ユーザ定義ルールを用いたXMLデータからの概念モデル抽出

Extraction of Conceptual Models from XML Data by User-Defined Rules

大河原 俊明♡　森嶋 厚行♢　杉本 重雄♠

Toshiaki OHKAWARA　Atsuyuki MORISHIMA
Shigeo SUGIMOTO

本論文では，XML データから概念モデルを抽出する一手法を提案する．本手法の特徴は，概念モデル抽出のための固定されたアルゴリズムを提供するのではなく，ユーザが概念モデル抽出のためのルールを記述するための枠組みと，与えられたルールを利用した概念モデル抽出の枠組みを提供することである．

We propose a method for extracting conceptual models from XML data. An important feature of the method is to give frameworks for allowing users to describe rules for the extraction and for using the rules to extract conceptual models, instead of just giving a fixed algorithm.

1. はじめに

近年の情報技術の急速な発展に伴い，ソフトウェアの実行環境の変化が速くなっている．また，Web サービスなどのソフトウェア間連携が現実のものになりつつあり，ソフトウェアの，動的な環境変化への対応が重要な問題となっている．我々はこれらを背景に，環境変化への対応において重要な問題の一つであるデータ変換に着目し，研究を進めている．

ここで我々の定義するデータ変換の問題とは，入力としてスキーマ S_A，S_B，およびスキーマ S_A のインスタンス I_A が与えられたとき，出力としてスキーマ S_B のインスタンス I_B を求めるものである．これを実現するためには，I_B を求めるための変換プログラムが必要であるが，その開発は一般に自明ではない．これまで，データ変換の問題は，情報統合などの文脈において研究が行われてきていたものの,「時折起こる問題」としてアプローチされることが通常であり，常に起こりうる問題としての注目はそれほど集められてこなかった．

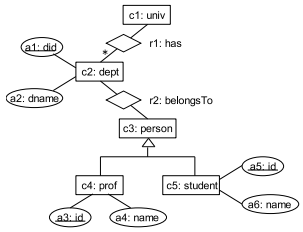
我々は，XML データを対象として，データ変換プログラムを効率良く開発するためのフレームワークの研究開発を行っている [1]．本フレームワークの特徴は，次の通りである．(1) データ変換プログラムの構築のための形式的モデルに基づくこと．(2) 単純なスキーママッチングに基づくデータ変換 [6] 以外のデータ変換にも対応するため，概念モデル (例えば，図 1 (a) で表される XML スキーマの概念モデルは図 1 (b) である) を利用したデータ変換プログラムの構築を行うこと．

我々が提案しているデータ変換モデル [2] では，データ変換を写像 $F : S_A \rightarrow S_B$ としてモデル化する (図 2 実線矢印)．また，本モデルでは XML から ER 図によって表される概念モデルのインスタンスへの写像 (**意味写像**と呼ぶ) を導入する．これにより，変換元のスキーマ S_A，および変換先のスキーマ S_B が与えられたとき，意味写像 $f : S_A \rightarrow D$ と $g : S_B \rightarrow D$ を用いて，S_A から S_B への変換は $F : S_A \rightarrow S_B \equiv f \circ g^{-1}$ と表現される (図 2 点線矢印)．ここで，D は概念モデルを表す ER 図 (例えば，図 1 (b)) によって表現される型である．

♡ 学生会員　筑波大学大学院 図書館情報メディア研究科
okwr@slis.tsukuba.ac.jp
♢ 正会員　筑波大学大学院 図書館情報メディア研究科
mori@slis.tsukuba.ac.jp
♠ 非会員　筑波大学大学院 図書館情報メディア研究科
sugimoto@slis.tsukuba.ac.jp

```
univ = (depts, people)
depts = (dept*)
dept = (@did:ID, dname)
people = (person*)
person = (@did:IDREF,
          (prof | student))
prof = (@id:ID, name)
person = (@id:ID, name)
```

(a)



(b)

図 1 XML スキーマ S_a(a) とその概念モデル D_a(b)
Fig. 1 XML schemaS_a (a) and its Conceptual Model D_a (b)

