ICOT Technical Memorandum: TM-0830

TM-0830

並列知識ベース処理実験システム Mu-Xのインタフェース

酒井　浩，武脇敏晃(東芝)

December, 1989

ICOT

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

(03) 456-3191~5
Telex ICOT J32964

Institute for New Generation Computer Technology

並列知識ベース処理実験システムＭｕ－Ｘのインタフェース

Logical Interface of Knowledge Base Machine Mu-X:
a proposal for knowledge information processing

酒井　浩
Hiroshi Sakai

武脇　敏晃
Toshiaki Takewaki

（株）東芝　総合研究所
Research & Development Center

情報通信システム技術研究所
Information & Communication Systems Lab.

Toshiba Corporation

e-mail: sakai, takewaki@isl.rdc.toshiba.co.jp

１．はじめに

１９７０年代後半から盛んになった知識情報処理分野では，エキスパートシステムを初めとする各種の知識利用システムが実用化の段階をむかえている。このようなシステムでは一般に扱う対象を限定した比較的小さな知識ベースとそれを組合せて有用な結論を得るための推論機構を含んでいる。この知識ベースと推論機構は，Prolog，LispまたはC 言語等を用いて，ワークステーション（ＷＳ）上に単一のプログラムとして実現されることが多い。この場合，知識ベースは推論機構や使用目的に適した知識表現を用いて記述されている。

一方，より広範囲を対象とする知識ベースを構築する試みがなされている。電子化辞書研究所では，日本語と英語について単語辞書や概念辞書を構築しつつある。また，ＭＣＣのＣＹＣ［Lenat 86］では百科辞典の内容を数百個の基本概念を用いて表現しようとしている。これらの知識ベースは多目的の利用を目指しており，その構築にあたっては，特定の推論機構を意識しているわけではない。

また，今後の計算機の利用形態を考えると，ネットワーク化の進展に伴い，知識情報処理分野でも分散処理が多くなるものと思われる。例えば，複数のＷＳ上で応用プログラムが動作し，知識ベース（長期記憶）や，より更新の頻度の高いワーキングメモリ（短期記憶）を共有して分散協調問題解決を行なうことが増えると予想される。

このような状況は，ＥＤＰ分野においてデータベースが発達してきた状況と非常に類似していると思われる。すなわち，初期のＥＤＰではプログラムにあわせてデータが作成されたため，ひとつの計算機上に同じデータが重複して格納されることが問題となった。そのためデータをプログラムから独立させることの重要さが認識され，データベースを開発するようになった。そして，現在，データベース分野では航空機の座席予約のようなオンライントランザクション処理の割合が増えつつある。

著者らは知識情報処理分野でも大規模な知識ベースや更新の多いワーキングメモリを格納・管理するためにＥＤＰ分野のデータベース管理システム（ＤＢＭＳ）あるいはデータベースマシンに相当するものが必要になると考えている。ただし，従来のＤＢＭＳをそのまま適用するのではデータモデル等の点で十分ではないと考えている。そこで，知識を表現するモデルとして項関係モデル［Yokota 86］，［Morita 86］を提案した。また，このモデルに基づいて知識を管理格納し，ホストマシンからの問合せを並列処理する試作機としてＭｕ－Ｘを開発した。ただし，Ｍｕ－Ｘではpure prolog に近い演算機能を実現しており，［Yokota 86］のモデルの拡張となっている。また，プログラミング言語からデータベースを利用する場合の問題点として従来から指摘されているインピーダンスミスマッチに対しては，Ｍｕ－Ｘにtuple-at-a-time のインタフェースを用意することにより，その解消を

はかっている。

本稿では，知識情報処理分野でのＤＢＭＳという位置づけで，Ｍｕ－Ｘのインタフェースについて述べる。

なお，本研究は通産省第五世代コンピュータプロジェクトの一環として，（財）新世代コンピュータ技術開発機構（ＩＣＯＴ）と共同で行なった。

2．知識情報処理分野へのＤＢＭＳ適用に関する検討

ここでは，電子化辞書のように大規模な知識ベースや更新頻度の高いワーキングメモリの管理・格納に必要なＤＢＭＳの機能について考察する。

最初に，多くのエキスパートシステムでは，知識ベースと推論機構を密に結合する形でひとつの計算機上にProlog，Lispあるいは C 等のプログラミング言語により実現する（これを一体化方式と呼ぶことにする）理由を考察する。

