CODASYL データベースシステムに対する関係インタフェースシステム (LDP-V 1.5) の設計と実現†

滝沢 誠†† 横塚 実†† 鈴木 信†††

本論文では，CODASYL データベースシステムに対する関係インタフェース (LDP-V 1.5) の設計と実現について論じる．関係インタフェースを考えるためには，CODASYL から関係スキーマへの変換と，逆に関係問合せ (QUEL 問合せ) を COBOL DML に変換し実行させる問合せ変換とが必要になる。ここでは，検索利用に限定して，上記問題の解決を行った．スキーマ変換では，互いのデータベースの内容を1対1に情報保存する変換アルゴリズムを示した．問合せ変換では，二つの言語の相違，データモデルとアクセス単位の相違を明らかにし，その解決手法を示した．まず，CODASYL モデルに基づいた非手続的問合せ (CODASYL 問合せ) を定義し，QUEL からこれへの変換を行う．次に，中間結果数を最少とし，アクセス実現値数をなるべく少なくするようにアクセスパスが生成され，必要な COBOL DML プログラムが生成される．このなかで，アクセスパスのコスト関数も同時に示した．これらは，LDP-V 1.5 として当協会の AIM と ADBS 上に実現されている．実現上問題となる LDP とコンパライラとのプロセス間通信は，計算機網 JIPNET によって実現した．本システムは，分散型データベースシステムの CODASYL システムの共通インタフェースになるとともに，一般ユーザの非手続的検索インタフェース，COBOL DML プログラム生産ツールともなる．

1. はじめに

CODASYL データベース・システム (以降 DBS と記す)[1] は，論理と物理構造が未分離な網型データ構造と，手続的な操作言語とを提供し，おもにパフォーマンス指向の大規模定型業務に用いられてきた．しかし，近年の DBS 応用の多様化と，非定型利用の増大によって，CODASYL DBS 上の非手続的インタフェースの必要性が高まってきている．分散型データベース・システム (DDBS と記す) においても，複数の異種 DBS を統合的に利用[2)~4)] するためには，まずこれらを同種化する共通関係モデルインタフェースの設定が必要となる[4)~6)]．このため，当協会では，CODASYL DBS 上に，非手続的検索機能を備えた関係インタフェース (ローカル・データベース・プロセッサ (LDP-V 1.5)) を開発した．本システムの目標は，以下の2点である．

(1) 開発中の JDDBS[4)~8)] における CODASYL DBS 上の共通インタフェースとなる．

(2) CODASYL DBS の一般ユーザ・インタフェースとともに，COBOL DML プログラムの開発支援ツールとなる．

現在までに，CODASYL DBS の関係インタフェースとしては，文献9)，10) 等が，更新演算用のスキーマ変換を論じているが，アクセス言語変換については，文献6) があるだけである．更新を考える上でも，更新すべきデータを見つけるための検索演算がまず必要となる．本論文では，文献6) に対して，関係問合せの結合として，主キーの等結合以外も許し，アクセスパス生成の改良を行い，システムの実現を行っている．

2章では，CODASYL スキーマから関係スキーマへの変換について述べ，3章では関係問合せの定義を行う．4, 5, 6 章では，おのおの，問合せの構造の変換，アクセスパス生成，DML の生成について論じる．最後に，7章で，LDP-V 1.5 の実現結果について述べる．

2. スキーマ変換

本章では，CODASYL スキーマから関係スキーマ (これをローカル概念スキーマ (LCS)[4),5)] とよぶ) への変換について述べる．

2.1 CODASYL モデル[1)]

CODASYL スキーマは，レコード型とセット型の集合である．レコード型 R は，データ項目集合 Ω_R $(=\{t_1, \cdots, t_r\})$ から成り，$R(t_1, \cdots, t_n)$ と記される．R の要素 r はレコード実現値とよばれる．R 内の各 r は，

† The Design and Implementation of LDP-V 1.5—The Relational Interface System over the CODASYL Database Systems by MAKOTO TAKIZAWA, MINORU YOKOTSUKA (Japan Information Processing Development Center, Development Department) and MAKOTO SUZUKI (Systech Co., Ltd.).

†† (財) 日本情報処理開発協会開発部

††† (株) システク

一般にそのデータ項目の値によって識別できず，その位置，すなわちデータベース・キー (db キー) によって識別できる．r の db キーを db-key (r) と記す．R の物理構造としては，実現値の格納配置を定める配置モードがある．R が配置モードとして CALC 項目をもつとき，この CALC 項目を，c-item(R) と記す．

セット型 S は，二つのレコード型 A と B 間の1対 N 関係性 ($S \subseteq A \times B$) を表し，$S(A, B)$ と記される．A と B を，おのおの S の親と子レコード型とよび，おのおの owner(S), member(S) と記す．S の物理構造としては，B 内の実現値の順序付け，DNA (Duplicates are not allowed) 指定，ポインタ種類がある．B の項目 t が S について DNA 項目のとき，同一の親実現値をもつすべての B 実現値の t の値は一意でなければならない．これらは，以下の記法を用いて表す．

s-item(S)≜S のソートキー項目 ($\subseteq \Omega_B$)
s-type(S)≜ソートの順 (昇 (A) または降 (D) 順)
D-item(S)≜S の DNA 項目 ($\subseteq \Omega_B$)
ptr(S)≜S を実現しているポインタの種類
$\in \{\{n\}, \{n,p\}, \{n,o\}, \{n,p,o\}\}$

ここで，n, p, o はおのおの next, prior, owner ポインタを表している．

2.2 スキーマ変換アルゴリズム

CODASYL から関係スキーマへの変換アルゴリズムを以下に示す．

(1) レコード型 $R(t_1, \cdots, t_n)$ から，関係 $E_R(\underline{@R}, a_{t_1}, \cdots, a_{t_n})$ を生成する．E_R と a_{t_i} は，おのおの R と t_i に対応した関係名と属性名である．$@R$ は，db キーを値としてとる仮想属性で，これを主属性とよぶ．また $@R$ は主キーでもある．

(2) セット型 $S(A, B)$ から，関係 $R_S(@A, \underline{@B})$ を生成する．R_S は S に対応した関係名である．$@A$ と $@B$ は，おのおの関係 E_A と E_B の主属性である．$@B$ は関係 R_S の主キーとなる．

