再帰有向超グラフに基づくデータモデル

宝珍輝尚† 都司達夫†

本論文では，マルチメディアデータの内容表現に使用することを目的として，有向グラフ，再帰グラフ，ならびに，超グラフの概念を導入したデータモデルを提案する．提案するデータモデルでは，データ実体を再帰有向超グラフとして表現する．このデータ実体を表現するグラフを実体グラフと呼ぶ．また，実体グラフの集まりを集積グラフとして扱う．さらに，集積グラフの構造を表現するシェイプグラフを導入する．演算はグラフの書き換えによるものであり，再帰的な問合せやパス上の正規表現による問合せを可能としている．本論文では，概説に続いて定義を示し，その後，実体グラフの枝の始終要素の深さを利用すると実体グラフを分割して表現できるか決定できることを示す．また，書き換え演算は複合値を扱うように拡張した datalog プログラムで記述できることを明らかにする．

A Data Model Based on Directed Recursive Hypergraphs

TERUHISA HOCHIN† and TATSUO TSUJI†

This paper proposes a data model incorporating the concepts of directed graphs, recursive graphs, and hypergraphs in order to represent the contents of multimedia data. In the proposed data model, an instance is represented with a directed recursive hypergraph. This graph is called an *instance graph*. A collection of instance graphs is managed as a graph named a *collection graph*. A *shape graph*, which represents the structure of a collection graph, is also introduced. An operation rewriting collection graphs is introduced to manipulate the collection graphs. This operation enables users to make recursive queries, and specify regular expressions on paths. This paper presents an illustrative example, and the formal definition of the proposed data model. It is clarified that whether the instance graph may be divided can be decided by using the depth of the initial and/or terminal elements of the edges of an instance graph. Moreover, the operation can be converted into the datalog program extended to treating complex values.

1. はじめに

近年，マルチメディアデータのデータベース (DB) での取り扱いが盛んに研究されている．マルチメディアデータの内容検索もこの中の一つである．マルチメディアデータの内容検索では，特徴量を用いて内容検索を行う方法[1),2)]もあるが，マルチメディアデータの内容をグラフを用いて表現しておき検索に役立てようという研究[3),4)]もある．CT 画像中の部位とその間の距離等をラベル付き有向グラフで表現する研究[3)]やビデオのシーンの内容を意味ネットワークで表現する研究[4)]がその例である．これらの研究ではラベル付き有向グラフが良く使用されるが，知識表現の一つである概念グラフでは，グラフのノードが再帰的にグラフに成り得るという再帰グラフであることが必要であり[5)]，また，ルールの表現に，n 個のノードの集まりを枝ととらえられる超グラフを用いる研究[6),7)]も行われてきている．従って，近い将来，マルチメディアデータの内容表現においても，単なる有向グラフではなく再帰グラフと超グラフの性質を持つグラフが必要となると考えられる．

ここで，グラフが再帰グラフや超グラフでなく単なる有向グラフの場合，グラフはオブジェクト指向データモデルやグラフに基づくデータモデル[8)~20)]で自然に表現することができる．しかし，当然ではあるが，再帰グラフや超グラフの性質を持つグラフは，これらのデータモデルでは自然に表現できない．

ハイパーノードモデル[21)]や Hy^{+}[22)]では再帰グラフの概念を導入し，再帰的な問合せやパス上の正規表現を用いた問合せの機能を提案している．この問合せ機能は非常に強力なものであるが，超グラフの表現能力を必要とする場合には自然にデータを表現できない．

一方，有向再帰ラベルノード超グラフモデル[23)]や

† 福井大学工学部
Faculty of Engineering, Fukui University

自己組織化意味関連モデル[24]では再帰グラフのみでなく超グラフの概念を導入している．これらのモデルの表現能力はマルチメディアデータの内容表現に十分であると考えられる．しかしながら，演算はグラフの構造の書き換えが主であり，再帰的な問合せやパス上の正規表現を用いた問合せを簡単に行うことができない．

そこで本論文では，マルチメディアデータの内容表現といった分野のデータを表現するのに十分な表現能力を持ち，かつ，再帰的な問合せやパス上の正規表現を用いた問合せを簡単に行うことを目的として，再帰有向超グラフに基づくデータモデルを提案する．提案するデータモデルは，再帰有向超グラフデータモデルと呼ぶもので，有向グラフ，再帰グラフ，ならびに，超グラフの概念を導入している．データを表現する基本単位は実体グラフと呼ぶグラフである．また，実体グラフの集まりを扱うものとして集積グラフを導入する．集積グラフは実体グラフを成分とするグラフである．さらに，集積グラフの構造を表現するシェイプグラフを導入する．操作は，書き換え演算 *rewrite* を用いて行う．この演算は集積グラフの書き換えを行う．この演算を用いることで再帰的な問合せを行うことができる．また，パス上の正規表現を記述できるように考慮している．

以下，2. では，例を用いて再帰有向超グラフデータモデルを説明し，3. で，本データモデルの定義を示す．そして，4. で，提案モデルの下でのデータの性質と問合せ能力について考察する．5. で関連する研究について言及し，最後に，6. でまとめる．

2. 概　　説

再帰有向超グラフデータモデルにおいて，データを表現する基本的な単位は実体グラフである．実体グラフには原始実体グラフと標準実体グラフがある．原始実体グラフは点である．標準実体グラフは再帰有向超グラフである．実体グラフは，識別子，名前，ならびに，データ値から構成されるラベルを持つ．

例 1 蝶が花にとまっている写真（図 1(a)）の内容表現を考えよう．写真には，蝶の前羽根と後羽根，胴，ならびに，5 本の足（前足 2 本，中足 1 本，後足 2 本）が写っている．蝶の 2 本の前足と 2 本の後足は，おのおの，別の花に接している．図 1(b) は，この写真の内容を実体グラフで表現した例である．図 1(b) では，原始実体グラフを小円で表し，標準実体グラフを俵型で表している．例えば，$n111$ や $n112$ は原子実体グラフである．$g1$，$g11$，ならびに，$g12$ は標準実体グラフである．また，枝は，始要素集合を破線で囲み，終要素集合を点線で囲み，この破線と点線をつなぐことによって表す．例えば，原子実体グラフ $n111$ と $n112$ は，枝 $e11$ により $n113$ と接続している．また，$g1$ は，$g11$, $g12$, $e13$, $e14$ を含み，$g11$ は，$n111$, $n112$, $e11$ 等を含む．