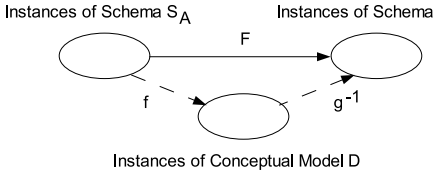


図 2 概念モデルを介したデータ変換
Fig. 2 Data Conversion through a Conceptual Model

概念モデルを介した写像による変換には次の利点がある．(1) 概念モデルレベルの知識 (オントロジなど) を変換写像の構築に利用できる可能性がある．(2) スキーマ間の直接の対応関係だけからでは自明でない関係を扱いやすい．例えば，図 1 (a) の XML スキーマでは prof 要素と student 要素は独立して存在するが，他のスキーマが同じ情報 (図 1 (b)) を表すとしても同じように表現されるとは限らない．例えば，これらは独立した要素として存在せず，person 要素だけが用意されており，prof と student の区別は属性の値によって行われるかもしれない．このような場合にでも，どちらのスキーマからも同じ概念モデルが抽出できれば，対応関係は明らかになる．

本稿では，上記における意味写像 f を半自動的に構築する一手法を提案する．我々は，これまでの研究で変換モデルの提案を行い [2]，概念モデル抽出支援システムの第一次プロトタイプシステムの開発を行ってきた [1]．しかしこれまで開発してきたシステムでは，概念モデル抽出のために利用されているルールは固定であり，かつ利用されていたアルゴリズムはそれらのルールに特化したものであった．本稿では，概念モデル抽出のためのルールの一般的な記述方法，および任意のルールが与えられたときの概念モデル抽出手法について提案する．

2. XML データからの概念モデル抽出手法の概要

2. 1 意味写像

我々が提案しているモデルでは，データ変換を意味写像の組合せとしてモデル化する．意味写像は $f : S \rightarrow D$ と表す．意味写像 $f : S \rightarrow D$ の定義域は，XML スキーマ S を満たす XML インスタンスの集合 $dom(S)$ である．意味写像 $f : S \rightarrow D$ の値域は，図 1 (b) で表されるような ER 図によって規定される値の集合 $dom(D)$ である．直観的には，$v \in dom(D)$ である値 v は，D 中の各ノード (クラス，関連，および属性) のインスタンス集合が持つタグ付きレコードである．例えば，$f : S_a \rightarrow D_a$(S_a は図 1 (a)，D_a は図 1 (b)) における D_a のインスタンスは，次の形式をしている．以下のタグ付きレコードで $c1$ や $c2$ などの各タグは，それぞれの ER

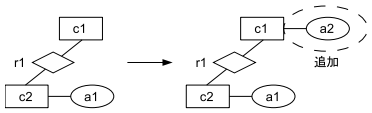


図 3 意味拡張操作子による D から D' への変更
Fig. 3 Changes by the Semantic Extension Operator

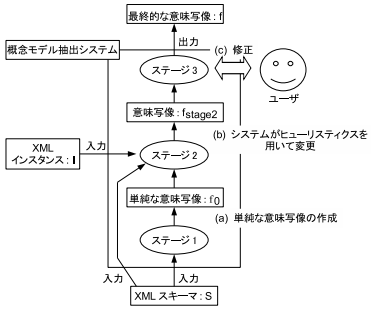


図 4 アーキテクチャ
Fig. 4 Architecture

図における各ノード (univ クラスや dept クラス) の ID である．

[$c1$: univ 要素, $c2$: dept 要素の集合, $c3$: person 要素の集合, ..., $r1$: ..., $r2$: dept 要素と person 要素の 2 項関係表, ..., $a6$: ...]

