データベースシステム 129－15
（２００３．１．２４）

XML データのスキーマ統合のための枠組の提案

石元 豪[1]　石原 靖哲[2]　藤原 融[2]

あらまし

XML ではスキーマ設計における自由度が高いため, 同じような意味を持つデータでも, それらのスキーマが異なる場合がある. そこで, 異なるスキーマをもつデータを統合することが望まれる. XML データのスキーマ統合法に関する研究はいくつか存在するが, どのようなスキーマで統合すべきかについて形式的に議論したものは筆者らの知る限り存在しない. 本稿では, "元のスキーマに対する任意の問合せと同じ意味の問合せが, 統合スキーマに対して, 元と同じ表現能力の問合せ言語で記述可能である" ことがよい統合スキーマの条件と考え, それにもとづき統合の枠組を提案する. そして, よいスキーマで統合可能となるための十分条件を示す.

A Framework for Integrating Schemas of XML Documents

Takeshi Ishimoto[1]　Yasunori Ishihara[2]　Toru Fujiwara[2]

Abstract

Since XML allows users to define schemas at will, documents conveying similar information may have different schemas. In that case, those different schemas should be integrated for the convenience of users. There are some researches on the method of integrating schemas of XML documents. However, as far as we know, no research discusses which schema is appropriate as the integrated schema. In this paper, we say that an integrated schema is appropriate if, for an arbitrary query of a given class of queries for original schemas, a query for the integrated schema with the same meaning exists in the same class. Based on this definition, we propose a framework for integrating schemas. Then we show a sufficient condition for schemas to be integrated into an appropriate schema.

1　まえがき

近年, XML は Web アプリケーション等のデータフォーマットとして広く使用され始めている. XML ではデータをテキスト形式で記述するため, OS やアプリケーションに依らずに処理することができ, データ交換が簡単で semistructured data [1] の処理にも適している. また, XML のデータの構造 (スキーマ) をユーザが自由に定義できる. しかし, スキーマ設計時の自由度が高いための欠点もある. 例えば, 同じような意味のデータでもスキーマが異なる場合がある. 問合せ式の構造はスキーマに依存するため, 同じような意味のデータを検索する場合でも, スキーマが異なれば別々に処理する必要が生じる. これらのデータを統一的に扱うには, すべてのスキーマを統合しなければならない. 統合の方法として, 元の複数のスキーマで表現できるすべての情報を統合スキーマで表現できる (情報が欠落しない) ような統合と, 多少の情報の欠落を許す統合とが考えられる. 本稿では, 前者の統合に注目する.

スキーマ統合に関する研究は古くから行われており [3]，近年は異種データベースの統合法 [5, 10] や, 統合スキーマに対する問合せの処理についての研究 [6, 9] が盛んである．XML データに特化したスキーマ統合法に関する研究もいくつか行われている [4, 7, 8]. 統合スキーマとしてなるべく単純なスキーマを採用するのが望ましいが，どのような統合スキーマが単純なのか，形式的な議論を行っている研究は，筆者らの知る限り存在しない．そこで本稿では，統合後のスキーマに対する問合せ式が統合前のスキーマに対する同じ意味の問合せ式と比べて複雑にならないことを "よい統合スキーマ" の条件と考え，それを形式的に定義する．そして, 統合スキーマが "よい統合スキーマ" であるための十分条件の提案をおこなう.

[1]大阪大学大学院基礎工学研究科情報数理系専攻
Department of Informatics and Mathematical Science, Graduate School of Engineering Science, Osaka University
[2]大阪大学大学院情報科学研究科マルチメディア工学専攻
Department of Multimedia Engineering, Graduate School of Information Science and Technology, Osaka University

```
<社員データ>
    <氏名>
        <姓> 大阪 </姓>
        <名> 太郎 </名>
    </氏名>
      <役職> 課長 </役職>
      <所属> 営業課 </所属>
<\社員データ>
```

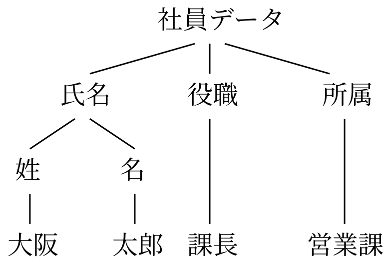

図 1: XML データベースとその木表現

なお，本稿における “統合” の定義は，[3] で紹介されている “behavioral equivalence” と同じ考え方に基づいている．

2 節では, XML 文書のデータモデルとスキーマ, 問合せ, データの変換器について説明する. 3 節では, 統合スキーマとよい統合スキーマを形式的に定義する. 4 節では, あるスキーマと, そのスキーマをもつ木集合をあるデータの変換器で変換した後の木集合がもつスキーマとの関係について述べる. 5 節で統合スキーマが “よい統合スキーマ” になるための十分条件を示し, 6 節でまとめと今後の課題を述べる.

2 諸定義

2.1 XML データ

XML データはラベルつき順序木で自然にモデル化される [2]. すなわち, 各頂点は対となるタグで囲まれた要素を表しており, タグ名もしくは値に対応するラベルがつけられている. 図 1 は, XML データとそれに対応する木を表している. また, この木を “社員データ (氏名 (姓, 名), 役職, 所属)” のような項で表現する.

本稿では, 参照や属性値を含まない XML データについて論じる. ある値をそのタグの属性値にするか, 要素にするかは任意であり, すべての属性値を要素により表現することも可能である. したがって属性値を含まないという仮定は一般性を失わない. また, 参照を含む XML データへの適用については今後の課題とする.

2.2 木

この節では木に関する定義をする. まず, 記号集合 Σ に対し, Σ 上のすべての有限記号列の集合を Σ^* で表す. また, Σ 上の木 (各頂点が Σ 中の記号でラベルづけされた木) の集合を $\mathcal{T}_\Sigma$ と書く. そして, 記号集合 Σ, Δ に対し, Δ 中の記号が葉のラベルとしてのみ現れるような, $\Sigma \cup \Delta$ 上の木の集合を $\mathcal{T}_\Sigma(\Delta)$ と表す.

次に頂点について定義する. 本稿では, 頂点を自然数の系列で表す. 空系列 ε で木の根頂点を表し, ui で頂点 u の i 番目の子頂点を表す. 形式的には, $t \in \mathcal{T}_\Sigma$ の頂点の集合 $Occ(t)$ を以下のように定義する. ここで, 自然数 k に対し, $[k]$ は $\{1, \ldots, k\}$ を表すとする.