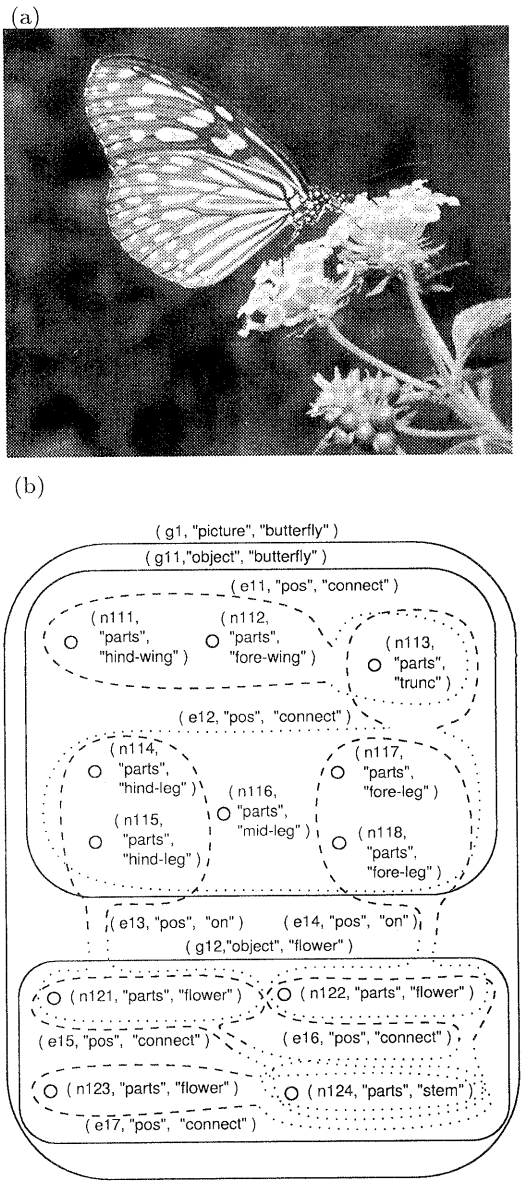


図 1　(a) 写真と (b) その内容を表現する実体グラフ
Fig. 1　(a) a picture and (b) an instance graph representing its contents

同じ構造を持つ実体グラフの集まりを集積グラフとしてとらえる．集積グラフは実体グラフを成分とするグラフである．

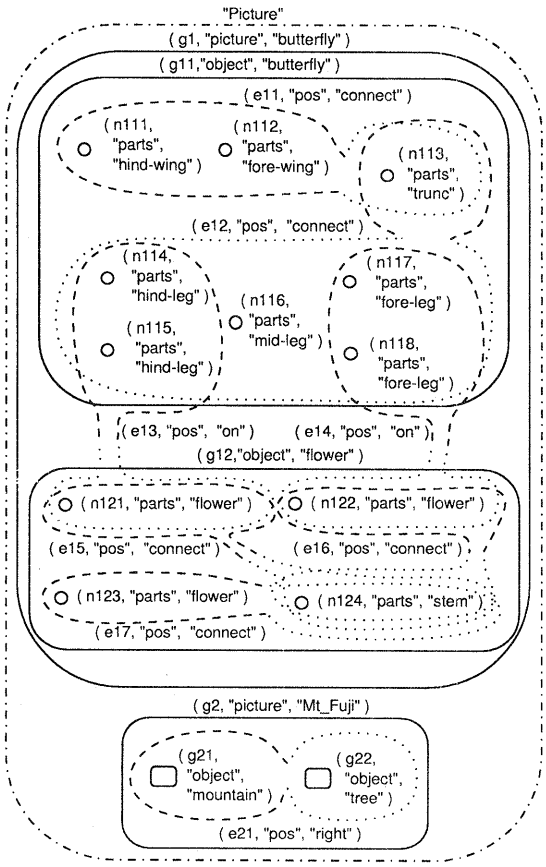


図 2 集積グラフの例
Fig. 2 An example of a collection graph

例 2 集積グラフの例を図 2 に示す．図 2 では，集積グラフを一点鎖線で表している．集積グラフは名前を持ち，その名前はデータベース内で一意である．図 2 に示した集積グラフの名前は Picture である．この集積グラフは 2 つの実体グラフを持っている．実体グラフ $g1$ は図 1(b) に示したものであり，$g2$ は他の写真のものである．これらの実体グラフを代表実体グラフと呼ぶ．

集積グラフの構造は，シェイプグラフと呼ぶグラフによって表現される．

例 3 図 3 は，図 2 に示した集積グラフ Picture のシェイプグラフである．実体グラフ picture は，標準実体グラフ object を含む．object は，枝 pos によって object と接続する．また，object は，原子実体グラフ parts を含み，parts も枝 pos によって parts と接続する．

シェイプグラフは硬シェイプ [26] の特性を持つ．すなわち，シェイプグラフは集積グラフの生成に先立って存在している必要はない．もちろん，生成に先立って存在しても構わない．集積グラフが存在するときは対応するシェイプグラフが存在しなければならない．実体グラフの生成，変更によってシェイプグラフが変更されることがある．ただし，一度生成されたシェイプグラフならびにその要素はユーザが陽に消去するまで消去されない．

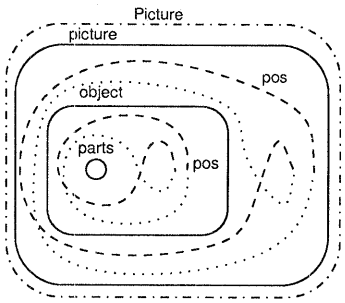


図 3 シェイプグラフ
Fig. 3 A shape graph

次に，演算について述べる．導入する演算は書き換え演算 *rewrite* であり，集積グラフを書き換える．本演算は，問合せならびに更新（挿入，削除，変更）の両方で使用する．本演算を問合せに使用する場合，問合せ結果の構造を表現する問合せグラフと問合せの条件を記述する問合せグラフを記述する．問合せグラフの構造は集積グラフの構造と同様である．問合せグラフのラベルは，識別子に対する変数，名前に対する変数，ならびに，値に対する変数の 3 つ組である．書き換え演算の結果は集積グラフである．

例 4 書き換え演算の例を図 4 に示す．集積グラフ Picture の中の標準実体グラフ picture から，標準実体グラフ object に含まれる原子実体グラフ parts が on という値を持つ枝 pos によって連結されているものを求め，集積グラフ My-picture としている．書き換え演算 rewrite の第一引数は問合せ結果の構造を表す問合せグラフである．第二引数は検索条件を指定する問合せグラフである．検索条件を満足する実体グラフが集積グラフ My-picture の実体グラフとなる．問合せグラフ中の X, Z1 といった変数は検索条件式中で使用され得る．図 4 では，例えば，ラベル X に対して，識別子に対する変数，名前に対する変数，ならびに，値に対する変数を，おのおの，X_{id}, X_{name}, X_{val} と表記している．

問合せグラフでは正規表現を使用することができる．

例 5 正規表現を含む書き換え演算の例を図 5 に示す．集積グラフ Picture の中の標準実体グラフ picture から，on という値を持つ 2 個以上の枝 pos によって連結されている原子実体グラフ parts を持つ標準実

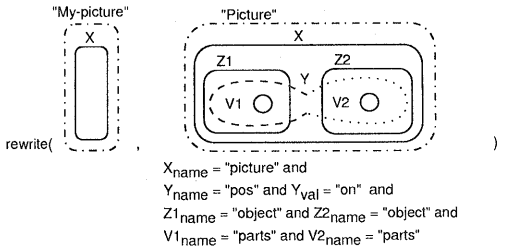


図 4 書き換え演算の例
Fig. 4 An example of the rewrite operation.

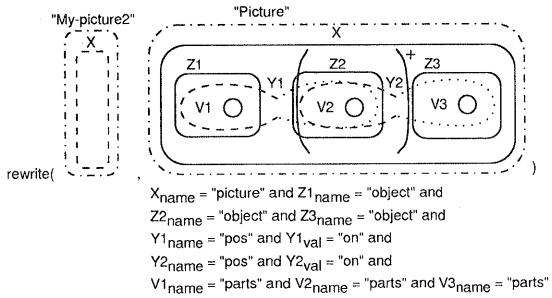


図 5 正規表現の記述例
Fig. 5 An example of specifying regular expression.