一般に，意味写像 $f : S \to D$ は，$w \in dom(S)$ なる XML インスタンス w から値 $v \in dom(D)$ を計算する．具体的には，f は，v 中の各ノードに対するインスタンス集合をそれぞれ計算するために，複数の XQuery 風の式で構成される．例えば，意味写像 $f : S_a \to D_a$ において，D_a の person クラス ($c3$) のインスタンス集合 (要素集合) を計算する式は，for $\$k$ in /univ/dept/person return $\$k$ である．

2. 2 意味写像の作成

本手法における概念モデルの抽出とは，スキーマ S を基に概念モデルを表す ER 図 D を導出し，意味写像 $f : S \to D$(図 2 左点線矢印 f) を作成することである．例えば，大学の情報を表す XML スキーマ S_a(図 1 (a)) のインスタンスを概念モデル D_a(図 1 (b)) のインスタンスに変換するための意味写像 $f : S_a \to D_a$ を作成する．

2. 3 写像操作系

本手法は，我々が定義したデータ変換モデル [2] に基づいている．本モデルは意味写像を操作するための写像操作系を定義しており，7 つの操作子 (逆写像 f^{-1}，意味合成 $f \circ g^{-1}$，ラベル変更，意味射影，意味拡張，値域制限，名前変更) を持つ．ここでは，本稿での議論で重要な意味拡張操作子について説明する．意味拡張操作子は，意味写像 $f : S \to D$ が与えられたとき，D に新たなノード n_i(クラス，関連，あるいは属性) を追加した D' への写像 $f' : S \to D'$ を作成する．図 3 に例を示す．この例における値域 D' は，D におけるクラス c_2 の属性 a_1 をクラス c_1 の属性 a_2 としてコピーすることによりノードの追加 (属性 a_2) が行われた新たな概念モデルである．

2. 4 概念モデル抽出システムのアーキテクチャ

提案手法を実現するためのシステムのアーキテクチャを図 4 に示す．本手法は，入力として XML スキーマ S とそのインスタンス I をとり，意味写像 $f : S \to D$ を作成する．本アーキテクチャの基本的な考え方は，概念モデル抽出のうち自動化できる部分はシステムが行い，それ以外はユーザが実行できるような環境を提供することである．概念モデル抽出は次の 3 段階で行われる．

(ステージ 1) 入力として XML スキーマ S をとり，単純な意味写像 f_0 を出力する (図 4 (a))．ここで単純な意味写像 $f_0 : S \to D_0$ の D_0 は，S と同型の構造を持つ ER 図である．すなわち，S 中の各 XML 要素に対して D_0 中の一つのクラスが対応し，S 中の

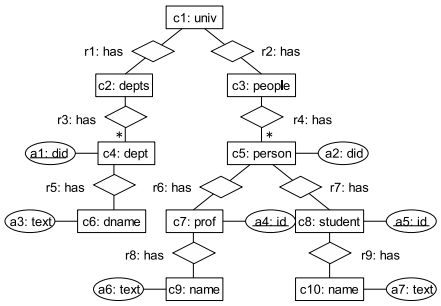


図 5 ステージ 1 が出力する D_0
Fig. 5 Output D_0 of Stage 1

要素の親子関係が D_0 の関連に直接対応する．例えば，図 1 (a) のスキーマを入力としたとき，ステージ 1 の出力 $f_0 : S \to D_0$ における D_0 は図 5 になる．

(ステージ 2) 入力としてステージ 1 の出力 f_0 と S のインスタンス I をとる．ヒューリスティクスを用いて写像操作子を f_0 に適用し，新たな意味写像 f_{stage2} を作成し，出力する (図 4 (b))．本論文で提案する手法はこの部分である．

(ステージ 3) 概念モデルを自動的に完全に抽出することは困難である．例えば，図 1 (b) の dept 要素と person 要素間の関連の名前が belongsTo であるということは，自動的にはわからない．そこで，ユーザによる修正が必要となる．本システムでは，ユーザが写像操作子を f_{stage2} に適用する (図 4 (c)) ことで，システムは最終的な意味写像 $f : S \to D$ を出力する．

3. ルールに基づいた概念モデル抽出

本章では，前章で説明したステージ 2 を実現する手法を説明する．具体的には，概念モデル抽出のためのルール記述法，および与えられたルールに基づいて概念モデルを抽出する手法について説明する．

3. 1 概念モデル抽出のためのルール記述

本手法におけるルールとは，ある意味写像 $f_i : S \to D_i$ に適用することにより，写像操作子を用いて，新しい意味写像 $f_{i+1} : S \to D_{i+1}$ を導出するものである．ここで，値域 D_{i+1} は D_i を拡張したものである．ここでは，次のヒューリスティクスを表すルールを例にとり説明する．

