Vol. 31 No. 2　　情　報　処　理　　Feb. 1990

解　説

2. 演繹データベースの形式的意味論†

勝　野　裕　文††

1. はじめに

演繹データベースは一般に個々の事実の集合体であるデータベースとデータに関する知識から構成され，その知識を使ってデータベースにはたくわえられていないデータを導き出し，従来のデータベースよりも柔軟な質問応答を行う．そこで，問題となるのは「新しいデータを導き出す」という意味をどう定式化するかという点である．この意味を明らかにしないと，同じ意味を表すはずの二つの演繹データベースが同じ質問に対して異なる答を返すことが考えられる．本解説ではこの意味に対するさまざまなアプローチを紹介し，演繹データベースの意味について説明する．

プログラムの意味については，公理的，手続き的，表示的意味などのさまざまな立場から研究が進められている．演繹データベースについてもその意味を，ある関数の不動点，論理式の集合，あるいはそのモデルで表したり，ある証明手続きと関連させて考えたりしている．また，新しいデータを導き出すのに使う論理式の集合は論理型プログラムのプログラムともみなせるので，演繹データベースの意味に関する議論は論理型プログラムの意味に関する議論とも考えることができる．事実，以下で紹介する内容はデータベースの国際会議と論理型プログラムの国際会議にまたがって活発に議論されている．

以下 **2.** では関係データベースの意味について説明し，**3.** では **2.** の説明をもとにして演繹データベースの意味について説明する．なお，参考文献はオリジナルなものよりはむしろサーベイ的性格をもつ文献を中心にあげておいた．

† Formal Semantics of Deductive Databases by Hirofumi KATSUNO (NTT Basic Research Laboratories).
†† NTT 基礎研究所

2. 関係データベースの意味

関係データベースの意味は通常，データが唯一の世界を表すように決められる．たとえば，**図-1** に示されているグラフを考えよう．このグラフを関係データベースでは，**図-2** の関係で表す．このとき，次の質問を考えてみよう．

［質問 1］ ノード a との間にアークがないノードはどれか．

［質問 2］ すべてのものとの間にアークがある節点はどれか．

質問1の答えは $\{a, c, d\}$ であるが，この答を出すためにはデータベースに格納されていない事実はすべて偽であるという暗黙の仮定と，a, c, d は b と違うものであるという仮定を使っている．前者の仮定を一般に閉世界仮説 (closed world assumption) と呼ぶ．後者の仮定は等号に関するもので単一名仮説 (unique name assumption)[15]と呼ぶ．

質問2の“すべてのもの”というのはどの範囲をさすのだろうか．通常，“すべて”はデータベース内のすべての個体を表す（すなわち今の場合は，a, b, c, d

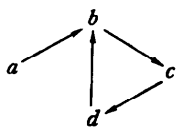

図-1 グラフ

Arc

From	To
a	*b*
b	*c*
c	*d*
d	*b*

図-2 関係データベース

を表す）と仮定することが多い．この仮定を表す論理式を領域閉包公理 (domain closure axiom) と Reiter は呼んでいる[15]．

これらの仮定によって図-2 の関係データベースは図-1 のグラフのみを表すことになる．この対応を関係データベースと表裏一体の関係にある一階述語論理との対応で考えると，関係データベースが一階述語論理の唯一の構造 (structure)* を表すことになる．言いかえると，関係データベース内のすべてのデータに対応する論理式の集合が唯一のモデル**をもつように，上で述べたようないくつかの仮定を行うわけである．このように不確定な情報を排除して（たとえば，a から c へのアークがあるかもしれないなどとは考えない），質問処理とデータの格納を簡単化したシステムが関係データベースであるという見方もできる．

Reiter は上で述べたそれぞれの仮定を一階述語論理の言葉で厳密に定義し，関係データベースを論理式の集合と位置づけた[15),16]．単純な表で表されているデータをわざわざ論理式の集合としてとらえ直す意義は次の点である．

1. 新しいデータを導く（演繹）のに使う知識を論理式として表せば，演繹データベースを論理式の集合と考えることができる．この結果，質問処理を定理の自動証明の問題に還元でき，すでに人工知能の分野で得られている成果を活用できる可能性がある．

2. モデル論的手法で $Q(a)$ または $Q(b)$ といった不確定な情報を表すと，同時に複数の（場合によっては無限個の）構造を考えなければならないが***，論理式の集合として演繹データベースを表すと，$Q(a)\vee Q(b)$ や $(\exists x)Q(x)$ といった論理式で簡単に表現できる．