- $t \in \Sigma$ のとき, $Occ(t) = \{\varepsilon\}$.
- $t \in \sigma(t_1 \cdots t_n)$ $(t_1, \ldots, t_n \in \mathcal{T}_\Sigma)$ のとき, $Occ(t) = \{\varepsilon\} \cup \bigcup_{i \in [n]} \{iu \mid u \in Occ(t_i)\}$.

また, 任意の木 $t \in \mathcal{T}_\Sigma$ と t 中の任意の頂点 u について, t 中の u でのラベルを $t[u]$ と表し, u の子頂点の数を $rank(t, u)$ と表す. そして, t 中の u 以下の部分木を t/u, t 中の u 以下を木の系列 $w \in \mathcal{T}_\Sigma^*$ で置き換えた木を $t[u \leftarrow w]$ で表す.

2.3 スキーマ

この節では, 正規木文法を用いて, XML のスキーマ (DTD) をモデル化する [2]. ここで, 記号集合 Δ 上のすべての正規言語の族を $Reg(\Delta)$ と表すものとする.

定義 1: スキーマとは正規木文法 $\mathcal{S} = (N, \Sigma, S, \sigma_0, P)$ である. ここで, N, Σ はそれぞれ, 非終端記号の有限集合, 終端記号 (ラベル) の有限集合である. S は N の要素で, 開始記号と呼ばれる. σ_0 は Σ の要素で, $\mathcal{S}$ をもつ木の根頂点のラベルと呼ばれる. $A \in N, t \in \mathcal{T}_\Sigma^*(Reg(N))$ とすると, P は $A \to t$ の形の式の集合である. ただし, $A = S$ ならば, t は木でかつ, $t[\varepsilon] = \sigma_0$ とする. $A \neq S$ ならば, t 中の頂点のラベルとして, σ_0 は現れないとする.

$\mathcal{S} = (N, \Sigma, S, \sigma_0, P)$ とし, $\xi_1, \xi_2 \in \mathcal{T}_\Sigma(Reg(N))$ とする. もし, ξ_1 中に正規言語 K がラベルづけされた頂点 u と P 中に式 $X_1 \to t_1$, ..., $X_n \to t_n$(ただし, $X_1 \cdots X_n \in K$) が存在して, u を $t_1, \ldots, t_n$ で置き換えることで ξ_2 が得られるならば, $\xi_1 \Rightarrow_\mathcal{S} \xi_2$ と書く. また, $\Rightarrow_\mathcal{S}$ の反射推移閉包を $\Rightarrow_\mathcal{S}^*$ と書く. そして, $\mathcal{S}$ により生成される木言語 $T(\mathcal{S})$ を $\{t \in \mathcal{T}_\Sigma \mid S \Rightarrow_\mathcal{S}^* t\}$ と定義する.

2.4 問合せ

この節では, [2] をもとに, 木への問合せを定義する. まずはじめに, 木から頂点を選び出す unary pattern と, 木から 2 頂点の組を選び出す binary pattern を定義する. Σ 上の unary pattern p とは $\mathcal{T}_\Sigma \times \mathcal{N}^*$ の部分集合で, Σ 上の binary pattern p' とは $\mathcal{T}_\Sigma \times \mathcal{N}^* \times \mathcal{N}^*$ の部分集合である ($\mathcal{N}$ は自然数の集合). $s \in \mathcal{T}_\Sigma$ とし, $u, v \in Occ(s)$ を s 中の頂点とする. $(s, u) \in p$ なら

ば, u は p にマッチするといい, $(s,u,v) \in p'$ ならば, (u,v) は p' にマッチするという.

p を unary pattern とする. 本稿では, 木 t を引数とし, t において指定した条件を満たすすべての頂点を求める問合せ $p(t)$ について考える. ここで, 問合せ $p(t)$ の実行結果 $E(p(t))$ を $\{u \mid (t,u) \in p\}$ と定義する. また, binary pattern については次節の木変換器の中で用いる.

木から頂点 (または, 2 頂点の組) を選び出す具体的な手法として, "木の頂点のたどり方を指定する" という方法がある. つまり, unary pattern, binary pattern として, 木の頂点のたどり方を指定する式を用いる. 本稿では, 木の頂点のたどり方を指定する式として正規表現を考える. 具体的には, x, y を変数, re を正規表現としたとき, unary pattern として $re \cdot x$, binary pattern として $x \cdot re \cdot y$ の形を考える. これら問合せ式の形式的な意味定義は以下の通りである. ここで, 正規表現 re により定義される言語を $L(re)$ とする.

- $re \cdot x = \{(t,v) \mid u,v \in Occ(t) \wedge u$ は v の祖先 $\wedge$ "u から v までのラベルの系列" $\in L(re)\}$
- $x \cdot re \cdot y =$ $\{(t,u,v) \mid u,v \in Occ(t) \wedge u$ は v の祖先 $\wedge$"u から v までのラベルの系列" $\in L(re)\}$

ただし, "u から v までのラベルの系列" には, u と v 自身のラベルも含む.

したがって, 問合せ $(re \cdot x)(t)$ は, ある頂点から木を葉の方向にたどっていき, たどった頂点のラベルの系列が $L(re)$ に含まれるような頂点の集合である.

2.5 木変換器

この節では, [2] をもとに, 木変換器を定義する. 本稿では, この木変換器を, 統合前と統合後のデータの対応を表現するために用いる.

Q を状態の有限集合する. Q, Σ 上の構造関数 f を, 任意の $q, q' \in Q$ について, "$q \neq q'$ ならば, $f(q) \cap f(q') = \emptyset$" が成り立つような, Q から Σ 上の binary pattern への写像と定義する. そして, Q, Σ 上のすべての構造関数の集合を $CF(Q,\Sigma)$ で表す.

定義 2: 木変換器 P は組 $(\Sigma, \Delta, Q, q_0, R)$ で表す. ここで,

- Σ は入力記号の有限集合.
- Δ は出力記号の有限集合.
- Q は状態の有限集合.
- $q_0 \in Q$ は初期状態.
- R は遷移規則 (q,p,t) の有限集合. ただし, $q \in Q$, p は Σ 上の unary pattern, $t \in T_\Delta(CF(Q,\Sigma))$.

