DEIM Forum 2011 B10-5

再帰有向超グラフデータモデルにおける特化・汎化関連の導入について

大平　雄貴† 　宝珍　輝尚† 　野宮　浩揮†

† 京都工芸繊維大学院工芸科学研究科情報工学専攻
〒 606–8585 京都府京都市左京区松ヶ崎御所海道町
E-mail: †ohirayuki@gmail.com, ††{hochin,nomiya}@kit.ac.jp

あらまし　本論文では、マルチメディアデータの内容をグラフを用いて表現する再帰有向超グラフデータモデルにおいて特化・汎化関連を表現可能とする。このために、再帰有向超グラフデータモデルに、一種のデータ定義を表現するシェイプグラフ間の特化・汎化関連を可能にするスキーマグラフを導入する。スキーマグラフの導入により、より実世界に近いデータ定義の系統化を目指す。

キーワード　データモデル、グラフ、特化、汎化、データ主導モデル

Introduction of specialization and generalization relationships to directed recursive hypergraph data model

Yuki OHIRA†, Teruhisa HOCHIN†, and Hiroki NOMIYA†

† Graduate School of Information Science, Kyoto Institute of Technology
Goshokaidocho Matsugasaki, Sakyo-ku, Kyoto-shi,Kyoto 606–8585 Japan
E-mail: †ohirayuki@gmail.com, ††{hochin,nomiya}@kit.ac.jp

Abstract　In this paper, specialization and generalization relationships can be expressed in directed recursive hypergraph data model which expresses contents of multimedia data by graphs. For this purpose, the schema graph is introduced to the directed recursive hypergraph data model. This graph can express specialization and generalization relationships between shape graphs which express a kind of data definition. Introducing the schema graph enables us to systematize data definition more realistic to the real world.

Key words　Data model, Graph, Specialization, Generalization, Instance-based model

1. はじめに

近年、データベース中に蓄えられたマルチメディアデータの検索について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は、このマルチメディアデータの検索分野に含まれている。この中で、マルチメディアデータの内容をグラフを用いて表現するアプローチがある。グラフ表現の中で、有向ラベル付きグラフは頻繁に用いられている。例えば、Petrakis らは、有向ラベル付けグラフを用いることにより、医学的イメージの内容の表現を提案している [1]。Uehara らはビデオクリップの場面の内容を表現するために、セマンティックネットワークを用いている [2]。筆者らも、マルチメディアデータ表現のグラフに基づくデータモデルを提案している [3]。このデータモデルは、再帰有向超グラフデータモデル (Directed Recursive Hypergraph Datamodel;DRHM) と呼ぶモデルである。このデータモデルでは、実体グラフ、集積グラフ、シェイプグラフという三つのグラフによりマルチメディアデータの内容表現を行う。実体グラフ (Instanse Graph) は、データを表現する際の基本的な単位である。集積グラフ (Collection Graph) は、構成要素として実体グラフを持つグラフである。そして、集積グラフのシェイプグラフ (Shape Graph) は、集積グラフの構造を表現するグラフである。DRHM はデータ主導のデータモデルである。すなわち、従来のスキーマ (データ定義) に相当するシェイプグラフから作成する必要はなく、シェイプグラフに対応しない実体グラフを作成できる。そして、この場合、その実体グラフに合うようにシェイプグラフが変更される。

ここで、近年のデータモデルでは、特化関連や汎化関連を導入しているものが多く見られる。特化関連は、ある実体集合とそれをより特殊化した実体集合の間に設けられる関連である。オブジェクト指向の考え方では、継承として扱われている関連である。汎化関連は、いくつかの実体集合とそれらを一般化した実体集合の間に設けられる関連である。このような特化関連や汎化関連を設けて実体集合を整理することにより、共通の情報を認識し、それをもとに操作や処理が可能となるため、情報

のモデリングにおいて特化関連や汎化関連は非常に有用である。しかしながら、DRHM には特化関連や汎化関連は導入されていない [3]。

そこで本論文では、DRHM に高度なモデリング能力を持たせることを目的として、DRHM への特化関連と汎化関連の導入について検討する。このために、本研究では、上記の三種のグラフに加えて、新たにスキーマグラフ (Schema Graph) というグラフを導入する。このグラフは、マルチメディアデータの内容表現のデータ定義を、より実世界に近く系統的になるようにすることを目指したものであり、このスキーマグラフにおいて、特化関連と汎化関連を記述可能とする。

以下、2. では DRHM について概説し、3. では特化・汎化関連について概説する。そして、4. でスキーマグラフを導入し、5. で特化・汎化関連設定時の振る舞いについて述べる。そして、6. でスキーマグラフに形式的な定義を与える。

2. 再帰有向超グラフデータモデル (DRHM)

ここでは、再帰有向超グラフデータモデルについて概説する。再帰有向超グラフデータモデルでは、マルチメディアデータの内容を三つのグラフを用いて表現する。すなわち、実体グラフ、集積グラフ、シェイプグラフである。それぞれのグラフについて以下で説明する。

2.1 実体グラフ (Instance Graph)

DRHM において、データを表現する際の最も基本的な構成要素は実体グラフである。実体グラフは再帰有向超グラフである。また、実体グラフは、識別子、名前、データ値を構成するラベルを持つ。ここで、実体グラフを単純な例を用いて述べる。

例 1. 図 1 に示される写真の表現を考える。この写真では、蝶は花の上にいる。蝶の頭、前の羽、後の羽と二つの前足、一つの中央の足、二つの後足が写真に写っている。二つの前足は花の上にあり、二つの後足は他の花にある。この写真の DRHM における内容表現を図 2 に示す。図 2 では、実体グラフは丸みを帯びた長方形で表現されている。例えば、n111,n112,g1,g11,g12 は実体グラフである。破線で要素の集合を表し、その間の関連を矢印で表現している。例えば、n111,n112 は矢印 e11 により n113 と関連を持っている。囲みの始点要素の集合が一つだけの要素を含み、終点要素の集合も一つだけの要素を含んでいる時、関連は単純な矢印で表現される。実体グラフ g12 の関連 e16 はこの表現の例である。再帰性の性質を持つので、実体グラフはその中に、実体グラフと枝を含んでいる。例えば、g1 は g11,g12,e14 を含んでいる。実体グラフの詳しい定義は、参考文献 [3] を参照されたい。

2.2 集積グラフ (Collection Graph)

実体グラフの集合は集積グラフで表現される。すなわち、集積グラフは実体グラフを構成要素とするグラフであり、その構造は実体グラフと同じである。

例 2. 集積グラフの例を図 3 に示す。この図では、集積グラフは一点鎖線により表現される。集積グラフはデータベースごとに固有の名前を持つ。図 3 に示される集積グラフの名前は Picture である。実体グラフ g1 は図 2 で示されているものである。実体グラフ g2 は他の写真の内容を表現したグラフである。これらの実体グラフを代表実体グラフ (representative instance graph) と呼ぶ。

