論　文

Technical Paper

演繹的網型データベースシステム

Deductive Network Database Systems

滝　沢　　誠*
Makoto TAKIZAWA

*　東京電機大学理工学部経営工学科
Dept. of Management and Systems Eng., Faculty of Science and Eng., Tokyo Denki University, Saitama 350-03, Japan.

1986年8月5日　受理

Keywords: deductive database systems, SLD resolution, Prolog, distributed database systems.

Summary

This paper presents the design and implementation of deductive database systems for conventional network database systems. Although almost all of the researchers have been trying to combine the relational model and the logic programming, we try to augment the conventional network database system with inference facilities of resolutions. In this paper, we define a formal system based on the first-order theory for the conventional network model. Then, we show an improvement of the SLD resolution by taking advantage of the network model. In our refutation procedure, meaningless backtracking for finding a refutation and redundant refutations for finding all answer substitutions are prevented. Also, in our procedure, for a given goals, procedures which can obtain a set of answer substitutions are generated by accessing the conventional network database. The procedures are written in an object-oriented manner.

1. まえがき

今後の情報システムでは，通信網で結合された種々の計算機サービスを，利用者はワークステーションを通してアクセスすることにより利用できる．こうした情報システムでは，データベースシステム(DBS)が最重要資源であり，一般利用者が，種々のDBSをこれらの異種性と分散性を意識せずに，容易に利用できる必要がある．ここでは，DBSはデータモデルを提供するシステムとする．データモデルはデータ構造とこれに対する操作演算の対で定まり，データモデルには，関係型[7]と網型[5][6]の二種の型がある．DBSの異種性をDBSのデータモデルの型の相違とする．各DBSに共通な関係型モデルを提供するインタフェースを設けることによりDBSの異種性を解決する試みは文献(9),(18)～(24)により既に行われている．

DBSを容易に利用するには，応用の必要とする視野が柔軟に定義できる必要がある．現在，視野はCOBOLなどの親言語により，DBSの応用プログラムとして実現されている．SQL[8]などによる従来の視野に対して，Prolog[14]などの論理型言語では視野を再帰的に定義できるとともに，応用の必要とする種々の手続きもデータと同じ形式で一様に書ける．本論文では，異種DBS上にPrologの視野を提供することにより，これらの異種DBSを利用者に一様に操作させることを試みる．

データベースを事実の集合，視野を規則として，新しい事実を推論規則により演繹するシステムは演繹的DBSである．関係型モデルと演繹的DBSの関連は，既に文献(13),(16)などの多くの研究者により研究されてきている．本論文では，商用DBSとして広く利用されている網型DBSに着目し，この上にPrologの視野を提供するインタフェースの設計と実現について論じる．また，目標節を網型データ構造に適したレコード単位の巡航的アクセス手続きに翻訳する方式を示す．本方式では，無意味な後戻りを行わずに反駁を求め，冗長な反駁を行わずに全答代入が求められる．

本論文では，2章で，従来の網型モデルの本質的部分を抽象化した順序網型データ構造と，これに対する

アクセス関数を定義する．3章では，網型モデルの第一階理論としての網型体系を定義する．4章では，論理式をホーン節に限定した演繹的網型データベースについて述べる．5章と6章では，目標節から網型データベースのアクセス関数の手続きへの翻訳方式を述べる．

2. 網型モデル

まず，従来の網型モデル[5][6][17]の論理的な構造を抽象化して定式化を行う．

2・1 網型データ構造

網型データ構造Nは，実体と実体間の関数関係の二種の要素から構成される．Nは，時間不変な論理構造を与える網型スキーマNと，これに実際にデータを与えた時間可変な網型データベースNの対である．Nは実体(E)型と関数(F)型の二種から成る．E型Eは $E=(@E, A_1, \ldots, A_m)$ $(m \geq 0)$ であり，@Eは識別子，A_i はデータ属性である．@Eと A_i をあわせて属性とする．属性Tに対して，dom(T)をTの定義域とする．Eに実際にデータを与えた実体(E)集合は，$E \subseteq dom(@E) \times dom(A_1) \times \ldots \times dom(A_m)$ である．E内の各要素eをE組といい，Eの属性Tに対して，T(e)はeのT値を示す．また，どの組も互いに異なった識別子を持つ．

E型 E_1 と E_2 間に関数の関係が存在するとき，関数(F)型 $F=(E_1, E_2)$ が定義できる．E_1 をFの定義域，E_2 を値域という．F型Fに実際のデータを与えた関数(F)集合Fは $F \subseteq E_1 \times E_2$ であり，関数：$E_1 \rightarrow E_2$ を示す．Fの要素をF組とする．

網型スキーマにNに対するE集合とF集合の族を，網型データベースNとする．NはE型を節点，F型を定義域を示す節点から値域の節点への有向辺とする有向グラフとして示せる．Fig. 1 に部(Dept)，社員(Emp)，プロジェクト(Proj)の網型スキーマを示す．たとえば，DEは，定義域をE型Emp，値域をDeptとするF型である．

2・2 順序網型データ構造とアクセス関数

レコード単位のデータ操作を行う従来のDML[17]の意味は明確でないので，これを明らかにする．網型データ構造Nに以下のように順序関係を導入することにより，順序網型データ構造Oを構成する．まず，E集合Eをある属性Tにより全順序付けたものを順序E(OE)集合E/Tとする．ここで，任意のe, e′∈Eに対して，T(e)≤T(e′)のとき，e′はeよりE/Tで大きい(e≤e′)とする．e<e′で，e<e″<e′なるe″がE/Tに存在しないとき，e′はeの直後，eはe′の直前であるという．E/Tに対して，関数 topE/T : → E, lastE/T : → E, nextE/T : E→ E, priorE/T : E→E を定義する．topE/T ( ) と lastE/T ( ) は，E/Tで最小(T値が最小)と最大(T値が最大)の順位E組を与える．各e∈Eに対して，nextE/T (e) と priorE/T (e) は，E/Tでeの直後と直前の順位のE組を与える．E/@Eは，従来のレコード型であり，@Eはdb鍵[5][6]である．