任意の $\xi, \xi' \in T_{\Delta \cup Q}(Occ(s))$ に対して, 以下を満たすような頂点 $u \in Occ(\xi)$ と遷移規則 (q,p,t) が R 中に存在すれば, $\xi \Rightarrow_{P,s} \xi'$ と表し, ξ から ξ' に遷移するという.

1. $\xi/u = q(v)$. ただし, $q \in Q, v \in Occ(s), (s,v) \in p$.
2. $\xi' = \xi[u \leftarrow \theta(t)]$. ただし, $\theta(t)$ は, 木 t 中の各 $f \in CF(Q,\Sigma)$ を, 以下を満たす木の系列 $q_1(v_1)\ldots q_m(v_m)$ で置き換えた木を表す.
    - $\{v_1,\ldots,v_m\} = \{v' \mid (s,v,v') \in \bigcup_{q \in Q} f(q)\}$
    - 任意の $i \in [m]$ に対して, $(s,v,v_i) \in f(q_i)$.
    - $v_1, v_2, \cdots, v_m$ は s における深さ優先探索の行きがけ順に並んでいる.

また, $\Rightarrow^*_{P,s}$ を $\Rightarrow_{P,s}$ の反射推移閉包とし, 任意の $s \in \mathcal{T}_\Sigma$ について, $P(s) = \{t \in \mathcal{T}_\Delta \mid q_0(\varepsilon) \Rightarrow^*_{P,s} t\}$ とする.

本稿では, 元のスキーマをもつ木 s を, 統合スキーマをもつ木 t に対応づけするために P を用いる. そしてこのとき, t は s に比べて, 情報が欠落していてはならないと考える. そこで以下では, 情報が欠落しないような P の性質を定義する. また, s 中の頂点と, t 中の頂点との写像についても定義する.

定義 3: top-down 木変換器
$P = (\Sigma, \Delta, Q, q_0, R)$ を木変換器とする. 以下の条件が成り立つとき, P を top-down 木変換器とよぶ.

- 任意の遷移規則 $(q,p,t) \in R$ について, ある $\sigma \in \Sigma$ が存在し, 任意の木 s と頂点 $v \in Occ(s)$ について, $(s,v) \in p$ ならば $s[v] = \sigma$.
- 任意の binary pattern p' と木 s, 頂点 $u,v \in Occ(s)$ について, $p' \in \{f(q) \mid$ "f は R 中の構造関数" $\wedge q \in Q\}$ かつ $(s,u,v) \in p'$ ならば, v は u の子頂点.

つまり, top-down 木変換器では変換過程において, 状態と頂点のラベルにより遷移規則が決まり, また, 必ず親頂点が出力された後に子頂点が出力される.

本稿では, これ以降, P を top-down 木変換器とする. 続いて, P の性質に関する定義をする.

定義 4: 線形性
$P = (\Sigma, \Delta, Q, q_0, R)$ において, 任意の $(q,p,t) \in R$ が, 次の条件のいずれかを満たすとき, P は線形であるという.

- 木 t 中に構造関数が 1 つしか存在しない.
- 木 t 中の異なる頂点にラベルづけされた任意の $f, g \in CF(Q,\Sigma)$ について, $(\bigcup_{q \in Q} f(q)) \cap (\bigcup_{q \in Q} g(q)) = \emptyset$ が成り立つ.

つまり, 線形性が成り立つ P では, 任意の入力木中のすべての頂点は, 変換過程において, 2 回以上出力されることはない.

定義 5: totally defined
$\mathcal{S}$ をスキーマとし, $P = (\Sigma, \Delta, Q, q_0, R)$ とする. 任意の $s \in T(\mathcal{S})$, $u \in Occ(s)$, $q \in Q$ について, $(s,u) \in p$ であるような $(q,p,t) \in R$ が存在するとき, P は $\mathcal{S}$ に対して totally defined であるという.

つまり, $\mathcal{S}$ に対して totally defined である P では, 任意の入力木 $s \in T(\mathcal{S})$ について, 少なくとも 1 つの出力木が定義されていることになる.

定義 6: non-deleting
$\mathcal{S}$ をスキーマとし, $P = (\Sigma, \Delta, Q, q_0, R)$ とする. 任意の $(q,p,t) \in R$ が, 次の条件を満たすとき, P は $\mathcal{S}$ に対して non-deleting であるという.

- 任意の $s \in T(\mathcal{S}), u \in Occ(s)$ について, $(s,u) \in p$ ならば, $\bigcup_{f \text{ appearing in } t, q \in Q} f(q) \supseteq \{(s,u,v) \mid v = ui(1 \leq i \leq rank(s,u))\}$
- $t[\varepsilon] \in \Delta$. つまり, $t[\varepsilon] \notin CF(Q,\Sigma)$.

つまり, $\mathcal{S}$ に対して non-deleting である P では, 任意の入力木 $s \in T(\mathcal{S})$ 中のすべての頂点は, 変換過程において, 少なくとも 1 回出力され, かつ, その頂点の親頂点 (ラベルは状態) とその頂点からなる部分木は, その後の変換過程において, あるラベル $\delta \in \Delta$ をもつ頂点と置き換えられる.

定義 7: 決定性
$\mathcal{S}$ をスキーマとし, $P = (\Sigma, \Delta, Q, q_0, R)$ とする. 任意の $s \in T(\mathcal{S})$, $u \in Occ(s)$, $q \in Q$ について, $(s,u) \in p$ かつ $(s,u) \in p'$ であるような異なる遷移規則 $(q,p,t), (q,p',t')$ が R 中に存在しないとき, P は $\mathcal{S}$ に対して決定性であるという.

つまり, $\mathcal{S}$ に対して決定性である P では, 任意の入力木 $s \in T(\mathcal{S})$ 中の任意の頂点について, 1 状態につき出力が多くとも 1 つしかない. P が決定性のとき, $P(s) = \{t\}$ を単に $P(s) = t$ と書く.

次に, 入力木の頂点と, 出力木の頂点との関係を表す頂点写像についての定義をする.