ヒューリスティクス 1: 概念モデルにおけるクラスの属性は，XML ではしばしば要素属性ではなく独立した要素として表現される．したがって，XML 要素 (図 3 の $c2$) がテキスト (図 3 の $a1$) だけを持ち，それ以外の子要素や属性を一切持たなければ，そのテキストの内容をそのクラスと 1 対 1 関連で接続される親クラス (図 3 の $c1$) の新たな属性 (図 3 の $a2$) とみなすことができる．

図 6 は**ヒューリスティクス 1** を表すルール記述である．ルール記述は if 節，extend 節，add constraints 節，derivation relationships 節から構成される．if 節にはルール適用の条件となる述語を記述する (図 7 に述語の一部を示す)．図 6 の if 節には次が記述されている．(1) D_i に関連名が has(関連名 has は XML 要素間の親子関係を表す) である関連 r1 で接続されているクラス $c1$，$c2$ があり，$c2$ は XML のテキストノードに対応する属性 $a1$ を持っている，(2) $r1$ は 1 対 1 関連である，(3) XML スキーマ S に，$c2$ および $a1$ に直接対応するノード (XML 要素や属性など．これらを実装ノードと呼ぶ) が存在し，かつ $a1$ は $c2$ が持つ唯一の実装ノードである．

extend 節には，D_{i+1} に新たに含まれるノードと，そのノードのインスタンスの計算方法を記述する．図 6 の extend 節に記述されていることは，(1) クラス $c1$ に属性 $a2$ を追加することと，(2) $a2$ のインスタンス集合は，関連 $r1$ と属性 $a1$ のそれぞれのインスタンス集合の結合演算によって計算された 2 項リレーションで

if:
class($c1$)
class($c2$)
rel($c1$, $c2$, $r1$)
name($r1$, "has")
attr($c2$, $a1$)
type($a1$, text)
multiplicity(1, $c1$, $r1$)
multiplicity(1, $c2$, $r1$)
hasOnlyOneImplNode($c2$, $a1$)

extend:
attr($c1$, $a2$) as $\pi_{ID,V}(r1 \bowtie a1)$

add constraints:

derivation relationships:
$c1c2r1a1 \rightarrow a2$
$c1 \rightarrow c2$
$c1c2 \rightarrow r1$
$c1c2r1a2 \rightarrow a1$

図 6 ルール記述例
Fig. 6 Example of Rule Description

述語	説明
class(c)	クラス c が存在
rel($c1$, $c2$, r)	クラス $c1$ とクラス $c2$ 間に関連 r が存在
attr(c, a)	クラス c に属性 a が存在
isa($c1$, $c2$)	クラス $c1$ isa クラス $c2$ である
name(n, "...")	要素 n(クラス，関連，あるいは属性) の名前は ... である
type(a, $type$)	属性 a のタイプ (ID，IDREF，etc.) は $type$ である
multiplicity(m, c, r)	ある関連 r のクラス c 側の多重度は m である
exclusion($r1$, $r2$, ..., rm)	ある関連 $r1$〜rm が排他制約を満たしている
hasOnlyOneImplNode(c, $node$)	あるクラス c は一つの実装ノード $node$ を持つ
hasNoAttr(c)	あるクラス c は属性を持たない

図 7 述語の説明
Fig. 7 Explanation of Predicates

ある，ということである．ここで，ID は $c1$ のキー値，V は $a1$ の属性値を表す．

add constraints 節には，D_{i+1} に新たに含まれる制約 (isa 関連など) を記述する．本例では新たな制約は追加されない．

derivation relationships 節には，ノード間の導出関係を記述する．ここで導出関係とは，どのノードのインスタンスから他のノードのインスタンスが計算可能かを表現した関係である．図 6 の derivation relationships 節には，クラス $c1$，クラス $c2$，関連 $r1$，および属性 $a2$ から属性 $a1$ が導出可能である，等が記述されている．