体グラフ picture を求め，集積グラフ My-picture2 としている．括弧の右肩に付けられた + は括弧内が1回以上繰り返されることを示す．

次に，書き換え演算を用いた実体グラフの挿入の例を示す．

例 6 実体グラフの挿入の例を図 6 に示す．この図では，実体グラフ picture が集積グラフ Picture に挿入されている．名前や値は変数を用いて指定している．

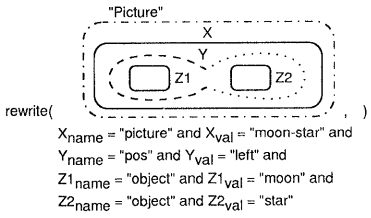


図 6 実体グラフの挿入
Fig. 6 Insertion of an instance graph.

実体グラフが挿入されると，システムが自動的に実体グラフに対する識別子を生成し付与する．書き換え演算の第一引数の問合せグラフ中の識別子に対する変数が既存の実体グラフを特定するために使用されている場合は，その実体グラフの変更を意味する．

3. 定　　義

ここでは，再帰有向超グラフデータモデルの定義を行う．まず，構造に関する定義を行い，次に，演算に関する定義を行う．

3.1 構　　造

3.1.1 実体グラフ，集積グラフ，シェイプグラフ

定義 1 実体グラフは以下のように定義される．

- 実体グラフには原子実体グラフと標準実体グラフがある．実体グラフはラベルを持つ．ラベルは3つ組 (d_{ID}, nm, d) で表される．ここで，d_{ID} は識別子，nm は名前，d はデータ型とデータ値の組である．
- 原子実体グラフは点である．
- 標準実体グラフ g は8つ組 $(V, E, L_v, L_e, \phi_v, \phi_e, \phi_{conn}, \phi_{comp})$ である．ここで，V は g に包含される実体グラフの集合，E は枝の集合，L_v は実体グラフのラベル集合，L_e は枝のラベル集合，ϕ_v は実体グラフ集合から実体グラフのラベル集合への写像：$V \rightarrow L_v$，ϕ_e は枝集合から枝のラベル集合への写像：$E \rightarrow L_e$，ϕ_{conn} は接続関係を表す写像：$2^V \times 2^V \rightarrow E$，$\phi_{comp}$ は構成関係を表す写像：$2^{V \cup E} \rightarrow V$ である．□

以降では，表 1 に示す表記を使用する．

次に，実体グラフに関する等価性を示す．

定義 2 2つのラベル l_1 と l_2 は，$id(l_1) = id(l_2)$ のとき同一である（$l_1 =_{id} l_2$）という．また，$val(l_1) = val(l_2)$ のとき値等価である（$l_1 =_{val} l_2$）といい，$nm(l_1) = nm(l_2)$ かつ $val(l_1) = val(l_2)$ のとき，等価である（$l_1 = l_2$）という．

2つのラベル集合 L_1 と L_2 が，$^{\forall}l_1 \in L_1\, ^{\exists}l_2 \in L_2(\ l_1 =_{id} l_2\ )$ かつ $^{\forall}l_2 \in L_2\, ^{\exists}l_1 \in L_1(\ l_2 =_{id} l_1\ )$ のとき，同一である（$L_1 =_{id} L_2$）という．また，$^{\forall}l_1 \in L_1\, ^{\exists}l_2 \in L_2(\ l_1 =_{val} l_2\ )$ かつ $^{\forall}l_2 \in L_2\, ^{\exists}l_1 \in L_1(\ l_2 =_{val} l_1\ )$ のとき，値等価である（$L_1 =_{val} L_2$）といい，$^{\forall}l_1 \in L_1\, ^{\exists}l_2 \in L_2(\ l_1 = l_2\ )$ かつ $^{\forall}l_2 \in L_2\, ^{\exists}l_1 \in$

表 1 表記
Table 1 Notations

表記	意味
$i(e)$	枝 e の始要素集合
$t(e)$	枝 e の終要素集合
$V(g)$	実体グラフ g と g に含まれる実体グラフの集合
$E(g)$	実体グラフ g に含まれる枝の集合
$id(l)$	ラベル l の識別子部分
$id(L)$	$\{\ id(l)\ \mid l \in L\ \}$，$L$ はラベルの集合
$nm(l)$	ラベル l の名前部分
$nm(L)$	$\{\ nm(l)\ \mid l \in L\ \}$，$L$ はラベルの集合
$val(l)$	ラベル l のデータ部分
$val(L)$	$\{\ val(l)\ \mid l \in L\ \}$，$L$ はラベルの集合

L_1($l_2 = l_1$) のとき，等価である ($L_1 = L_2$) という．

2つの実体グラフ $g_1 = (V_1, E_1, L_{v1}, L_{e1}, \phi_{v1}, \phi_{e1}, \phi_{conn_1}, \phi_{comp_1})$ と $g_2 = (V_2, E_2, L_{v2}, L_{e2}, \phi_{v2}, \phi_{e2}, \phi_{conn_2}, \phi_{comp_2})$ は，$l_1 = \phi_{v1}(g_1)$，$l_2 = \phi_{v2}(g_2)$ としたとき $l_1 =_{id} l_2$ であるならば同一である ($g_1 \equiv g_2$) という．また，$V_1 = V_2$，$E_1 = E_2$，$L_{v1} = L_{v2}$，$L_{e1} = L_{e2}$，$\phi_{conn_1} = \phi_{conn_2}$，かつ，$\phi_{comp_1} = \phi_{comp_2}$ のとき等価である ($g_1 = g_2$) という．さらに，ψ_v を $V(g_i)$ から V_s への全単射，ψ_e を $E(g_i)$ から E_s への全単射とするとき，$\phi_{conn_i}(V, U) = e \Rightarrow \phi_{conn_s}(\psi_v(V), \psi_v(U)) = \psi_e(e)$ かつ $\phi_{comp_i}(V, U) = v \Rightarrow \phi_{comp_s}(\psi_v(V), \psi_v(U)) = \psi_v(v)$ であるならば，同形である ($g_1 \approx g_2$) という．また，$V(g_1) \subseteq V(g_2)$，$E(g_1) \subseteq E(g_2)$，ϕ_{conn_1} が ϕ_{conn_2} と同じか ϕ_{conn_2} の限定，かつ，ϕ_{comp_1} が ϕ_{comp_2} と同じか ϕ_{comp_2} の限定であるとき，g_1 は g_2 の部分グラフである ($g_1 \subseteq g_2$) という． □

次に，実体グラフの構成要素を定義する．

定義 3 実体グラフ $g = (V, E, L_v, L_e, \phi_v, \phi_e, \phi_{conn}, \phi_{comp})$ において，$G_i \subseteq 2^{V \cup E}$ なる G_i に対して，$\phi_{comp}(G_i) = v$ であるとき，v は G_i から直接構成されているという．また，G_i を v の直接構成要素集合と呼び，$\phi^{-1}_{comp}(v)$ と表記する．v の直接構成要素集合の中の実体グラフのみからなる集合（すなわち，$\phi^{-1}_{comp}(v) \cap V$）を v の直接構成実体グラフ集合といい，その要素を直接構成実体グラフという．同様に，v の直接構成要素集合の中の枝のみからなる集合（すなわち，$\phi^{-1}_{comp}(v) \cap E$）を v の直接構成枝集合といい，その要素を直接構成枝という．