一体化方式では，知識ベースと推論機構を分離した形で実現する（これを分離方式と呼ぶことにする）に比べ，推論機構間で知識ベースやワーキングメモリの内容を共有するには向かないものの，下記の長所をあげることができる。

①一体化方式ではプログラミング言語が一般にチューリング完全であるため，何かある処理ができずに困るということはほとんどない。また，プログラミング言語は種々の知識表現を実現できる。
分離方式では，知識はある一定のモデルに従って格納され，可能な演算はモデルによって規定されるため，必要な演算が簡単に行なえないことも多い。

②推論機構と知識ベースは，ゼロから試行錯誤的に少しずつ開発することが多い。この場合，ひとつの言語で両方を実現する一体化方式の方が柔軟性が高く，開発に都合が良い。

③一体化方式では推論機構と知識ベースの間でデータ形式の変換が不要で，かつ同一のプログラム中であるため処理が高速である。分離方式の場合，データ形式の変換およびマシンやタスク間のデータ転送が必要なため，処理速度の点で不利である。

④効率の良い推論を実現するには，きめ細かな制御が必要である。例えば，あるデータの内容に応じて次にどのデータを参照すべきかを判断しなければならないことが多い。つまり，知識ベースをtuple-at-a-time で高速に参照する必要性が高い。その点，一体化方式ではプログラミング言語で知識ベースを実現するため，特に問題はない。
分離方式の場合もtuple-at-a-time の処理が高速であれば問題はない。しかし，set-at-a-timeだけが得意なデータベースではインピーダンスミスマッチを生ずる。

一体化方式はこのように多くの長所を有している。従って，分離方式によりＤＢＭＳに知識ベース等を共有する場合にも，その長所はなるべく継承する方が望ましいと考えられる。ただし，一体化方式と分離方式では推論機構と知識ベースの機能分担は同じである必要性はないと考えられる。例えば，エキスパートシステムがＤＢＭＳに格納されている知識ベースを利用する場合を考えると，システム起動時にＤＢＭＳからホストマシンに必要な知識を一括してロードし，その後は従来と同様，ホストマシン単体上でシステムが動作すれば良いと考えられる。

以上の考察の結果，ＤＢＭＳを知識情報処理に適用するには，ＤＢＭＳは下記の要求を満たすべきであると考えられる。

①明確で強力なモデルを持つこと
知識ベースを格納するには，知識を簡潔に曖昧さなく表現できることが必要である。また，演算に関してできるだけ豊富（できればチューリング完全）な方が良い。

②プログラミング言語と親和性があること
プログラミング言語がサポートする基本的なデータ型および演算が必要である。これは，ホストマシン上で構築した知識ベースを後でＤＢＭＳに格納する時には特に必要であると考えられる。また，効率の良いtuple-at-a-timeのデータ参照も，プログラミング言語との親和性を高める上で重要と考えられる。

また，複数の推論機構で知識ベースやワーキングメモリの内容を共有するためには，従来の

ＤＢＭＳと同じく，下記の機能が必要であると思われる。

③知識の管理機能
利用者や推論機構から，どのような知識がどのような形式で格納されているか調べられなければならない。そのためには，例えばデータ辞書に，ある一定の書式で格納されている各知識について記述される必要がある。

④同時実行制御
知識ベースやワーキングメモリを複数の推論機構で共有する場合，更新処理のコンシステンシを保障するために同時実行制御が不可欠である。同時実行制御をロック機構で実現する場合には，ロックの単位を大きくする場合（例えば知識ベース全体）と，小さくする場合（例えばワーキングメモリのメモリセル）の両方が必要と考えられる。

⑤信頼性と障害復旧
信頼性が高く，たとえ障害が発生しても影響が最小限ですむ方が望ましい。

知識情報処理に関連したデータベースの研究としては，従来，演繹データベース，オブジェクト指向データベース等が提案され，システムの試作が行われている。

演繹データベースは，データベースと推論機能を結合しようとする試みであり，主な課題は再帰問合せの処理と否定の扱いである。データモデルには，構造を持たない（function free）データしか扱えない正規型関係モデルを採用する場合も多い。