図 1 蝶と花の写真

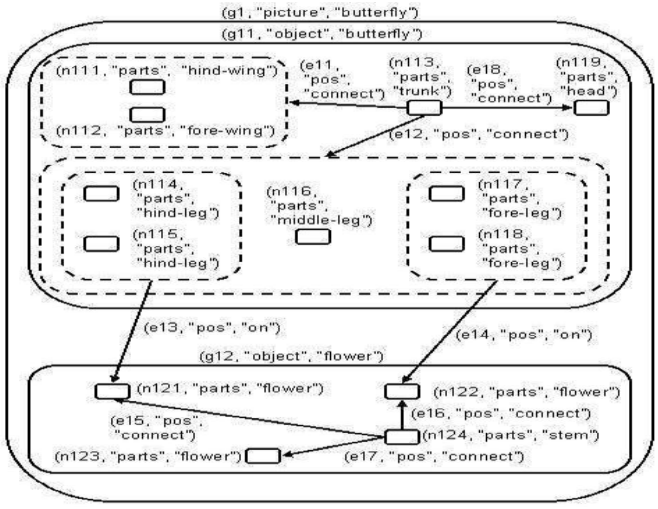

図 2 蝶と花の写真の内容を表す実体グラフ

2.3 シェイプグラフ (Shape Graph)

集積グラフ内の実体グラフの構造を表現するためにシェイプグラフが導入されている。すなわち、シェイプグラフはデータ

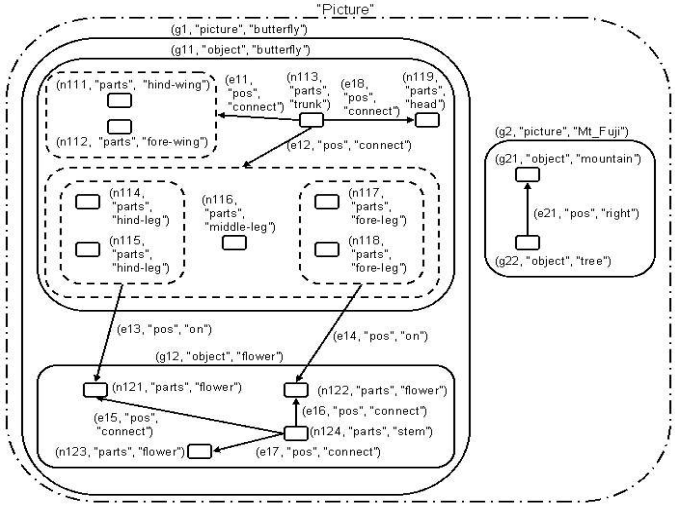

図 3 集積グラフ

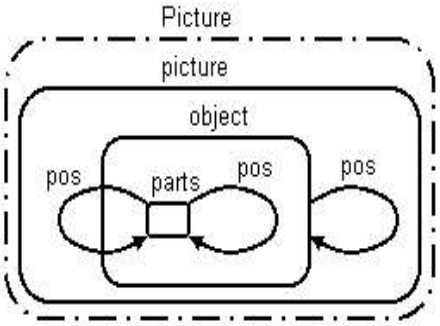

図 4　シェイプグラフ

定義のためのグラフである。

例 3. 図 4 に図 3 に示した集積グラフ Picture のシェイプグラフを示す。このシェイプグラフは以下の構造により表現される。実体グラフ picture は実体グラフ object を含んでいる。実体グラフ object は実体グラフ pos により実体グラフ object につながっている。実体グラフ object は実体グラフ parts を含んでいる。実体グラフ parts は、実体グラフの内側か外側の囲み pos により実体グラフ parts につながっている。

シェイプグラフは集積グラフの作成より前に存在している必要はない。しかし、集積グラフの作成の前に存在しても良い。また、シェイプグラフは集積グラフが存在している間存在しなければならない。実体グラフの定義をするシェイプグラフがまだ存在していない時は、実体グラフの挿入の結果、シェイプグラフが作成される。また、実体グラフの更新の結果においても、シェイプグラフが作成されることがある。以上より、DRHM はデータ主導のデータモデルと言える。しかし、一度シェイプグラフが作成されると、それらは実体グラフの消去では消去されない。シェイプグラフは、シェイプグラフの消去によってのみ消去できる。シェイプグラフのエッジはシェイプエッジと呼ばれる。同じ名前を持つ実体グラフはシェイプグラフにマッピングされ、 その名前は実体グラフと同じになる。集積グラフ内のエッジで, 同じ名前を持つものは, 始点要素が同じシェイプグラフ $sg_1^{init}, \cdots$, and sg_n^{init} にマッピングされ, 終点要素は同じシェイプグラフ $sg_1^{term}, \cdots$, and sg_m^{term} にマッピングされる。 また、エッジもシェイプエッジにマッピングされる。 そのシェイプエッジの名前はエッジの名前と同じになり、始点要素はシェイプグラフ $sg_1^{init}, \cdots$, and sg_n^{init} で、終点要素はシェイプグラフ $sg_1^{term}, \cdots$, and sg_m^{term} となる。

シェイプグラフの構造は 9 つ組 $(nm_{sg}, V_s, E_s, L_{v_s}, L_{e_s}, \phi_{v_s}, \phi_{e_s}, \phi_{connect_s}, \phi_{comp_s})$ で表現される。ここで、それぞれ以下である。

nm_{sg}:シェイプグラフの名前、

V_s:シェイプグラフの集合、

E_s:シェイプエッジの集合、

L_{v_s}:シェイプグラフのラベルの集合、

L_{e_s} シェイプエッジのラベルの集合、

ϕ_{v_s}:シェイプグラフの集合からシェイプグラフのラベルの集合への関数 $(\phi_{v_s} : V_s \to L_{v_s})$、

ϕ_{e_s}:シェイプエッジの集合からシェイプエッジのラベルの集合への関数 $(\phi_{e_s} : E_s \to L_{e_s})$、

$\phi_{connect_s}$:シェイプグラフの集合間のつながりを表す部分関数 $(\phi_{connect_s} : E_s \to 2^{V_s} \times 2^{V_s})$、

ϕ_{comp_s}:包含関連を表す部分関数 $(\phi_{comp_s} : V_s \cup \{g\} \to 2^{V_s \cup E_s})$

また、シェイプグラフもしくはシェイプエッジのラベルは 3 つ組 (s_{id}, nm_d, DT) で表される。ここで、s_{id} は識別子で、nm_d はシェイプグラフとシェイプエッジの名前である。そして、DT はデータ型の集合である。このラベルはシェイプラベルと呼ばれる。