(3) $n(>1)$ 個のレコード型 $R_1, \cdots, R_n$ 間の n 項関係性は，**図1**のようなレコード型 $L(t_1, \cdots, t_m)$ とセット型 $S_i(R_i, L)(i=1, \cdots, n)$ によって表せる．このような L を，とくにリンクレコード型とよぶ．リンクレコード型 L からは，関係 $R_L(\underline{@R_1}, \cdots, \underline{@R_n}, a_{t_1}, \cdots, a_{t_m})$ が生成される．各 $@R_i$ は，関係 E_{R_i} の主属性である．R_i と R_j ($i \neq j$) とが同一のレコード型のときには，関係 R_L の主属性 $@R_i$ と $@R_j$ とを区別するために，おのおのセット型 S_i と S_j を前置 (i.e. $S_i@R_i, S_j@R_j$) した名前を主属性名として用いる．

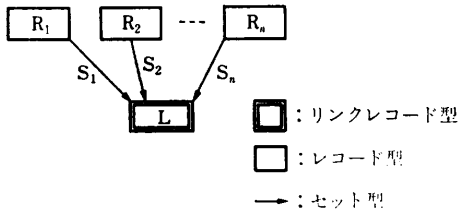


図 1 リンクレコード型
Fig. 1 Link record-type.

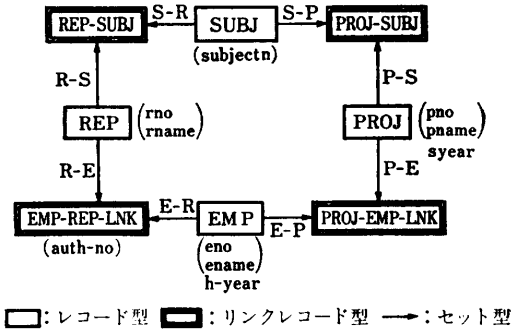


図 2 CODASYL スキーマ
Fig. 2 CODASYL Schema.

主属性を一つだけ有する関係を，E (または事象) 関係とよぶ．(1) で生成された関係は，E 関係である．主属性を複数有し，各主属性はある E 関係の主属性である関係を R(関係性) 関係とよぶ．(2),(3) で生成された関係は，R 関係である．意味的に，R 関係 $R(\underline{@E_1}, \cdots, \underline{@E_n}, a_1, \cdots, a_n)(n>1)$ は，n 個の E 関係 $E_1, \cdots, E_n$ 間の n 項関係性を表している．すなわち，

$$\forall r \in R \quad \exists e_1 \in E_1 \cdots \exists e_n \in E_n$$
$$e_1[@E_1] = r[@E_1] \wedge \cdots$$
$$\cdots \wedge e_n[@E_n] = r[@E_n]$$

このため，主属性間の結合としては，＝と≠のみが許される．また，主属性に対する算術演算は許されず，aggregate 関数[11] としては，count と any のみが許される．また，R 関係 R の各主属性 $@E_i$ に対して，e-rel($@E_i$) は，E 関係 E_i を表すものとする．

ここで，関係 ER に対して，以下の記法を導入する．

patt(ER)≜ER の主属性の集合
pkey(ER)≜ER の主キー

検索利用に限定した場合，二つのスキーマのデータ

SUBJ (@J, subjectn)
PROJ (@P, pno, pname, sycar)
EMP (@E, eno, ename, h-year)
REP (@R, rno, rname)
PROJ-SUBJ (@P, @J)
PROJ-EMP-LNK (@P, @E)
EMP-REP-LNK (@E, @R, auth-no)
REP-SUBJ (@R, @J)

図3 図2の LCS
Fig. 3 The LCS of Fig. 2.

ベースのおのおのの要素が1対1対応するならば，二つのスキーマは，互いの情報が保存されるので等価であるといえる．CODASYL スキーマ C と，これから生成された LCS S では，おのおののデータベースの組と実現値とが1対1対応していることは容易にわかる．よって，C と，生成された S とは，等価であるといえる．

2.3 例

図2は，プロジェクトとメンバ，メンバと論文，論文およびプロジェクトとテーマとの情報を有するデータベースの CODASYL スキーマである．**図3**は，これから生成される LCS を示している．

3. LCS 問合せ

ローカル概念スキーマ (LCS) 内の関係に対する問合せを LCS 問合せとよぶ．これは，関係計算言語 QUEL[11] を用いて，次のように記述される．

range $(l_1, L_1)\cdots(l_m, L_m)$;
retrieve into $R(a_1=ae_1, \cdots, a_k=ae_k)$ where
qual;

L_i は LCS 関係で，l_i はその組変数である．R は結果関係で，a_j はその属性である．問合せ内の変数は，$l.a$ (l は組変数，a はその関係の属性) と記され，値として組 l の a 値をとる．ae_j は，条件式 qual を満足する組 $l_1, \cdots, l_m$ 上の算術式で，結果属性 a_j の値を与える．各 ae_j 内の変数を目標属性とよぶ．

条件式 qual は，制限および結合述語から成る論理式である．組変数 x に関する制限式 $rp(x)$ は QUEL と同じである．組変数 x と y の間の結合述語 $jp(x, y)$ には，主結合と非主結合述語とがある．

i) 主結合述語 $pjp(x, y) \equiv x.a\,\theta\,y.b$

a と b は，おのおの x と y の関係の主属性で，互いの定義域は等しい．2.2 節で述べたように，$\theta \in \{=, \neq\}$ である．

ii) 非主結合述語 $njp(x, y) \equiv ac(x)\,\theta\,ae(y)$

これは，通常の結合である．x と y に関する算術式 $ae(x)$ と $ae(y)$ の値が，任意の $\theta \in \{<, \leq, =, \geq, >, \neq\}$ について比較される．ただし算術式 $ae(x)$, $ae(y)$ は，ともに，主属性を含んではならない．

簡単にするために問合せは，max, count といった集積関数を含まないものとする．

条件式は，次のように積正規化されているとする．

$\text{qual} \equiv c_1 \wedge \cdots \wedge c_n$
$c_i \equiv rc_i(x) \mid jc_i(x, y)$
$rc_i(x) \equiv rp_{i1}(x) \vee \cdots \vee rp_{il_i}(x)$ $(i=1, \cdots, n)$
$jc_i(x) \equiv pjp(x, y) \mid njp_{i1}(x, y) \vee \cdots \vee njp_{il_i}(x, y)$

ここで，x と y は組変数である．各 c_i は，同一組変数を参照する述語の和でなければならない．x の制限式 $rc_i(x)$ の積を $rf(x)$ と記す．x と y の結合式 $rc_i(x, y)$ の積を $jf(x, y)$ と記すと，次の形式を有している．