ＩＣＯＴのＫａｐｐａ[Yokota 88]は，自然言語処理と定理証明を目的としている。自然言語の辞書を格納するため非正規型関係モデルを採用している。

ＭＣＣのＬＤＬ[Tsur 86]は，Prologと類似の構文を持つ演繹データベースのインタフェース用の言語であり，Prologがtuple-at-a-timeの処理を行なうのに対して，set-at-a-timeの処理を行なう。また，データとして集合やリスト構造を扱うことができる。

東大大須賀研のＫＡＵＳ（多層論理に基づく推論機構，知識ベース，マン・マシン・インタフェースなどを備えた総合システム）の一部をなすＫＡＵＳデータベース[高須 87]では，多層論理のデータ構造を採用しており，集合が基本的なデータ型となっている。

このように知識情報処理をターゲットとするデータベースでは，知識を表現するために記述能力の高いデータモデルを採用している。

Ｍｕ－Ｘでも３章に述べるように記述性の高いモデルを採用するとともに，先に示したＤＢＭＳに対する要求をふまえた機能の検討およびインタフェースの設計を行なった。

３．項関係モデル

Ｍｕ－Ｘでは，知識を表現するため，項関係モデル[Yokota 86]を採用している。ただし，Ｍｕ－Ｘの項関係モデルは[Yokota 86]と比較して大幅に機能強化を行なっている。ここでは項関係モデルをＭｕ－Ｘで実現している機能に基づいて，再定義を行なう。

３．１　項と演算に関する定義

ここでは，項関係モデルの構成要素である項，および項に対する操作について述べる。

(1) 項

項は，下記の規則により構成される。

- 定数記号は項である。
- 変数記号は項である。
- ｎ引数関数記号ｆと項t1, t2, . . tnから構成されるf(t1, t2, . . tn)は項である。
- 上記の規則によって構成されるものだけが項である。

また，定数記号には数値や文字列が含まれ，それらに対する各種の演算が可能である。また，項の一種としてリストがあり，項関係モデルでは値の集合を表現する場合にも使用する。

なお，整数の全体集合をINT，文字列の全体集合をSTR，リストの全体集合をLIST，項の全体集合をTERMで表わすことにする。

次に項に対する主要な演算について説明する。

(2) 等号

$TERM^2$ から真理値集合への写像であり，項どうしが（変数記号を含め）完全に一致すれば真，そうでなければ偽とする。

(3) 引数抽出演算子

複合項のｉ番目の引数を得る関数π_iを定義する。

例　$\pi_i\ f(t1, t2, .., tn) = ti$

(4) 置換演算子

項ｐに含まれる変数記号を別の項で置換する演算子を定義する。

例　$\theta = \{x/a, y/b\}$ とすると

$\theta f(x, y, z) = f(a, b, z)$

なお，変数記号をリネーミングするだけの置換演算子を恒等置換演算子と呼ぶことにする。

(5) 項に対するその他の演算

項関係モデルでは，項に対して計算可能な演算はすべて含まれる。主な演算は，$TERM^n$ から真理値集合への写像と$TERM^n$ からTERMへの写像である。

Ｍｕ－Ｘが実際に備えている演算は，Prolog, Lisp, c 等のプログラミング言語の式（副作用の無いものに限る）に相当するものである。具体的な演算については付録に示す。

(6) 項の集合の集合に対する演算

項関係モデルでは，項の集合の集合に対する計算可能な演算はすべて含まれる。Ｍｕ－Ｘでは，max, min, avg など集約演算，与えられた項集合の各要素から構成されるリストを生成する演算を備えている。