定義 8: 写像
$\mathcal{S}$ をスキーマとする. P は線形で, $\mathcal{S}$ に対して totally defined, non-deleting, 決定性であるとする. s を $T(\mathcal{S})$ 中の任意の木とし, $t = P(s)$ とおく. 任意の $v \in Occ(s)$ について以下を満たす遷移規則 $(q,p,t') \in R$, $\xi, \xi' \in \mathcal{T}_{\Delta \cup Q(Occ(s))}$ および $u \in Occ(\xi)$ が一意に定まる.

- $q_0(\varepsilon) \Rightarrow^*_{P,s} \xi \Rightarrow_{P,s} \xi' \Rightarrow^*_{P,s} t$.
- $\xi/u = q(v)$.
- $(s,v) \in p$.
- $\xi' = \xi[u \leftarrow \theta(t')]$.

このとき, u を $\lambda_{P,s}(v)$ と表す. 直観的に $\lambda_{P,s}(v)$ は, P によって対応づけられる $P(s)$ 中の頂点である. 頂点集合 $V \subseteq Occ(s)$ について, $\lambda_{P,s}(V) = \{\lambda_{P,s}(v) \mid v \in V\}$ とする.

また, 写像 $\mu_P(\sigma) : \Sigma \to 2^{\Delta}$ を $\mu_P(\sigma) = \{P(s)[\lambda_{P,s}(v)] \mid s \in T(\mathcal{S}) \wedge v \in Occ(s) \wedge s[v] = \sigma\}$ と定義する. 直観的に $\mu_P(\sigma)$ は, P によって σ と対応づけられる $P(s)$ 中の頂点のラベルの集合である.

定義 9: P が次の条件を満たすとき, P をラベル重複しない木変換器と呼ぶ.

- 任意の $\sigma, \sigma' \in \Sigma$ について, $\mu_P(\sigma) \cap \mu_P(\sigma') = \emptyset$ を満たす.
- $\Gamma = \bigcup_{\sigma \in \Sigma} \mu_P(\sigma)$, $\Omega = \Delta - \Gamma$ とすると, 任意の遷移規則 (q,p,t) 中の t は $t[u] \in \Omega \cup CF(Q,\Sigma)(u \in Occ(t), u \neq \varepsilon)$ を満たす.

以降, P が線形で, ラベル重複せず, かつ, スキーマ $\mathcal{S}$ に対して, 決定性, totally defined, non-deleting ならば, P を $\mathcal{S}$ に対する統合用木変換器と呼ぶ.

3 提案する枠組み

本稿では, 入力として, スキーマ $\mathcal{S}_1, \mathcal{S}_2$, 問合せ式のクラス L と以下で述べる集合 EQ が与えられたとき, 統合用木変換器 P_1, P_2 を求め, 木の集合 $T' = \{P_1(t) \mid t \in T(\mathcal{S}_1)\} \cup \{P_2(t) \mid t \in T(\mathcal{S}_2)\}$ を $T(\mathcal{S})$ が含むようなスキーマ $\mathcal{S}$ を統合スキーマとして出力することを考える. ここで, EQ は, $\mathcal{S}_1, \mathcal{S}_2$ 間で同じ意味の頂点集合を指定する問合せの組の集合であり, ある $L' \subseteq L$ に対して $EQ \subseteq L' \times L'$ である. この節では, まず "統合スキーマ" を定義し, そして, "統合スキーマ" の部分集合である "よい統合スキーマ" を定義する.

3.1 統合スキーマの定義

"統合スキーマ" を定義する前に, 2 つのスキーマ間における "包摂" 関係を定義する.

定義 10: $\mathcal{S}, \mathcal{S}_1$ をスキーマとする. また, L を unary pattern の部分クラスとする. 以下の条件が成り立つとき, $\mathcal{S}$ は $\mathcal{S}_1$ を L に関して "包摂" するという. ここで, P は統合用木変換器である.
$\forall q_1 \in L, \forall t \in T(\mathcal{S}_1), \exists P, \exists q \in L:$
$$[\lambda_{P,t}(E(q_1(t))) = E(q(P(t))) \wedge P(t) \in T(\mathcal{S})]$$
つまり, $\mathcal{S}_1$ に対する L 中のどの unary pattern についても, それと同じ意味の unary pattern が, 元と同じクラス (L) で, $\mathcal{S}$ に対して記述できる.

$\mathcal{S}$ が $\mathcal{S}_1$ と $\mathcal{S}_2$ を包摂しているとき, $\mathcal{S}$ を, $\mathcal{S}_1$ と $\mathcal{S}_2$ の "統合スキーマ" とよぶ.

3.2 よい統合スキーマの定義

一般に, 元の複数のスキーマには同じ意味の情報が存在する. 統合スキーマ中でこれらの情報が散在していると, この同じ意味の情報への問合せが簡潔にならず, 統合によるメリットが十分に得られない. 本稿では, 元の 2 つのスキーマにおける任意の同じ意味の問合せ式について, スキーマの統合後, それと同じ意味の問合せ式を元と同じクラスで記述できるような統合スキーマを "よい統合スキーマ" と呼ぶことにする.

定義 11: $\mathcal{S}_1, \mathcal{S}_2$ をスキーマとし, $\mathcal{S}$ を $\mathcal{S}_1, \mathcal{S}_2$ の L に関する統合スキーマとする. また, "統合スキーマ" の条件を満たすような P_1 の集合を Π_1, P_2 の集合を Π_2 とする. $\mathcal{S}$ が以下の条件を満たすとき, $\mathcal{S}$ は $\mathcal{S}_1$ と $\mathcal{S}_2$ の L, L' に関する "よい統合スキーマ" であるという.

$\forall(q_1,q_2)\in EQ, \forall t_1\in T(\mathcal{S}_1), \forall t_2\in T(\mathcal{S}_2),$
$\exists P_1\in\Pi_1, \exists P_2\in\Pi_2, \exists q\in L':$
$[\lambda_{P_1,t_1}(E(q_1(t_1))) = E(q(P_1(t_1)))$
$\wedge\lambda_{P_2,t_2}(E(q_2(t_2))) = E(q(P_2(t_2)))$
$\wedge P_1(t_1), P_2(t_2)\in T(\mathcal{S})]$

4　統合用木変換器の性質

問合せ式のクラス L を正規表現とする．この節ではまず, 統合用木変換器 P 上での, スキーマ $\mathcal{S}_1$ をもつ木 s に対する問合せ式から, $P(s)$ に対する問合せ式への変換を定義する．そして, その変換を用いて, $\{P(s)\mid s\in T(\mathcal{S}_1)\}\subseteq T(\mathcal{S})$ なる任意の $\mathcal{S}$ は, $\mathcal{S}_1$ を正規表現に関して包摂することを示す.