$jf(x, y) \equiv pjp(x, y)[\wedge njf(x, y)] \mid njf(x, y)$

$njf(x, y)$ は，$pjp(x, y)$ ではない $jc_i(x, y)$ の積である．

正規化された問合せは，LCS 問合せグラフ (LQG) とよばれるグラフによって表せる．これは，問合せの記述と理解に役立つ．LQG は，問合せ内の組変数を表す節点集合 LN と，結合式を表す辺集合 LE とから成る．各節点 x は，情報として $(x, rf(x), tl(x))$ を有している．x は節点名で，対応する組変数名である．$rf(x)$ は x の制限式であり，$tl(x)$ は x の目標属性集合である．E と R 関係を表す節点を，おのおの E と R 節点とよぶ．

節点 x と y 間の辺は結合式を表している．結合式の種類に応じて，i) 主，ii) 主非等，iii) 非主の3種の辺がある．i) と ii) はともに主結合述語 $x.a\theta y.b$ を表し，i) は θ が $=$ で，ii) は $\neq$ のときである．iii) は非主結合式 $njf(x, y)$ を示している．

LQG は，各節点と辺の表す条件式の積を条件式とし，各節点 x の $tl(x)$ の和を目標属性集合とする LCS 問合せを表している．

例として，図3の LCS に対する問合せ「プロジェクトの開始以前に雇用されたメンバが，第一著者として書いた論文の主題と，プロジェクトの主題とが異なっている論文とそのプロジェクトの番号を求めよ」を考える．これは，QUEL で次のように記される．

```
RANGE (E, EMP) (P, PROJ)
      (R, REP);
RANGE (J, SUBJ);
RANGE (PJ, PROJ-SUBJ) (RJ, REP-SUBJ);
```

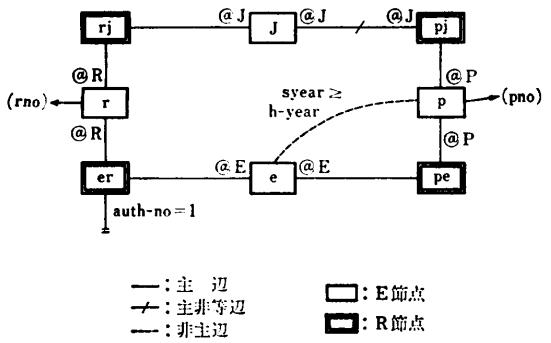


図 4 LQG
Fig. 4 LQG.

```
RANGE (PE, PROJ-EMP-LNK)
        (ER, EMP-REP-LNK);
RETRIEVE INTO ANTIO 10 (P.PNO, R.RNO)
WHERE
P.@P=PE.@P AND PE.@E=E.@E AND
E.@E=ER.@E AND
ER.@R=R.@R AND R.@R=RJ.@R AND
P.@P=PJ.@P AND PJ.@J NE J.@J AND
J.@J=RJ.@J AND
P.SYEAR GE E.H-YEAR AND
ER.AUTH-NO=1;
```

図 4 は，これの LQG である．

4. 問合せの構造変換

LCS 問合せから COBOL DML を生成するためには，以下の2点の相違を解決せねばならない．

(i) データモデルの相違 i.e. 関係モデルと CODASYL モデル

(ii) アクセス単位の相違 i.e. 集合単位の非手続的アクセスと，実現値単位の手続的アクセス

われわれのアプローチは，まず，LCS 問合せから，CODASYL モデルに基づいた非手続的問合せ (CODASYL 問合せとよぶ) を生成し，(i)を解決する．次に最適なアクセスを生成し，(ii)を解決する．前者を構造変換，後者をアクセスパス生成とよぶ．本章では，構造変換について論じる．

4.1 CODASYL 問合せ

CODASYL 問合せ (CQ と記す) は，QUEL と同様に記述される．

range $(r_1, R_1)\cdots(r_m, R_m)$;

retrieve into $R(a_1=ae_1, \cdots, a_k=ae_k)$

where qual;

R_i はレコード型で，r_i はその実現値変数である．r_i はその値として R_i 内の実現値をとる．R は結果関係名で，a_j はその属性である．CQ 内の変数は，$r.a$ (r は実現値変数，a はそのレコード型のデータ項目) と記され，値として実現値 r の a の値をとる．ae_j は，条件式 qual を満足する実現値 $r_1, \cdots, r_m$ 上の算術式で，結果属性 a_j の値を与える．各 ae_j 内の変数を目標項目とよぶ．

条件式は，制限と結合述語とから成る論理式である．ここで，$\theta \in \{<, \leq, =, \geq, >, \neq\}$, $\theta' \in \{=, \neq\}$ とする．

i) 制限述語 $rp(x) ::= ae(x)\,\theta\, v \mid ae(x)\,\theta\, ae(y)$

$ae(x)$ は，実現値変数 x を参照する算術式である．

ii) 結合述語

これは，問合せの構造を表す主結合述語と，たんに値の関連を表す非主結合述語とから成る．

ii-1) 主結合述語

$$pjp(x, y) ::= sjp(x, y) \mid nsp(x, y)$$

a) セット述語 $sjp(x, y) ::= S(x, y)$

S はセット型で，X と Y をおのおの S の親，子レコード型とする．このとき，$sjp(x, y)$ は次の意味をもつ．

$$S(x, y) \triangleq \begin{cases} \text{true if } \underline{\text{実現値 } x \text{ が，} y \text{ の } S \text{ についての親}} \\ \text{false } \underline{\text{otherwise}} \end{cases}$$

b) 非セット述語 $nsp(x, y) ::= x\,\theta'\,y$

これは x と y のとる実現値を，θ' で比較する．

ii-2) 非主結合述語 $njp(x, y) ::= ae(x)\,\theta\, ae(y)$

これは，通常の結合で，値の比較を行う．

問合せの条件式は，LCS 問合せと同様に積正規化される．ただし，CQ では，$jc_i(x, y)$ は次のようである．

$$jc_i(x, y) ::= sjp(x, y) \mid nsp(x, y) \mid njp_{i_1}(x, y) \vee \cdots \vee njp_{i_{l_i}}(x, y)$$

セット述語の否定は，次のように正規化される．

$$\sim S(x, y) ::= x \neq x' \wedge S(x', y) \mid S(x, y') \wedge y' \neq y \mid x \neq x' \wedge S(x', y') \wedge y' \neq y$$