３．２　項関係モデルの定義

(1) 項関係

項関係は，関数記号および引数の個数が同一である項の集合である。ここでは項関係の要素のことをタプルと呼ぶ。

(2) 拡張制約演算

項関係Ｐ，TERMから真理値集合への写像Ｃ，TERMからTERMへの写像Ｆから項関係Ｒを生成する演算であり，式１により定義される。

$R = \{F(p) \mid p \in P,\ C(p)\}$　… 式１

例　関係モデルの射影演算は，

$R = \{F(p) \mid p \in P\}$

と表わせるので，拡張制約演算に含まれる。

例　関係モデルの制約演算は，

$R = \{p \mid p \in P,\ \pi_i\ p = t\}$

ただし，ｔ は定数記号

と表わせるので，拡張制約演算に含まれる。

例　関係モデルの選択演算は，

$R = \{p \mid p \in P,\ \pi_i\ p = \pi_j\ p\}$

と表わせるので，拡張制約演算に含まれる。

例　[Yokota 86] の単一化制約は，

$R = \{\theta_1\ p \mid p \in P,\ {}^{\exists}\theta_1\ {}^{\exists}\theta_2\ (\theta_1\ \pi_i\ p = \theta_2\ \pi_j\ t)\}$

ただし，θ_1 と θ_2 は最汎化置換（ｍｇｕ）

と表わせるので，拡張制約演算に含まれる。

(3) 拡張結合演算

項関係ＰとＱ，$TERM^2$ から真理値集合への写像Ｃ，$TERM^2$ からTERMへの写像Ｆから項関係Ｒを生成する演算であり，式２により定義される。

$R = \{F(p, q) \mid p \in P,\ q \in Q,\ C(p, q)\}$　… 式２

例　関係モデルの結合演算は

$R = \{F(p, q) \mid p \in P,\ q \in Q,\ \pi_i\ p = \pi_j\ q\}$

と表わせるので，拡張結合演算に含まれる。

例　単一化結合演算[Yokota 86] は，

$R = \{F(\theta_1\ p,\ \theta_2\ q) \mid p \in P,\ q \in Q,\ {}^{\exists}\theta_1\ {}^{\exists}\theta_2\ (\theta_1\ \pi_i\ p = \theta_2\ \pi_j\ q)\}$

ただし，θ_1 θ_2 は置換演算子

と表わせるので，拡張結合演算に含まれる。

(4) 拡張集合演算

項関係ＰとＱから項関係Ｒを生成する演算であり，下記のとおり３種類の演算がある。

①積集合

$R = \{p \mid p \in P,\ q \in Q,\ {}^{\exists}\iota\ (p = \iota\ q)\}$

②和集合

$R = \{p \mid p \in P \vee p \in Q\}$

③差集合

$R = \{p \mid p \in P,\ q \in Q,\ {}^{\exists}\iota\ (p = \iota\ q)\}$

ただし，ι は恒等置換演算子

(5) 項関係に対する拡張集約演算

項集合P，項集合の集合からTERMへの写像Fから，項集合Rを生成する演算である。

$R = \{F(P)\}$

例　関係データベースの関係全体に対して，i番目の属性の最大値を求める演算は，

$F(P) = r(a) \quad s.t. \quad a = \max(\pi_i p)$

とすることにより，求めることができる。

(6) ネスト演算（項関係の同値類に対する拡張集約演算）

項集合P，$TERM^2$ から真理値集合への写像で反射律，対称律，推移律を満たすC（同値関係）及び項の集合の集合からTERMへの写像Fから，項集合Rを生成する演算である。

$R = \{F(Q) \mid Q$ はPのCによる同値類$\}$

この演算は，非正規型関係モデルのネスト演算に相当している。ただし，属性値に格納する関係は，本モデルの場合，リストとして表現される。

例　関係データベースの関係全体に対して，k番目の属性値によりgroup-byを行ない，各groupについてi番目の属性の最大値を求めるには，

$C(t,u) \triangleq \pi_k t = \pi_k u$

$F(P) \triangleq r(a) \quad s.t. \quad a = \max_{p \in P}(\pi_i p)$

とすればよい。

(7) アンネスト演算

項関係P，TERMから項の集合の集合を生成する演算Fから，項関係Rを生成する演算である。

$R = \{r \mid p \in P, \ r \in F(p)\}$

この演算は，非正規型関係モデルのネスト演算に相当している。

4．Mu－Xのインタフェース

ここでは，Mu－XはホストマシンとしてのPSIに対して提供しているインタフェースについて述べる。

4．1　インタフェースの特徴

(1) 項形式のインタフェース

Mu－Xでは，ホストマシンとの間の問合せ／結果の通信にも，項関係に格納しているのと同形式の項（タグつきデータ）を用いている。この方式は，送信側で項からテキストに変換し，受信側でテキストから項に再変換するのと比較して，