Eと属性Tに対して，v∈ dom (T) ごとに集合 {e∈E|T (e) =v} の@Eによる全順序集合をE[v]とし，その族をE%Tとする．ここで，任意のe, e′ ∈E[v]に対して，@E(e)≤@E(e′)なるとき，e′はeよりE%Tで大きい(e≤e′)とする．e<e′でe<e″<e′なるe″がE%Tに存在しないとき，e′はeの直後，eはe′の直前であるという．E%Tに対して，関数 topE%T : dom (T)→ E, lastE%T : dom (T)→ E, nextE%T : E→E, priorE%T : E→Eを定義する．各v∈dom (T)に対して，topE%T (v) と lastE%T (v)は，E[v]内で@Eが最小と最大順位のE組を与える．各e∈Eに対して，nextE%T (e) と priorE%T (e)は，E[v]内でeの直後と直前のE組を与える．あるTについてE%Tが定義されるとき，Tを直接属性とする．Tは従来のCALC項目[5][6]である．また，E/ @E=E%@Eである．

F型F=(E, G)のF集合Fと Eの属性Tを考える．G内の各E組gに対して，{<e, g>∈F} をT(e)の順に全順序付けた集合をF/T[g]とする．各f =

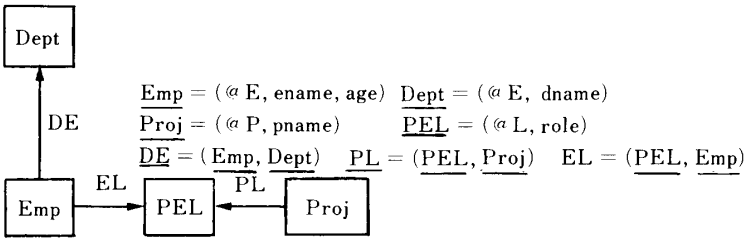


Fig. 1 Network Sckema.

$<e, g>, f' = <e', g> \in F$に対して，$T(e) \leq T(e')$のとき，$f'$はfよりF/Tで大きい$(f \leq f')$とする．直前，直後の関係もE%Tと同様に定義する．F/T [g]の族を順序F (OF) 集合F/Tとする．F/Tに対して，関数funcF/T : E→ G, ftopF/T : G→ E, flastF/T : G→ E, fnextF/T : E→ E, fpriorF/T : E→Eを定義する．各$e \in E$に対して，funcF/T (e) は，$<e, g> \in F$なるgを与える．各$g \in G$に対して，ftopF/T (g) とflastF/T (g) は，F/T [g] 内の最小と最大順位のE組を与える．各$e \in E$に対して，fnextF/T (e) とfpriorF/T (e) は，F/T [funcF/T (e)]内でeの直後と直前の順位のE組を与える．F/Tは従来の親子型である．

以上定義した関数をアクセス関数とする．アクセス関数はDML [17]に対応している．たとえば，ftopDE/@Eは，COBOL DMLのfind first E within DEである．

3．網型体系

前章で定義した網型データ構造の第一階理論としての網型体系を定義する．

3・1 網型言語と解釈

網型スキーマNに対する網型言語Lは，関数記号を含まず，述語記号として，各E型$E = (@E, A_1, \ldots, A_m)(m \geq 0)$に対する$m+1$項述語記号Eと，各F型$F = (E_1, E_2)$に対する二項述語記号Fと，各属性Tに対する一項述語記号Tを含む第一階言語である．変数を大文字，定数を小文字の英数字で示す．Fig. 1で，Dept (X, a) は述語，Xは変数，aは定数である．&&は論理積，||は論理和を示すとする．

網型言語Lの解釈Iについて，①Iの定義域D内の各個体dに対して，各々異なった定数dが対応し，②D内の個体は網型データベースN内の値だけであると仮定する．①と②は一意名仮定，定義域閉包仮定である．

3・2 公　　理

次に網型体系の特殊公理を定義する．

〔特殊公理〕　$E = (@E, A1, \ldots, Am)$と$G = (@G, B_1, \ldots, B_n)$をE型，$F = (E, G)$をF型，Lの網型解釈Iの定数集合を$C = \{d_1, \ldots, d_q\}$とする．このとき以下の論理式 (wff) は公理である．

(1) 〔完全公理〕 Gを網型体系Tの具象wff集合としたとき，各E型Eに対して，$GE = \{<k, e_1, \ldots, e_m> | E(k, e_1, \ldots, e_m) \in G\}$，各F型Fに対して，$GF = \{<k_1, k_2> | F(k_1, k_2) \in C-G\}$を各々EとFのTでの外延とする．ここで，$GE = \{e_j | e_j = <k_j, e_{1j}, \ldots, e_{mj}>, j=1, \ldots, r\}$, $GF = \{f_i | f_i = <k_{1i}, k_{2i}>, i=1, \ldots, p\}$とする．各E型Eと各F型Fに対して，

$\forall K \forall X_1 \ldots \forall X_m (E(K, X_1, \ldots, X_m) \rightarrow (K=k_1 \,\&\&\, X_1=e_{11} \,\&\&\, \ldots \,\&\&\, X_m=e_{m1}) \| \ldots \| (K=k_r \,\&\&\, X_1=e_{1r} \,\&\&\, \ldots \,\&\&\, X_m=e_{mr}))$.

$\forall X \forall Y (F(X, Y) \rightarrow (X=k_{11} \,\&\&\, Y=k_{21}) \| \ldots \| (X=k_{1p} \,\&\&\, Y=k_{2p}))$.

(2) 〔一意名公理〕 $(d_i \neq d_j)$ (ただし，$i \neq j$).

(3) 〔定義域閉包公理〕 $\forall X (X=d_1 \| \ldots \| X=d_q)$.