4.1　問合せ式の変換

定義 12: $\mathcal{S}$ をスキーマとし, P を $\mathcal{S}$ に対する統合用木変換器とする．X, X', X'', Y を入力記号 Σ 上の正規表現, $\sigma\in\Sigma$ とし, 問合せ式 $X\cdot x$ の P 上での変換 $\Psi_P(X,\varepsilon)\cdot x$ を次のように定義する．ここで, 木 t において, 根頂点から構造関数 f の親までをたどったときのラベルの連接を $path(t,f)$ と表す. P は線形なので, $path(t,f)$ は一意に定まる．また, $h(Y)=\{\sigma\mid\sigma\cdots\in L(Y)\}$ とする.

- $X=(X')^*$ の時.
  $\Psi_P(X,Y)=(\Psi_P^*(X',X')\cdot\Psi_P(X',Y))+\varepsilon.$
- $X=X'+X''$ の時.
  $\Psi_P(X,Y)=\Psi_P(X',Y)+\Psi_P(X'',Y).$
- $X=X'\cdot X''$ の時.
  $\Psi_P(X,Y)=\Psi_P(X,X'')\cdot\Psi_P(X'',Y).$
- $X=\sigma$ かつ $Y\neq\varepsilon$ の時.
  $\Psi_P(X,Y) = path(t_0,f_0)+path(t_0,f_1)+\cdots+path(t_1,f_2)+\cdots+path(t_m,f_n).$
  ただし, $path(t_i,f_j)$ は次を満たすものとする.
  - f_j は構造関数で, t_i 中に存在する.
  - $t_i[\varepsilon]\in\mu_P(\sigma).$
  - 任意の $s\in T(\mathcal{S})$, 任意の $v,u\in Occ(s)$ について, ある $q'\in Q$ が存在して, $(s,v,u)\in f_j(q')\Rightarrow s[u]\in h(Y).$
- $X=\sigma$ かつ $Y=\varepsilon$ の時.
  $\Psi_P(X,Y)=t_0[\varepsilon]+\cdots+t_m[\varepsilon].$
  ただし, $\{t_0[\varepsilon],\ldots,t_m[\varepsilon]\}=\mu_P(\sigma).$

4.2　正規表現に関する包摂

補題 1: 任意のスキーマ $\mathcal{S}$ と, $\mathcal{S}$ に対する統合用木変換器である任意の P について以下が成り立つ. 任意の $s\in T(\mathcal{S})$ と $v\in Occ(s)$, 入力記号 Σ 上の正規表現 X, Y について,

1. $v\in E((X\cdot x)(s))\Leftrightarrow$
   $\lambda_{P,s}(v)\in E(\Psi_P((X,\varepsilon)\cdot x)(P(s))).$
2. 任意の $y\in h(Y)$ について,
   $v\in E((X\cdot y\cdot x)(s))\Leftrightarrow$
   $\lambda_{P,s}(v)\in E((\Psi_P(X,Y)\cdot\Psi_P(y,\varepsilon)\cdot x)(P(s))).$

[証明]
$*, +, \cdot$ の個数が k 個であるような入力記号 Σ 上の問合せ式を X_k と表し, k に関する数学的帰納法によって証明する．また, 証明の中では, 問合せ式の引数 $(s), (P(s))$, 変数 x は省略する．つまり, $(X\cdot x)(s)$ を X と表す.
[基底段階]
$k=0$ の時, $X_0=\sigma$(ただし, $\sigma\in\Sigma$) である.

1. μ_P の定義より, $v\in E(\sigma)$ ならば, $\lambda_{P,s}(v)$ のラベルは $\mu(\sigma)$ に属する． よって, $\lambda_{P,s}(v)\in E(\Psi_P(X_0,\varepsilon))$ である． 逆に, $\lambda_{P,s}(v)\in E(\Psi_P(X_0,\varepsilon))$ ならば, ラベル重複なしの条件より, v のラベルは σ である． よって, $v\in E(X_0)$．以上より, $v\in E(X_0)\Leftrightarrow\lambda_{P,s}(v)\in E(\Psi_P(X_0,\varepsilon))$ が成り立つ.
2. $\Rightarrow$
   $v\in E(\sigma\cdot y)$ とし, v の親頂点を u とする. そうすると, $s[u]=\sigma, s[v]=y$ である. また, 次を満たすような遷移規則 $(q,p,t_1),(q',p',t_2)$ が一意に定まる (決定性より).
   - この頂点 u,v に対して, $(s,u)\in p, (s,v)\in p'$ (totally defined より).
   - t_1 において $(s,u,v)\in f(q')$ を満たす構造関数 f が存在してかつ一意に定まる (線形性, non-deleting より).

   よって, $P(s)$ 上の $\lambda_{P,s}(u)$ から $\lambda_{P,s}(v)$ へのラベルの系列は, $path(t_1,f)\cdot t_2[\varepsilon]$ である．また, Ψ_P の定義より, $path(t_1,f)\in E(\Psi_P(\sigma,Y))$, $t_2[\varepsilon]\in E(\Psi_P(Y,\varepsilon))$ であるから, $\lambda_{P,s}(v)\in E(\Psi_P(X_0,Y)\cdot\Psi_P(y,\varepsilon))$.
   $\Leftarrow$
   $\lambda_{P,s}(v)\in E(\Psi_P(\sigma,Y)\cdot\Psi_P(y,\varepsilon))$ とする. このとき, Ψ_P の定義より, 以下を満たす t_1, t_2, f が存在する.
   - $\lambda_{P,s}(v)\in E(path(t_1,f)\cdot t_2[\varepsilon]).$
   - ある $q\in Q$ と, “任意の $v\in Occ(s)$ について, $(s,v)\in p\Rightarrow s[v]=\sigma$” を満たす p が存在して, $(q,p,t_1)\in R.$
   - ある $q'\in Q$ と, “任意の $v\in Occ(s)$ について, $(s,v)\in p'\Rightarrow s[v]=y$” を満たす p' が存在して, $(q',p',t_2)\in R.$

   よって, $t_1[\varepsilon]\in\mu_P(\sigma), t_2[\varepsilon]\in\mu_P(y)$ である. このことより, $v\in E(\sigma\cdot y)$ が成り立つ.