①データ形式の変換に伴う曖昧さを生じない
②変換が必要無いので処理時間が短い
③メッセージサイズが小さい

という利点がある。これは，Mu－XとホストマシンでデータのPSI内部表現がほぼ等しいために生ずる利点である。

(2) Prologのゴール列と類似の問合せ言語

項関係に対する問合せは，Prologゴール列と類似の構文で記述する。各サブゴールは項関係または評価可能述語であり，それらを組合せることができるため，記述能力は高い。ただし，Prologと異なりサブゴールとしてルールを指定することはできないため，append/3等の述語を自由に定義することはできない。Mu－Xではこの問題を解決するため，文字列やリストに対する評価可能述語を豊富に備えている。

問合せ言語については4．4で説明する。

(3) ホストマシンからの柔軟なタプルアクセスの実現

項関係モデルは，潜在能力としてはタプルの引数として論文の本文全体や画像を格納できる。そこで例えば，著者名，タイトル，論文の本文全体を3属性の項関係として格納する場合を考える。すると，著者名とタイトルは高速検索を行なう必要性が高いので磁気ディスクに格納し，大量になる論文の本文全体は光ディスクに格納するというように，物理的にわけて格納する必要があると思われる。

Mu－Xでは，ホストマシンが項関係を属性方向に分割したり，分割した片方のタプルから対応する他方のタプルに高速にアクセスできるようにすべきであると考え，定数記号の一種としてタプルの格納位置情報を用意した。そしてタプルの格納位置情報を求める評価可能述語や格納位置情報をもとにタプルをアクセスする評

価可能述語を用意した。なお，格納位置情報には特別なタグが付されており，特定の演算以外で不正に使用されることのないようにしている。

また，項の検索を高速化するため，ＳＳＣＷ［森田 86］等の機能も実現している。

(4) tuple-at-a-time のタプル操作

ホストマシンからＭｕ－Ｘ内の項関係に対して，tuple-at-a-time のタプル操作を実現することにより，ホストマシンのプログラミング言語との間のインピーダンスミスマッチの解消をはかっている。ホストマシンは項関係に対してタプル単位での読込みと書出し，およびキー値の指定によるタプルの参照，追加，削除，更新を効率良く行なうことができる。

(5) データ辞書による項関係の管理

Ｍｕ－Ｘでは，格納している各々の項関係をデータ辞書で管理している。データ辞書自身も項関係であり，利用者やホストマシンが問合せコマンドを用いてデータ辞書にアクセスできるだけでなく，Ｍｕ－Ｘの管理ソフトウェアもデータ辞書を問合せコマンドの解析に使用する。データ辞書を項関係としたため，見易さと処理の高速性を両立させることができた。

(6) 項関係およびタプル単位のロック機構

同時実行制御を行なうためのロックの単位を項関係全体および個々のタプルの両方のレベルでかけられるように，意図ロック［Gray 75］を採用した。

また，項関係は，各ホストマシンから共有されるパーマネントなものと，セッションの中でのみ有効なテンポラリなものとを区別して扱うようにした。この場合，同名の項関係があれば，テンポラリのものを優先するようにした。パーマネントな項関係をコピーしてテンポラリな項関係を作成することにより，ホストマシンがその内容を変更してその影響を調べるといった実験が可能となる。

４章の残りの部分では，Ｍｕ－Ｘが提供するインタフェース述語を機能別に説明する。

４．２　トランザクション制御用述語

①begintr 述語

トランザクションの開始を指示する。

②endtr 述語

トランザクションの終了を指示する。

③aborttr 述語

トランザクションのアボートを指示する。

４．３　項関係の管理用述語

①define述語

テンポラリな項関係のスキーマを定義する。

②drop述語

テンポラリまたはパーマネントな項関係について，定義および項関係全体を削除する。

③catalog 述語

テンポラリな項関係をパーマネント化する。

４．４　項関係操作用述語

ここで述べる述語は，すべてset-at-a-time の処理を行なう。

①retrieve述語

３章の項関係モデルで定義した演算を行なう。

retrieve(Result, Query) という形式で使用され，Prologのbagof(Result, Query, Ans) の Ans に相当する結果をテンポラリな項関係として生成する。