(4) 〔網型公理〕 各E型Eと各F型Fに対して，

$\forall K \forall X_1 \ldots \forall X_m (E(K, X_1, \ldots, X_m) \rightarrow @E(K) \,\&\&\, A_1(X_1) \,\&\&\, \ldots \,\&\&\, A_m(X_m))$.

$\forall K \forall X_1 \ldots \forall X_m \forall Y_1 \ldots \forall Y_m (E(K, X_1, \ldots, X_m) \,\&\&\, E(K, Y_1, \ldots, Y_m) \rightarrow X_1=Y_1 \,\&\&\, \ldots \,\&\&\, X_m=Y_m)$.

$\forall K_1 \forall K_2 (F(K_1, K_2) \rightarrow @E(K_1) \,\&\&\, @E_2(K_2))$.

$\forall K_1 \forall K_2 \exists X_1 \ldots \exists X_m \exists Y_1 \ldots \exists Y_n (F(K_1, K_2) \rightarrow E_1(K_1, X_1, \ldots, X_m) \,\&\&\, E_2(K_2, Y_1, \ldots, Y_n))$.

$\forall K_1 \forall K_2 \forall K_3 (F(K_1, K_2) \,\&\&\, F(K_1, K_3) \rightarrow K_2=K_3)$.□

更に，閉世界仮定 (CWA) を設ける．分離規則と一般化規則の推論規則と以上の公理集合からなる第一階理論 [10]を網型体系とする．網型体系は第一階理論であることから，完全で健全である．

4．演繹的網型データベース

網型体系のwffをホーン節の形式に制限した演繹的網型データベース (DNB) を定義する．DNBは，事実ベース (FB) と規則ベース (RB) から成る．FBは事実公理 (3・2節) の基底単位節の集合である．Fig. 2にFig. 1の網型データベースを示し，Fig. 3にそのFB示す．

規則ベース (RB) は，本体が空でないホーン節である規則節の集合である．たとえば，次の節は規則節である．

41) Projemp (X, Y) : −Proj (K, X), PL (L, K), PEL (L, Z), EL (L, M), Emp (M, Y, T).

```
1)  Dept (d1, dev).      2)  Dept (d2, cmp).
3)  Emp (e1, a, 30).     4)  Emp (e2, b, 28).     5)  Emp (e3, c, 31).
6)  Emp (e4, d, 38).     7)  Emp (e5, e, 26).     8)  Emp (e6, f, 35).
9)  PEL (11, 1d).        10) PEL (12, mb).        11) PEL (13, mb).
12) PEL (14, 1d).        13) PEL (15, mb).        14) PEL (16, mb).
15) PEL (17, mb).        16) PEL (18, mb).
17) Proj (p1, db).       18) Proj (p2, cn).
19) DE (e1, d1).         20) DE (e2, d1).         21) DE (e3, d1).
22) DE (e4, d1).         23) DE (e5, d1).         24) DE (e6, d1).
25) EL (11, e1).         26) EL (12, e1).         27) EL (13, e2).
28) EL (14, e3).         29) EL (15, e4).         30) EL (16, e5).
31) EL (17, e5).         32) EL (18, e6).
33) PL (11, p1).         34) PL (12, p2).         35) PL (13, p1).
36) PL (14, p1).         37) PL (15, p2).         38) PL (16, p1).
39) PL (17, p2).         40) PL (18, p2).
```

Fig. 2 Fact base.

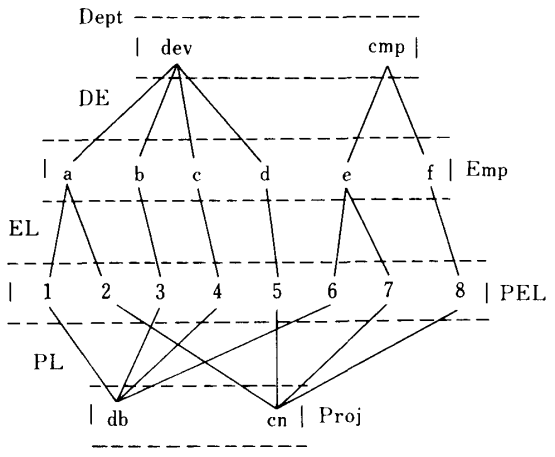


Fig. 3 Network database.

規則節の左辺は，FB内の述語記号を含まないとする．規則節は網型データベースの視野を与える．関係型モデル[7]と異なり，DNBでは，視野は再帰的に定義できる．たとえば，「プロジェクトXと推移的に関係のある社員Y」を示す視野Pempは，次のように書ける．

42) Pemp (X, Y) : −Projemp (X, Y).

43) Pemp (X, Y) : −Projemp (X, Z), Pemp (Z, Y).

ここで，41)−43)の集合がRBの例である．RB内の規則節の左辺の述語記号を視野述語記号とする．

質問「プロジェクトdbのメンバYを求めよ」は目標節：−Projemp (a, ?Y)で，Yを目標変数という．

5. 反駁手続き

次に，演繹的網型データベース(DNB)と目標節の導出について考える．

5・1 Prolog反駁手続き

演繹的網型データベース(DNB) Dと目標節Gの推論規則としてSLD導出[14]があり，健全で完全であることが知られている[1][15]．$:-A_1, \ldots, A_m$を目標節H, $A :-B_1, \ldots, B_n$をD内の節Cとする．$A_i\theta = A\theta$なる代入θが存在するとき，$:-(A_1, \ldots, A_{i-1}, B_1, \ldots, B_n, A_{i+1}, \ldots, A_m)\theta$をHとCのSLD導出形，Cを入力節という．$A_i$は選択基本式であり，Hから計算規則Rにより選択される．SLD導出の列をSLD演繹，空節を導き出すSLD演繹をSLD反駁という．FをDU {G}のSLD反駁とし，ここでの代入の合成$\theta_1 \ldots \theta_p$を目標変数に制限したものを答代入という．DU {G}の可能な演繹は，Fig. 4に示すSLD木で示される．節点は目標節，枝は入力節，根は目標節Gを示し，根から葉への路はSLD演繹を示す．葉が空節の路はSLD反駁を示す．葉が空節の節点には，答代入が書いてある．選択基本式には下線が引いてある．葉がXである路は失敗演繹を示す．