シェイプグラフ $(nm_{sg}, V_s, E_s, L_{v_s}, L_{e_s}, \phi_{v_s}, \phi_{e_s}, \phi_{connect_s}, \phi_{comp_s})$ と集積グラフ $(nm_{cg}, V, E, L_v, L_e, \phi_v, \phi_e, \phi_{connect}, \phi_{comp})$ には以下の関係がある。

- $nm_{cg} = nm_{sg}$
- $nm(L_{v_s}) \supseteq nm(L_v)$,
- $nm(L_{e_s}) \supseteq nm(L_e)$,
- マッピング $\theta_v : V \to V_s$ s.t. $^{\forall} v \in V \ ^{\exists} v_s \in V_s (nm(\phi_v(v)) = nm(\phi_{v_s}(v_s)) \wedge \theta_v(v) = v_s)$,
- マッピング $\theta_e : E \to E_s$ s.t. $^{\forall} e \in E \ ^{\exists} e_s \in E_s (nm(\phi_e(e)) = nm(\phi_{e_s}(e_s)) \wedge \theta_e(e) = e_s)$, $\phi_{connect}(e) = (U, W) \Rightarrow \phi_{connect_s}(\theta_e(e)) = (\Theta_v(U), \Theta_v(W))$, ここで、$\Theta_v(U)$ は実体グラフの集合 $U = \{v_1, \cdots, v_n\}$ におけるシェイプグラフの集合 $\{\theta_v(v_1), \cdots, \theta_v(v_n)\}$ を意味する。
- $\phi_{comp}(v) = U \cup Z \Rightarrow \phi_{comp_s}(\theta_v(v)) = \Theta_v(U) \cup \Theta_e(Z)$, ここで、$\Theta_e(Z)$ はエッジの集合 $Z = \{e_1, \cdots, e_n\}$ における シェイプエッジ $\{\theta_e(e_1), \cdots, \theta_e(e_n)\}$ を意味する。

3. 特化・汎化関連

特化・汎化関連は、基本的なデータ関連の一つであり、複数のデータ間における意味的階層性を生成する上でなくてはならない関連である。特化関連とは、あるオブジェクトと、そのオブジェクトの特徴を持ちながら、より詳細な特徴を有するオブジェクトとの間の関連である。一方、汎化関連とは、複数のオブジェクトと、それらが共通に有する特徴を持った、より汎用的なオブジェクトとの関連である。

特化・汎化関連は多くのデータモデルに導入されている。例えば、ER モデルでは、実体間の関連において、ISA 関連を導入することにより特化・汎化関連の表現が可能となっている [4]。Abiteboul らが提案している IFO モデルでは、データの基本的な単位であるオブジェクトにおけるスーパタイプとサブタイプ間に ISA 関連の二つのタイプとして、特化関連と汎化関連を定義している。このデータモデルのグラフでは、二種類の矢印により両者を区別している [5]。ここでは、ER モデルと IFO モデルを挙げたが、これらのモデルは、データ挿入に従ってデータ定義を行わなければならないスキーマ主導のデータモデルである。

一方、Tanaka らは、データ主導のデータモデルである Obase モデルを提案している [6]。Obase モデルでは、Obase オブジェクトの中に、サブオブジェクトの集合を持ち、その間にスー

パーオブジェクト・サブオブジェクト関連が導入されている。それにより、オブジェクトの持つプロパティを上方または下方へ継承できる。また、継承演算子 (上 (下) 方への継承、左 (右) への継承) を導入することにより、階層構造の表現により幅を持たせている。上 (下) 方への継承は、ISA 関連に相当する継承である。左 (右) への継承は構成関係を表す IS-PART-OF 関連を持つオブジェクト間の継承である。また、Obase モデルはスキーマレスのモデルであり、非常に柔軟なデータモデルである [6]。

4. スキーマグラフ

2. で述べた三つのグラフに加えて、本論文では、新たにスキーマグラフという名称のグラフを導入する。このグラフはスキーマレベルのグラフであり、対象とするのは複数のシェイプグラフである。このグラフは、対象としているマルチメディアデータの内容表現の中から、階層的に包含関係を持つデータ定義群をグループ化するグラフである。より容易な管理を目指すために、スキーマグラフでは特化関連と汎化関連を定義可能とする。この機能により、より実世界に近いデータ定義の系統化を目指している。

4.1 特化枝と汎化枝

スキーマグラフでは、シェイプグラフ間の特化関連を直線の矢印で表記し、汎化関連を点線の矢印で表現する。

図 5 に特化関連の例を示す。Student と Teacher は Person を特化したものであることを表している。なお、図中の Person、Student、および、Teacher はシェイプグラフである。Student と Teacher は Person を特化したものであるので、Person が持つ要素を Student と Teacher も持っている。

図 6 に汎化関連の例を示す。Person は Student と Teacher を汎化したものであることを表している。Person は Student と Teacher が共通に持つ要素を持っている。

スキーマグラフでは、複数のシェイプグラフが階層構造を形成しているが、それらのシェイプグラフの中で、実際に集積グラフにより実体グラフを保持できるのは、最下層のシェイプグラフだけとする。また、一つ上の階層のシェイプグラフ X と同じ構造を持つシェイプグラフ_X を必ず持つものとする。集積グラフ_X には、シェイプグラフ X の下位の_X 以外のすべてのシェイプグラフに対応する集積グラフに属さない実体グラフが格納される。例えば、図 5 の場合、Student でも Teacher でもない Person の実体グラフは集積グラフ_Person に格納される。実際にスキーマグラフを使用する際には、例えば、あるシェイプグラフ A を特化元とし、シェイプグラフ B を特化先とすると、(シェイプグラフ A に対応する) 集積グラフ A に保持されていた実体グラフはシェイプグラフ_A に対応する集積グラフに移行されるので、そこから集積グラフ B に必要な実体グラフを移行するという手続きを取る必要がある。

4.2 ラ ベ ル

スキーマグラフは、シェイプグラフ間の特化・汎化関連を行うグラフであるので、個々のシェイプグラフは、そのグラフと関連が形成される相手方のシェイプグラフの情報を保持しなければならない。相手方のグラフの情報を保持するため、シェイプグラフのラベルを 4 つ組 (id,PID,name,DT) に拡張する。ここで、id は識別子、PID は、関連が形成されるあるシェイプグラフにおいて、それと関連を形成しているシェイプグラフの識別子の集合を保持するもので、PID は PID=({ id } , PID) という形式である。ここでの id は最直近で関連を形成しているシェイプグラフの識別子である。また、PID は入れ子構造となっていて、関連の相手方が関連しているグラフなど、そのグラフとは最直近で関連を形成していないが、関連を形成しているシェイプグラフの識別子まで保持する。name は名前、DT はデータ型の集合である。