[帰納段階]
$k\leq m$ の時, 補題が成り立つと仮定する.
$k=m+1$ の時.

- $X_{m+1}=X_i\cdot X_j$ の時 (ただし, $i,j\leq m, i+j=m$).

1. 仮定より, 任意の $y \in h(X_j)$ について, $v \in E(X_i \cdot y) \Leftrightarrow \lambda_{P,s}(v) \in E(\Psi_P(X_i, X_j) \cdot \Psi_P(y, \varepsilon))$. また, $v \in E(X_j) \Leftrightarrow \lambda_{P,s}(v) \in E(\Psi_P(X_j, \varepsilon))$ が成り立つ. よって, $v \in E(X_i \cdot X_j) \Leftrightarrow \lambda_{P,s}(v) \in E(\Psi_P(X_i \cdot X_j, \varepsilon))$, つまり, $v \in E(X_{m+1}) \Leftrightarrow \lambda_{P,s}(v) \in E(\Psi_P(X_{m+1}, \varepsilon))$ が成り立つ.
2. 仮定より, 任意の $y' \in h(X_j)$ について, $v \in E(X_i \cdot y') \Leftrightarrow \lambda_{P,s}(v) \in E(\Psi_P(X_i, X_j) \cdot \Psi_P(y', \varepsilon))$. また, 任意の $y \in h(Y)$ について, $v \in E(X_j \cdot y) \Leftrightarrow \lambda_{P,s}(v) \in E(\Psi_P(X_j, Y) \cdot \Psi_P(y, \varepsilon))$ が成り立つ. よって, 任意の $y \in h(Y)$ について, $v \in E(X_i \cdot X_j \cdot y) \Leftrightarrow \lambda_{P,s}(v) \in E(\Psi_P(X_i \cdot X_j, Y) \cdot \Psi_P(y, \varepsilon))$, つまり, $v \in E(X_{m+1} \cdot y) \Leftrightarrow \lambda_{P,s}(v) \in E(\Psi_P(X_{m+1}, Y) \cdot \Psi_P(y, \varepsilon))$ が成り立つ.

- $X_{m+1} = X_i + X_j$ の時 (ただし, $i, j \leq m, i+j = m$).
  上と同様に示せる.
- $X_{m+1} = X_m^*$ の時.
  X_m の繰り返し回数 i に関する数学的帰納法で示せる.

以上より, $k = m+1$ の時にも補題は成り立つので, 任意の $k \geq 0$ について, 補題は成り立つ.

定理 1: スキーマ $\mathcal{S}_1$ に対する統合用木変換器である任意の P について, $\{P(s) \mid s \in T(\mathcal{S}_1)\} \subseteq T(\mathcal{S})$ を満たす $\mathcal{S}$ は, $\mathcal{S}_1$ を正規表現に関して包摂する.

[証明]
補題 1 より, $\mathcal{S}_1$ に対する任意の正規表現の問合せと同じ意味の問合せが, $\mathcal{S}$ に対して正規表現で記述できる. よって, $\mathcal{S}$ は "包摂" の定義の中の条件を満たす.

5 よいスキーマで統合可能となるための十分条件

この節では, 問合せ式のクラス L を正規表現とし, EQ 中の問合せ式のクラス L' を連接のみからなる正規表現としたとき, よい統合スキーマに統合可能となるために EQ が満たすべき条件を定義する. そして, 入力として, その条件を満たす EQ と任意のスキーマ $\mathcal{S}_1, \mathcal{S}_2$ が与えられたとき, 統合用木変換器 P_1, P_2 を求めることができ, 木の集合 $T' = \{P_1(t) \mid t \in T(\mathcal{S}_1)\} \cup \{P_2(t) \mid t \in T(\mathcal{S}_2)\}$ を $T(\mathcal{S})$ が含むような "よい統合スキーマ"$\mathcal{S}$ が存在することを示す.

5.1 EQ の条件

定義 13: 2 つのスキーマ $\mathcal{S}_1, \mathcal{S}_2$ 間で同じ意味の頂点集合を指定する問合せの組の集合 EQ が以下の 4 条件を満たすとき, EQ は無矛盾であるという. ここで, スキーマの定義より, $\mathcal{S}_1$ をもつ木の根頂点のラベルはちょうど 1 つ存在するので, そのラベルを σ_1 とする. 同様に, $\mathcal{S}_2$ をもつ木の根頂点のラベルを σ_2 とする. また, $\mathcal{S}_1$ をもつ木の頂点のラベルの集合を Σ', $\mathcal{S}_2$ をもつ木の頂点のラベルの集合を Σ'' とする.

1. $(\sigma_1 \cdot x, \sigma_2 \cdot x) \in EQ$.
2. $(q_1, q_2) \in EQ$ ならば, どの q_1, q_2 も $q_1 = \sigma_1 \cdot \sigma'_1 \cdots \sigma'_m \cdot x, q_2 = \sigma_2 \cdot \sigma''_1 \cdots \sigma''_n \cdot x$ の形をしている. ただし, $\sigma'_1, \ldots, \sigma'_m \in \Sigma'$, $\sigma''_1, \ldots, \sigma''_n \in \Sigma''$, $m, n \geq 1$ とする.
3. 任意の $(p_1 \cdot x, p_2 \cdot x), (p'_1 \cdot x, p'_2 \cdot x) \in EQ$ について, 以下を満たすような p''_1, p''_2 が存在すれば $(p''_1 \cdot x, p''_2 \cdot x) \in EQ$. ただし, $\sigma'_1, , \sigma'_2 \in \Sigma'$, $\sigma''_1, \sigma''_2 \in \Sigma''$ である.
   - $p_1 = p''_1 \cdot \sigma'_1 \cdots, p_2 = p''_2 \cdot \sigma''_1 \cdots$.
   - $p'_1 = p''_1 \cdot \sigma'_2 \cdots, p'_2 = p''_2 \cdot \sigma''_2 \cdots$.
   - $\sigma'_1 \neq \sigma'_2, \sigma''_1 \neq \sigma''_2$.
4. 任意の $(p_1 \cdot x, p_2 \cdot x), (p'_1 \cdot x, p'_2 \cdot x) \in EQ$ について, p'_1 が p_1 の真の接頭語ならば, p'_2 は p_2 の真の接頭語. またその逆も成り立つ.