問題はSLD木から反駁をいかに見付けるかであり，このための手続きを反駁手続きという．多くのPrologシステムでは，目標節の最左の基本式を選択し，入力節を決められた順に選択し，SLD木を縦型に探索する反駁手続きが用いられている．

5・2 SL反駁と順序網型データベースの結合

次に，SLD導出と2章で定義した順序網型データベースOに対するアクセスをいかに結合するかを考える．

最初の問題は，いつOをアクセスするかである．SLD導出で，入力節Cが事実ベース(FB)内の節であるとき，Cが指示するE組またはF組をOから取り出す必要がある．例として，Fig. 3に対する目標節 :−Emp (X, N, A), Dept (Y, ?D), DE (X, Y) のPrologの計算規則PRと他の計算規則Rによる SLD木をFig. 4と5に示す．両図を比較すると，Fig. 5のSLD木のほうが節の数が少ない．入力節の順序をO内の順序に一致させる．たとえば，Emp/@Eでは<dev>,<cmp>と，DE/@Eでは<dev>に対して

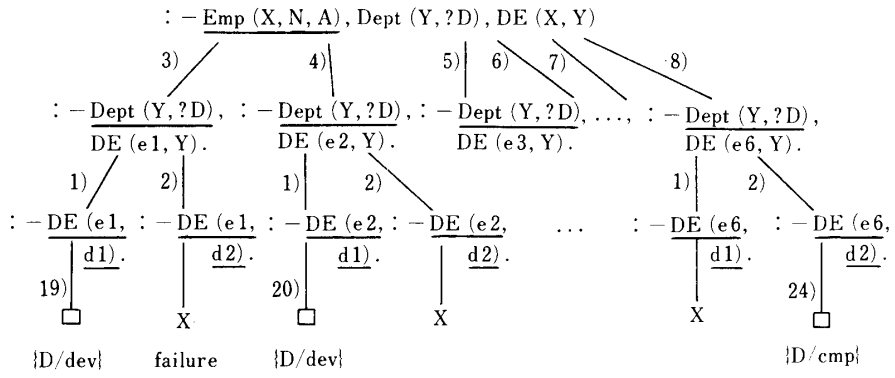


Fig. 4 SLD tree.

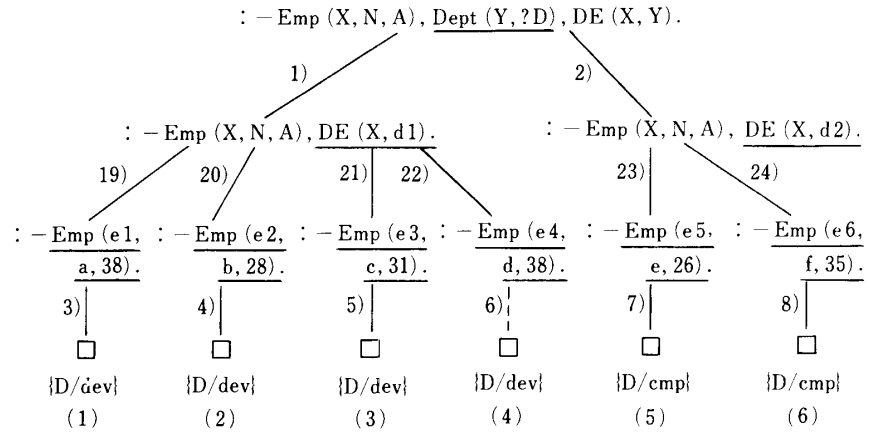


Fig. 5 SLD tree.

< a, 30 >, < b, 28 >, < c, 31 >, < d, 38 >と順序付けられているとする. Fig. 5 の SLD 木に対して, 以下のアクセス関数を用いて, 答代入を求められる.

```
d=topDept/@D ();
while (d≠⊥) {e=ftopDE/@E (d);
while (e≠⊥) {output (dname (d));
      e=fnextDE/@E (e);}
      d=nextDept/@D (d);}
```

この例は, 順序網型データ構造に則して基本式を選択すれば, SLD 木を縮小し, アクセス関数により木内の全答代入を求められることを示している.

第 2 の問題は, SLD 木の縦型探索の後戻りに関してである. 後戻りは, ①導出に失敗したときと, ②反駁に成功しほかの反駁を見付けるときに用いられる. ①について考える. たとえば, FB 内の述語記号 A, B, Cに対して, 目標節 :－A (X, Y), B (X, ?N), C (Y, M) を考える. Prolog 反駁手続きでは, H_2= :－ (B (X, ?N), C(Y, M)) θ_1が, 次にH_3=:－C(Y, M) θ_1 θ_2 が導出される. θ_1はXと Y, θ_2はNに対する基底代入である.

ここで, H_3での導出に失敗すると, H_2に後戻りし, 選択基本式 B (X, ?N) θ_1に対する次の入力節を FB の中から探す. しかし, この手続きは, H_2の SLD 導出による代入θ_2はYの代入を変化させないので, 反駁を見付けるという点からは意味がない. したがって, B (X, ?N) θ_1についてほかの代入を求めても, :－ C (Y, M) $\theta_1\theta_2$の SLD 導出は失敗する. ここで, H_3から H_2を飛び越して H_1に後戻りするなら, ほかの代入が求まる可能性がある.

以上より, 導出Dに失敗したら, 目標節からDまでの導出Eの中で, 選択基本式が共有変数を持つ一番Dに近いEに後戻りするなら, 無意味な導出を防げることがわかる.

次に, Fig. 5 を考えてみる. (1)で反駁が求まったとき, (2)－(4)は(1)と同じ答代入 {D/dev} を持っているので, 後戻りにより(2)－(4)を求めることは冗長である. 他の答代入 {D/cmp} を求めるには, (1)から直接, 目標変数を持つ節点, この場合は根に後戻りすればよい. 以上より反駁が求まったとき, 目標変数を持つ節点の中で一番葉に近い節点に直接後戻りすることにより冗長な反駁を防止できる.