ここで，x', y' は，おのおの x と y と同じレコード型に対する新たな実現値変数である．$S(x, y)$ に対するアクセスパスを CODASYL DBS はセット型としてサポートしているが，$\sim S(x, y)$ はサポートされていないから，上記のように正規化する．

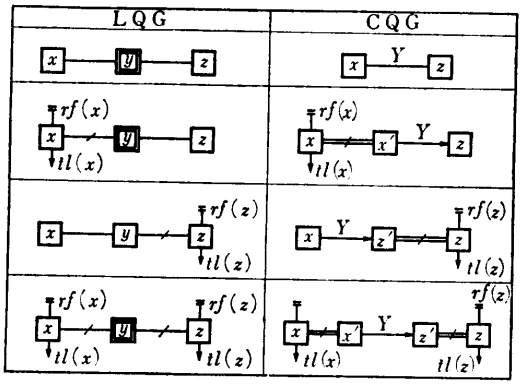
x の制限式 $rf(x)$ も LCS 問合せと同じである．x と y 間の結合式 $jf(x, y)$ は次の形式をもつ．

$$jf(x, y) ::= sjp(x, y) \wedge njf(x, y) \mid njf(x, y) \mid nsp(x, y) \wedge njf(x, y)$$

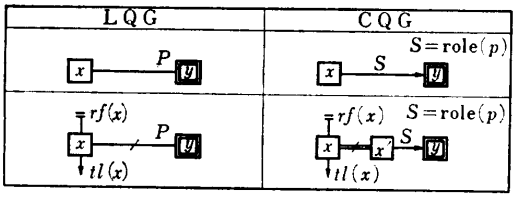
正規化された問合せは，CODASYL 問合せグラフ (CQG) とよばれるグラフによって表せる．CQG は，

表 1 LQG と CQG の対応
Table 1 The correspondence between LQG and CQG.

1) Y はセット型



2) Y はリンクレコード型



実現値変数を表す節点集合 CN と，結合式を表す辺集合 CE とから成る．CN 内の節点 x は，情報$(x, rf(x), tl(x))$ を有している．x は節点名で，対応する実現値変数名である．$rf(x)$ と $tl(x)$ とは，おのおの x の制限式と目標属性集合である．

節点 x と y 間の辺は，結合述語の種類に応じて，i）セット，ii）等，iii）非等，iv）非主の4種の辺から成る．おのおの，$S(x,y)$，$x=y$，$x \neq y$，$njf(x,y)$ を表す．

CQG は，節点と辺が表す条件式の積を条件式とし，各節点 x の $tl(x)$ の和を目標属性集合とする CODASYL 問合せ (CQ) を表している．

例として，図2のスキーマに対する問合せ「プロジェクト開始前に入社したメンバを求めよ」は，次のように記述される．

range (p, PROJ) (e, EMP)
$(pe, \text{PROJ-EMP-LNK})$;
retrieve into $R(e.\text{ename})$ where
$P\text{-}E(p, pe)$ and $E\text{-}P(e, pe)$ and $p.\text{syear} \geq e.\text{h-year}$;

4.2 構造変換

LCS 問合せから CODASYL 問合せ (CQ) への構造変換について論じる．x と z を LQG の E 節点，y を R 節点とする．a と d をおのおの x と z の主属性，b と c をおのおの x と z に対応する y の主属性とする．このとき，構造変換アルゴリズムは，以下のように表せる．

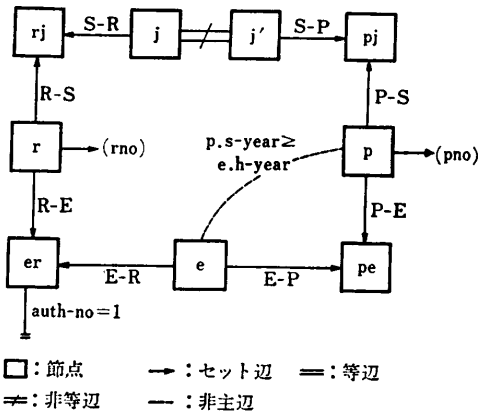


図 5 図4の CQG
Fig. 5 The CQG of Fig. 4.

（1） LQG 内のすべての E 節点 x から CQG 節点 x を作る．

（2） R 節点 y と関連する辺から，**表1**に基づいて CQG 辺を生成する．

（3） 目標属性，制限式，非主結合式内の属性を，対応するデータ項目に置換する．

LCS の E組とレコード実現値，また R 組と親子セット実現値/リンクレコード実現値とは，1対1対応している．したがって，このアルゴリズムによって生成された CQG と LQG との表す問合せの意味は等価である．

例として，図4の LQG は，**図5**の CQG に変換される．

5. アクセスパス生成

非手続的な CODASYL 問合せグラフ (CQG) から，実現値単位のアクセスを行うためのアクセスパス生成について論じる．

5.1 目　標

アクセスパスは次の目標を達成するように生成される[6]．

i） 中間結果数の最少化
ii） アクセスされる実現値数の最少化

CODASYL モデルでは，関係モデルと異なり，導出された結果を再度 DML によってアクセスできない．

CODASYL モデルが提供するアクセス機構を用いてアクセスを行うためには，一連の巡航によって結果を得る必要がある．巡航が断たれるとそこに中間結果を生成せねばならない．この数の増大は，CODASYL モデルとは独立な格納と操作のためのファイル管理負荷の増大をもたらす．以上より，なるべく巡航を断たずに，CODASYL モデルとは独立な中間結果の生成を減少させねばならない．

次に，よい応答性をもたらさねばならない．われわれは，データベースの全処理時間は，アクセスされる実現値数に比例すると仮定した．したがって，アクセス実現値数の最少化が必要になる．

5.2 DF アルゴリズム (DFA)

アクセスパスは，CQG を縦型サーチして得られる木（アクセス木 (AT)）として表される．木は DFA[6] と呼ばれるヒューリステクスによって生成される．x と y を CQG 節点，l を x と y 間の辺とする．acc (l, y) は，x の次の節点 y を l 経由でアクセスしたときの評価関数である (5.5.5 項)．最初すべての CQG 節点と辺は，F(ree) と印されているとする．

(1) F 印のすべての CQG 節点 x を調べて，acc$(-, x)$ が最少の x を見つける．$t_0 \leftarrow \Lambda$; push down (Λ, Λ) (if found); 見つからないときは，終了．

(2) x に対応した AT 節点 t を作る．$t_0 \neq \Lambda$ ならば，t_0 のいちばん右の子として t を結合する．

(3) x が F 印ならば，U(sed) と印し，$pt(x) \in t$;