例　retrieve(r(X, Y), (p(X, Y, Z), Z<1)

の場合，関数記号p で３引数の項関係をＰとすると，下記に示すテンポラリな項関係Ｒを生成する。

$R = \{r(\pi_1 p, \pi_2 p) \mid p \in P, \pi_3 p < 1\}$

例　retrieve(r(X, Y), (p(X, Y, Z), Z=f(U))

の場合，関数記号p で３引数の項関係をＰとすると，下記に示すテンポラリな項関係Ｒを生成する。

$R = \{\theta_1 r(\pi_1 p, \pi_2 p) \mid p \in P, {}^{\exists}\theta_1 {}^{\exists}\theta_2 \ (\theta_1 \pi_3 p = \theta_2 f(U))\}$

Ｍｕ－Ｘのインタフェースでは，上の例に示すように演算子”＝”は，Prologと同じく単一化演算子として扱われる。３章で定義した等号を表わす場合は，演算子”＝＝”を使用する。

②lock述語

項関係を単位とする排他制御を行なう。

③append
結果をテンポラリな項関係に追加することを除いてretrieveと同じ処理を行なう。
④delete
項関係から条件を満たすタプルを削除する。

4．5　タプル入出力用述語
①get 述語
テンポラリまたはパーマネントな項関係に含まれるタプルを１つずつホストマシンに送る。
②put 述語
テンポラリな項関係に対してタプルを１つずつ追加する。
③getaslist 述語
テンポラリまたはパーマネントな項関係に含まれるタプルを転送できる範囲でいくつかまとめてホストマシンに送る。
④putaslist 述語
テンポラリな項関係に対してタプルを何個かずつ追加する。

4．6　単一タプル処理用述語
項関係（パーマネントとテンポラリのいずれも可）に対して，主キーの値の指定することによりtuple-at-a-time の操作を行なう。主キーには複合項は許されるが，変数記号が含まれていてはならない。
①insert述語
項関係にタプルを挿入する。
②locktpl 述語
タプル単位で排他制御を行なう。
③erase 述語
項関係からタプルを削除する。
④find述語
項関係中のタプルをホストマシンに送る。
⑤change述語
項関係中のタプルを更新する。主キーの値を変更する場合としない場合の２通りのインタフェースを備えている。

5．文献情報処理への適用
著者らは，ホストマシンＰＳＩからこのインタフェースを使用するアプリケーションとして科学論文に関する情報処理を行なった。知識ベースとして約2,000 件の論文に関する情報とそれらの参考文献約23,000件の情報を格納した。そして，「最も多くの論文が参考文献としてあげている論文を求める」や「著者別に論文数（共著は1/著者数とする）を求める」等，興味のある問合せを列挙し，それらを問合せを本稿のインタフェースで記述した。その結果，ほとんどの問合せはretrieve述語で簡単に記述することができた。また，複雑な問合せの場合には２～３個のretrieve述語の組合せ（例えば，アンネスト，ネスト，ソーティングの組合せ）で記述することができた。
この実験には構造データが扱えるカード型のユーザインタフェース［武脇 88］を使用することにより，視覚的にわかりやすい形で問合せを記述することができた。

6．おわりに
Ｍｕ－Ｘの項関係モデルとそのためのインタフェース言語について報告した。Ｍｕ－Ｘでは，ＤＢＭＳを知識情報処理分野に適用するという立場で，知識を記述する能力やホストマシンのプログラミング言語との親和性を重視した設計を行なっている。

謝辞
研究を御指導いただいたＩＣＯＴ各位に深謝いたします。

参考文献
[Gray 75] Gray, J., N., 'Granularity of Locks in a Large Shared Data Base', Proc. of the 1st Int'l Conf. on VLDB, 1975.
[Lenat 86] Lenat, D., et al, 'Cyc: Using Common Sense Knowledge to Overcome Brittleness and Knowledge Acquisition Bottlenecks', The AI Magazine, 6, No.4, pp.66-85, 1986.
[Monoi 88] Monoi, H., et al, 'Unification-Based Query Language for Relational Knowledge Bases and Its Parallel Execution', Proc. of the Int'l Conf. on FGCS'88, Tokyo, 1988.
[Morita 86] Morita, Y., et al, 'Retrieval-By-Unification Operation on a

Relational Knowledge Base', Proc. of the 12th Int'l Conf. on VLDB, Kyoto, 1986.