次に，g の n 次の構成要素集合 $V^n_{ce}(g)$ を以下の漸化式により定義する．

$$V^0_{ce}(g) = \{g\}$$
$$V^{i+1}_{ce}(g) = \bigcup DC(v),\ v \in V^i_{ce}(g) \cap V$$

ここで，$DC(v)$ は v の直接構成要素集合である．g の n 次の構成要素集合 $V^n_{ce}(g)$ の中の実体グラフ（枝）のみからなる集合を g の n 次の構成実体グラフ集合（構成枝集合）といい，その要素を g の n 次の構成実体グラフ（構成枝）という．

そして，$V^\infty_{ce}(g)$ を g の構成要素集合という．g の構成要素集合の中の実体グラフ（枝）のみからなる集合を g の構成実体グラフ集合（構成枝集合）といい，その要素を構成実体グラフ（構成枝）という． □

例 7 図 1 の実体グラフ $g1$ の直接構成要素は，$g11$, $g12$, $e13$, $e14$ である．$g11$ と $g12$ は $g1$ の直接構成実体グラフであり，$e13$, $e14$ は $g1$ の直接構成枝である．

次に，実体グラフの集合をとらえるために集積グラフを導入する．

定義 4 集積グラフは実体グラフを成分に持つグラフである．すなわち，集積グラフ cg の成分である実体グラフを，$g_1 = (V_1, E_1, L_{v1}, L_{e1}, \phi_{v1}, \phi_{e1}, \phi_{conn_1}, \phi_{comp_1})$, $\cdots$, $g_n = (V_n, E_n, L_{vn}, L_{en}, \phi_{vn}, \phi_{en}, \phi_{conn_n}, \phi_{comp_n})$ とすると，cg は9つ組 $(nm, V, E, L_v, L_e, \phi_v, \phi_e, \phi_{conn}, \phi_{comp})$ で表される．ここで，nm は集積グラフの名前，$V = V_1 \cup V_2 \cup \cdots \cup V_n$，$E = E_1 \cup E_2 \cup \cdots \cup E_n$，$L_v = L_{v1} \cup L_{v2} \cup \cdots \cup L_{vn}$，$L_e = L_{e1} \cup L_{e2} \cup \cdots \cup L_{en}$，$\phi_v : V \to L_v$，$\phi_e : E \to L_e$，$\phi_{conn} : 2^V \times 2^V \to E$，$\phi_{comp} : 2^{V \cup E} \to V$ である．そして，集積グラフ cg の成分である実体グラフ $(g_1, g_2, \cdots, g_n)$ を代表実体グラフと呼ぶ．データベースは集積グラフの集合である．集積グラフの名前はデータベース内で一意でなければならない． □

最後に，集積グラフ中の実体グラフの構造を表現するシェイプグラフを導入する．

定義 5 シェイプグラフは以下のように定義される．

- シェイプグラフには原子シェイプグラフと標準シェイプグラフがある．シェイプグラフはスキーマラベルを持つ．スキーマラベルは名前とデータ型の組で表される．
- 原子シェイプグラフは点である．
- 集積グラフ $cg = (nm, V, E, L_v, L_e, \phi_v, \phi_e, \phi_{conn}, \phi_{comp})$ の標準シェイプグラフ sg は9つ組 $(nm, V_s, E_s, L_{v_s}, L_{e_s}, \phi_{v_s}, \phi_{e_s}, \phi_{conn_s}, \phi_{comp_s})$ で表される．ここで，nm は集積グラフの名前，V_s は sg に包含されるシェイプグラフの集合，E_s はスキーマ枝の集合，L_{v_s} はシェイプグラフのラベル集合，L_{e_s} はスキーマ枝のラベル集合，$\phi_{v_s} : V_s \to L_{v_s}$，$\phi_{e_s} : E_s \to L_{e_s}$，$\phi_{conn_s} : 2^{V_s} \times 2^{V_s} \to E_s$，$\phi_{comp_s} : 2^{V_s \cup E_s} \to V_s$．ただし，スキーマ枝はスキーマラベルをもつ枝である．さらに，集積グラフとそれに対応するシェイプグラフの間には以下の関係がある．
  - ◇ $nm(L_{v_s}) \supseteq nm(L_v)$,
  - ◇ $nm(L_{e_s}) \supseteq nm(L_e)$,
  - ◇ V 中のどの実体グラフに対しても V_s 中のシェイプグラフに対応付ける写像 θ_v が存在する．また，実体グラフの集合 $V = \{v_1, \cdots, v_n\}$ に対しては，$\{\theta_v(v_1), \cdots, \theta_v(v_n)\}$ を $\theta_v(V)$ と表記する．
  - ◇ E 中のどの枝に対しても E_s 中のスキーマ枝に対応付ける写像 θ_e が存在する．また，枝の集合 $E = \{e_1, \cdots, e_n\}$ に対しては，$\{\theta_e(e_1), \cdots, \theta_e(e_n)\}$ を $\theta_e(E)$ と表記する．
  - ◇ $\phi_{conn}(V, U) = e \Rightarrow \phi_{conn_s}(\theta_v(V), \theta_v(U)) = \theta_e(e)$

◇ $\phi_{comp}(V \cup W) = g \quad \Rightarrow$

$$\phi_{comp_s}(\theta_v(V) \cup \theta_e(W)) = \theta_v(g)$$

実体グラフが挿入され上記の関係が満足されない場合，上記の関係を満足するようにシェイプグラフが変化しなければならない．

□

3.2 演　　算

実体グラフの操作のために書き換え演算 rewrite を定義する．本演算を用いた問合せ記述は，視覚的問合せ言語[11),12),25)]と同様な視覚性を有する．まず，集積グラフに対応する問合せグラフを定義する．

定義 6 問合せグラフは以下のように定義される．

- 問合せグラフは，原子問合せグラフまたは標準問合せグラフである．問合せグラフは問合せラベルを持つ．問合せラベルは，3つ組 $(v_{id}, v_{name}, v_{val})$ である．ここで，v_{id} は識別子に対する変数，v_{name} は名前に対する変数，v_{val} は値に対する変数である．
- 原子問合せグラフは点である．
- 標準問合せグラフ qg は9つ組 $(cgn, V_q, E_q, L_{v_q}, L_{e_q}, \phi_{v_q}, \phi_{e_q}, \phi_{conn_q}, \phi_{comp_q})$ である．ここで，cgn は集積グラフの名前，V_q は qg 中に含まれる問合せグラフの集合，E_q は qg 中に含まれる問合せ枝の集合，L_{v_q} は問合せグラフのラベル集合，L_{e_q} は問合せ枝のラベル集合，$\phi_{v_q} : V_q \to L_{v_q}$，$\phi_{e_q} : E_q \to L_{e_q}$，$\phi_{conn_q} : 2^{V_q} \times 2^{V_q} \to E_q$，$\phi_{comp_q} : 2^{V_q \cup E_q} \to V_q$ である．ここで，問合せ枝とは問合せラベルを持つ枝である．cgn が既存の集積グラフの名前の場合は，問合せグラフの構造はその集積グラフのシェイプグラフの構造に従わなければならない．□