(4) x が U 印ならば，$x, t, t'(=pt(x))$ を C 印．

(5) x が C 印ならば，t を C 印．

(6) CQG 内の F 印のセットリンクのなかで，acc(l, y) の最少の l と y を見つける．見つかったならば，l を C 印し，pushdown(x, t); $t_0 \leftarrow t$; $x \leftarrow y$; go to(2);

(7) 見つからないとき，popup(x, t); $x \neq \Lambda$ ならば，(2) へ．$x = \Lambda$ ならば，生成された AT に対して DML を生成する．go to (1);

生成された AT 枝と CQG辺とは1対1対応する．しかし，CQG がループを含むときには，ある CQG 節点 x に対して複数の AT 節点 $t_1, \cdots, t_n$ ($C(x)$ と記す) が存在し得る．$n \geq 2$ のとき，$C(x)$ 内の AT 節点を x の合流節点とよぶ．

5.3 アクセス木 (AT)

アクセス木 (AT) は，実現値変数を示す節点集合 AN と，セット辺を表す枝集合 AB とから成る．各節点 a は，対応する CQG 節点 (cn(a) と記す)，制限式 $rf(a)$，目標項目集合 $tl(a)$ とから成る．枝 $(a, b) \in AB$ は次のセット述語 $S^+(a, b)$ を表す．

$$S^+(a, b) \triangleq \begin{cases} S(a, b) \text{ if } a \text{ は } b \text{ の owner} \\ S(b, a) \text{ otherwise} \end{cases}$$

a の子節点の集合を child (a) と記す．また，AT 内のある二つの節点間に，CQ に対応して非主辺 $njf(a, b)$，非等辺 $a \neq b$ を示す辺が存在する．

節点 a とセット型 S_i^+ で結合された子節点を b_i とする．各節点 x のとる実現値を x' と記す．a の実現値 a' に，S_i^+ を介して結合された b_i の実現値集合を $S_i^+(a')$ と記す．a' が条件を満足するとは，次の述語 cond が真のときである．

$$\text{cond}(a') \triangleq \begin{cases} \text{true if } rf(a') \wedge \forall b_i \in \text{child}(a) \\ \quad \exists b_i' \in S_i^+(a') \text{ cond}(b_i') \\ \text{false otherwise} \end{cases}$$

AT 根節点を a とすると，a のある一つの実現値 a' が cond を満足すれば，問合せは解をもつことになる．

b_i がセット型 S_i の子のとき，これを N 節点，親のとき，O節点とよぶ．また，$S_i^+(a')$ 内の実現値は，セット型 S_i に基づいて，全順序づけられている．

CODASYL DML は，CALC 項目による直接アクセスと，順序集合 $S_i^+(a')$ 内の実現値をその順にたどる逐次アクセスとがある．ここで以下を定義する．

$S_i^+(a', k) \triangleq S_i^+(a')$ 内の k 番目の b_i 実現値

par $(b_i) \triangleq b_i$ の親節点 $(=a)$

sptr $(b_i) \triangleq b_i$ の右隣りの兄弟節点 $(=b_{i+1})$，なければ ϕ

cptr $(b_i) \triangleq b_i$ のいちばん左の子節点 $(=b_1)$，なければ ϕ

l-next$(b_i) \triangleq b_i$の先祖のなかで，最も近くにいる N 節点，なければ，TERM

このとき，各 AT 節点における実現値アクセスは，**図6** のようになる．図中で，frtrn, srtrn, next は次のようである．

frtrn $(b_i) \triangleq l$-next (b_i)

$$\text{srtrn}(b_i) \triangleq \begin{cases} \text{sptr}(b_i) \text{ if sptr}(b_i) \neq \phi \\ l\text{-next}(b_i) \text{ otherwise} \end{cases}$$

$$\text{next}(b_i) \triangleq \begin{cases} \text{cptr}(b_i) \text{ if cptr}(b_i) \neq \phi \\ l\text{-next}(b_i) \text{ otherwise} \end{cases} \quad (b_i \text{ は葉節点})$$

図7 は，図5の節点 e を根としたときの AT である．

5.4 合流節点

CQG 節点 x の合流節点を $c(x) = \{t_1, \cdots, t_n\}$ とす

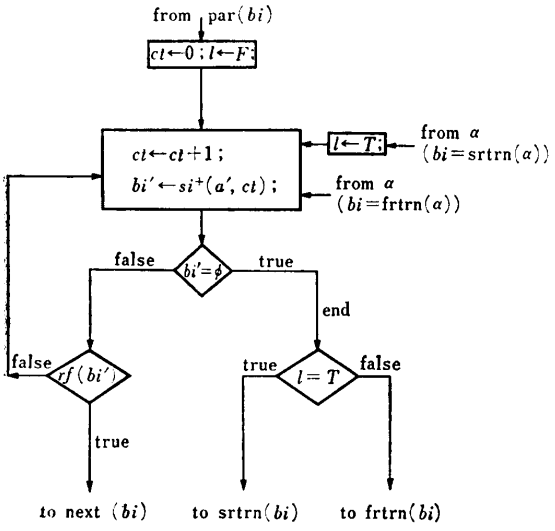


図 6 AT 節点 b_i の DML
Fig. 6 The DML program of b_i.

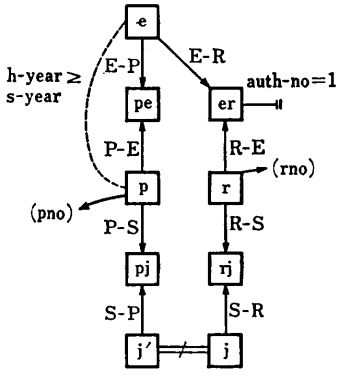


図 7 図5のアクセス木
Fig. 7 The access tree of Fig. 5.

