知識ベース管理システムKappa
―非正規形モデルと演繹データベース―

3M-8

横田 一正
(財) 新世代コンピュータ技術開発機構

1. はじめに

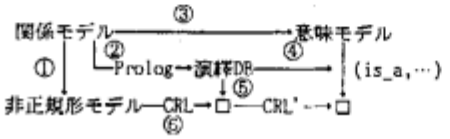
非正規形モデル(nested relational model)は，実世界の表現や処理効率などの点で，関係モデルに比べて好ましい点を多くもっている．しかし関係モデルのような明確な意味の定義が難しいために，非正規形モデルを論理の枠内で考えること，つまり非正規形モデルへの演繹的アプローチはこれまであまり試みられなかった．

Kappa(横田 86，87)では，非正規形モデルを内部モデルに採用しており，その上に知識情報処理システム，とりわけ自然言語処理や定理証明処理に必要とされ演繹機能を付加しようとしている．そのためにネストしたレコード構造に基づいた論理型言語CRL を考え，非正規形モデルへの演繹的アプローチ(横田 87a)をおこなっている．

本発表では，非正規形モデルの演繹DBための言語CRL の概要を例と共に説明する．

2. CRLの位置づけ

CRLは以下のように位置づけることができる:



①はDBのレベルでの関係モデルの拡張であり，AIM-PやVersoを始めとして多くの研究開発がおこなわれている．②は関係モデルを述語論理の立場から一階理論として捉え，関係の意味およびそれに対する操作(推論)を拡張したもので，Prologとの融合がもっとも多く研究されている．③は実世界の意味記述を強化したもので，ERモデルやIFO モデルなど多くの研究がある．これまでこれらは独立に研究されることが多かったが，演繹DBのための論理型言語の枠内で，意味モデルでの意味記述を考えたり(④)，非正規形モデルを表現すること(⑤)が試みられてきている．

しかし非正規形モデルは一般的には

R⊂E1×…×En，Ei::=D|P(R')

と定式化されるので，演繹DBのように関係(テーブル)を述語に対応させると，指数になってしまい手に負えなくなる．そこでCRL では，非正規レコードを表現する形式的枠組をまず与え，それに基づいた演繹DBとの統合(⑤)，さらには意味モデルの一部との統合を考えている．そのための言語がCRLである．

3. 非正規レコードの表現

CRL での領域は，通常の値集合ではなくそのベキ集合としている．つまり上の式でEi::=P(D)|P(R') としている．単一の値をそれだけを要素とする集合(singleton) と考えればすべてを集合として扱えるので都合がよいからである．つまり集合の構成子を導入するのではなく，単一化の自然な拡張としてネストを扱いたいからである．明らかに関係モデルの拡張になっており，非正規レコードはより一般的なレコード構造となっている．

非正規レコードの特徴は，横方向の列ネスト(colunn-nest)と縦方向の行ネスト(row-nest)である．CRLでは(属性，値)-対を拡張した項(NAV: nested AV-pair) としてネスト構造をもったレコードを表現している．

関係モデルのタプルは属性集合から値集合への関数としてていぎするが，列ネストをもったレコードは属性集合を木領域に拡張しその葉から値領域への関数として定義する．

(葉,値)-対の組合せ

(葉'値)＊…＊(葉'値)

で列ネストを表現している．属性値が単一値の集合であれば括弧はsyntax sugarとして省略する．葉は属性の結合

属性'…'属性

と表現する．CRL は非正規レコードに限定しているので，NAV 中の葉が一意的であること，および2つの任意の葉a1'leaf'a2，b1'leaf'b2に対a1=b1,a2=b2という制限を置いている．これは無限木の制限にもなっている．

[例1] 名前，出身地からなるレコード

名前'(姓''金枝上"＊名''敦史")＊出身地''神戸"，
名前'(姓''河村"＊名''元夫")＊出身地''名古屋"．

値は属性名(葉)によって修飾されているので，位置はどこにあってもよい．

次に行ネストはネストする属性を部分木の集合に対応させて表現できる:

属性'…'属性'{部分木,…}．

多値のレベルのネストは，

(葉'値集合)＊…＊(葉'値集合)