4.3 重 複 要 素

継承を許可すると、特化元 A の要素 x と同じ名前を持つ特化先 B の要素 x について考えなければならない。この場合、(1) 特化先の要素のみを持つか、(2) どちらの要素を持つかを指定させるという方法が採られることが多い。

これに対し、ここでは、特化元 A の要素 x も特化先 B の要素 x も持つこととする。要素 x は、要素名のみでは識別できないので、A.x と B.x として識別する。A が S の特化先となっていて S にも要素 x がある場合は、S.A.x も持つことになる。そして、それぞれに異なる値を保持可能とする。

4.4 データ型の木

ここで、名前が同じである複数のシェイプグラフを特化 (汎化) させたときに、データ型はどう扱うのかという問題が生じる。例えば、数字を扱う場合、同じ名前の要素を、データによっては int で扱っていたり、float で扱っていたりすることも起こり得る。この問題に対処するため、スキーマグラフでは、データ型の木を導入する。図 7 にデータ型の木の一部を示す。木の根は全てのデータ型を表す Value であり、根に近くなるにつれてより一般的になっている。この木に基づいてデータ型の特化・汎化関連は対処される。例えば、同じ名前を持つが、違うデータ型 (例 Int と Float) を持つ複数のグラフを特化あるいは汎化させた場合、木における共通の枝元にあるデータ型 (例 Number) を持つことになる。

4.5 検 索

スキーマグラフの階層の中で、上位の階層のシェイプグラフを検索すれば、それより下位に存在するすべてのシェイプグラフも検索の対象となる。あるシェイプグラフ X の下位に存在するすべてのシェイプグラフを検索の対象にしたくない場合、その下位に存在するシェイプグラフ_X を検索の対象とすればよい。

5. スキーマグラフにおける特化・汎化関連の振る舞い

ここでは、スキーマグラフの特化・汎化関連の振る舞いを示す。

5.1 汎化の振る舞い

5.1.1 初めて汎化する時の振る舞い

新たに汎化元から汎化されたシェイプグラフと汎化元との間に汎化関連が出来る。このシェイプグラフを X とすると、X の

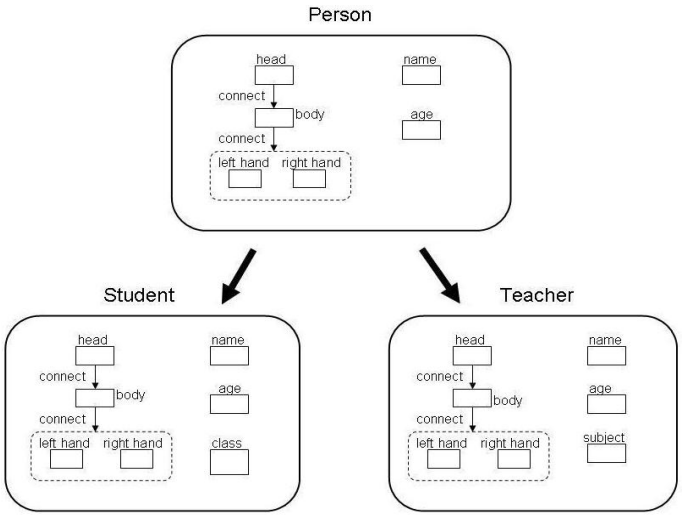

図 5　スキーマグラフ (特化)

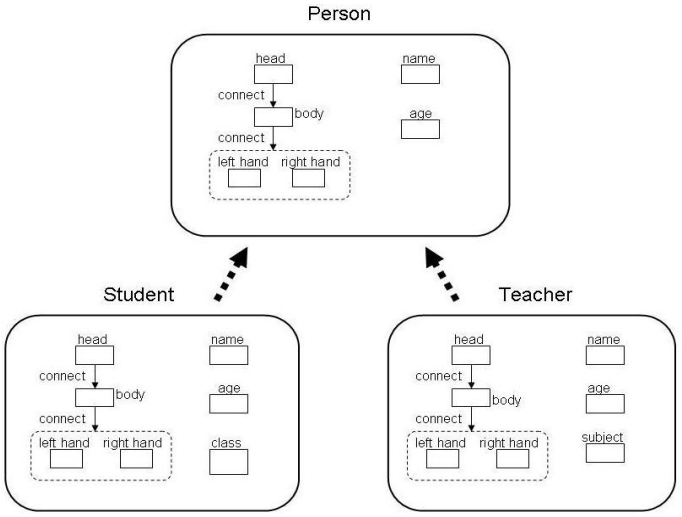

図 6　スキーマグラフ (汎化)

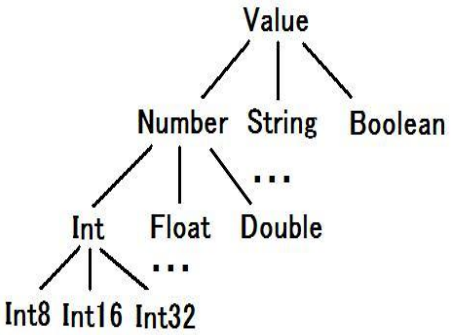

図 7　データ型の木

汎化元として X と同じ構造を持つシェイプグラフ_ X が作成される。

5.1.2　汎化関連から汎化先が削除された時の振る舞い

(1) 一つの汎化先が削除される時:　汎化先が消えたとしても、汎化元には影響はない。

(2) 汎化先の要素が削除される時:　汎化先の要素は削除できない。

5.1.3　汎化元の変更における注意

汎化関連を形成しているスキーマグラフにおいて、ある汎化先のシェイプグラフ X と、それと同じ構造の汎化元のシェイプグラフ_ X がある場合を考える。ここで、汎化元の変更により、汎化先のシェイプグラフ X が変更された場合、シェイプグラフ_ X には、その変更が適用されないため、シェイプグラフ X とシェイプグラフ_ X の要素が異なってしまう場合がある。ここで、汎化元の変更の場合により振る舞いが異なる。

(A) 汎化元に要素が追加される場合:　この場合は、全汎化元のシェイプグラフ (シェイプグラフ_ X も含む) に共通の要素を持たなければ、汎化先のシェイプグラフ X は変更されない。つまり、シェイプグラフ X にその要素の追加をしたければ、シェイプグラフ_ X にも同様の要素を追加する必要がある。

(B) 汎化元に新たなシェイプグラフが追加される場合や、汎化元の要素が削除された結果、汎化先のシェイプグラフ X の要素が削除される場合:　この場合は、以下に示す二つの場合により振る舞いが異なる。

(i) 汎化元のシェイプグラフ_ X に対応する集積グラフに実体グラフがある時:　まず、シェイプグラフ_ X の名前を変更する。これを行わない限り、汎化関連の更新は行えないこととする。そして、名前の変更後、汎化をし直すこととする。その結果、新シェイプグラフ_ X が形成される。