以上の考察をもとに, 目標節から順序網型データベースOを操作するアクセス手続きの生成方法について考える.

6. アクセス手続きの生成

本章では，与えられた目標節から順序網型データベースOをアクセス関数で操作して，答代入集合を求める手続きの生成について考える．

6・1 基本定義

演繹的網型データベースD，目標節G，計算規則RによるSLD演繹の集合をSPとする．まず，SLD演繹間の類似関係を定義する．

［定　義］　SP内の演繹FとF′が次の条件を充足するとき，FとF′は類似するという．

① FとF′は同じ長さである．

② FとF′内の各i番目の導出D_iとD'_i (Fig. 6)に対して，

㋐ $i=0$のとき，$\theta_0=\theta'_0=\varepsilon$(恒等代入)．

㋑ $i>0$のとき，$k=k'$，$j=j'$ (i.e. $A_j=A'_{j'}$)，$m=m'$．
$\theta_{i+1}=\theta_i\sigma_i$．$\theta'_{i+1}=\theta'_i\sigma'_i$．

σ_iをD_iの代入，θ_iをD_iまでの代入合成とする．

$k=k'=0$ならば，AとA′は同一述語記号の基本式である．

$k=k'>0$ならば，入力節C_iとC'_iは同じ規則節である．□

［定　義］　演繹Fのiとj番目の導出をD_i(Fig. 6)とD_j($j<i$)とし，各々の選択基本式をA_iとA_jとする．A_iの変数XとA_jの変数Yに対して，$X\theta_i=Y\theta_i$で，$Y\sigma_i$が具象化しているとき，A_iとA_jは変数Xを共有するという．また，A_jをXの具象点とする．□

類似したFとF′では，各導出で選択基本式が同じで，入力節が規則節ならば同じ入力節を持ち，事実節ならば互いに同じ述語の例である．Fig. 4内の全演繹は類似している．類似関係は明らかに同値関係であるので，SPは同値類SP_1, SP_2, ...に分割できる．各SP_iを類似類という．各類似類SP_iの全答代入を，順序網型データベースをアクセス関数により操作して求める手続きを生成する．このとき，次の2つの手続きが必要になる．

① SP_i内のある反駁Fを見付ける手続き．

② ①のFに対して，SP_i内の全反駁を見付ける手続き．

①では，いかに失敗する演繹を防ぐかが問題になる．ここで，無意味な後戻りを定義する．

［定　義］　SLD木Tで，AをBから根までの路内の節点とし，AとBでの選択基本式を各々PとQとする．このとき，PとQが共通変数Xを持ち，XはAで具象化されるとき，BからAへの後戻りは意味があるという．そうでなく，かつAとB間に意味のある後戻り先の節点がないとき，無意味であるという．Bからの後戻りが意味のある節点の中で，一番Bに近い節点を意味のある後戻り節点とする．□

［命　題1］　類似類SP_iの演繹F_1のi番目の導出D_iが失敗したとき，j番目($j<i$)の導出への無意味な後戻りにより見付けられる演繹F_2でもi番目の導出D'_iは失敗する．

［証　明］　F_1とF_2の$j-1$番目の代入合成は同じθ_{j-1}である．jから$i-1$までの導出の代入は，i番目の導出の選択基本式内の変数を含まないので，D'_iも失敗する．□

導出に失敗したときの無意味な後戻りを防止できれば，反駁手続きの効率を向上できる．

次に，②の類似類SP_i内の答代入を有効に求める問題を考える．SP_i内の冗長な反駁を定義する．

［定　義］　SP_i内の反駁FとF′の代入の合成を各々θとθ'とする．目標変数に対する代入の合成を各々θ_1とθ_1'とし，$\theta=\theta_1\,\theta_2$と$\theta'=\theta_1'\,\theta_2'$とする．ここで，$\theta_1=\theta_1'$であるとき，$\theta'$と$\theta$は互いに冗長であるという．□

冗長な反駁FとF′は同じ答代入を持つことから，Fが求まったら冗長な反駁を避けることが望ましい．

本論文では，無意味な後戻りを行わず，冗長な反駁をなるべく行わず，類似類内の全答代入を求めるアクセス手続きを合成する．

6・2 巡　航　木

各類似類SP_iの代表元をいかに見付けるかを考え

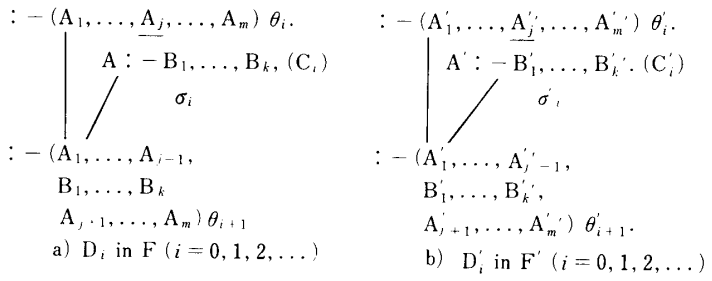


Fig. 6　Similar deductions.