3. 関係データベースの処理で暗黙に仮定されていたことがはっきりとする．

ここで，具体的に Reiter がどのように上で述べた仮定を形式化したか示す．彼の形式化は次章で説明する演繹データベースの議論においても，自然に拡張できる．まず，データベースに格納されていない事実はすべて偽であるという閉世界仮説について述べる．Reiter の方法では，データベース内の事実に対応する基底アトム (ground atom)**** の集合を T とし，T から証明できない基底アトムの否定からなる集合，

$$Neg(T)=\{\neg Q(a_1,\cdots,a_n)\mid T\nvdash Q(a_1,\cdots,a_n)\}^*$$

を暗黙に仮定された否定情報と考え，$Neg(T)$ を T に追加する．図-2 のデータベースの場合，

$$\{\neg Arc(a,a),\ \neg Arc(b,a),\cdots,\neg Arc(d,d)\}$$

という論理式の集合を追加することになる．

否定情報を表すもう一つの方法は真になるのはこれだけといった情報を論理式で，たとえば今の例では

$$Arc(x,y)\equiv(x=a\wedge y=b)\vee(x=b\wedge y=c)\vee\cdots\vee(x=d\wedge y=b)\qquad(1)$$

と書く方法である．この方法は今の例のように事実しかない場合はうまく記述できるが，演繹のための知識が導入された場合に問題点が生じる．この点については，**3.1.2** で述べる．

単一名仮説はすべての定数は違うという論理式の集合

$$a\neq b,\ a\neq c,\cdots,c\neq d$$

になる．以後，単一名仮説で仮定されるこれらの論理式の集合を UNA で表すことにする．また，領域閉包公理は

$$(\forall x)(x=a\vee\cdots\vee x=d)$$

と表せる．以後，この論理式を DCA で表す．これらの論理式では等号 (=) を使うので，等号に関する一般的な公理（以後，EQ で表す）** も関係データベースの形式化のために必要である．

3. 演繹データベースの形式的意味

演繹データベースでは，$Arc(a,b)$ といった基底アトムで表される事実以外に「x から y へのアークがあり，y から z への道があれば，x から z への道がある」といった知識を表す論理式

$$Arc(x,y)\wedge Path(y,z)\rightarrow Path(x,z)\qquad(2)$$

も取り扱う．演繹データベースの分野では，(2) 式のような演繹に使う知識に対応する論理式の集合を IDB (Intensional Database)，通常のデータベースに対応する基底アトムの集合を EDB (Extensional Database) と呼び，演繹データベースは IDB と EDB から構成されていると考えることが多い．また，IDB の論理式は新しい事実を生成するための規則と考え，IDB の論理式をルールと呼ぶことも多い．

IDB で扱う論理式として任意の論理式を許すと質問処理は複雑になり，一般には決定不能な問題にな

---

* 構造の定義は本誌の別の解説[1] に述べられている．

** 論理式の集合のモデルとはその集合のすべての論理式を真とするような構造のことをいう．

*** たとえば $Q(a)$ だけが真の世界，$Q(b)$ のみが真の世界，両方とも真になる世界．

**** 変数を含まない原子論理式．

* $T\nvdash Q$ は「T から Q が証明できない」を表すとする．

** 詳細は参考文献 6) を参照のこと．

る．そこで，扱う論理式の形，あるいは論理式で扱う世界を制限する研究がなされてきた．その制限は個々の研究成果によって微妙に違うので，各研究成果を比較するときには注意が必要である．

論理式の形を制限する場合，ホーン節や後で述べる層状プログラム (stratified program) に制限することが多い．これらの制限を行うと，$Q(a)$ または $Q(b)$ といった不確定情報を扱えなくなるので，質問処理は扱いやすくなる．

論理式で扱う世界を制限する方法には，

(制限 1) 関数記号は有限個の定数しかないとする

(制限 2) 単一名仮説またはその拡張を仮定する

(制限 3) 領域閉包公理またはその拡張を仮定する

といった方法がある．なお，以下では上の制限 1，2，3 を仮定する．

これらの制限以外にも，暗黙に仮定された否定情報をどのように形式化しているかという点はそれぞれの結果を理解するうえで重要である．

3.1 Datalog

Datalog とは Data と Prolog からつくられた造語で，関係データベースに Prolog によって実現されるような演繹機能を付加したデータベースを考えるというのがこの造語の由来である．

Datalog では，IDB の論理式を確定ホーン節と呼ばれる論理式に制限し，かつ一般の関数記号は扱わず，有限個の定数のみからなる言語を考える．確定ホーン節とは，(2)式のように前提部がアトムの論理積で書かれ，結論部が唯一のアトム* で表される論理式である．したがって，IDB は

$$Arc(x, y) \rightarrow Path(x, y) \qquad (3)$$

と(2)式からなり，EDB は図-1 に対応する基底アトムの集合からなる演繹データベースは Datalog の一例である．

3.1.1 Datalog の不動点意味

上の例をもとにして Datalog の意味を考える．まず，IDB の知識を使って EDB のデータからどのように新しいデータが得られるか考えてみよう．(3)式と $Arc(b, c)$ というデータから $Path(b, c)$ というデータが得られる．次に，(2)式と $Arc(a, b)$, $Path(b, c)$ から $Path(a, c)$ が得られる．このように，次次と新しいデータを生成する手続きを形式的に述べるために，基底アトムの集合 S を別の基底アトムの集合 $T_P(S)$ に変換する関数 T_P を定義する．ここで，S は今までに得られたすべてのデータに対応し，$T_P(S)$ は IDB の知識と EDB と S を使って新しく得られるデータに対応する．