(ii) シェイプグラフ_ X に対応する集積グラフに実体グラフがない時:　汎化先のシェイプグラフ X の更新と同時に、汎化元のシェイプグラフ_ X の更新も行われる。

5.1.4　汎化関連に汎化元が追加された時の振る舞い

(1) 一つの汎化元が追加される時:　汎化元を更新するために汎化し直す。

(2) 一つの汎化元の要素が追加される時:　汎化元を更新するために汎化し直す。

5.1.5　汎化関連から汎化元が削除された時の振る舞い

(1) 一つの汎化元が削除される時:　汎化元を更新するために汎化し直す。

(2) 一つの汎化元の要素が削除される時:　汎化元を更新するために汎化し直す。

5.2　特化の振る舞い

5.2.1　特化関連に特化先が追加された時の振る舞い

(1) もともと存在していたシェイプグラフが特化先として追加される時:　もともと存在していたシェイプグラフが特化先として追加される時は、特化元とその特化先に特化関連が形成される。特化元のシェイプグラフに対応する集積グラフ X に保持されている実体グラフは、集積グラフ_ X に移行する。

(2) 新たにシェイプグラフを作成して、それを特化先として追加する時:　新たに生成されたシェイプグラフに、特化元の要素が付加されてから、特化関連が形成される。特化元のシェイプグラフに対応する集積グラフ X に保持されている実体グラフは、集積グラフ_ X に移行する。

(3) 特化先の要素が追加される時:　特化元にはその要素は追加されない。

5.2.2　特化関連から特化先が削除された時の振る舞い

(1) 一つの特化先が削除される時:　一つの特化先が削除されたとしても、特化元はそのまま残る。

(2) 全ての特化先が削除される時: 全ての特化先がなくなったとしても、特化元はそのまま残る。
(3) 一つの特化先の要素が削除される時: (A) その要素が、特化元の中にある要素の場合は、削除できない。 (B) 削除される要素が特化元の中にない要素の場合は、削除でき、特化元は何も変更されない。

5.2.3 特化関連に特化元が追加された時の振る舞い
(1) 一つの特化元が追加される時: 新たな特化元が追加されると、その特化元の中にある要素がもともとあった特化先に付加される。新たに追加した特化元のシェイプグラフに対応する集積グラフ X に保持されている実体グラフは、集積グラフ_X に移行する。
(2) 一つの特化元の要素が追加される時: 特化先の対応する部分にその要素が追加される。

5.2.4 特化関連から特化元が削除された時の振る舞い
(A) 特化先のシェイプグラフに対応する集積グラフに実体グラフがない時:
(1) 一つの特化元が削除される時: 削除された特化元の中にあった要素が、特化先から削除される。
(2) 全ての特化元が削除される時: それに応じて、特化先も削除される。
(3) 一つの特化元の要素が削除される時: 特化先の対応する要素が削除される。
(B) 特化先のシェイプグラフに対応する集積グラフに実体グラフがある時:
削除できない。同等のことをしたい場合については、5.3.2 を参照のこと

5.3 シェイプグラフ進化への対応について

5.3.1 汎化関連について
汎化関連を形成している汎化先のシェイプグラフ A と汎化元のシェイプグラフ B、C があるとする。シェイプグラフ B の要素で、シェイプグラフ B では必要なくなったが、シェイプグラフ A で必要な場合は、次の方法を採ることができる。シェイプグラフ B をシェイプグラフ A の汎化元から外し、シェイプグラフ B からその要素を削除し、シェイプグラフ B とシェイプグラフ C をもとに新たなシェイプグラフ A' を汎化により作成する。

5.3.2 特化関連について
特化関連を形成している特化元のシェイプグラフ A と特化先のシェイプグラフ B、C があるとする。シェイプグラフ A の要素で、シェイプグラフ A には必要なくなったが、シェイプグラフ B、C には必要な要素がある時がある。例えば、シェイプグラフ A とシェイプグラフ B、C に同じ名前の要素があるが、保持している内容が違う場合である。この場合、次の二つの方法を取ることが出来る。
[方法 1] 要素を持っているシェイプグラフ A と要素を持っていない変更後のシェイプグラフ (A') を並列に存在させる。手順として、まず、シェイプグラフ A' を作り、シェイプグラフ A' とシェイプグラフ B、C を関係づけて、必要に応じて値をコピーする。
[方法 2] シェイプグラフ A を特化先とするシェイプグラフ A' を作成する。

6. スキーマグラフの形式的な定義

ここでは、スキーマグラフの形式的な定義を示す。スキーマグラフ scg が n 個のシェイプグラフ $sg_i = (nm_{sgi}, V_i, E_i, L_{vi}, L_{ei}, \phi_{vi}, \phi_{ei}, \phi_{connecti}, \phi_{compi})(i = 1, \cdots, n)$ を持つとする。このとき、スキーマグラフ scg は5つ組 $scg = (SG, E_{汎化}, E_{特化}, \phi_{E_{汎化}}, \phi_{E_{特化}})$ である。ここで、
SG : シェイプグラフの集合 ($SG = sg_1 \cup \cdots \cup sg_n$)、
$E_{汎化}$: 汎化関連のエッジの集合、
$E_{特化}$: 特化関連のエッジの集合、
$\phi_{E_{汎化}}$: 汎化関連のあるシェイプグラフの集合の要素を示す関数 ($\phi_{E_{汎化}} : E_{汎化} \rightarrow 2^{SG} \times SG$)、
$\phi_{E_{特化}}$: 特化関連のあるシェイプグラフの集合の要素を示す関数 ($\phi_{E_{特化}} : E_{特化} \rightarrow 2^{SG} \times 2^{SG}$)、

以降では、ある汎化枝 $e \in E_{汎化}$における $E_{汎化}$の始点集合の要素となる l 個の汎化元のシェイプグラフを $sg_{g(init)1}, \cdots, sg_{g(init)l}$とし、$E_{汎化}$の終点となる汎化先のシェイプグラフを $sg_{g(term)}$とする。また、ある特化枝 $e \in E_{特化}$における $E_{特化}$の始点集合の要素となる m 個の特化元のシェイプグラフを $sg_{s(init)1}, \cdots, sg_{s(init)m}$とし、$E_{特化}$の終点集合の要素となる p 個の特化先のシェイプグラフを $sg_{s(term)1}, \cdots, sg_{s(term)p}$とする。

また、pnd(a) という表記を、要素 a に付けられたラベルから PID と name と DT を取り出す関数とすると、以下に出てくる ^{corr} という表記を、ある要素集合 A に対して $A^{corr} = \{a |\ pnd(a) = pnd(a_i) \ \wedge \ a_i \in A\}$ という意味で用いる。つまり、要素集合 A の要素に付けられたラベルと name,PID,DT が同じであるラベルを持つ要素からなる集合を A^{corr} と表記する。