とできる．

[例2] 名前と趣味からなるレコード

名前''金枝上"＊趣味'{"スキー","テニス","読書"}，
名前''河村"＊趣味'{"音楽鑑賞","スキー"}

この行ネスト操作はある代数構造(下記参照)の下での部分因数旗開に対応し，表現の一意性は保証されないので(意味制約によって属性間に順序づけができれば，つまり非正規レコードの正規形がつねに求められれば可能)，このネストの意味論としては，行(アン)ネスト演算によるDBの変換によって意味を変えないことにしている:

レコード ＝ レコード＋…＋レコード，

たとえば例2の2番目のレコードは

名前''河村"＊趣味''音楽鑑賞"，
名前''河村"＊趣味''スキー"

と複数のレコードで表現することができる．CRLのNAVの意味論はCIL(向井 87)での部分タグ木(PTT) の集合としている．つまり1つのレコードもDB全体もPTTの集合として同じ意味をもっている．

[例3] 列ネストと行ネストをもったレコード

学生'(名前'(姓''斉藤"＊名'ω)＊所属''テニス同好会"
＊趣味'{"野球","音楽"}
＊出身校'(学校名''三田"＊
所在地'(県名''東京"＊市名''港")))．

このように列ネストと行ネストを組合わせることによって非正規レコードを表現できる．"ω"は値が不明なことを表

Knowledge Base Management System Kappa
- Nested Relational Model and Deductive Database
Kazumasa YOKOTA
Institute for New Generation Computer Technology

わしている．

しかし通常の意味での集合では，行（アン）ネスト演算を充分に反映させることはむずかしい（つまり通常の集合演算以上の操作が必要とされる），そこで記法上は集合であるがそれはsyntax sugarであり，その要素間は以下の代数構造の'＋'を満たしている：

$$a1^{\frown}(a2^{\frown}t)=(a1^{\frown}a2)^{\frown}t,$$
$$a0^{\frown}(a1^{\frown}t1*\cdots*an^{\frown}tn)=a0^{\frown}(a1^{\frown}t1)*\cdots*a0^{\frown}(an^{\frown}tn),$$
$$a0^{\frown}(a1^{\frown}t1+\cdots+an^{\frown}tn)=a0^{\frown}(a1^{\frown}t1)+\cdots+a0^{\frown}(an^{\frown}tn),$$
$$t1*t2=t2*t1,$$
$$t1+t2=t2+t1,$$
$$t1*(t2*t3)=(t1*t2)*t3,$$
$$t1+(t2+t3)=(t1+t2)+t3,$$
$$t+t=t,$$
$$t1*(t2+t3)=(t1*t2)+(t1*t3).$$

この代数の下での行ネストに注意する必要がある．つまり

a^X*b^Y*c^{c1,c2},
a^X*b^Y*c^{c2,c3}

の行ネストは，AIM-Pのような集合の構成

a^X*b^Y*c^{{c1,c2},{c2,c3}}

ではなく，集合の合併に対応する

a^X*b^Y*c^{c1,c2,c3}

であり，葉には値集合の部分集合がつねに取られる．

4. NAVの単一化とプログラム

次に単一化のためにNAVの標準形とNAV間の両立性の概念を導入する．そのために'＋'を含まないNAVをSAV (simple AV-pair)いうことにする．任意のNAVはSAVの和に変換可能で，それをNAVの標準形と呼ぶ．例2の

名前^"柯村"*趣味^{"音楽鑑賞","スキー"}=
　名前^"柯村"*趣味^"音楽鑑賞"
　＋名前^"柯村"*趣味^"スキー"

の右辺は標準形の例である．

SAV が両立するとは，その属性木が重ね合わせ可能なことである（たとえばa^bとc^b，あるいはa^bとa^c^bは非両立）．NAVの標準形を構成するSAVはすべて両立するので，NAVの両立性も定義できる．両立可能なNAVの和(集合)をテーブル，テーブルの集合をDBとして定義する．この両立性は，１つのテーブル内の各レコードの不明情報も規定している(つまり明示されていない葉はunknown value＝ω＝特殊な集合をもつ)．テーブルに共通なこの属性の木領域がスキーマである．