る．AT の根を a とし，a' を cond を満足する実現値とする．この a' に対して，各 t_i の条件を満足する実現値集合を T_i とする．$t_1, \cdots, t_n$ が x の合流節点であることは，各 T_i が同一でなければならないことを意味している．われわれの DFA によって生成された AT では，$c(x)$ 内のある節点を t_i とすると，t_i を根節点とする部分木内に $c(x)$ のすべての節点が含まれる．この性質は重要である．t_i の実現値 t_i' から図 6 によってアクセスされる $c(x)$ 内のすべての $t_j(\neq t_i)$ の各実現値 t_j' に対して，t_i' との同一性の判定を加えるだけで，合流節点の条件を調べられる．もし AT がこの性質を有しなければ，T_i と T_j の格納のための中間ファイルと，互いの共通部分を求める集合演算とが必要になってしまう．したがって，われわれの DFA では，中間結果を必要としないアクセスパスを

表 2 $rf(a)$ の選択度
Table 2 The selectivity of $rf(a)$

$rf(a)$	選択度 $st(rf(a))$
$r.a=v$	card(a)/card(r)(=St_{ra} と記す)
$r.a\neq v$	$1-St_{ra}$
$r.a\left\{\begin{matrix}>\\<\end{matrix}\right\}v$	$(1-St_{ra})/2$
$r.a\left\{\begin{matrix}\geq\\\leq\end{matrix}\right\}v$	$(1+St_{ra})/2$
$r.a=v_1\vee\cdots\vee r.a=v_n$	$n\cdot St_{ra}$
$rf_1\wedge\cdots\wedge rf_n$	$St_{f1}\cdot St_{f2}\cdots\cdots St_{fn}$ ここで $St_{fi}=rf_1$ の選択度
$rf_1\vee\cdots\vee rf_n$	$1-(1-st_{f1})\cdots\cdots(1-St_{fn})$

(rf_i は互いに異なった属性を参照している)

生成できる．

5.5 アクセスコスト

アクセスパスのコストについて考える．

5.5.1 選 択 度

実現値変数 x の制限式 $rf(x)$ の選択度を st_x と記す．これは，存在する x 実現値のなかで，条件 $rf(x)$ を満足するものの割合の期待値である[12]．**表 2** は，$rf(x)$ のパターンと選択度を示している．不等述語については，データベースごとに経験的に選択度を定めることが有効と考えている．

5.5.2 結 合 度

セット型 $S(A, B)$ に対して，以下を定義する．

$$\mathrm{cnt}(S)\triangleq A \text{ に対する } B \text{ の結合度}$$
$$\triangleq\sum\nolimits_{\forall a\in A}|\{b|(a,b)\in S\}|/|A|$$
$$\mathrm{icnt}(S)\triangleq B \text{ に対する } A \text{ の結合度}$$
$$\triangleq\sum\nolimits_{\forall b\in B}|\{a|(a,b)\in S\}|/|B|$$
$$=\begin{cases}1 & \underline{\text{if}}\ S \text{ のメンバシップクラスが Mandatory/Automatic (M/A)}\\ \leq 1 & \underline{\text{otherwise}}\end{cases}$$

cnt (S) は，一つの親実現値 a に，平均何個の子 b が結合されているかを示している．icnt (S) は，子実現値 b が S の親をもつ確率を与えている．

5.5.3 セット型アクセス

セット型 $S(A, B)$ で，B は制限式 rf_B を有しているとする．まず，A から S 経由で B をアクセスするとする．A の実現値 a に結合された S の子 B の実現値集合を $S(a)$ と記す．$S(a)$ のなかから，条件 rf_B を満足するすべての実現値を見つけるためにアクセスされねばならない平均実現値数を acn (S) とする．acn (S) は，rf_B の形式によって，**表 3** のようになる．たとえば，条件 $B.a\leq 5$，a が S の昇順ソート項目のとき (2)，next ポインタをたどって，$B.a>5$ となった時点でアクセスを打ち切ってよい．このとき，acn (S) は，結合度 cnt (S) の半分となる．この

表 3 主制限式のパターン
Table 3 The primary restrictions.

$rf(B)=prf(s)\wedge nrf(s)$

主制限式 $prf(s)$	条件	acn(S)
1) $B.a=v$	a=S-item(S)∧ $a\neq$D-item(S)	cnt(S)・(1+St$_{Ba}$)/2 St$_{Ba}$ は $prf(S)$ の選択度
2) $B.a\theta v$	S-type(S)=ST∨ (S-type(S)=ST′∧ $p\in$ptr(S))	cnt(S)/2
3) $B.a=v$	a=D-item(S) $a\neq$S-item(S)	cnt(S)/2
4) なし		cnt(S)

θ	ST	ST′
<, ≤	D	A
>, ≥	A	D

ように，アクセスする子実現値数を減少できる制限述語を主制限とよび，$prps$ と記す．

次に，BからAへのアクセスを考える．iacn(S) を B 内の実現値 b からその親 a を見つけるためにアクセスされる B の平均実現値数とすると以下のようになる．

$$\text{iacn}(S)\triangleq\begin{cases}1 & \underline{\text{if}}\ o\in\text{ptr}(S)\ \text{(owner ポインタを有している)}.\\ \text{cnt}(S)/2 & \underline{\text{otherwise}}\end{cases}$$

owner ポインタを有していれば，ただちに親に達せられるが，なければ子をたどって親に達せねばならない．

5.5.4 アクセス木 (AT) のアクセスコスト

a を根節点とするアクセス木 T を考える [図 8]．$b_1,\cdots,b_n$ は a の n 個の子節点でこの順に結合されている．各 b_i は部分木 T_i の根であり，a とはセット型 S_i^+ によって結合されている．a は選択度 sta の制限式をもつ．NOC(T) を，一つの a 実現値 a' からアクセスされる T 内の全実現値数の期待値とする．これは次のように再帰的に定義される．

$$\begin{aligned}\text{NOC}(T)\triangleq 1+&st_a\cdot(ct_{ab_1}\cdot\text{NOC}(T_1)\\&+tst_{T_1}\cdot(ct_{ab_2}\cdot\text{NOC}(T_2)+\cdots\\&+tst_{T_{n-1}}\cdot(ct_{ab_n}\cdot\text{NOC}(T_n))\cdots))\end{aligned}$$

ここで，

$$\text{tst}_T\triangleq\begin{cases}\text{st}_a\cdot\prod_{i=1}^{n}\text{tst}_{T_i} & \underline{\text{if}}\ \text{child}(a)\neq\phi\\ \text{st}_a & \underline{\text{otherwise}}\end{cases}$$

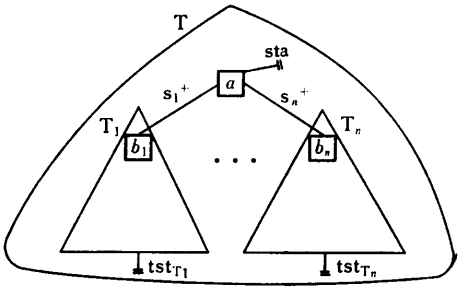


図 8 アクセス木 T
Fig. 8 The access tree T.