る．まず，目標節に対して目標グラフを定義する．

［定　義］　目標節$G = :-A_1,\ldots,A_m$に対して，次の手続きで形成されたグラフをGの目標グラフTGとする．

① 各基本式A_iに対して，節点A_iを設ける．

② 基本式A_iとA_jが共通変数Xを持ち，両者が共にF型でなければ，節点A_iとA_jの間に辺Xを設ける．□

目標グラフは，選択基本式間の変数の共有関係を示している．たとえば，:−PEL (X, S), EL (X, Y), Emp (Y, ?N, A), PL (X, Z), Proj (Z, P) の目標グラフは Fig. 7 となる．ここで，F集合を示す基本式 EL (X, Y) と PL (X, Z) の間に辺は存在しない．

目標グラフ TG から，基本式の選択順と基本式間の変数の共有関係を示す巡航木を定義する．

〈巡航木生成手続き〉　目標グラフ TG から，以下の手続きにより構成された木を巡航木Tとする．

① TGの節点と辺を無印とし，コスト関数 cost (X) が最小なE型の節点Xを選ぶ．Xを木Tの根とする．XとX内の無印な全変数を印付け，印付けられた変数Vに対して，XはVの具象点である．この印付け操作を mark (X) と書く．

② TG 内で，Xと無印点Yを結合する無印辺Vの具象点がYでない辺から，cost (Y) が最小な辺を選ぶ．mark (Y)．Yの隣接辺のなかで両端が無印の辺を印する．YがE型またはF型ならば，YをXの最後の子としてTに加える．YをXとして，②へ．Yが規則ならば，規則節Cを規則ベース (RB) から選び，TG からYを除き，Cを加え，②へ．

③ Xが根で，TG 内の全節点が有印のときは終了．TG 内に無印点があれば，無印点Yの中で，cost (Y) が最小のYを選び，mark (Y)．YをXの最後の子としてTに加える．YをXとして，②へ．Xが根で，②の条件の隣接辺がないとき，Xの親をXとして，②へ．□

cost (X) は，基本式Xと単一化可能な入力節を見付けるコスト関数である．コストの単位は，順序網型データベースOに対するアクセス関数の適用回数とする．cost (X) は，Xの種類により次のように与えられる．

① XがE型の基本式$E(K, A_1,\ldots,A_m)$で，K, A_iは変数または定数であるとき，

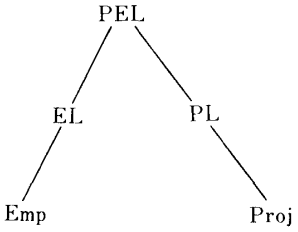


Fig. 8　Navigational tree.

㋐ Kまたは直接属性A_iが定数または有印変数のとき，アクセス関数 dtopE/@Eまたは dtopE/A_iでアクセスできるので，cost (X) = | E | * s (A_i) である．| E | はEの基数，s (A_i) はA_iの選択度[12]である．

㋑ そうでないとき，E/@Eを先頭から逐次アクセスしなければならないので，cost (X) =E/2である．

② XがF型の基本式 F (A, B) のとき，cost (X) はAが有印ならばs，Bが有印ならばc．ここで，cはFの値域の各E組に対応する定義域のE組の平均数で，Fの結合度である．sは，定義域の各E組がFにより関連した値域のE組を持つ確率で，Fの逆結合度である[19]．

③ Xが規則のとき，cost (X) は定数Mを与える．Mの値を大きくすることにより，規則基本式が選択されることを遅延できる．□

以上の巡航木Tは，次の性質を持つ巡航木である．

① Gの節点とTの節点は1対1対応する．

② Tの各節点の子は，左から右に順序付けられている．

③ G内で節点XとY間に辺があれば，T内でXとYは同一の根から葉への路内にある．

Fig. 8 に Fig. 7 の巡航木の例を示す．巡航木Tの節点の縦型探索順は，基本式の選択順を与え，子の親は，子の意味のある後戻り節点である．Tの縦型探索により，節点AがBより先にみつかるとき，AがBより順位が高いといい，A > Bと書く．Fig. 8 が示す SLD 演繹を Fig. 9 に示す．

目標変数の具象点である節点を目標点とする．Fig. 8 で，Emp (Y, ?N, A) は目標点であり，PEL (X, S) は変数XとSの具象点である．

6・3　アクセス手続き

アクセス手続きPはアクセス関数の系列である．巡

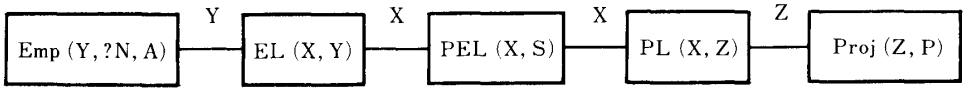


Fig. 7　Goal Graph.

```
:- (PEL(X, S), EL (X, Y), Emp (Y, ?N, A), PL (X, Z), Proj (Z, P))θ0
   |  PEL (_, _).
   /
:- (EL (X, Y), Emp (Y, ?N, A), PL (X, Z), Proj (Z, P)) θ1
   |  EL (_, _).
   /
:- (Emp (Y, ?N, A), PL (X, Z), Proj (Z, P)) θ2
   |  Emp (_, _, _) .
   /
:- (PL (X, Z), Proj (Z, P)) θ3
   |  PL (_, _).
   /
:- Proj (Z, P) θ4
   |  Proj (_, _).
   /
□   θ5
```

Fig. 9 Refutation of Fig. 8.

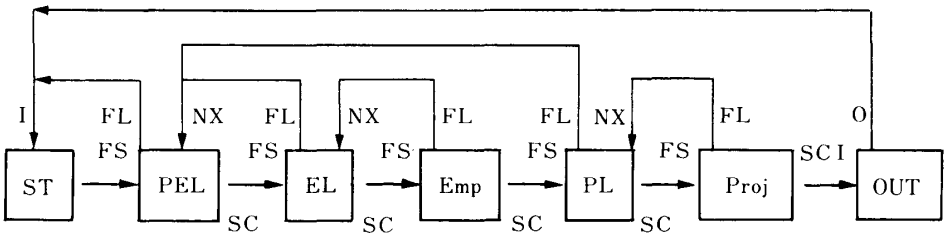


Fig. 10 Simple access procedure.

航木TからPを生成する問題を考える.