3. 2 ルールに基づいた 3 段階概念モデル抽出

本手法におけるステージ 2 は，図 8 で示すようにフェイズ 1〜フェイズ 3 の 3 段階の操作から構成される．以下で詳細に説明する．

(フェイズ 1) ステージ 1 で出力されるのは単純な意味写像 $f_0 : S \rightarrow D_0$ であり，D_0 は S と同型の構造を持つ．この $f_0 : S \rightarrow D_0$ に 3. 1 節で説明したルールを適用すれば，D_0 に新しくノードを追加した D_1 と，D_1 を値域に持つ $f_1 : S \rightarrow D_1$ が得られる．フェイズ 1 では，このようなルール適用を繰り返し，最終的には，もうルールの適用が行えない状態の意味写像 $f_{max} : S \rightarrow D_{max}$ を作成する．同時に，この過程で適用されたルールに記述された導出関係 (例えば，図 6 の derivation relationships 節) を得る．

(フェイズ 2) フェイズ 2 では，フェイズ 1 で得られた導出関係を利用して，D_{max} 中のノードのうち，必要最小限のノードのみを発見する．具体的には，D_{max} 中のノード集合の部分集合のうち，導出関係に関する minimal covers を求めることにより行われる．ここで言う minimal cover とは，導出関係を利用することでそのノード集合のみで D_{max} のノードをすべて導出できるようなノード集合である．minimal covers は一般に複数存在する (図 8 の $D_{mc\_1}, \ldots, D_{mc\_k}$).

フェイズ 2 の操作を行うアルゴリズムを図 9 に示す．このアルゴリズムは，意味写像の値域を表す ER 図 erd と，導出関係の集合 d を入力とし，erd 中のノード集合 $erd.nodes()$ の d に関する

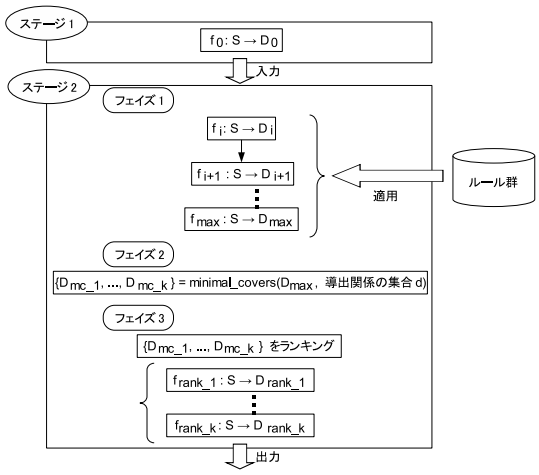


図 8 ステージ 2 の概要
Fig. 8 Overview of Stage 2

minimal covers を出力する．アルゴリズムの詳細は以下の通りである．

図 9 (a) はアルゴリズムの本体であり，以下の手順で出力となる minimal covers を求める．(1): 引数として ER 図 erd と導出関係の集合 d をとり，erd に存在するノード集合 ($erd.nodes()$ メソッドにより計算される) から d に存在するノード集合 ($d.nodes()$ メソッドにより計算される) との差集合をとることで，erd を構成するノード集合のうち，他のノード集合からでは決して導出できないノード集合 $default$ を求める (図 9 (a)3 行目)．定義により，$default$ はすべての minimal covers に含まれるはずである．(2): $d.nodes()$ のある部分ノード集合 n が，導出関係 d を利用することで $d.nodes()$ を導出可能かを $covers$ 関数により判定する (図 9 (a)7 行目)．(3): (2) が真のとき，$candidate$(n と $defalut$ の和集合) 中のノードのみから構成される ER 図が，ER 図として不適切でないかを判定する．その結果が真ならば，$candidate$ を minimal cover とする (図 9 (a)8 行目)．(4): $d.nodes()$ の各部分ノード集合 n 対して (2)，(3) を繰り返し，複数の minimal covers を得る (図 9 (a)4〜11 行目)．