6.1 汎化の振る舞い

$SG_{g(init)} = \{sg_{g(init)i} | 1 \leqq i \leqq l\}$ とする。また、$V_{g(init)} = V_{g(init)1} \cup \cdots \cup V_{g(init)l}$、$E_{g(init)} = E_{g(init)1} \cup \cdots \cup E_{g(init)l}$ とする。

6.1.1 初めて汎化する時の振る舞い
初めて汎化するとは、$V_{g(term)} = (V_{g(init)1} \cap \cdots \cap V_{g(init)l})^{corr}$、$E_{g(term)} = (E_{g(init)1} \cap \cdots \cap E_{g(init)l})^{corr}$ を持つ、新たなシェイプグラフ $sg_{g(term)}$ を生成し、$V^{corr}_{g(term)}$ と $E^{corr}_{g(term)}$ を持つシェイプグラフ sg_{other} を生成し、$\phi_{汎化}(e) = (SG_{g(init)}, sg_{g(term)})$ なる e について $E_{汎化} = E_{汎化} \cup \{e\}$ とすることである。

6.1.2 汎化関連から汎化先が削除された時の振る舞い
(1) 一つの汎化先が削除される時: シェイプグラフ $sg_{g(term)}$ を汎化枝 e$\in E_{汎化}$ の汎化先から削除するとは、$SG = SG - \{sg_{g(term)}\}$ となり、$E_{汎化} = E_{汎化} - \{e\}$ となることである。
(2) 汎化先の要素が削除される時: 汎化先の要素は削除できない。

6.1.3 汎化関連に汎化元が追加された時の振る舞い
(1) 一つの汎化元が追加される時: あるシェイプグラフ

$sg_{g(init)x}(x=1,\cdots,l)$ を汎化枝 e∈ $E_{汎化}$ の汎化元に追加するとは、$\phi_{E\ 汎化}=(SG_{g(init)},sg_{g(term)})$ が $\phi_{E\ 汎化}=(SG_{g(init)}\cup\{sg_{g(init)x}\},sg_{g(term)})$ となり、$V_{g(term)}=V^{corr}_{g(init)}\cap V^{corr}_{g(init)x}$、$E_{g(term)}=E^{corr}_{g(init)}\cap E^{corr}_{g(init)x}$ となることである。

(2) 一つの汎化元の要素が追加される時: 汎化枝 e∈ $E_{汎化}$ において、あるシェイプグラフ $sg_{g(init)x}(x=1,\cdots,l)$ の要素 ($v\in V_{g(init)x}$ または、$edge\in E_{g(init)x}$) を追加するとは、$V_{g(init)x}=V_{g(init)x}\cup\{v\}^{corr}$ 、または、$E_{g(init)x}=E_{g(init)x}\cup\{edge\}^{corr}$ となり、$V_{g(term)}=V^{corr}_{g(init)}\cap V^{corr}_{g(init)x}$、$E_{g(term)}=E^{corr}_{g(init)}\cap E^{corr}_{g(init)x}$ となることである。

6.1.4 汎化関連から汎化元が削除された時の振る舞い

(1) 一つの汎化元が削除される時: あるシェイプグラフ $sg_{g(init)x}$ を汎化枝 e∈ $E_{汎化}$ の汎化元から削除するとは、$\phi_{E\ 汎化}(e)=(SG_{g(init)},SG_{g(term)})$ が $\phi_{E\ 汎化}(e)=(SG_{g(init)}-\{sg_{g(init)x}\},sg_{g(term)})$ となり、$V_{g(term)}=V^{corr}_{g(init)}\cap V^{corr}_{g(init)x}$、$E_{g(term)}=E^{corr}_{g(init)}\cap E^{corr}_{g(init)x}$ となることである。

(2) 一つの汎化元の要素が削除される時: 汎化枝 e∈ $E_{汎化}$ において、あるシェイプグラフ $sg_{g(init)x}(x=1,\cdots,l)$ の要素 ($v\in V_{g(init)x}$ または、$edge\in E_{g(init)x}$) を削除するとは、$V_{g(init)x}=V_{g(init)x}-\{v\}^{corr}$ 、または、$E_{g(init)x}=E_{g(init)x}-\{edge\}^{corr}$ となり、$V_{g(term)}=V^{corr}_{g(init)}\cap V^{corr}_{g(init)x}$、$E_{g(term)}=E^{corr}_{g(init)}\cap E^{corr}_{g(init)x}$ となることである。

6.2 特化の振る舞い

$SG_{s(init)}=\{sg_{s(init)i}|1\leqq i\leqq m\},SG_{s(term)}=\{sg_{s(term)i}|1\leqq i\leqq m\}$ とする。また、$V_{s(init)}=V_{s(init)1}\cup\cdots\cup V_{s(init)m},V_{s(term)}=V_{s(term)1}\cup\cdots\cup V_{s(term)p}$, $E_{s(init)}=E_{s(init)1}\cup\cdots\cup E_{s(init)m},E_{s(term)}=E_{s(term)1}\cup\cdots\cup E_{s(term)p}$ とする。

6.2.1 特化関連に特化先が追加された時の振る舞い

(1) もともと存在していたシェイプグラフが特化先として追加される時: あるシェイプグラフ $sg_{s(term)x}$ を特化枝 e∈ $E_{特化}$ の特化先に追加するとは、$\phi_{E\ 特化}(e)=(SG_{s(init)},SG_{s(term)})$ が $\phi_{E\ 特化}(e)=(SG_{s(init)},SG_{s(term)}\cup\{sg_{s(term)x}\})$ となり、$V_{s(term)x}=V_{s(term)x}\cup V^{corr}_{s(init)},E_{s(term)x}=E_{s(term)x}\cup E^{corr}_{s(init)}$ となることである。

(2) 新たにシェイプグラフを作成して、それを特化先として追加する時: 特化枝 e∈ $E_{特化}$ において、新たにシェイプグラフを特化先に追加するとは、$V_{s(term)x}=V^{corr}_{s(init)},E_{s(term)x}=E^{corr}_{s(init)}$ なる新たなシェイプグラフ $sg_{s(term)x}$ と $V^{corr}_{s(init)}$ と $E^{corr}_{s(init)}$ を持つ sg_{other} を作成し、$\phi_{E\ 特化}(e)=(SG_{s(init)},SG_{s(term)})$ が $\phi_{E\ 特化}(e)=(SG_{s(init)},SG_{s(term)}\cup\{sg_{s(term)x},sg_{other}\})$ となることである。