〔1〕 **単純アクセス手続き**

アクセス手続きは，通信路で結合されたセルの集合である．セルXは，データの順序集合D(X)とアクセス関数から構成される．通信路は，セルXの出力ポートX：OからYの入力ポートY：Iの間に存在でき，X：O→Y：Iと書く．通信の単位をトークンという．各セルXは入力ポートX：FS (FIRST) とX：NX (NEXT)，出力ポートX：SC (SUCCESS) とX：FL (FAILURE) を持つ．セルXはT内の節点Xに対応している．Pは開始 (ST) と出力 (OUT) という特別なセルを持つ．両セルは出力Oと入力Iポートを持つ．巡航木Tと節点Xに対して以下の記法を導入する．

parent (X) = Xの親．root (T) =Tの根．

next (X) = T内でXの縦型順の次の順序の節点．

last (T) = T内で縦型順の最後の順位の節点．

Tから，ある答代入を見付ける単純アクセス手続きSPを構成する．SPではセルは次のように結合される．

①ST：O → root (T)：FS, root (T)：FL → ST：I. ②X：SC → next (X)：FS. ③X：FL → parent (X)：NX. ④ last (T)：SC → OUT：I. ⑤ OUT：O → ST：I.

各セルXのD(X)は，順序網型データベースO内のある順序集合（たとえば，Emp/@E）である．セルXの動作について述べる．まず，セルSTは，ST：Oより恒等代入εをトークンとして送出する．Xは，X：FSより$<\theta>$を受けるとD(X)の最小順位の組d，たとえば，D(X)がEmp/@Eならばd= topEmp/@E()と基本式Xθで単一化を行う．d=X$\theta\theta'$なる組dが見付かるまで，D(X)を順々に調べ，見付かればX：SCより$<\theta\theta'>$を送信する．見付からなければ，X：FLより$<\theta>$を送信する．X：NXより$<\theta>$を受信すれば，D(X)の最後にアクセスされた組の次順位の組から以上を繰り返す．OUTは，OUT：Iで$<\theta>$を受信すれば，θより答代入を利用者に出力し，$<\theta>$をOUT：Oより送信する．STは，ST：Iよりトークンを受信すれば全セルを停止させる．Fig. 10にFig. 8の単純アクセス手続きを示す．

単純アクセス手続きの構成方法より以下が成り立つことは明らかである．

［**命　題2**］　単純アクセス手続きSPは，無意味な後戻りを行わずに反駁を見付けられる．□

〔2〕 **一般アクセス手続き**

巡航木Tの全答代入を冗長反駁を行わずに求めることを試みる．T内の各節点Xに対し次の記号を導入する．

tail (X) = Xを根とする部分木の最右の葉．

rtarg (X) = X > Yで，X > Z > Yの目標点Zがない目標点Y．

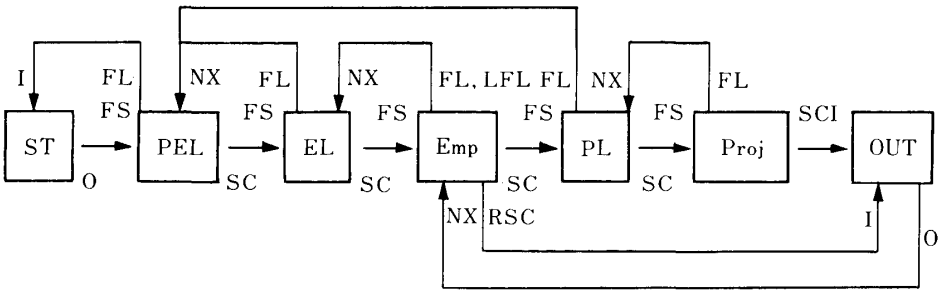


Fig. 11 General access procedure.

ltarg (X) = X < Yで，X < Z < Yの目標点Zがない目標点Y.

rjoin (X) = Xと rtarg (X) の共通の先祖Yで，他のどの先祖Zに対しても，Z > Yなるもの.

tlast (T) = 任意の目標点Xに対し，X > Yなる目標点Y.

単純アクセス手続き SP の各セルXに，X : LFL と X : RSC の2つの出力ポートを加え，新たに次の通信路を設ける.

① OUT : O → tlast (T) : NX (SP から OUT : O → ST : Iを除く).

② X : RSC → rjoin (X) : FS, X : LFL → rtarget (X) : NS.

トークンは組< typ, to, from, 代入 >で typ∈{F (IRST), N (EXT)} で，to と from にはセルの識別子が入る．まず，ST はトークン< F, root (T), last (T), ε >を ST : Oより root (T) に出力する．各セルXが，typ=F のトークンを受けたなら，単純アクセス手続き SP と同様に動作する．OUT が<..., θ >を OUT : Iで受信したとき，答代入を出力し，< N, NULL, NULL, θ >を OUT : Oから送信する．X : FS または X : NX で< N, Z, Y, θ >を受信した場合を考える．Z=Y=NULL なら，from をX, to を tail (X) にして，SP と同様に動作する．Y=Xならば，X : RSC に< N, NULL, NULL, θ' >を送信する．X<Z ならば，from を Z, to を tail (Z) とし，SP の動作をする．Z ≥ X ≥ Yならば，SP の動作を行う.

以上を一般アクセス手続き GP とする．GP では，巡航木が示す類似類内の全ての答代入を，冗長な反駁をなるべく行わずに求められる．Fig. 11 に Fig. 8 の巡航木の一般アクセス手続きを与える.

[命 題 3] 類似類 SP_i に対して，①一般アクセス手続き GP は，反駁を見付けるための無意味な後戻りを行わず，②Prolog 手続きより少ない導出により SP_i 内の全答代入を求められる.

[証 明] ①は命題2より明らかである．Prolog 手続きは，冗長な反駁を行うが，GP は冗長反駁を避けられるので②がいえる．□

7. むすび

本論文では，異種 DBS をホーン節の形式により標準的に扱うことと，従来の DBS に推論機能を付加する問題について議論した．特に，商用に広く利用されている網型 DBS 上に Prolog による視野を与え，SLD 導出と網型 DBS の巡航的アクセスを結合する有効な反駁方法を示した.

この方法では，無意味な後戻りを行わずに反駁を見付け，これと類似した反駁集合の全答代入を冗長な反駁を行わずに求められる手続きが生成される．また，この方法は，従来の計算機の物理的 I/O がレコード単位の巡航的操作であることから，Prolog システムを関係型 DBS の上ではなく，物理的 I/O 層の上に直接実現する方法として利用できる.