NAV を構成する変数の領域は値集合のベキ集合であり，この制限の下でのNAV の単一化を考える．NAV1とNAV2の単一化は，NAV1＝S11＋…＋S1n，NAV2＝S21＋…＋S2mとすると，S11＋…＋S1n＝S21＋…＋S2mの解を求めることである．すべてのSij は両立するので，その属性木を構成する葉をl1,…,lp,Sij＝l1^Vij1*…*lp^Vijpとすると，単一化はA+C+Iの下で方程式

$$lk:\ \sum_{i=1}^{n} V1ik = \sum_{j=1}^{m} V2jk$$

($1\leq k\leq p$) の解を求めることに帰着できる．この条件でのmguは一意的ではないが，現在CRL ではNAVの葉に変数集合あるいは変数と値の集合はないので，ΣVijkに変数x が含まれていればΣVijk＝xで，mguの一意的は保証される．

次にこの非正規レコード上の演繹機能を実現するために，Prolog同様にプログラムを定義する．このプログラムは，関係完備に対応した操作を含ませるためにstratifiedのクラスの否定をもっている．プログラム節は，NAV t と制約の集合C＝{C1,…,Cm}とリテラル(NAV tまたは¬t) の有限集合B＝{t1,…,tn}の3つ組(t,C,B) として定義する．ここでの制約は，上記の代数構造に関する条件を表わしている．プログラム節の集合がプログラムである．プログラミング言語としての真書的意味論はPTT の集合の集合，手続き的意味論はSLDNF である．しかしDBへの質問の評価機構としては，別の評価アルゴリズムとなる．このプログラム節は，DB の記述のほか，制約規則，生成規則，導出規則のためにある．プログラミング言語としてのCRL ，質問評価機構としてのCRL の詳細については現在さらに検討中である．

5. CRLの例

この節では，上で定義したCRLの例を示す．

〔例4〕親子関係のレコード

親^{"孝","桜"}*子供^{"明","薗","優"},
親^{"明","薫"}*子供^{"純"},
親^{"徹","薗"}*子供^φ,
子供^{"孝"},
親^{"勇","夏"}*子供^{"桜","博"}

３番目のレコードは子供がいないことを表わしている．そして４番目のレコードは親がわからないことを表わしており，親^ω*子供^{"孝"}と同じ意味である．このように空値を，"存在しない"と"不明"の２種類で表現する．φは空集合ではなくωと同じく特殊な集合として扱っている．

〔例5〕親子関係への質問

通常のPrologと同じく，例4の各レコードを単位節(ファクト)と考え，質問をゴールとして与える．

・"明"の親を求める：　←親^X*子供^"明".
・"明"と"薗"の共通の親を求める：
　←親^X*子供^"明",親^X*子供^"薗".
　(DBは一意的な表現ではないので
　←Y⊃{"明","薗"}|親^X*子供^Yでないことに注意)
・"明"または"桜"の親を求める：
　←親^X*子供^{"明","桜"}.
・子供のいない親を求める：　←親^X*子供^{}.
・親の不明の子供を求める：　←親^ω*子供^Y.

答えはそれぞれ，X＝{"孝","桜"}，X＝{"孝","桜"}，X＝{"孝","桜"}かつ{"勇","夏"}，X＝{"徹","薗"}，Y＝{"孝"}が返される．この質問の特徴は，集合値のほかに空値で質問することができること，また結果が集合として返ってくることである．集合値といっても全解ではない．

〔例6〕再帰型の質問

Prolog同様に，先祖関係はレコードを使ったプログラムとして書くことができる：

　先祖^X*子孫^Y ← 親^X*子供^Y.
　先祖^X*子孫^Y ← 親^X*子供^Z, 先祖^Z*子孫^Y.

すると"純"の先祖を求める質問は次のようになる：

　←先祖^X*子孫^"純".

答えはXに，{"明","薫"}，{"孝","桜"}，{"勇","夏"}が返される．このようにレコード構造に基づくプログラムを自由に書くことによって複雑な質問を処理することができる．再帰型質問の評価についてはまだ詳細に検討していないが，関係モデルに基づく演繹DBでのコンパイル／最適化技法がそのまま流用できると予想している．

参考文献

〔横田 86〕横田，金枝上，"PSI上の知識ベース管理ソフトウェアKappa"，第4回第5世代コンピュータに関するシンポジウム，27-28，May，1986．

〔横田 87〕横田，"知識ベース管理基本ソフトウェアKappa"，第5回第5世代コンピュータに関するシンポジウム，Jun.,9-10，1987．

〔横田 87a〕横田，"非正規形モデルへの演繹的アプローチ"，情報処理学会DB研究会，87-DB-58，Mar.,16，1987．

〔向井 87〕K.Mukai,"Anadic Tuples in Prolog"，ICOT-TR，1987．