1 つ目の条件は, $\mathcal{S}_1, \mathcal{S}_2$ をそれぞれもつ 2 つの木は根頂点同士が同じ意味であると指定されていなければならないということを意味している.

2 つ目の条件は, 根頂点からあるラベルの系列をたどった (根頂点を先頭とするラベルの連接により指定される) 頂点集合同士しか, 同じ意味として指定できないということを意味している. 本稿で同じ意味とは, 頂点がもつ情報の意味と頂点の構造的意味 (木におけるその頂点の位置) をいう. この構造的意味を根頂点からのラベルのたどり方を指定することで表すことにする.

3 つ目の条件は, ラベルの連接が途中で枝分かれするような 2 つの問合せ式が EQ 中に存在すれば, その枝分かれまでのラベルの連接を表す問合せ式も EQ 中に存在することを保証するための条件である. これは次の条件の記述を簡単化するために導入している.

4 つ目の条件は, 構造的に矛盾なく集合 EQ が与えられている (上下関係が逆転した頂点同士が同じ意味と指定されてはいない) ことを保証するための条件である.

5.2 統合用木変換器を求めるアルゴリズム

この節では, 入力された任意のスキーマ $\mathcal{S}_1, \mathcal{S}_2$ と無矛盾な $EQ \subseteq L' \times L'$ について, $\mathcal{S}_1, \mathcal{S}_2$ をよい統合スキーマで統合するための統合用木変換器を求めるアルゴリズムを示す.

このアルゴリズムは前半部分 (アルゴリズム 1) と後半部分 (アルゴリズム 2) に分けられる. アルゴリズム 1 では, 元の 2 つのスキーマに対する同じ意味の 2 つの問合せ式を, 同じ 1 つの問合せ式に変換する. アルゴリズム 2 では, アルゴリズム 1 での変換後の問合せ式と, 元の 2 つの問合せ式との意味がそれぞれ同じになるような木変換器を求める. そして, EQ で指定

された頂点集合への統合スキーマに対する問合せが L' に属することを示すことで, この統合スキーマは "よい統合スキーマ" であることを示す.

アルゴリズム 1

1. 入力を, スキーマ $\mathcal{S}_1, \mathcal{S}_2$ と, 無矛盾な集合 $EQ = \{(q_{10}, q_{20}), (q_{11}, q_{21}), \ldots, (q_{1r}, q_{2r})\}$ (ただし, $r \geq 0$) とする. ここで, $\mathcal{S}_1$ の終端記号の有限集合を $\Sigma, \mathcal{S}_2$ の終端記号の有限集合を Σ' とすると $\Sigma \cap \Sigma' = \emptyset$ が成り立つものとする. また, 任意の $i, j \geq 0$ について, $i \leq j$ ならば, $|q_{1i}| \leq |q_{1j}|$($|q|$ で, q の連接の長さを表す) とする. そして, Δ を $(\Sigma \cup \Sigma') \cap \Delta = \emptyset$ を満たす記号の集合とする. また, $\sigma_0, \sigma_1 \ldots \in \Sigma, \sigma'_0, \sigma'_1 \ldots \in \Sigma', \delta_0, \delta_1 \ldots \in \Delta$ とし, 任意の $i, j \geq 0$ について, $i \neq j$ ならば $\delta_i \neq \delta_j$ とする.
2. 集合 $\{(q'_{10}, q'_{20}), (q'_{11}, q'_{21}), \ldots, (q'_{1r}, q'_{2r})\}$ を EQ' とおく. ただし, 任意の $0 \leq i \leq r$ について, $q'_{1i} = q_{1i} \wedge q'_{2i} = q_{2i}$ とする.
3. 集合 EQ' 要素を前から順にみていく. 1 番目の要素 (q'_{10}, q'_{20}) は, それぞれの根頂点を返す問合せの組である. つまり, それぞれの根頂点のラベルを σ_0, σ'_0 とすると, $(\sigma_0 \cdot x, \sigma'_0 \cdot x)$ である. EQ' 中の要素のすべての問合せ式について, σ_0, σ'_0 の部分を δ_0 におきかえた集合を新たな EQ' とする.
4. $k = 1$ とする.
5. EQ' の $k+1$ 番目の要素 (q'_{1k}, q'_{2k}) は $(re \cdot \delta_l \cdot \sigma_1 \cdots \sigma_m \cdot x, re \cdot \delta_l \cdot \sigma'_1 \cdots \sigma'_n \cdot x)$ の形をしている. ただし, re は Σ, Σ', Δ 上の連接で, $l < k$ かつ $m, n \geq 1$ である. EQ' 中の要素のすべての問合せ式について, $re \cdot \delta_l \cdot \sigma_1 \cdots \sigma_m$ と $re \cdot \delta_l \cdot \sigma'_1 \cdots \sigma'_n$ の部分を $re \cdot \delta_l \cdot \sigma_1 \cdots \sigma_{m-1} \cdot \sigma'_1 \cdots \sigma'_{n-1} \cdot \delta_k$ におきかえた集合を新たな EQ' とする.
6. $k = k+1$ とし, $k \leq r$ ならば, 5 へ. そうでなければ終了.

補題 2: 任意の無矛盾な集合 EQ に対して, アルゴリズム 1 が適用でき, 任意の $0 \leq i, j \leq r$ について集合 EQ' は以下を満たす.

- $q'_{1i} = q'_{2i}$.
- $q'_{1i}(q'_{2i})$ から $\Sigma'(\Sigma)$ 中の記号を削除し, Δ 中の記号をアルゴリズムのステップ 3 や 5 において対応する $\Sigma(\Sigma')$ 中の記号に置き換えると, $q_{1i}(q_{2i})$ が得られる.
- $q_{1i} = p_{1i} \cdot x$, $q'_{1i} = p'_{1i} \cdot x$, $q_{1j} = p_{1j} \cdot x$, $q'_{1j} = p'_{1j} \cdot x$ とすると, p'_{1i} が p_{1j} の真の接頭語ならば, p'_{1i} は p'_{1j} の真の接頭語. q'_{2i}, q'_{2j} についても同様.