現在，M-380 Q の網型 DBMS AIM/DB 上に，反駁を行うシステム LIP (Logic Interface Processor)

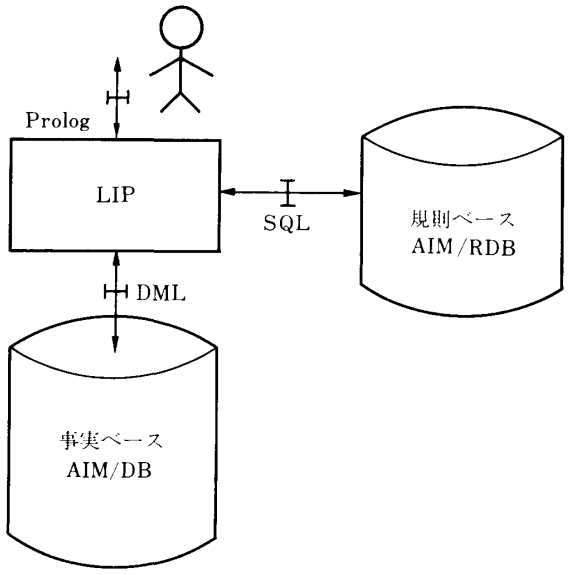


Fig. 12 LIP

の実現をUTI−LISPにより行っている(Fig. 12). 更に，ワークステーション内に，目標のDBSに対するアクセス手続きを生成するシステムも実現中である．

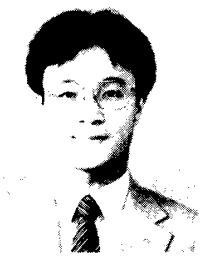
本論文により，既存の異種DBSをPrologにより統合的に操作できることが示された．現在，データベースに対する更新方法，Prologによるトランザクション記述，複数の演繹的DBS間の分散演繹の方法などを検討している．

◇参考文献◇

(1) Apt, R. and van Emden, M. H. : Contributions to the theory of logic programming, JACM, Vol. 29, No. 3, pp. 841-862 (1982).
(2) Cambell, J. A. : Implementations of Prolog, Ellis horwood limited (1982).
(3) Chang, C. L. : On Evaluation of Queries Containing Derived Relations in a Relational Data Base, Login &Database, pp. 235-260, Plenum Press (1981).
(4) Clocksin, W. F. and Mellish, C. S. : Programming in Prolog, Springer-Verlag (1984).
(5) CODASYL DDL Committee : CODASYL Data Description Language, Journal of Development (1973).
(6) CODASYL DDL Committee : CODASYL Data Description Language, Journal of Development (1978).
(7) Codd, E. F. : A relational model of data for large shared data bank, CACM, Vol. 13, No. 6, pp. 337-387 (1970).
(8) Data, C. J. : An Introduction to Database Systems, Addison-Wesley (1981).
(9) Dayal, U., *et al.* : Query optimization for CODASYL database systems, Proc. of the ACM SIGMOD, pp. 138-150 (1982).
(10) Endertom, H. : Mathematical Introduction to Logic, Academic Press (1972).
(11) Henschen, L. J. and Naqvi, S. A. : On compiling queries in recursive first-order databases, JACM, Vol. 31, pp. 47-107 (1984).
(12) Hevner, A. and Yao, S. B. : Query processing on a distributed databases, Proc. of the 3rd Berkeley Workshop, pp. 91−107 (1978).
(13) Kobayashi, I. : Classification and Transformations of Binary Relationship Schemata, Sunno Institute of Business Administration (1985).
(14) Kowalski, R. : Logic for Problem Solving, Research Studies Press (1979).
(15) Lloyd, D. : Foundation of Logic Programming, Springer-Verlag (1985).
(16) Ohsuga, S. : Developing a deductive relational database for uniform handling of complex queries, JIP (IPSJ), pp. 123-137 (1983).
(17) Olle, T. : The CODASYL Approach to Data Base Management, John Wiley&Sons (1978).
(18) Takizawa, M., *et al.* : The Four-schema concept as gross architecture of distributed databases and heterogeneous problems, JIP, Vol. 2, No. 3, pp. 134-142 (1979).
(19) Takizawa, M., *et al.* : Query translation in distributed databases, Proc. of the IFIP, pp. 451-456 (1980).
(20) Takizawa, M. : Distribution problems in distributed databases, JIP, Vol. 5, No. 3, pp. 139-147 (1982).
(21) 滝沢，野口：CODASYLデータベースシステムに対する非手続的更新インタフェースの基本概念，情報処理学会論文誌, Vol. 24, No. 6, pp. 587-866 (1983).
(22) Takizawa, M. : Distributed Database System-JDDBS, JARECT, Ohm-sha and North-holland, Vol. 7, pp. 262-283 (1983).
(23) 滝沢，野口：CODASYL DML上の非手続的グラフ問合せの設計と実現，情報処理学会論文誌, Vol. 25, No. 5, pp. 764-774 (1984).
(24) Takizawa, M., Itoh, H. and Moriya, K. : Logic interface system on CODASYL database system, Proc. of the Logic Programming Conf., pp. 111-117 (1986).
(25) 野口，滝沢：知識工学基礎論，オーム社(1986).
(26) Ullman, J. D. : Implementation of logical query languages for databases, TODS, Vol. 10, No. 3, pp. 289-321 (1985).

〔担当編集委員：大須賀節雄〕

著者紹介



滝沢　誠(正会員)

昭和48年東北大学工学部応用物理学科卒業．昭和50年同大学院工学研究科応用物理学専攻修士課程修了．工学博士．同年(財)日本情報処理開発協会入社．昭和61年東京電機大学理工学部経営工学科講師．主に分散型データベースシステム，通信網の研究に従事．著書「知識工学基礎論」(共著)．情報処理学会，電子情報通信学会，ACM各会員．