正規表現は，グラフに基づくデータベースにおいて問合せに良く使用される[11),12),25)]．ここでも，正規表現が可能にすることを考える．まず，問合せパスを定義する．

定義 7 問合せパスは以下により再帰的に定義される．

- 問合せ枝はパスである．
- 問合せ枝とパスの連接はパスである．
- パス p に対して，$(p)^+$ はパスである．
- パス $p1, p2$ に対して，($p1 \mid p2$) はパスである．
- パス p に対して，$\neg p$ はパスである．

問合せパスは，問合せパスラベルを持つことができる．問合せパスラベルは，識別子に対する変数，名前に対する変数，値に対する変数，正閉包を表すフラグ，ならびに，否定を表すフラグの5つ組である．□

次に，標準問合せグラフに問合せパスを導入した正規問合せグラフを定義する．

定義 8 正規問合せグラフ qg は14項組 $(cgn, V_q, E_q, P_q, L_{v_q}, L_{e_q}, L_{p_q}, \phi_{v_q}, \phi_{e_q}, \phi_{p_q}, \phi_{conn_q}, \phi_{comp_q}, \phi_{path_{conn}}, \phi_{path_{union}})$ である．ここで，P_q は qg 中の問合せパスの集合，L_{p_q} は問合せパスラベルの集合，$\phi_{p_q} : P_q \to L_{p_q}$，$\phi_{path_{conn}}$ はパスの接続関係を表す写像 : $\Pi(E_q \cup P_q) \to P_q$，$\phi_{path_{union}}$ はパスの和を表す写像 : $\Pi P_q \to P_q$ であり，他は標準問合せグラフと同じである．ただし，$\Pi P_i \equiv P_1 \times P_2 \times \cdots \times P_n$ である．□

次に，書き換え演算の主要素である問合せ仕様と割り当て仕様を定義する．

定義 9 問合せ仕様は組 (rqg, rc) である．ここで，rqg は正規問合せグラフ，rc は rqg 中の変数集合 $\mathtt{X}$ に関する検索条件式である．割り当て仕様は組 $(nqg.va)$ である．ここで，nqg は標準問合せグラフ，va は nqg 中の変数に対する値割り当ての集合である．□

以上により，書き換え演算 rewrite は以下のように定義できる．

定義 10 書き換え演算 $rewrite(target, sl)$ は，sl で指定された集積グラフ中で sl を満足する実体グラフを，$target$ で指定した実体グラフの構造を持つように書き換え，$target$ で指定した集積グラフとする．ここで，$target$ は割り当て仕様であり，sl は問合せ仕様のリストである．また，$\mathtt{X}_t$ を $target$ 中の変数の集合，$\mathtt{X}_s$ を sl 中の変数の集合としたとき，$\mathtt{X}_t \subseteq \mathtt{X}_s$．$target$ に指定された集積グラフが既存の集積グラフであるとき，既存の集積グラフが書き換えられる．そうでないときは，$target$ に指定された名前をもつ集積グラフを生成する．□

4. 考　　察

4.1 実体グラフの性質

ここでは，議論を簡単にするために，枝の始要素集合と終要素集合には，おのおのひとつの要素のみが存在するものとする．以下の議論は，要素が複数の場合も適用できる．

定義 11 $V(g_1) \cap V(g_2) = \emptyset$，かつ，$E(g_1) \cap E(g_2) = \emptyset$ であるとき，g_1 と g_2 は独立であるという．また，$^{\exists}e \in E(g_1)$ ($i(e) \subseteq V(g_2) \vee t(e) \subseteq V(g_2)$)，または，$^{\exists}e \in E(g_2)$ ($i(e) \subseteq V(g_1) \vee t(e) \subseteq V(g_1)$) であるとき，g_1 と g_2 は連結しているという．□

例 8 次の2つの実体グラフ g_a と g_b を考えよう．$g_a = (\{v_1, v_2, g_a\}, \{e_1\}, L_v, L_e, \phi_v, \phi_e, \phi_{conn_a}, \phi_{comp_a})$，ここで，$\phi_{conn_a}(\{v_1\}, \{v_2\}) = e_1$，$\phi_{comp_a}(\{v_1, v_2, e_1\}) = g_a$ であり，$g_b = (\{v_2, v_3, g_b\}, \{e_2\}, L_v, L_e, \phi_v, \phi_e,$

$\phi_{conn_b}, \phi_{comp_b}$)，ここで，$\phi_{conn_b}(\{v_2\},\{v_3\}) = e_2$，$\phi_{comp_b}(\{v_2, v_3, e_2\}) = g_b$．このとき，$t(e_1) = i(e_2) = \{v_2\}$ であり，g_a と g_b は連結している．

ある実体グラフ g の全ての構成要素が他の実体グラフ，ならびに，その構成要素と接続されていない場合，g は他の要素からくくり出して表現可能である．これを利用すると，ある実体グラフをそのまま一つのグラフとして表現する必要はなく，いくつかのより小さな実体グラフに分割して表現できる．これにより，複合オブジェクトを分割して格納するアプローチ[28]と同様，実体グラフの取り扱いを容易にすることができる．ある実体グラフ g の全ての構成要素が他の実体グラフ，ならびに，その構成要素と接続されていない場合，g は構造的に整っているといえる．

定義 12 実体グラフ g の構成実体グラフ集合を VC とすると，VC の要素を始要素集合または終要素集合に含む全ての枝に対して，枝の始要素集合が VC の部分集合であり，かつ，枝の終要素集合が VC の部分集合であるとき，g は構造的に整っているという．□

例 9 実体グラフ $g = (\{v_1, v_2, v_3, v_4, g_1, g_2, g\}, \{e_1, e_2, e_3\}, L_v, L_e, \phi_v, \phi_e, \phi_{conn}, \phi_{comp})$ を考える．ここで，$\phi_{conn}(\{v_1\},\{v_2\}) = e_1$，$\phi_{conn}(\{v_3\},\{v_4\}) = e_2$，$\phi_{conn}(\{g_1\},\{v_3\}) = e_3$，$\phi_{comp}(\{v_1, v_2, e_1\}) = g_1$，$\phi_{comp}(\{v_3, v_4, e_2\}) = g_2$，$\phi_{comp}(\{g_1, g_2, e_3\}) = g$．実体グラフ g_1 の構成実体グラフ集合は $\{v_1, v_2\}$ である．実体グラフ v_1，v_2 を始要素集合または終要素集合に含む枝は e_1 のみである．e_1 の始要素集合 $\{v_1\}$ は g_1 の構成実体グラフ集合 $\{v_1, v_2\}$ の部分集合である．また，e_1 の終要素集合 $\{v_2\}$ は g_1 の構成実体グラフ集合 $\{v_1, v_2\}$ の部分集合である．従って，g_1 は構造的に整っている．一方，実体グラフ g_2 の構成実体グラフ集合は $\{v_3, v_4\}$ である．実体グラフ v_3，v_4 を始要素集合または終要素集合に含む枝は e_2 と e_3 である．ここで，e_3 の始要素集合 $\{g_1\}$ は g_2 の構成実体グラフ集合 $\{v_3, v_4\}$ の部分集合ではないので，g_2 は構造的に整っていない．