[Shibayama 88] Shibayama, S., et al, 'Overview of Knowledge Base Mechanism', Proc. of the Int'l Conf. on FGCS'88, Tokyo, 1988.

[Tsur 86] Tsur, S., Zaniolo, C., 'LDL: A Logic Based Data-Language', Proc. of the 12th Int'l Conf. on VLDB, Kyoto, 1986.

[Yokota 86] Yokota, H., et al, 'A Model and an Architecture for a Relational Knowledge Base, Proceedings of the 12th International Symposium on Computer Architecture, Tokyo, 1986.

[Yokota 88] Yokota, K., et al, 'Overview of the Knowledge Base Management System (Kappa)', Proc. of the Int'l Conf. on FGCS'88, Tokyo, 1988.

[酒井 89a] 酒井，武脇，" 並列知識ベース処理実験システムＭｕ－Ｘの開発"，第７回第五世代コンピュータに関するシンポジウム予稿集，1989.

[酒井 89b] 酒井，武脇，'知識ベースマシン'，東芝レビュー，44巻10号，1989.

[柴山 88] 柴山他，" 知識ベースマシンＭｕ－Ｘ"，DE88-6，電子情報通信学会

[高須 87] 高須，大須賀，" ＫＡＵＳデータベース"，昭和６２年度人工知能学会全国大会予稿集，1987.

[武脇 88] 武脇他，" 知識ベースマシン Mu-X(3)　～表型言語から項型言語への変換～"，第３７回情報処理学会全国大会予稿集，1988.

[森田 86] 森田他，" スーパーインポーズドコードを用いた構造体の検索方式"，第３３回情報処理学会全国大会予稿集，1986.

付録　Ｍｕ－Ｘが用意している演算

(1) 定数記号に対する演算

① INT* から INTへの写像
整数に対する通常の算術演算がある。

② INT* から真理値集合への写像
数値間の大小比較がある。

③ STR* から真理値集合への写像
文字列間の辞書式順序での大小比較，ワイルドカードを含むパターン照合がある。

(2) 項に対する演算

①TERM* から真理値集合への写像
- t ∈TERMが整数ならば真，そうでなければ偽
- t ∈TERM，u ∈TERMの間でθ_1 t ＝θ_2 u となるような置換演算子θ_1，θ_2 が存在すれば真，そうでなければ偽（単一化可能）
- t ∈TERM，u ∈TERMの間でπ_1 t ＝π_2 u が成立てば真，そうでなければ偽

②TERM* からTERMへの写像
- f(t)∈TERMに対して，g(t, a, x)を生成する。ただし，t ∈TERM，a は定数記号，x は項f(t)に含まれない変数記号とする。
- f(a, b)∈TERMに対してg(c)を生成する演算。ただし，a とb はともに数値であり，c はa とb の和であるとする。

③TERMから INTへの写像
項どうしの単一化可能性の必要条件を高速に判定する方法であるＳＳＣＷ［森田 86］等の特性値を求める関数が用意されている。

④整数の集合の集合から INTへの写像
整数の集合に対して，その最大値，最小値，平均値等を求める集約演算が用意されている。

⑤項の集合の集合から INTへの写像
項の集合に対して，その要素を要素とするリストを生成する演算が用意されている。

(6) リストに対する演算
下記のような演算が用意されている。
- l ∈LISTに対して，その要素数を与える関数
- t ∈LIST，u ∈LISTに対して，リストのアペンドを生成する関数
- t ∈LIST，u ∈TERM，TERM^2 から真理値集合へ写像Ｃに対して，t の要素v でＣ(v, u) となるものがあれば真，そうでなければ偽