(3) 特化先の要素が追加される時: 特化枝 e∈ $E_{特化}$ において、あるシェイプグラフ $sg_{s(term)x}(x=1,\cdots,p)$ の要素 ($v\in V_{s(term)x}$、または、$edge\in E_{s(term)x}$) を追加するとは、$V_{s(term)}(e)=V_{s(term)}(e)\cup\{v\}$、または、$E_{s(term)}(e)=E_{s(term)}(e)\cup\{edge\}$ となることである。

6.2.2 特化関連から特化先が削除された時の振る舞い

(1) 一つの特化先が削除される時: あるシェイプグラフ $sg_{s(term)x}(x=1,\cdots,p)$ を特化枝 e∈ $E_{特化}$ の特化先から削除するとは、$\phi_{E\ 特化}(e)=(SG_{s(init)},SG_{s(term)})$ が $\phi_{E\ 特化}(e)=(SG_{s(init)},SG_{s(term)}-\{sg_{s(term)x}\})$ となることである。

(2) 全ての特化先が削除される時: 特化枝 e∈ $E_{特化}$ の特化先から全てのシェイプグラフを削除するとは、$SG_{s(term)}=\{\}$ となり、$E_{特化}=E_{特化}-\{e\}$ となることである。

(3) 一つの特化先の要素が削除される時: 特化枝 e∈ $E_{特化}$ の特化先のシェイプグラフ $sg_{s(term)x}(x=1,\cdots,p)$ の要素 ($v\in V_{s(term)x}$ または、$edge\in E_{s(term)x}$) を削除する場合、 (A) 特化元の中にある要素 $v\in V_{s(init)}$、または、$edge\in E_{s(init)}$ ならば、v、または、edge は削除できない。 (B) 特化元の中にない要素 $v\notin V_{s(init)}$ 、または、$edge\notin E_{s(init)}$ ならば、$V_{s(term)}=V_{s(term)}-\{v\}^{corr}$ または、$E_{s(term)}=E_{s(term)}-\{edge\}^{corr}$ となることである。

6.2.3 特化関連に特化元が追加された時の振る舞い

(1) 一つの特化元が追加される時: あるシェイプグラフ $sg_{s(init)x}$ を特化枝 e∈ $E_{特化}$ の特化元に追加するとは、$V^{corr}_{s(init)x}$ と $E^{corr}_{s(init)x}$ を持つシェイプグラフ sg_{other} を作成し、$\phi_{E\ 特化}(e)=(SG_{s(init)},SG_{s(term)})$ が $\phi_{E\ 特化}(e)=(SG_{s(init)}\cup\{sg_{s(init)x}\},SG_{s(term)}\cup sg_{other})$ となり、$V_{s(term)i}=V_{s(term)i}\cup V^{corr}_{s(init)x},E_{s(term)i}=E_{s(term)i}\cup E^{corr}_{s(init)x}(i=1,\cdots,p)$ となることである。

(2) 一つの特化元の要素が追加される時: 特化枝 e∈ $E_{特化}$ において、あるシェイプグラフ $sg_{s(init)x}(x=1,\cdots,m)$ の要素 ($v\in V_{s(init)x}$、または、$edge\in E_{s(init)x}$) が追加されるとは、$V_{s(init)x}=V_{s(init)x}\cup\{v\}$、または、$E_{s(init)x}=E_{s(init)x}\cup\{edge\}$ となり、$V_{s(term)i}=V_{s(term)i}\cup\{v\}^{corr}$ または、$E_{s(term)i}=E_{s(term)i}\cup\{edge\}^{corr}(i=1,\cdots,p)$ となることである。

6.2.4 特化関連から特化元が削除された時の振る舞い

(A) 特化先のシェイプグラフに対応する集積グラフに実体グラフがない時

(1) 一つの特化元が削除される時: あるシェイプグラフ $sg_{s(init)x}(x=1,\cdots,m)$ を特化枝 e∈ $E_{特化}$ の特化元から削除するとは、$\phi_{E\ 特化}(e)=(SG_{s(init)},SG_{s(term)})$ が $\phi_{E\ 特化}(e)=(SG_{s(init)}-\{sg_{s(init)x}\},SG_{s(term)})$ となり、$V_{s(term)i}=V_{s(term)i}-V^{corr}_{s(init)x}$、$E_{s(term)i}=E_{s(term)i}-E^{corr}_{s(init)x}(i=1,\cdots,p)$ となることである。

(2) 全ての特化元が削除される時: 特化枝 e∈ $E_{特化}$ の特化元から全てのシェイプグラフを削除するとは、$SG=SG-SG_{s(init)}-SG_{s(term)}$となり、$E_{特化}=E_{特化}-\{e\}$ となることである。

(3) 一つの特化元の要素が削除される時: 特化枝 e∈ $E_{特化}$ において、あるシェイプグラフ $sg_{s(init)x}(x=1,\cdots,m)$ の要素 ($v\in V_{s(init)x}$ または、$edge\in E_{s(init)x}$) が削除されるとは、$V_{s(init)x}=V_{s(init)x}-\{v\}$、または、$E_{s(init)x}=E_{s(init)x}-\{edge\}$ となり、$V_{s(term)i}=V_{s(term)i}-\{v\}^{corr}$、または、$E_{s(term)i}=E_{s(term)i}-\{edge\}^{corr}(i=1,\cdots,p)$ となることである。

(B) 特化先のシェイプグラフに対応する集積グラフに実体グラ

フがある時
あるシェイプグラフ $sg_{s(init)x}(x = 1, \cdots, m)$ を特化枝 e∈ $E_{特化}$ の特化元から削除することや、$sg_{s(init)x}$ の要素 ($v \in V_{s(init)x}$ または、$edge \in E_{s(init)x}$) を削除することは出来ない。

7. 適 用 例

汎化関連の適用例を図 8 と図 9 に示す。図 9 では図 8 に示した Student と Teacher というシェイプグラフをもとに Person というシェイプグラフを作成した例である。Student でも Teacher でもない実体グラフのためのシェイプグラフ_ Person が作成されている。

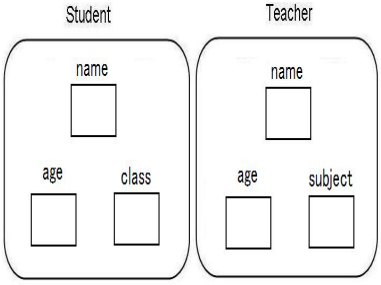

図 8 汎化関連の適用例 (a)