次に，実体グラフが構造的に整っているかを別の基準で判断するために，枝の深さを定義する．

定義 13 枝 e を直接構成枝とする実体グラフを g とする．このとき，$i(e) \subseteq \bigcup_{i=1}^{n_{init}+1} V_{ce}^i(g)$ なる最小の数 n_{init} を枝 e の始要素の深さと呼ぶ．同様に，$t(e) \subseteq \bigcup_{i=1}^{n_{term}+1} V_{ce}^i(g)$ なる最小の数 n_{term} を枝 e の終要素の深さと呼ぶ．さらに，始要素の深さと終要素の深さの大きい方を枝の深さと呼ぶ．□

例 10 図 1 の実体グラフ $g1$ の枝 $e11$ は，実体グラフ $g11$ の直接構成枝である．$e11$ の始要素集合は $\{n111, n112\}$ であり，これらは $g11$ の直接構成実体グラフである．つまり，$i(e11) \subseteq V_{ce}^1$ である．従って，$e11$ の始要素の深さは 0 である．同様に，$e11$ の終要素の深さも 0 である．よって，枝 $e11$ の深さは 0 である．一方，$e13$ は $g1$ の直接構成枝である．$i(e13) = \{n114, n115\}$ であり，$i(e13) \subseteq V_{ce}^1 \cup V_{ce}^2$ である．従って，$e13$ の始要素の深さは 1 である．同様に，$e13$ の終要素の深さも 1 であり，枝 $e13$ の深さは 1 である．

ある枝の始要素（終要素）の深さが 0 であるということは，当然ではあるが，その枝は始要素（終要素）の構成要素を始要素（終要素）としていないということであり，直観的には，始要素（終要素）である実体グラフの境界を横切って構成実体グラフを始要素（終要素）とはしていないということである．一方，ある枝の始要素（終要素）の深さが k（$k > 0$）であるということは，その枝を直接構成枝とする実体グラフ v の $k+1$ 次の構成実体グラフを始要素（終要素）としているということであり，直観的には，実体グラフの境界を k 回横切って v の $k+1$ 次の構成実体グラフを始要素（終要素）とするものが 1 つはあるということである．このように考えると，枝の始要素（終要素）の深さを利用すると実体グラフが構造的に整っているかを判断できると考えられる．

定理 1 構造的に整っている実体グラフ g の n 次の構成実体グラフ g_{n_i} は，$1 \leq k \leq n$ なるすべての k について，g_{n_i} および g_{n_i} の構成要素が g の k 次の構成枝の始終要素でないか，または，g_{n_i} および g_{n_i} の構成要素が g の k 次の構成枝の始要素（終要素）であるならば高々深さ $(n-k)$ の始要素（終要素）である場合，構造的に整っている．□

（証明）まず，g の構成枝が k 次（$1 \leq k \leq n$）の構成枝のみであるとして考える．

g_{n_i} および g_{n_i} の構成要素が g の k 次の構成枝の始終要素でない場合，明らかに，k 次の構成枝で g_{n_i} および g_{n_i} の構成要素を始終要素とするものはない．従って，この場合，k 次の構成枝は g_{n_i} の構成実体グラフを始終要素とする枝ではなく，g_{n_i} は構造的に整っている．

次に，g_{n_i} および g_{n_i} の構成要素が g の k 次の構成枝の始要素（終要素）であって高々深さ $(n-k)$ の始要素（終要素）である場合を考える．始要素（終要素）の深さの定義より，この場合，枝の始要素（終要素）集合は g の $(n-k+1)$ 次までの構成要素の和集合の部分集合である．g_{n_i} の m 次（$0 \leq m$）の構成要素は，g の $n+m$ 次の構成要素であるが，$n-k+1 \leq n$ かつ $n \leq n+m$ であるので，常に $n-k+1 \leq n+m$

が成り立つ．ここで，等号が成り立つのは $k=1$ かつ $m=0$ の場合であり，この場合，g_{n_i} は g の1次の構成枝の要素であり，g_{n_i} 自身を始要素または終要素としている．従って，この枝は g_{n_i} の構成要素を始終要素としていない．これ以外の場合は不等号が成り立つ．$m>1$ なる m の場合とは g_{n_i} の構成要素を表すので，k 次の構成枝で g_{n_i} の構成要素を始終要素とするものはないということである．これらより，g_{n_i} は構造的に整っている．

以上が，$1 \leq k \leq n$ なるすべての k について成り立てば，g の n 次までの構成枝で g_{n_i} の構成要素を始終要素とするものはない．また，g は構造的に整っているので，g および g の構成要素は g の外の枝の始終要素とはなっていない．従って，g_{n_i} は構造的に整っている．（証明終り）

この定理を利用すると，ある代表実体グラフ g が与えられたときに☆，g の構成要素を次数に従って順に解析するのみで g の構成要素の中で構造的に整っているものを求めることができる．

4.2 問合せ能力

ここでは，書き換え演算の能力を示す次の定理を示す．

定理 2 書き換え演算は複合値を扱うように拡張した datalog プログラム [27] で記述することができる．□

(証明) 書き換え演算は $rewrite((qg_t, va), (qg_{q_1}, rc_{q_1}) \cdots (qg_{q_n}, rc_{q_n}))$ と記述できる．そこで，qg，rc，ならびに，va が複合値を扱うように拡張した datalog プログラムで記述できることを示す．

(1) 正規問合せグラフは以下のように変換できる．

(1-1) 問合せラベル $(X_{id}, X_{name}, X_{val})$ を持つ正規問合せグラフ qg

qg の直接構成要素の集合を束縛する変数 Q を導入し，写像 $\phi_{comp}: 2^{V \cup E} \rightarrow V$ に対応する述語 $contain_{qg}(X_{id}, Q)$ をルールの本体に加える．さらに，qg の直接構成要素を表す変数 $Y_i (i=1,\cdots,n)$ を導入して，述語 $Q \ni Y_i$ をルールの本体に加える．

(1-2) 問合せラベル $(Y_{id}, Y_{name}, Y_{val})$ を持つ問合せ枝 e

e の始要素集合を束縛する変数 S と e の終要素集合を束縛する変数 T を導入し，$\phi_{conn}: 2^V \times 2^V \rightarrow E$ に対応する述語 $connect(Y_{id}, S, T)$ をルールの本体に加える．さらに，e の始要素を表す変数 $U_i (i=1,\cdots,m)$ を導入して，述語 $S \ni U_i$ をルールの本体に加える．同様に，e の終要素を表す変数 $W_j (j=1,\cdots,l)$ を導入して，述語 $T \ni W_j$ をルールの本体に加える．

(1-3) 問合せラベル $(X_{id}, X_{name}, X_{val})$ を持つ原子問合せグラフ

変数 X_{id} は標準問合せグラフまたは問合せ枝の要素として出現するので，特に行うことはない．

(1-4) 問合せ枝と問合せパスの連接

まず，2つの問合せ枝 e_1 と e_2 の連接について考える．問合せ枝 e_1 は，(1-3) に述べたように，変数 Y_1，S_1，T_1 を用いて述語 $connect_{e_1}(Y_1, S_1, T_1)$ に変換できる．同様に，問合せ枝 e_2 は述語 $connect_{e_2}(Y_2, S_2, T_2)$ に変換できる．e_1 の終要素が e_2 の始要素の場合，述語 $T_1 \ni \alpha_1$ と述語 $S_2 \ni \alpha_1$ をルールの本体に加える．問合せ枝と問合せパスの連接も同様に変換できる．一般に，問合せ枝の連接 $e_1, e_2, \cdots, e_n$ は，$connect_{e_1}(Y_1, S_1, T_1), \cdots, connect_{e_n}(Y_n, S_n, T_n)$, $T_1 \ni \alpha_1, S_2 \ni \alpha_1, \cdots, T_{n-1} \ni \alpha_{n-1}, S_n \ni \alpha_{n-1}$ という述語に変換できる．