P は EDB と IDB の和集合であるとする．基底アトム Q が $T_P(S)$ に含まれるための必要十分条件は「P のなかのある確定ホーン節のすべての変数に適当な定数を代入すると，結論部が Q の形をした

$$Q_1 \wedge \cdots \wedge Q_m \rightarrow Q$$

という確定ホーン節が得られ，$Q_1, \cdots, Q_m$ はすべて S に含まれる」である．

このように T_P という関数を定義すると

$$T_P(\phi) = \text{EDB}$$

がいえる．また，$T_P^n(S)$ は S に T_P を n 回適用して得られる基底アトムの集合とする．直観的には，$T_P^n(\phi)$ は EDB に高々 $n-1$ 回 IDB のルールを適用して得られる基底アトムの集合を表す．

T_P は単調関数，すなわち

$$S \subseteq S' \text{ ならば } T_P(S) \subseteq T_P(S')$$

という性質をもつので，$T_P^n(\phi)$ は n に関して集合の包含関係の意味で単調増加，すなわち

$$\phi \subseteq T_P(\phi) \subseteq T_P^2(\phi) \subseteq \cdots$$

が成り立つ．

したがって，

$$\lim_{n \to \infty} T_P^n(\phi)$$

が存在する．この極限を $T_P \uparrow \omega$ で表すと，$T_P \uparrow \omega$ は EDB に IDB の知識を適用して得られるデータ全体を表すことになる．よって，基底アトム Q が $T_P \uparrow \omega$ に含まれていれば，Q が真であると考え，含まれていなければ，偽であると考える．このとき，$T_P \uparrow \omega$ は T_P の最小不動点* であるという特徴づけもできるので，$T_P \uparrow \omega$ を P の不動点意味という．また，$T_P \uparrow \omega$ は P のエルブランモデル**のなかで，真となる基底アトムの集合が最小（最小エルブランモデル）であるという特徴づけもできる．なお，上の例では $Path(a, c)$ は真であるが，$Path(c, a)$ は偽になる．

3.1.2 Datalog の論理的意味

上で述べた $T_P \uparrow \omega$ は Reiter の閉世界仮説を使うと次のように特徴づけることができる．まず，2. と同じようにして，$Neg(P)$ を定義し，

$$P \cup Neg(P) \cup \text{DCA} \cup \text{UNA} \cup \text{EQ}$$

という論理式の集合を DDB で表すと，$T_P \uparrow \omega$ は

---

* このアトムの述語記号は等号でないとする．

* $T_P(S) = S$ となる集合 S のなかで，集合の包含関係の意味で一番小さい集合．

** 今の場合，一般の関数は考えていないので，領域が定数の集合からなるモデルをエルブランモデルという．

DDB の唯一のエルブランモデルである．したがって，$T_P\uparrow\omega$ は DDB から証明できる基底アトム全体に等しい．また，基底アトム Q が $T_P\uparrow\omega$ に含まれていなければ，DDB から $\neg Q$ が証明できる．

一方，暗黙に仮定された否定情報を(1)式の拡張の立場で記述すると，グラフの例の場合 *Arc* に関する(1)式と *Path* に関する

$$Path(x,y)\equiv Arc(x,y)\vee [(\exists z)Arc(x,z)\wedge Path(z,y)] \quad (4)$$

という論理式が必要になる．(4)式の直観的意味は「もし x から y への道があるならば，その道は IDB のルールから導き出される」に対応している．(4)式のように「導けるものは，EDB に IDB を適用して得られるものだけ」という意味に対応する論理式を Clark の Completion Axiom と呼び，$Comp(P)$ で表す．このとき，

$$P\cup Comp(P)\cup \mathrm{DCA}\cup \mathrm{UNA}\cup \mathrm{EQ}$$

という論理式の集合を DDB′ で表すと，$T_P\uparrow\omega$ は DDB′ から証明できる基底アトム全体にも等しくなる．しかし，基底アトム Q が $T_P\uparrow\omega$ に含まれていないからといって，DDB′ から $\neg Q$ が証明できるとは限らない．DDB′ から証明できる否定情報は SLD 導出と呼ばれる定理証明法と密接に関連していることが知られている[6),18)]．

