of programs, Comm. of ACM, Vol. 21, No. 8, pp. 613-641 (1978).
10) Ashcroft, E. A. and Wadge, W. W.: Lucid, a Non procedural Language with Iteration, Comm. of ACM, Vol. 20, No. 7, pp. 517-526 (July 1977), 他.
11) Dennis, J. B. and Misunas, D. P.: A Preliminary Architecture for a Basic Data Flow Processor, Proc. 2nd Annual Symposium on Computer Architecture, pp. 126-132 (Jan. 1975), 他.
12) Bobrow, D. G. (chaired): A Panel on Knowledge Representation Proc. 5th IJCAI pp. 983-992 (1977).
13) Winograd, T.: Towards a Procedural Understanding of Semantics, Stanford Univ. AIM-292.
14) Hendrix, G. G.: Expanding the Utility of Semantic Networks Through Partitioning, 4th Int'l Joint Conf. on AI (1975).
15) Mylopoulos, J., Cohen, P., Borgida, A. and Sugar, L.: Semantic Networks and the Generation of Context 4th Int'l Joint Conf. on AI (1975).
16) Nagao, M. and Tsujii, J.: S-Net: A Foundation for Knowledge Representation Language 6th Int'l Joint Conf. on AI 1979 (to appear).
17) Randall, D., Bunchanan, B. and Shortliffe, E.: Production Rules as a Representation for a Knowledge-Based Consultation Program, Artificial Intelligence, Vol. 8, No. 1, pp. 15-46 (1977).
18) Chang, C. L. and Lee, R. C. T.: Symbolic Logic and Mechanical Theorem Proving, Academic Press (1973).
19) Robinson, J. A.: A Machine Oriented Logic Based on the Resolution Principle JACM Vol. 12 (Jan. 1965).
20) van Emden: Programming with Resolution Logic, Machine Intelligence, Vol. 8, pp. 266-299.
21) W. V. O. クワイン (中村, 大森訳)：論理学の方

法，岩波書店，他.
22) 大須賀節雄：意味処理と知識利用のシステムについて，日本語情報処理シンポジウム前刷，(1978).
23) 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. 2, pp. 100-109 (1976).
24) 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).
25) Ohsuga, S.: Semantic Information Processing in Man-Machine Systems, Proc. 1977 IEEE Conf. on Decision & Control, pp. 1351-1358 (Dec. 1977).
26) 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，pp. 695-696 (1979).
27) Codd, E. F.: A Relational Model of Data for Large Shared Data Banks, Comm. ACM, Vol. 13, No. 6, pp. 377-397 (1970)，他.
28) Smith, J. M. and Chang, P. Y. T.: Optimizing the Performance of a Relational Algebra Databases Interfare CACM Vol. 18, No. 10, pp. 568-579 (1975).
29) 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，pp. 697-698 (1979).
30) Huet, G. P.: A Mechanization of Type Theory, 3rd IJCAI, pp. 139-146 (1973).
31) Ohsuga, S.: Theoretical Basis for a Knowledge Representation System, Proc. Int'l Joint Conf. on AI (to appear).

(昭和 54 年 8 月 29 日受付)