(1-5) 正閉包 $(qp)^+$

まず，簡単のために，問合せ枝 e_1 の正閉包を考える．問合せ枝 e_1 は，(1-3) で述べたように，述語 $connect_{e_1}(Y_1, S_1, T_1)$ に変換できる．正閉包のための第一のルールを以下に示す．

$$p_{e_1}(Y, S, T) \leftarrow connect_{e_1}(Y, S, T).$$

第二のルールを以下に示す．

$$p_{e_1}(Y, S, T) \leftarrow p_{e_1}(Y, S, T_x), connect_{e_1}(Y_x, S_x, T), T_x \ni \alpha, S_x \ni \alpha.$$

以上，問合せ枝について述べてきたが，一般の問合せパスについても同様の変換が行える．

(1-6) 和 ($qp1 \mid qp2$)

問合せパス qp_1 と qp_2 の和は以下に示す2つのルールとする．

$$p_{union} \leftarrow p_1.$$
$$p_{union} \leftarrow p_2.$$

(1-7) 否定

問合せパス qp の否定は以下のように変換する．

$$p_{neg} \leftarrow \neg p.$$

(2) 検索条件 rc は容易に述語に変換できる．

(3) 値割り当て va も容易に述語に変換できる．

（証明終り）

例えば，図4に示した書き換え演算は以下のように表現できる．

$$X \leftarrow contain(X, Q),\ Q \ni Y, Q \ni Z1, Q \ni Z2,$$
$$contain(Z1, R1),\ R1 \ni V1,$$
$$contain(Z2, R2),\ R2 \ni V2,$$

☆ 代表実体グラフは構造的に整っている

$connect(Y, S, T), S \ni V1, T \ni V2,$
$X_{name} = "picture",$
$Y_{name} = "pos", Y_{val} = "on",$
$Z1_{name} = "object", Z2_{name} = "object",$
$V1_{name} = "parts", V2_{name} = "parts".$

5. 関連研究

ここでは，再帰グラフのみでなく超グラフの概念を導入し，マルチメディアデータの内容表現に十分であると考えられる有向再帰ラベルノード超グラフモデル[23]，ならびに，自己組織化意味関連モデル[24]と本論文で提案した再帰有向超グラフデータモデルを比較する．

有向再帰ラベルノード超グラフモデル[23]は知識表現を目的として提案されており，超グラフ，再帰グラフ，ならびに，ラベルノードグラフの概念を導入している．ここで，ラベルノードグラフとは，ノードとノード間を結ぶ有向枝という二つの概念を一つの概念で表現するために導入されたものであり，ノードとしても枝としても振る舞える．例えば，Fukui は city であるという関係 (isa Fukui city) は，図 7(a) のように記述できる．図で四角形で示しているのがラベルノードである．図の有向枝の始要素は，従来の枝に対応するラベルノードである．図の有向枝と交わるラベルノード，ならびに，有向枝の終要素は，従来の枝が連結するノードである．この有向枝により，超グラフとしての枝が表現でき，超グラフ内の要素を順序付けることができる．また，LISP の式 (PLUS (TIMES 3 6 2) 4) は図 7(b) のように記述できる．このように，ラベルノード内にラベルノードの記述が可能であり，再帰グラフの性質を持つ．有向再帰ラベルノード超グラフモデルと再帰有向超グラフデータモデルの最も大きな違いは，前者が知識表現モデルであるのに対し，後者はデータモデルであるということである．有向再帰ラベルノード超グラフモデルでは，基本的に一つの連結グラフで対象世界の知識を表現する．従って，データ定義もデータ実体も一つの連結グラフで表される．一方，再帰有向超グラフデータモデルでは，一つのマルチメディアデータに対してその内容を表現する一つの実体グラフを考え，その実体グラフが多数存在することを前提としている．このため，同種の実体グラフをまとめる集積グラフを導入し，集積グラフの構造を表現するシェイプグラフを導入している．さらに，シェイプグラフが表す構造を手がかりとして集積グラフに対する問合せを可能としている．

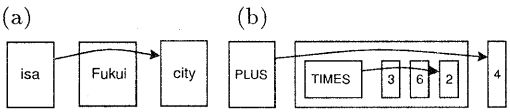


図 7 ラベルノードの例
Fig. 7 Examples of labelnodes.

自己組織化意味関連モデル[24]は，概念の相対性の操作を可能とする概念構造の表現モデルである．このために，通常のデータベース演算の挿入・削除・更新・検索・巡航やグラフの統合／分割，グループ化／アングループ化という演算に加えて，デュアル (dual) 演算を導入している．デュアル演算は，ある局所ビュー内のノードと枝の役割を反転させる演算である．これにより，ある概念を，ある場合は実体としてとらえ，ある場合は関連としてとらえることが可能である．再帰有向超グラフデータモデルでは，いったん枝としたものは実体グラフとしてとらえることはできない．また，実体グラフとしたものは枝としてとらえることはできない．自己組織化意味関連モデルのデュアル演算に相当する機能のサポートは今後の課題である．

6. おわりに

本論文では，概念表現やマルチメディアデータの内容表現といった分野のデータを表現するのに十分な表現能力を持ち，かつ，再帰的な問合せやパス上の正規表現を用いた問合せを簡単に行うことを目的として，有向グラフ，再帰グラフ，ならびに，超グラフの概念を導入したデータモデル（再帰有向超グラフデータモデル）を提案した．データを表現する基本単位である実体グラフ，実体グラフの集まりを扱う集積グラフ，ならびに，集積グラフの構造を表現するシェイプグラフを導入した．演算はグラフの書き換えによるものであり，再帰的な問合せやパス上の正規表現による問合せが可能である．また，実体グラフの枝の始終要素の深さを利用して実体グラフを分割して表現できるか否かが決定できることを示した．さらに，書き換え演算は複合値を扱うように拡張した datalog プログラムで記述できることを明らかにした．

本論文では，書き換え演算が複合値を扱うように拡張した datalog プログラムで記述できることを示す際に，連結関係を表す写像と構成関係を表す写像を述語として使用した．これは問合せの処理効率を考慮したものではない．著者らは，有向グラフに基づくデータモデルのデータベースに対して検索効率の良い格納構造を提案してきた[29]．この格納構造を再帰有向超グラフデータモデルに対しても適用可能とし，効率の良

い問合せ処理を実現することは今後の課題である．また，自己組織化意味関連モデルのデュアル演算に相当する機能のサポートや，利用者インタフェースの開発も今後の課題である．