今の例では，$\neg Path(b,a)$ は DDB′ から証明できず，DDB′ の意味はわれわれの直観とは合わない．

3.2 論理和演繹データベース

Datalog では IDB で扱う論理式の形は確定ホーン節に制限されていた．この制限によって，Datalog の意味は直観的に分かりやすくなり，質問処理の問題も扱いやすくなっている．しかし，確定ホーン節だけでは「$Q(a)$ または $R(b)$」といった不確定な情報を表すことができない．そこで，IDB で扱う論理式の形を

$$Q_1\wedge\cdots\wedge Q_m\rightarrow R_1\vee\cdots\vee R_k \quad (5)$$

(ただし，$m\geq 0$ かつ $k\geq 1$) というように，結論部に論理和($\vee$)が書けるように拡張して考える．このような論理式で IDB が記述されている演繹データベースを特に論理和演繹データベース (disjunctive deductive database) と呼ぶことにする．

Datalog で表される演繹データベースの意味は $T_P\uparrow\omega$ で表される唯一の世界（モデル）によって表された．しかし，論理和演繹データベースでは不確定な情報を扱うので複数のモデルを同時に考えなければいけない．

論理和演繹データベース (DDB) の意味の定式化に関してはいくつかの提案がある．それらの提案では暗黙に仮定された否定情報をどのように取り扱うかが異なるが，肯定情報に関しては次のような共通の意味を考えている．ϕ を基底アトムの論理和の形 ($Q_1\vee\cdots\vee Q_i$) をした論理式とする．このとき，ϕ が DDB から導ける意味は DDB のすべてのエルブランモデル上で ϕ の解釈が真となることである．また，この条件は DDB のすべての極小エルブランモデル* 上で ϕ の解釈が真となるという条件に等しいことが知られている．

暗黙に仮定された否定情報の定式化については次の提案がある．

1. Minker による一般化閉世界仮説 (Generalized Closed World Assumption)[7)]
2. McCarthy による極小限定 (Circumscription)[5)]
3. Ross と Topor による定式化[17)]

一般化閉世界仮説では，基底アトム Q がすべての極小エルブランモデルで偽となるとき $\neg Q$ を暗黙に仮定された否定情報とみなす．この否定情報の集合を Neg とすると，DDB の論理的意味は DDB, Neg, DCA, UNA, EQ からなる論理式の集合で表されているとする．

極小限定では，暗黙に仮定された否定情報をつけ加えるという立場をとらず，すべての極小モデルで成り立つことが DDB の実際の意味に対応していると考える．

一般化閉世界仮説と極小限定の違いはたとえば次の点に現れる．$Q(a)\vee R(b)$ からなる演繹データベースに一般化閉世界仮説を適用すると Neg は空集合となり，

$$(\neg Q(a)\wedge R(b))\vee(Q(a)\wedge\neg R(b))$$

という論理式は一般化閉世界仮説のもとで真とも偽とも決められない．一方，極小限定の定式化ではこの論理式は真となる．

Ross と Topor の定式化** では，次のようにして否定情報を DDB につけ加える．まず，IDB 内の(5)式の形の各論理式を

$$Q_1\wedge\cdots\wedge Q_m\rightarrow R_i\ (1\leq i\leq k)$$

という k 個の確定ホーン節の集合に変換してできる

---

* ここで，極小とは真となる基底アトムの集合が包含関係の意味で極小となることである．一般に，論理和演繹データベースの場合 $T_P\uparrow\omega$ のような最小エルブランモデルは存在しない．

** Rajasekar らも同じ概念を別の方法で定式化している．参考文献14)参照．

Datalog の形をした演繹データベースを P' とする．そして，基底アトム Q が $T_{P'}\uparrow\omega$ に属していなければ，$\neg Q$ を暗黙に仮定された否定情報とみなす．この定式化では論理的に等価な二つの IDB が異なる意味をもつ場合がある．たとえば，$Q(a)\vee R(b)$ と $Q(a)$ からなる IDB_1 と $Q(a)$ のみからなる IDB_2 は論理的に等価であるが，IDB_1 では $\neg R(b)$ が仮定されず，IDB_2 では仮定される．

このように否定情報の取り扱いをめぐってさまざまな提案があるのは，Datalog の場合と違って論理和情報があるときに人間がどのような暗黙の仮定を行っているかがはっきりしないからである．今後，論理和情報を使った演繹データベースの応用分野が定まってくると，その応用分野に依存した別の定式化が考え出される可能性がある．

また，論理和演繹データベースでは複数のモデルを同時に考えなければならないので質問処理は複雑になり，「a または b」といった不確定な答の概念[15]も導入しなければならない．