$$\text{ct}_{ab_i}\triangleq\begin{cases}\text{acn}(S_i) & \underline{\text{if}}\ b_i\ \text{は}\ a\ \text{の子}\\ \text{iacn}(S_i) & \underline{\text{otherwise}}.\end{cases}$$

$$\text{st}_a\triangleq\begin{cases}1/|A| & \underline{\text{if}}\ a\ \text{は，2番目以降の合流節点}\\ rf(a)\ \text{の選択度} & \underline{\text{otherwise}}\ \text{[表 2]}.\end{cases}$$

tst$_T$ を，木 T の選択度とよぶ．これは，根 a の実現値 a' が条件を満足する (cond(a')=true) 確率である．

$C(T)$ を T 内でアクセスされる全実現値の期待数とする．T の根節点 a へのアクセス方法としては，i)CALC 項目による直接アクセス，ii) 単項セット型 S による逐次アクセス，iii) 領域内の逐次アクセスの3種がある．

i) のとき，a の制限式は次のようであるとする．$rf(a)=:prf(a)\wedge nrf(a)$，$prf(a)=:a.c=v_1\vee\cdots\vee a.c=v_n$．$c$ は a の CALC 項目である．$prf(a)$ と $nrf(a)$ の選択度を，おのおの pst$_a$, nst$_a$ とする．$|A|$ を，レコード型 A のカーディナリティとすると，a 内のアクセスされる実現値数は，$|A|\cdot\text{pst}_a$ となる．ii) では，ctSYSTEM $_a$ となる．iii) では，a が全面サーチされるので，$|A|$ 個の実現値がアクセスされる．

$$C(T)\triangleq\begin{cases}\text{i)}\ \text{pst}_a\cdot|A|\cdot(1+\text{nst}_a\cdot(\text{ct}_{ab_1}\cdot\text{NOC}(T_1)\\ \quad+\text{tst}_{T_1}\cdot(\text{ct}_{ab_2}\cdot\text{NOC}(T_2)+\cdots\\ \quad+\text{tst}_{T_{n-1}}\cdot(\text{ct}_{ab_n}\cdot\text{NOC}(T_n))\cdots))\\ \text{ii)}\ \text{ctSYSTEM}_a\cdot\text{NOC}(T)\\ \text{iii)}\ |A|\cdot\text{NOC}(T)\end{cases}$$

5.5.5 評価関数

5.2 節で述べた，CQG 節点 x に対する評価関数 acc(l,y) について考える．acc としては1段階のサーチを行う acc 1 と，2段階までの acc 2 とがあり，次のように定義される．

i) acc 1 $(l,y)\triangleq\text{ct}_{xy}$.

ii) acc 2 $(l,y)\triangleq\text{ct}_{xy}\cdot(1+\text{st}_y\cdot\min_{\text{すべての印のない}y\text{の隣接ノード}z}\text{ct}_{yz})$.

ii) を用いるとき，子節点のない y が優先的に選ば

れる（$\because ct_{y'x}=0$）のを防ぐために，次の方法を用いる．

（1） acc 1 (x, y') が最少な y' を選ぶ．

（2） y' が子節点をもつ（i.e. $ct_{y'x}\neq 0$）ならば，子をもつすべての y のなかから acc 2 (l, y) が最小の y を選ぶ．

（3） y' が子をもたねば，これを選ぶ．

現在までの実験[13]では，サーチレベルよりもむしろ選択度，結合度といった統計情報の，実際のデータベース・アクセスに対する適合性が問題となっている．統計情報の確かさを高めるために，結合度については，各実行ごとに，実際にアクセスした結果による動的なチューニングを行っている．

6. DML 生成

COBOL DML は，アクセス木（AT）の枝 (S_i^+) と子節点 (b_i) を前置順にたどりながら，$(S_i^+, b_i, rf(b_i), tl(b_i), \mathrm{srtrn}(b_i), \mathrm{frtrn}(b_i))$ をパラメータとして生成される．詳細は，文献 14) を参照されたい．

CODASYL モデルでは，セット型，レコード型，領域，実行単位毎の現在子に基づいて巡航的にアクセスがなされる．したがって，以下の AT 節点では，これへのアクセスの前後に現在子の復帰と退避が必要になる．

（i） 同一レコード型が，異なった実現値変数をもつとき，各変数に対応した AT 節点

（ii） 合流節点

同一レコード型が，一つのパス内で複数回異なったセット実現値についてアクセスされるので，その都度，現在子が保持される必要がある．また，非主辺と非等辺を処理するための DML も同時に生成される．

7. LDP-V 1.5 の実現

LDP-V 1.5 は，現在当協会 M-170 F の AIM と Acos-700 の ADBS 上に実現されている．LDP は，PL/I で記述され，ソース文数約 15,000，オブジェクト約 200 KB である．

LDP のシステム構成を**図 9** に示す．LDP で生成された COBOL プログラムを，翻訳/リンクし，DBMS 上で実行させるために，LDP と OS とのプロセス間通信が必要になる．このために，われわれの開発した JIPNET の ITP (対話プロトコル) を用いて，必要な JCL をリモートバッチシステム (RES) に送り，COBOL プログラムを実行させている．結果も同様に