図 9 (b) における $covers$ 関数は，d を利用することで，引数としてとる n から all が導出可能かを以下の手順で判定する．(1): 導出関係の集合 d のある導出関係 d_i に対して，n が d_i の左辺のノードをすべて含む集合であるとき，d_i の右辺のノードを n に追加する (図 9 (b)4〜6 行目)．(2): 導出関係の集合 d の各導出関係 d_i に対して (1) を繰り返し，n がこれ以上増えないという状態にする (図 9 (b)2〜8 行目)．(3): n が all と等しいノード集合かを判定する (図 9 (b)9 行目)．

(フェイズ 3) 前述したように，フェイズ 2 で得られる minimal covers は一般に複数である．フェイズ 3 では，複数の minimal covers $D_{mc\_1}, \ldots, D_{mc\_k}$ を，概念モデルとして相応しいと考えられる順 $D_{rank\_1}, \ldots, D_{rank\_k}$ にランキングし，それらを値域に持つ意味写像 $f_{rank\_1}, \ldots, f_{rank\_k}$ を出力する．ランキングを行うために，図 10 に示す表を用いて，各 minimal cover にスコア付けを行う．スコア付けはこの表に従い，例えば，クラスが一つ存在するごとに 3 点減点するというような減点法で行う．ランキングでは，スコアが高い方がランクが高いとみなす．例えば，図 6 の derivation relationships 節で表される導出関係の場合，minimal covers として $c1a2$，$c1c2a1$ という 2 つのノード集合が得られるが，この表を用いてスコア付けを行うと，$c1a2$ は-5 点，$c1c2a1$ は-8 点となり $c1a2$ の方がランクが高くなる．

3. 3 適用例

図 1 (a) の XML スキーマを入力としてステージ 1 で出力される単純な意味写像 $f_0 : S \rightarrow D_0$ の値域 D_0 を表す概念モデル図 5 に

```
1  Set[Set[Node]] phase2(ERDiagram erd, Set[D-Relationship] d) {
2     Set[Set[Node]] ans = φ;
3     Set[Node] default = erd.nodes() − d.nodes();
4     for each Set[Node] n ∈ P(d.nodes()) {
5        Set[Set[Node]] candidate = n ∪ default;
6        if(!∃ m ∈ ans (m ⊂ candidate)) {
7           if(covers(d, n, d.nodes()) {
8              if(isValid(candidate, erd)) ans = ans ∪ {candidate};
9           }
10       }
11    }
12    return ans;
13 }
```

(a)

```
1  boolean covers(Set[D-Relationship] d, Set[Node] n, Set[Node] all) {
2     while(true) {
3        Set[Node] n′ = n;
4        for each d_i ∈ d {
5           if(d_i.left() ⊆ n)) n = n ∪ d_i.right();
6        }
7        if(n == n′) break;
8     }
9     return (n == all);
10 }
```

(b)

図 9 フェイズ 2 の操作を行うアルゴリズム
Fig. 9 Algorithm for Phase 2

減点対象	減点数
クラス	3
サブクラス	0
属性	2
キー属性	0
関連	1

図 10 スコア表
Fig. 10 Score Table

対し，本提案手法を適用すると次のようになる．

フェイズ 1: フェイズ 1 では，D_0 に新しい概念モデルのノード集合が追加され，出力として図 11 を値域とする $f_{max} : S \rightarrow D_{max}$ が得られる．また，ノード間の導出関係も得られる．

フェイズ 2: フェイズ 2 では，フェイズ 1 で得られた導出関係を利用して，図 9 で示すアルゴリズムにより minimal covers を求めると，$c1c4c5a1(r12 \mid a2)(a8 \mid c6a3)(c11a15a16 \mid a11a12 \mid c7a4a9 \mid c7c9a4a6)(c12a17a18 \mid a13a14 \mid c8a5a10 \mid c8c10a5a7)(r10 \mid c2c3)$ のようなノード集合の組合せが得られる．ここで $(x \mid y)$ という記述は，x か y のどちらかであることを示す．したがって本例では，$2^3 \times 4^2 = 128$ 通りの minimal covers が得られる．

フェイズ 3: フェイズ 3 では，フェイズ 2 で出力された minimal covers を図 10 に示す表を用いてスコア付けし，ランキングする．本例において，スコアが最高となるのは $c1c4c5c11c12r10r12a1a8a15a16a17a18$ であり，これらのノードから構成される概念モデルは図 1 (b) となる (ただし，関連 $r2$ などノードの名前を除く)．したがって，ステージ 2 ではこの概念モデルを値域に持つ意味写像 $f_{rank\_1} : S \rightarrow D_{rank\_1}$ が出力される．