3.3 層状プログラム

節点 x から節点 y への道があるが，y から x への道はないとき，x から y への一方通行だと考えて，*One-way*(x, y) と表すことにする．この *One-way* という述語を論理式で記述すると，

$$Path(x, y)\wedge\neg Path(y, x)\rightarrow One\text{-}way(x, y) \qquad (6)$$

と書ける．この論理式は

$$Path(x, y)\rightarrow Path(y, x)\vee One\text{-}way(x, y)$$

と等価である．前者の論理式は，普通人間が意図するまず *Path* の概念があって，次にすでにもっているその *Path* を使って *One-way* を定義するという新しい概念の生成方法に合致しているようにみえる．一方，後者の論理式はそのような人間の意図を無視して不自然な形で論理和情報を導入しているようにみえる．

そこで，述語記号の集合を $\mathcal{P}_1,\cdots,\mathcal{P}_N$ という N 個の階層に分割する．$\overline{\mathcal{P}}_i$ は $\mathcal{P}_1,\cdots,\mathcal{P}_i$ の和集合を表すとする．直観的には，$\mathcal{P}_1$ の述語記号はあらかじめ与えられた概念を表し，$\mathcal{P}_i$ の述語記号は自分と同じ階層 ($\mathcal{P}_i$) あるいは自分よりも下の階層 ($\overline{\mathcal{P}}_{i-1}$) を使って定義されると考える．そして，IDB で扱う論理式を

$$Q_1\wedge\cdots\wedge Q_m\wedge\neg R_1\wedge\cdots\wedge\neg R_k\rightarrow Q \qquad (7)$$

という形に制限し，さらにアトム Q の述語記号が $\mathcal{P}_i$ に属していれば，各 $Q_j(1\leq j\leq m)$ の述語記号は $\overline{\mathcal{P}}_i$ に属し，各 $R_j(1\leq j\leq k)$ の述語記号は $\overline{\mathcal{P}}_{i-1}$ に属さなければならないという条件を課する．この条件は，肯定情報を使うときは自分と同じ階層までの情報を使えるが，否定情報を使うときは自分よりも下の階層しか使えないという制限を表している．

IDB がこのような制限を満たす論理式からなる演繹データベースを層状プログラム (Stratified Program)[1] と呼ぶ．たとえば，IDB が(2)，(3)，(6)式からなる演繹データベースは，$\mathcal{P}_1$ を {*Arc*, *Path*}，$\mathcal{P}_2$ を {*One-Way*} と考えると層状プログラムである．

層状プログラムの意味は次のように下の階層から順番に Datalog の場合と同じようにして決まる．まず，結論部の述語記号（上の Q の述語記号）が階層 $\mathcal{P}_i$ に属している IDB の論理式の集合と，述語記号が $\mathcal{P}_i$ に属している EDB の基底アトムの集合の和からなる集合を P_i とする．このとき，P_1 は確定ホーン節の集合なので，Datalog の不動点意味を定めるさいに使った T_{P_1} という関数を用いて $\mathcal{P}_1$ の述語記号をもつ基底アトム Q の真偽値を

$$Q\text{が真}\Longleftrightarrow Q\in T_{P_1}\uparrow\omega$$

という条件で定めることができる．

次に，T_{P_2} を使って，$\mathcal{P}_2$ の述語の基底アトム Q の真偽値を定めるが，P_2 は確定ホーン節の集合ではないので，(5)式の形をした論理式を扱えるように T_{P_2} という関数の定義を拡張する必要がある．すなわち，基底アトム Q が $T_{P_2}(S)$ に含まれるための必要十分条件は「$Q\in T_{P_1}\uparrow\omega$」または「$P_2$ のある論理式のすべての変数に適当な定数を代入すると，結論部が Q である

$$Q_1'\wedge\cdots\wedge Q_m'\wedge\neg R_1'\wedge\cdots\wedge\neg R_k'\rightarrow Q$$

という形の論理式になり，かつ $Q_1',\cdots,Q_m'$ はすべて S に含まれ，$R_1',\cdots,R_k'$ は S に含まれない」である．

このとき，P_2 に表れる否定記号はすべて $\mathcal{P}_1$ の述語記号に関するものだけなので，確定ホーン節の場合と同様にして $T_{P_2}^{n}(T_{P_1}\uparrow\omega)$ が n に関して単調増大であることが示せるので，

$$lim_{n\to\infty}T_{P_2}^{n}(T_{P_1}\uparrow\omega)$$

が存在する．この極限を M_{P_2} で表すと，M_{P_2} は $T_{P_1}\uparrow\omega$ を含む．$\mathcal{P}_2$ の述語記号をもつ基底アトム Q の真偽値を

$$Q\text{が真}\Longleftrightarrow Q\in M_{P_2}$$

という条件で定める．

このように下の階層から順番に意味を定めていく

と，M_{PN} が層状プログラムの意味を表すことになる．

Datalog の場合には最小エルブランモデルが存在し，そのモデルが Datalog の意味を表した．層状プログラムの場合には一般に最小エルブランモデルは存在せず，複数の極小エルブランモデルが存在する．IDB のなかの否定記号の現れかたをもとにして，極小エルブランモデルのあいだにある順序を導入できる．この順序に関して最小となる極小エルブランモデルを完全モデル (perfect model) と呼ぶが，M_{PN} は Pの完全モデルであることが知られている[11),12)]．