謝辞 有益なコメントを頂いた査読者の方々に感謝致します．なお，本研究は，一部，文部省科学研究費補助金（課題番号 09780258）による．

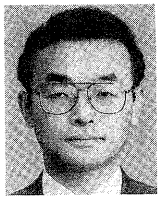
参考文献

1) Gudivada, V. N. and Raghavan, V. V. : Content-Based Image Retrieval Systems, *COMPUTER*, Vol. 28, No. 9, pp. 18–22 (1995).
2) Flickner, M., Sawhney, H., Niblack, W., Ashley, J. Huang Q., Dom, B., Gorkani, M., Hafner, J., Lee, D., Petkovic, D., Steele, D., and Yanker, P. : Query by Images and Video Content: The QBIC Project, *COMPUTER*, Vol.28, No. 9, pp. 23–32 (1995).
3) Petrakis, E. G. M., and Faloutsos, C. : Similarity Searching in Medical Image Databases, *IEEE Trans. on Knowledge and Data Eng.*, Vol. 9, No. 3, pp. 435–447 (1997).
4) Uehara, K., Oe, M., and Maehara, K. : Knowledge Representation, Concept Acquisition and Retrieval of Video Data, *Proc. of Int'l Symposium on Cooperative Database Systems for Advanced Applications*, pp.218–225 (1996).
5) Sowa, J. F. : *Conceptual Structures: Information Processing in Mind and Machine*, Addison-Wesley Publishing Company.
6) Ramaswamy, M., Sarkar, S., and Chen, Y.-S. : Using Directed Hypergraphs to Verify Rule-Based Expert Systems, *IEEE Trans. on Knowledge and Data Eng.*, Vol. 9, No. 2, pp. 221–237 (1997).
7) Puigsegur, J., Schorlemmer, W. M., and Agusti, J. : From Queries to Answers in Visual Logic Programming, *Proc. of 1997 IEEE Conf. on Visual Languages*, pp. 102–109 (1997).
8) Gyssens, M., Parendaens, J., Van den Bussche, J., and Gucht, D. V. : A Graph-Oriented Object Database Model, *IEEE Trans. on Knowledge and Data Eng.*, Vol. 6, No. 4, pp. 572–586 (1994).
9) Paredaens, J., Peelman, P. and Tanca, L. : G-Log: A Graph-Based Query Language, *IEEE Trans. on Knowledge and Data Eng.*, Vol. 7, No. 3, pp. 436–453 (1995).
10) Su, S. Y. W., Guo, Mingsen and Lam, H.: Association Algebra: A Mathematical Foundation for Object-Oriented Databases, *IEEE Trans. on Knowledge and Data Eng.*, Vol. 5, No. 5, pp. 775–798 (1994).
11) Curz, I. F., Mendelzon, A. O. and Wood, P. T., G^+: Recursive Queries Without Recursion, *Proc. of 2nd Int'l Conf. on Expert Database Systems*, pp. 355–368 (1988).
12) Consens, M. P. and Mendelzon, A. O., GraphLog: a Visual Formalism for Real Life Recursion, *Proc. of 9th ACM PODS*, pp. 404–416 (1990).
13) Kunii, H. S. : *Graph Data Model and Its Data Language*, Springer-Verlag (1990).
14) Rosenberg, A. L. : Addressable collection graphs, *J. ACM*, Vol. 19, No. 2, pp. 309–340 (1972).
15) Gutiérrez, A., Pucheral, P., Stehffen, H., and Thévenin, J.-M. : Database Graph Views: A Practical Model to Manage persistent Graphs, *Proc. of the 20th Int'l Conf. on Very Large Data Bases*, pp. 391–402 (1994).
16) Lucarella, D. and Zanzi, A. : A Graph-Oriented Data Model, *Proc. of 7th Int'l Conf. on Database and Expert Systems Applications*, pp. 197–206 (1996).
17) Ayres, R. and King, P. J. H. : Querying Graph Databases Using a Functional language Extended with Second Order Facilities, *Proc. of the 14th British National Conf. on Databases (BNCOD14)*, pp. 189–203 (1996).
18) Guting, R. H. : GraphDB : Modeling and Querying Graphs in Databases, *Proc. of the 20th Int'l Conf. on Very Large Data Bases*, pp. 297–308 (1994).
19) Goldman, R., and Widom, J. : DataGuides: Enabling Query Formulation and Optimization in Semistructured Databases, *Proc. of 23rd Int'l Conf. on Very Large Data Bases*, pp. 436–445 (1997).
20) Buneman, P., Davidson, S., Fernandez, M., and Suciu, D. : Adding Structure to Unstructured Data, *Proc. of 6th Int'l Conf. on Database Theory*, pp. 336–350 (1997).
21) Levene, M. and Loizou, G. : A Graph-Based Data Model and its Ramification, *IEEE Trans. on Knowledge and Data Eng.*, Vol. 7, No. 5, pp. 809–823 (1995).
22) Consens, M. P. and Mendelzon, A. O. : Hy^+ : A Hypergraph-based Query and Visualization System, *Proc. of 1993 ACM SIGMOD Int'l Conf. on Management of Data*, pp. 511–516 (1993).
23) Boley, H. : Directed Recursive Labelnode Hypergraphs: A New Representation-Language, *Artificial Intelligence*, Vol. 9, pp. 49–85 (1977).
24) Fujiwara, Y. and Gotoda, H. : Representation

Model for Relativity of Concepts, *Int'l Forum on Information and Documentation*, Vol. 20, No. 1, pp. 22–30 (1995).
25) Houchin, T. : DUO: Graph-based Database Graphical Query Expression, *Proc. of 2nd Far-East Workshop on Future Database Systems*, pp.286–295 (1992).
26) Nakata, M., Hochin, T., and Tsuji, T. : Bottom-up Scientific Databases Based on Sets and Their Top-down Usage, *Proc. of International Database Engineering & Applications Symposium*, pp. 171–179 (1997).
27) Abiteboul, S., Hull, R., and Vianu, V. : *Foundation of Databases*, Addison-Wesley Publishing Company (1995).
28) Valduriez, P., Khoshafian, S. N., and Copeland, G. P. : Implementation Techniques of Complex Objects, *Proc. of the 12th International Conference on Vary Large Data Bases*, pp. 101–110 (1986).
29) Hochin, T., and Tsuji, T. : A Storage Structure for Graph-Oriented Databases Using an Array of Element Types, *Proc. of 8th Great Lakes Symposium on VLSI*, pp.452–457 (1998).

(平成 11 年 9 月 20 日受付)
(平成 11 年 12 月 20 日採録)

(担当編集委員 藤原 譲)



宝珍 輝尚 (正会員)

昭和 57 年名古屋工業大学電気工学科卒業. 昭和 59 年同大大学院修士課程修了. 同年, 日本電信電話公社入社. 平成 5 年 7 月より福井大学工学部情報工学科助手. 現在, 情報・メディア工学科助教授. 博士 (工学). マルチメディアデータ管理, 柔構造データベース, 拡張可能 DBMS, グラフに基づくデータモデルの研究に従事. 電子情報通信学会, IEEE, ACM, 日本情報考古学会各会員.

都司 達夫 (正会員)

昭和 48 年大阪大学基礎工学部電気工学科卒業. 昭和 53 年同大大学院博士課程修了. 同年, 福井大学工学部情報工学科講師. 現在, 情報・メディア工学科教授. 工学博士. データベースシステム, プログラミング言語の研究に従事. 著書"Optimizing Schemes for Structured Programming Language Processors" (Ellis Horwood). 電子情報通信学会, IEEE, 日本情報考古学会各会員.