4. 関連研究

XML から UML のクラス図や ER 図などの概念モデルを抽出する研究はいくつか行われている [4] [5] が，これらは，(1) 変換規則が固定である，(2) インスタンスの変換を考慮していない，という点で本手法と異なる．特に (1) の理由により，これらの手法の設計時に想定されていないようなスキーマを対象とした場合に対応することは，簡単ではない．一方，本手法ではルールの追加は比較的容易にできる．

データベースを対象としたリバースエンジニアリングの支援が可能な CASE ツールも存在する [3] が，これは特に自動的なリバースエンジニアリングを想定したものではない．

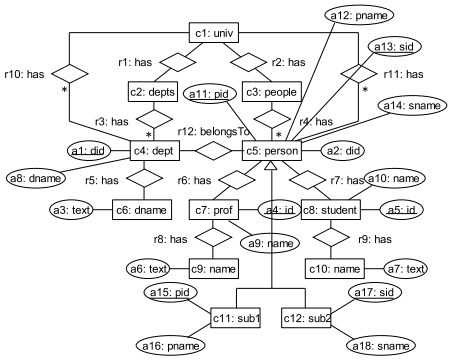


図 11 フェイズ 1 が出力する D_{max}
Fig. 11 Output D_{max} of Phase 1

5. おわりに

本稿では，データ変換プログラムの構築支援を目的とした，XML データからの概念モデル抽出の一手法を提案した．本手法の特徴は，概念モデル抽出の際に利用されるルールを，ユーザなどが自由に追加できることである．今後の課題としては，共通に利用可能なルールを集めたルールリポジトリの構築や，本手法の有効性を示すためのベンチマークの開発などが挙げられる．

[謝辞]

ゼミなどでご議論いただきました筑波大学図書館情報メディア研究科田畑孝一教授，阪口哲男助教授，永森光晴講師に感謝いたします．本研究の一部は日本学術振興会科学研究費補助金若手研究 (B)(課題番号 15700108) による．

[文献]

[1] 古川夏子，上村匡稔，大河原俊明，森嶋厚行，杉本重雄．XML データからの意味情報抽出支援プロトタイプシステムの実装．日本データベース学会 Letters，Vol.3，No.1，pp.129-132，2004 年 6 月．

[2] 森嶋厚行．データ変換プログラムのモデル化と構築手法の提案．日本データベース学会 Letters，Vol.3，No.1，pp.53-56，2004 年 6 月．

[3] Jean-Luc Hainaut, Jean Henrard, Didier Roland, Vincent Englebert, Jean-Marc Hick: Structure Elicitation in Database Reverse Engineering. WCRE 1996: 131-140.

[4] Mikael R. Jensen, Thomas H. Moller, Torben Bach Pedersen: Converting XML DTDs to UML diagrams for conceptual data integration. Data Knowl. Eng. 44(3): 323-346 (2003).

[5] Murali Mani, Dongwon Lee, Richard R. Muntz: Semantic Data Modeling Using XML Schemas. ER 2001: 149-163.

[6] L. Popa, Y. Velegrakis, R. J. Miller, M. A. Hernandez, R. Fagin: Translating Web Data. VLDB 2002: 598-609.

大河原 俊明 Toshiaki OHKAWARA
筑波大学大学院 図書館情報メディア研究科博士前期課程在学中．日本データベース学会学生会員．

森嶋 厚行 Atsuyuki MORISHIMA
筑波大学 図書館情報メディア研究科助教授．1998 年 筑波大学大学院 工学研究科修了．博士 (工学)．ACM，IEEE-CS，情報処理学会，電子情報通信学会，日本データベース学会各正会員．

杉本 重雄 Shigeo SUGIMOTO
筑波大学 図書館情報メディア研究科教授．京都大学大学院 工学研究科電子工学専攻博士後期課程修了．工学博士．ACM，IEEE-CS，情報処理学会他会員．