層状プログラムの論理的意味は Reiter の閉世界仮説を階層ごとに使うことによっても表せる．まず，一番下の階層の暗黙に仮定される否定情報を Reiter の閉世界仮説を使って

$$CWA(P_1)=P_1\cup Neg(P_1)$$

が $\mathcal{P}_1$ の情報を表していると考える．

他の階層では，暗黙に仮定された否定情報を自分よりも下の階層の情報を使って定義する．$CWA(P_{i-1})$ が階層 $i-1$ までの情報を表すと考えると，

$$NEG(P_i)=\{\neg Q(t_1,\cdots,t_n)\mid CWA(P_{i-1})\cup P_i\not\vdash Q(t_1,\cdots,t_n)\}$$

が階層 i の暗黙に仮定された否定情報を表す．したがって，階層 i までの情報は

$$CWA(P_i)=P_i\cup NEG(P_i)\cup CWA(P_{i-1})$$

で表せると考える．

このとき，完全モデル M_{PN} は

$$CWA(P_N)\cup DCA\cup UNA\cup EQ$$

の唯一のモデルと特徴づけることができる[3)]．

3.4 正当なモデル (Well-Founded Model)

IDB の記述力をますため，層状プログラムを拡張することを考える．一つの方向は，結論部に論理和 ($\vee$) を許すことである．すなわち，

$$Q_1\wedge\cdots\wedge Q_m\wedge\neg R_1\wedge\cdots\wedge\neg R_k\rightarrow S_1\vee\cdots\vee S_j$$

という形の論理式を考え，否定記号の現れかたに関しては層状プログラムと同じ制限，またはより弱い制限を課する場合である[12)]．

この場合は論理和演繹データベースの拡張にもなっているので，その妥当な意味の定式化，質問処理はいっそう難しくなる．そこで，論理和を許さず，一つのモデルで意味が表せるという性質をできるだけ保つ方法を考える．

たとえば，**図-3** に示した，二人ゲームの局面遷移図を考えよう．このグラフで各ノードは局面を表し，ノード x からノード y へのアークは局面 x を局面 y に変える手があることを示している．あるプレーヤが自分の指す番であるにもかかわらず，指す手がなければそのプレーヤの負けであるとする．たとえば，図-3 のグラフで局面 d で指す順番がまわってきたプレーヤは負けである．

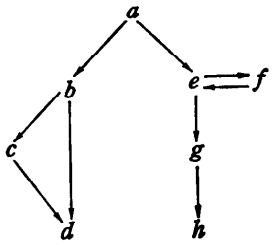

図-3 局面遷移図

今，*Move* と *Win* という二つの述語記号は次の意味を表すと考える．$Move(x,y)$ が真ならば，x から y へ動かす手がある．$Win(x)$ が真ならば，局面 x で先手必勝，すなわち後手がどんな手を打っても先手がうまい手を打ち続ければ先手が必ず勝つ．この場合，$Win(x)$ が偽ならば，局面 x で後手必勝になる．したがって，局面 x を局面 y へ動かす手があり，局面 y が後手必勝ならば，x で先手必勝であるという論理式

$$Move(x,y)\wedge\neg Win(y)\rightarrow Win(x)\qquad(8)$$

が真となるべきである．

(8)式では，結論部と同じ述語記号に関する否定が前提部に現れているので，層状プログラムでは扱えない．しかし，局面 x から移れる局面を x の子供と考え，x の子孫を x の下の階層と考えるならば，(8)式は下の階層の否定情報を使って x が先手必勝かどうかを決めているともみなせる．このように，IDB が(7)式の形をした論理式の集合で，層状プログラムの場合に仮定した否定記号の現れかたに関する制限をいっさい考えないときにどのような意味を与えるかを説明する．

層状プログラムのこのような拡張に関してはさまざまな提案がなされたが，ここでは今までの提案を包含し，かつ今の時点でもっとも有望だと考えられる正当なモデル (Well-founded Model)[13)]を紹介する．

まず，互いに共通部分をもたない基底アトム集合の対 $\langle T, F\rangle$ を I で表し，3値の解釈と呼び，

$Q\in T\Longleftrightarrow Q$ が真

$Q\in F\Longleftrightarrow Q$ が偽

$Q\notin T\cup F\Longleftrightarrow Q$ の真偽値はまだ不明

という対応を考える．今，I は演繹の途中ですでに得

られた情報を表していると考え，層状プログラムの節(3.3)で考えた T_{P_I} に対応する次の T_I と F_I という関数を考える．

$I=\langle T, F\rangle$, S_I を基底アトムの集合，Qを基底アトムとする．Qが $T_I(S_I)$ に含まれるための必要十分条件は，「$Q\notin T$ であり，かつ Pのある論理式のすべての変数に適当に定数を代入すると，結論部がQである

$$Q_1'\wedge\cdots\wedge Q_m'\wedge\neg R_1'\wedge\cdots\wedge\neg R_k'\rightarrow Q$$