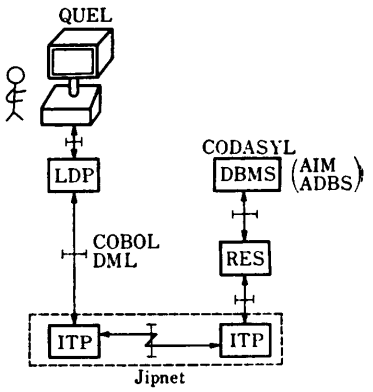


図 9 LDP-V 1.5 の構成
Fig. 9 LDP-V 1.5.

```
READY
EXEC MMUI.CLIST(LDP17G0)

*** JDDBS LDP-V1.5 START ...10.05.13.361

#00010 RANGE (E, EMPLOYEE)(P, PROJECT)(R, REPORTR);
#00020 RANGE (PJ, PROJ-SUBJ)(RJ, REP-SUBJ);
#00030 RANGE (PE, PROJ-EMP-LNK)(ER,EMP-REP-LNK);
#00040 RANGE (J, SUBJECT);
#00050 RETRIEVE INTO R ( P.PNO, R.RNO) WHERE
#00060 P.BP=PE.BP  AND  PE.BE=E.BE  AND
#00070 BBBBBBE.BE=ER.BE AND ER.BR=R.BR AND
#00080 R.BBBBR=RJ.BR    AND RJ.BJ NE PJ.BJ AND
#00090 PJ.BP=P.BP  AND ER.BAUTH-NO=1;
#00100 GO;
GO.............10.11.58.609

THE ELAPSE TIME FOR RETRVP OM.. 17797
THE ELAPSE TIME FOR RQGP.......13735
THE ELAPSE TIME FOR DQGP.......  642
THE ELAPSE TIME FOR DFA ....... 1314
THE ELAPSE TIME FOR DMLG.......20860


-- THE EST. NUMB. OF ACCESSED OCCURRENCES    656.260
-- THE EST. TOTAL SELECTIVITY                  0.200

***  JIPNET START...10.15.48.828

( PNO ((RNO)))
C858   ITOT 75
REKI81 HAMAE78A
       HAMAE78B
       HAMAE79
       HAMAE80
       JDDBS78
       JDDBS79
MMUI80 JDDBS80A
       TAKIM78
       TAKIM79A
       TAKIM79B
       TAKIM79C
       TAKIM79D
       TAKIM80A
       TAKIM80B
       TAKIM80C
       TAKIM80D
         :
```

図 10 LDP-V 1.5 の使用例
Fig. 10 Example of LDP-V 1.5 usage.

ITP 経由で LDP に返される．

図 10 は，LDP を通しての使用例である．他に，LDP はビュー定義と検索機能を有している．

8. むすび

本論文では，CODASYL DBS に対する関係イン

タフェース LDP-V 1.5 におけるスキーマ変換と，問合せの変換方法，およびその実現について述べた．本システムの実現によって，手続的DBS に対して，非手続インタフェースの提供を可能とすることができた．現在，DDBS の各サイトの共通インタフェース化，COBOL DML プログラムの作成ツール化，レポート生成処理と更新機能の組み込みを行っている．LDP-V 1.5 の性能評価は，別に論じる[13]．

謝辞 JIPNET との結合に協力いただいた当協会の長沢義昭氏，データベース保守等に助力をいただいた当プロジェクトの塚本元子氏，また日頃ご指導をいただいている当協会開発部次長の小関重美氏に感謝します．

参考文献

1) CODASYL DBTG: *CODASYL Data Description Language Journal of Development*, (1973).
2) Smith, J.M. et al.: Multibase-Integrating Heterogeneous Distributed Database Systems, *AFIPS Conf. Proc.*, Vol. 50, pp. 487-499 (1981).
3) Moore, R.C.: Handling Complex Queries in a Distributed Database, *SRI Technical Note*, 170, (1979).
4) Takizawa, M., Hamanaka, E. and Ito, T.: Resource Integration and Data Sharing on Heterogeneous Resource Sharing System, Proc. of the ICCC'78, pp. 253-258 (1978).
5) Takizawa, M. and Hamanaka, E.: The Four-Schema Concept as the Gross Architecture of Distributed Databases and Heterogeneity Problems, *JIP*, Vol. 2, No. 3, pp. 134-142 (1979).
6) Takizawa, M. and Hamanaka, E.: Query Translation in Distributed Databases, Proc. IFIP'80, pp. 451-456 (1980).
7) Takizawa, M.: Distribution Problems in Distributed Databases—Integration and Query Decomposition, to appear in *JIP* (1982).
8) 滝沢 誠：分散型データベースシステム JDDBS-II と通信処理，信学会 AL 81-22 (1981).
9) Zaniolo, C.: Design of Relational Views over Network Schemas, Proc. ACM SIGMOD, pp. 179-190 (1979).
10) Wong, E. and Katz, R, H.: Logical Design and Schema Conversion for Relational and DBTG Databases, Proc. E-R Conf., pp. 311-321 (1979).
11) Held, G. et al.: INGRES—A Relational Data Base System, AFIPS Conf. Proc., pp. 409-416 (1976).
12) Hevner, A.R. and Yao, S.B.: Query Processing on a Distributed Database, Proc. Berkeley Workshop, pp. 91-107 (1978).
13) 横塚 実，滝沢 誠，鈴木 信：LDP-V 1.5 の評価，JIPDEC TR 81/10 (1981).
14) JIPDEC：マンマシンユーザインタフェースの調査研究，JIPDEC 55-S-002 (1981).

付記 用語，記号定義

(1) CODASYL スキーマ

$S(A, B)$ ：親レコード型 A，子レコード型 B とするセット型 S

s-item(S) ：S のソートキー項目

s-type(S) ：S のソートの順 ($\in \{A$(昇)，D(降)$\}$)

D-item(S) ：S の DNA 項目

ptr(S) ：S を実現しているポインタの種類 $\in \{\{n\}\ \{n,p\}\ \{n,o\}\ \{n,p,o\}\}$

ここで n：NEXT ポインタ
p：PRIOR ポインタ
o：OWNER ポインタ

$R(t_1, \cdots, t_n)$：データ項目 $t_1, \cdots, t_n$ から成るレコード型 R

(2) ローカル概念スキーマ (LCS)

@R ：主属性．関係の属性で，値として db キーをとる属性

E関係：主属性を一つだけもつ関係，レコード型に対応する

R関係：主属性を複数有し，各主属性はあるE関係の主属性である関係．セット型とリンクレコード型に対応する

ER：関係

patt (ER)：ER の主属性の集合

pkey (ER)：ER の主キー

(3) LCS 問合せ

主結合：主属性間の結合で，＝とキのみが許される．

非主結合：主結合でない結合

LQG ：LCS 問合せグラフ

(4) CODASYL 問合せ

実現値変数：レコード実現値を値としてとる変数

セット述語 $S(x, y)$：実現値変数 x と y のとる実現値がおのおの S の親と子関係にあるとき真となる述語

CQG：CODASYL 問合せグラフ

(5) アクセスパス生成

AT：アクセス木，CQG から生成されるアクセスパスを表す木

$S^+(a,b)$：{セット述語 $S(a,b)$ if a は b の親 / $S(b,a)$ otherwise

合流節点：一つの CQG 節点 x に対して，複数の AT 節点 $(t_1,\cdots,t_n)$ が存在するとき，これらの AT 節点を合流節点 $c(x)$ とよぶ

選択度：実現値変数 x の制限式 $rf(x)$ を満足する実現値の割合

結合度：セット型によって結合される実現値数の平均値

T：アクセス木

NOC(T)：T の根節点 a の一つの実現値からアクセスされる T 内の平均実現値数

C(T)：T 内でアクセスされる全実現値数

(昭和 56 年 11 月 9 日受付)
(昭和 57 年 5 月 19 日採録)