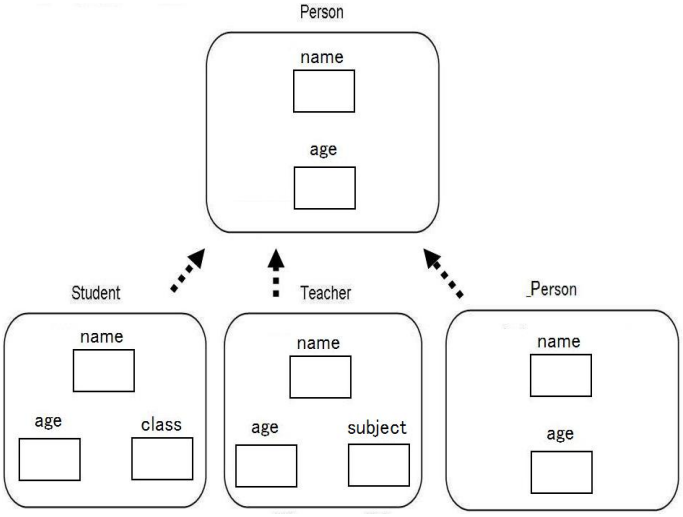

図 9 汎化関連の適用例 (b)

次に、特化関連の適用例を図 10〜図 13 に示す。まず、Person(図 10) を特化したシェイプグラフ Student を作成している (図 11)。ここでは、_ Person も作成されている。次に、Teacher を特化先としている (図 12)。さらに、Student と Teacher を特化した TA を作成している。これと同時に、_ Student と_ Teacher が作成されている (図 13)。

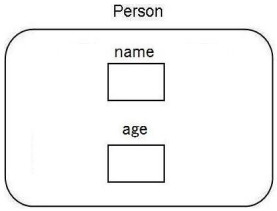

図 10 特化関連の適用例 (a)

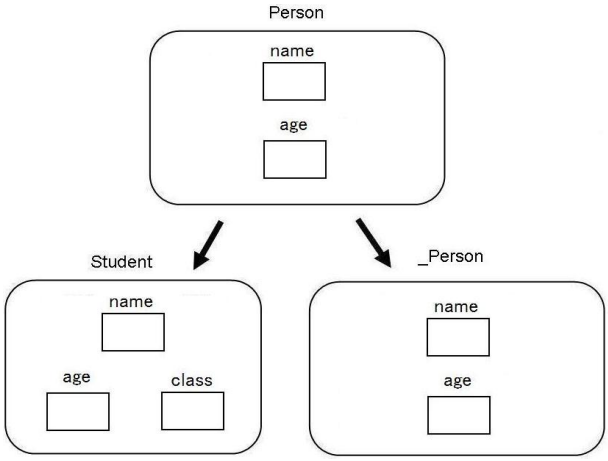

図 11 特化関連の適用例 (b)

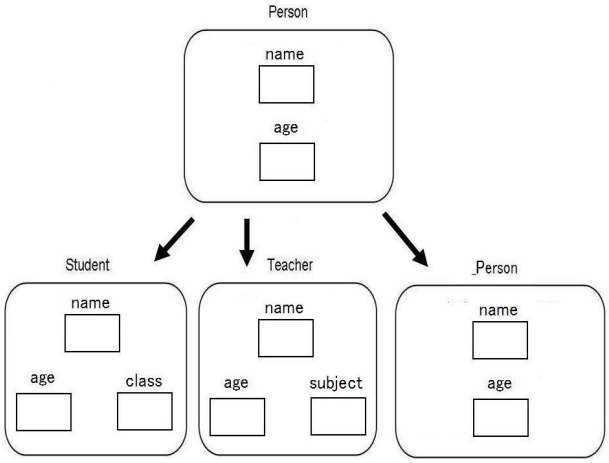

図 12 特化関連の適用例 (c)

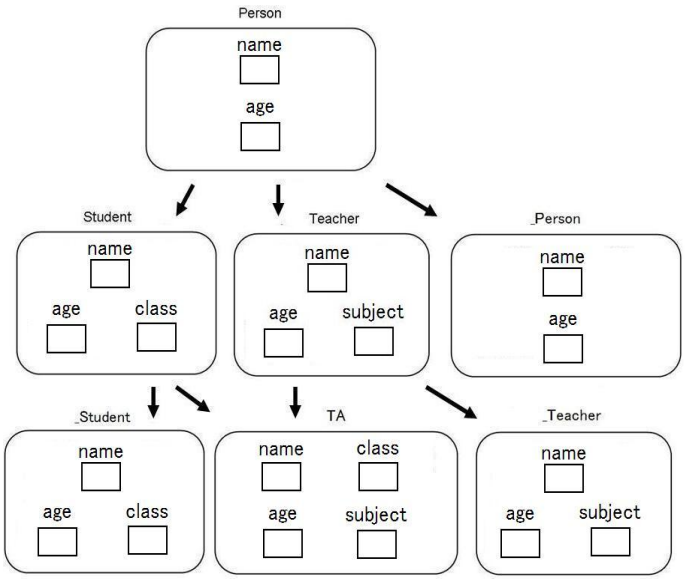

図 13 特化関連の適用例 (d)

8. おわりに

本論文では、再帰有向超グラフデータモデルに高度なモデリング能力を導入することを目的として、スキーマグラフと呼ぶグラフを導入した。スキーマグラフは、データ定義レベルのグラフであるシェイプグラフに、特化・汎化関連を導入したもの

である。スキーマグラフについて概説した後、特化・汎化関連に関する振る舞いについて整理した。その後、スキーマグラフの形式的な定義を示した。

今後は、スキーマグラフにおける特化・汎化関連と他のデータモデルにおける特化・汎化との比較や現実問題への適用が課題である。

謝　　辞

本研究は、一部、文部科学省科学研究費補助金 (課題番号:20300037) による。

文　　献

[1] Petrakis, E. G. M. and Faloutsos, C., Similiarity Searching in Medical Image Databases, IEEE Trans. on Know. and Data Eng.,Vol.9(1997)435-447.

[2] Uehara, K, Oe, M and Maehara, K, Knowledge Reprensentation, Concept Acquisition and Retrieval of Video Data, Proc. of Int'l Symposium on Cooperative Database Systems for Advanced Applications (1996)218-225.

[3] 宝珍輝尚, 都司達夫, 再帰有向超グラフに基づく一データモデル, 情報処理学会論文誌：データベース, Vol.41, No.SIG 1 (TOD 5)(2000)11-21.

[4] 増永良文, リレーショナルデータベース入門 [新訂版]-データモデル・SQL・管理システム-, サイエンス社 (2003)15.

[5] Serge Abiteboul and Richard Hull, IFO: A Formal Semantic Database Model, ACM Transactions on Database Systems, Vol.12, No.4(1987)525-565.

[6] Katsumi Tanaka, Shojiro Nishio, Masatoshi Yoshikawa, Shinji Shimojo, Jyun-ya Morishita and Tsuneo Jozen, Obase Object Database Model: Towards a More Flexible Object-Oriented Database System, Proc. of Int'l. Symp. on Next Generation Database Systems and Their Applications (NDA'93) (1993)159-166.