という形の論理式が得られ，かつ $Q_1', \cdots, Q_m'$ はすべて $S_I\cup T$ に含まれ，$R_1', \cdots, R_k'$ は F に含まれる」である．

同様に基底アトムの集合 S_I に対して，Qが $F_I(S_I)$ に含まれるための必要十分条件は，「$Q\notin F$ であり，かつPの任意の論理式に対して，変数への適当な代入の結果，結論部がQの形の論理式，すなわち

$$Q_1'\wedge\cdots\wedge Q_m'\wedge\neg R_1'\wedge\cdots\wedge\neg R_k'\rightarrow Q$$

が得られるならば，ある Q_i' が $S_I\cup F$ に含まれるか，ある R_I' がTに含まれるかのどちらかが成り立つ」である．

層状プログラムの場合には，T_{P_I} で導けない基底アトムはその否定が成り立つという考えかたを採用したが，今の場合 F_I を使ってもっと積極的に偽になる場合を求めている．

たとえば，図-3 のアークに対応する *Move* の基底アトムの集合を EDB とし，演繹データベース P が EDB と(8)式からなるとする．最初は，何も情報がない，すなわち $I_1=\langle\phi, \phi\rangle$ という3値の解釈を考える．そして，H_P を基底アトム全体とし，

$$T_{I_1}\uparrow\omega=lim_{n\rightarrow\infty}T_{I_1}^n(\phi)$$
$$F_{I_1}\downarrow\omega=lim_{n\rightarrow\infty}F_{I_1}^n(H_P)$$

という極限を考えると，直観的には前者は EDB，後者はどのように考えても真となる可能性がない基底アトム集合に対応する．今の例では $T_{I_1}\uparrow\omega$ は EDBに等しく，

$$F_{I_1}\downarrow\omega=\{Move(c_1, c_2)\mid Move(c_1, c_2)\notin \text{EDB}\}$$

となる．

次に，$I_2=I_1\cup\langle T_{I_1}\uparrow\omega,\ F_{I_1}\downarrow\omega\rangle$ として上と同様な極限を考えると，

$$T_{I_2}\uparrow\omega=\phi \quad F_{I_2}\downarrow\omega=\{Win(d),\ Win(h)\}$$

となり，局面 d, h で後手必勝という結果が得られる．

さらに，$I_3=I_2\cup\langle T_{I_2}\uparrow\omega,\ F_{I_2}\downarrow\omega\rangle$ として上と同様な極限を考えると，

$$T_{I_3}\uparrow\omega=\{Win(b),\ Win(c)\} \qquad F_{I_3}\downarrow\omega=\phi$$

となり，局面 b, c で先手必勝という結果が得られる．

$I_4=I_3\cup\langle T_{I_3}\uparrow\omega,\ F_{I_3}\downarrow\omega\rangle$ として，ふたたび極限を考えると，

$$T_{I_4}\uparrow\omega=\phi \qquad F_{I_4}\downarrow\omega=\phi$$

となり，もはや新しい情報は得られない．

そこで，I_4 を P の正当なモデルと呼ぶ．このとき，次の点に注意する必要がある．

- 正当なモデルは3値のモデルなのですべての基底アトムの真偽値が定まるわけではない．上の例では $Win(a)$, $Win(e)$, $Win(f)$ の真偽値は未定義である．これは，局面 a, e, f では先手必勝，後手必勝ともいえないことに対応している．
- 3値のモデルの解釈を用いると普通の場合と意味が変わる場合がある．$Win(a)\vee\neg Win(a)$ という論理式は通常は真と考えるが，3値の解釈では未定義になる．
- Pが層状プログラムであればPの正当なモデルは完全モデルと一致する．

正当なモデルはいくつかの別の方法で特徴づけられることが知られている．これらの方法については参考文献 13)，21)を参照していただきたい．

4. おわりに

3.では，一般の関数記号は使わず，有限個の定数のみを使って演繹データベースが記述される場合について述べた．関数記号を導入するといくつかの問題点が生じるが，基本的には上で述べたことがほとんどそのまま成立する．関数記号の導入はさらに一般的に考えると，演繹データベースの世界に構造をもった対象を導入するというとらえかたもできる．こういった見かたに関しては本誌の別の解説[9]を参照していただきたい．

本解説では，人間にとって直観的に分かりやすい意味を与えられた情報からどのように求めるかという立場で，演繹データベースの意味を説明した．しかし，実際に計算機で処理することを考えれば，いかに妥当な意味を定めようとも，質問処理に時間がかかり過ぎて結局答が求まらないのでは，そのような意味を考える意義が薄れてしまう．逆に，いかに高速に処理可能な質問処理アルゴリズムがあっても，われわれが意図した意味と異なる答を返すのであれば，そのアルゴリズムは使いものにならない．一般に，計算機で処理しやすい意味は人間が意図する意味と異なる傾向があるので，処理しやすさと妥当性という二つの側面を念頭において研究を進めることが重要である．なお，演繹

データベースの質問処理に関しては本誌の別の解説[10]や参考文献 19), 20) を参照していただきたい.

データベースの研究分野の一つに質問言語の表現能力に関する分野がある. この分野は一見演繹データベースとは直接関係がないようにみえるが, 演繹のための規則が質問言語に埋め込めるという見かたをすれば密接に関連する. この分野の動向に関しては参考文献 2)を参照していただきたい.

また, 本解説では論理的な意味を表すのに Reiter の閉世界仮説を用いたが, 閉世界仮説よりも記述力が強い極小限定を用いることも可能である. この点に関しては Przymusinski の一連の論文[3),12),13)]や参考文献 11)を参照していただきたい.

謝辞 本解説の草稿に対して貴重なコメントをくださった宮崎収兄氏, 三浦孝夫氏, 外山芳人氏, 内藤昭三氏, 白柳潔氏に深謝いたします.

参考文献

1) Apt, K. R., Blair, H. and Walker, A.: *Towards a Theory of Declarative Knowledge.* In 8), pp. 547-623.
2) Chandra, A. K.: *Theory of Database Queries*, Proc. of ACM-PODS, pp. 1-9 (1988).
3) Gelfond, M., Przymusinska, H. and Przymusinski, T.: *On the Relationship between Circumscription and Negation as Failure*, Artif. Intell., Vol. 38, No. 1, pp. 75-94 (1989).
4) 小林功武: 古典データベースから演繹データベースへ, 情報処理, Vol. 31, No. 2, pp. 189～197 (1990).
5) Lifschitz, V.: *Computing Circumscription*, Proc. of IJCAI, pp. 121-127 (1985).
6) Lloyd, J. W.: *Foundations of Logic Programming*. 2nd Eds., Springer-Verlag, Berlin and New York (1987).
7) Minker, J.: *On Indefinite Databases and the Closed World Assumption*, In Springer's Lecture Note in Computer Science, Vol .138, pp. 292-308.
8) Minker, J. Ed.: *Foundations of Deductive Databases and Logic Programming*, Morgan Kaufmann Publishers (1988).
9) 三浦孝夫, 塩谷 勇: 演繹データベースと複合オブジェクト, 情報処理, Vol. 31, No. 2, pp. 206～215 (1990).
10) 宮崎収兄, 世木博久: 演繹データベースの問合せ処理, 情報処理, Vol. 31, No. 2, pp. 216～224 (1990).
11) 中川裕志: 極小限定の研究動向, 情報処理, Vol. 30, No. 6, pp. 684-693 (1989).
12) Przymusinski, T. C.: *On the Declarative Semantics of Deductive Databases and Logic Programs*, In 8). pp. 193-216.
13) Przymusinski, T. C.: *Every Logic Program has a Natural Stratification and an Iterated Least Fixed Point Model*, Proc. of ACM-PODS. pp. 11-21 (1989).
14) Rajasekar, A., Lobo, J. and Minker, J.: *Weak Generalized Closed World Assumption*, J. of Automated Reasoning, Vol. 5, pp. 293-307 (1989).
15) Reiter, R.: *Equality and Domain Closure in First-Order Databases*, J. ACM, Vol. 27, No. 2, pp. 235-249 (1980).
16) Reiter, R.: *Towards a Logical Reconstruction of Relational Database Theory*, In On Conceptual Modeling, Brodie, M. and et al., Eds., Springer-Verlag, Berlin and New York, pp. 191-238 (1984).
17) Ross, A. K. and Topor, R. W.: *Inferring Negative Information from Disjunctive Databases*, J. of Automated Reasoning, Vol. 4, pp. 397-424 (1988).
18) Shepherdson, J. C.: *Negation in Logic Programming*, In 8), pp. 19-88.
19) Ullman, J. D.: *Principles of Database and Knowledge-Base Systems* Vol. 1, Computer Science Press, Maryland (1988).
20) Ullman, J. D.: *Principles of Database and Knowledge-Base Systems* Vol. 2, Computer Science Press, Maryland (1989).
21) Van Gelder, A.: *The Alternating Fixpoint of Logic Programs with Negation*, Proc. of ACM-PODS, pp. 1-10 (1989).

(平成元年 10 月 6 日受付)