アルゴリズム 2

1. 以下を満たすような, $\mathcal{S}_1$ に対する統合用木変換器を $P_1 = (\Sigma, \Sigma \cup \Sigma' \cup \Delta, Q, q_0, R)$ とし, 以下と同様の条件を満たすような, $\mathcal{S}_2$ に対する統合用木変換器を $P_2 = (\Sigma', \Sigma \cup \Sigma' \cup \Delta, Q', q'_0, R')$ とする.
   - 任意の $s \in T(\mathcal{S}_1)$ について, $s = P_1(s)$.
   - $Q = \{q_0, q_1, \ldots\} \cup$ $\{re \mid re$ はΣ上の連接で $1 \leq |re| \leq max\{|p_1| \mid (p_1 \cdot x, p_2 \cdot x) \in EQ\}\}$.
   - $\mathcal{S}_1$ をもつ木の根頂点のラベルを σ とすると, $(q_0, \sigma \cdot x, t_0) \in R$ の t_0 中にある構造関数は状態 σ でのみ定義されている. また, 異なる遷移規則にこの構造関数は現れない.
   - $(q, \sigma \cdot x, t) \in R$ 中の q がΣ上の連接で $1 \leq |q| < max\{|p_1| \mid (p_1 \cdot x, p_2 \cdot x) \in EQ\}$ ならば, t 中にある構造関数は状態 $q \cdot \sigma$ でのみ定義されている. また, 異なる遷移規則にこの構造関数は現れない.
   - $(q, \sigma \cdot x, t) \in R$ が上のどの条件にも当てはまらないならば, t 中にある構造関数は $\{q_0, q_1, \ldots\}$ の要素の状態でのみ定義されている.
2. EQ' の要素 $(q'_{10}, q'_{20}), \ldots, (q'_{1r}, q'_{2r})$ すべてと, EQ の要素 $(q_{10}, q_{20}), \ldots, (q_{1r}, q_{2r})$ とをそれぞれ順に比較する. それに基づいて, P_1, P_2 を変更していく. 以下, P_1 の変更についてのみ述べる.
3. $k = 0$ とする.
4. $q_{1k} = \sigma_0 \cdot \cdots \cdot \sigma_m \cdot x$, $q'_{1k} = re \cdot x$ とすると, 連接 re 中の $\delta \in \Delta$ と $\sigma \in \Sigma$ の個数の和は $m+1$ である. ここで, 連接 q_{1k} 中の $\sigma_n (0 \leq n \leq m)$ を前から順に, 連接 re 中の δ または σ に対応づけをする. この対応づけを $\{(\sigma_0 : \gamma_0), \ldots, (\sigma_m : \gamma_m)\}$ と表すことにする. ただし, $\gamma_0, \ldots, \gamma_m \in \Delta \cup \Sigma$ とする. 補題 2 よりこの対応づけが可能である.
5. P_1 に入力 $s = \sigma_0(\sigma_1(\cdots(\sigma_m)\cdots))$ を与えたとき適用される遷移規則の系列を $(q_0, \sigma_0 \cdot x, t_0)$, $(\sigma_0, \sigma_1 \cdot x, t_1), \ldots, (\sigma_0 \cdots \sigma_{m-1}, \sigma_m \cdot x, t_m)$ とする. P_1 の性質より, この系列はちょうど 1 つ存在する.
6. 遷移規則 $(q_0, \sigma_0 \cdot x, t_0)$ の t_0 の根頂点のラベルを γ_0 に置き換え, 新たな R とする. $l = 0$ として, 8 へ.
7. 遷移規則 $(\sigma_0 \cdots \sigma_{l-1}, \sigma_l \cdot x, t_l)$ の t_l の根頂点のラベルを γ_l に置き換え, 新たな R とする.
8. $re = \cdots \gamma_l \cdot \sigma'_1 \cdots \sigma'_i \cdot \gamma_{l+1} \cdots$ ならば 9 へ. $re = \cdots \gamma_l \cdot \gamma_{l+1} \cdots$ ならば 10 へ.
9. 遷移規則 $(\sigma_0 \cdots \sigma_{l-1}, \sigma_l \cdot x, t_l)$ の t_l の構造がこのアルゴリズムで 1 度も変更されていなければ, t_l 中にある $f'(\sigma_0 \cdots \sigma_l) = x \cdot \sigma_l \cdot \sigma_{l+1} \cdot$

y と定義されている構造関数 f' の頂点を木 $\sigma'_1(\cdots(\sigma_i(f'))\cdots)$ で置き換えた木を新たな t_l とする.

10. $l = l + 1$ とし, $l \leq m - 1$ ならば, 7 へ.
11. $k = k + 1$ とし, $k \leq r$ ならば, 4 へ. そうでなければ終了.

補題 3: アルゴリズム 2 によって置き換えされた P_1, P_2 は統合用木変換器である.

[略証]

top-down 木変換器 アルゴリズム 2 中での木変換器の変更点は, top-down 木変換器の条件を満したままなので, P_1, P_2 は top-down 木変換器である.

線形性 アルゴリズム 2 中では, 任意の遷移規則 (q, p, t) の t 中の構造関数は変更しないので, P_1, P_2 は線形のままである.

totally defined アルゴリズム 2 中では, 任意の遷移規則 (q, p, t) の q, p については変更しないので, P_1, P_2 は totally defined のままである.

non-deleting アルゴリズム 2 中では, 任意の遷移規則 (q, p, t) の t 中の構造関数は変更しない. また, $t[\varepsilon] \in CF(Q, \Sigma)$ となるような t の変更もないので, P_1, P_2 は non-deleting のままである.

決定性 アルゴリズム 2 中では, 任意の遷移規則 (q, p, t) の q, p については変更しない. また, t 中の構造関数についても変更しないので, P_1, P_2 は決定性のままである.

ラベル重複 任意の $\sigma \in \Sigma$ について, $\mu_{P_1}(\sigma) = \{\sigma\} \cup \{\gamma \mid (\sigma : \gamma)\}$ である. ここで, $\gamma = \sigma$ もしくは, $\gamma = \delta_k (1 \leq k \leq r)$ である. アルゴリズム 1 より, 任意の $i, j \geq 0$ について, $i \neq j$ ならば $\delta_i \neq \delta_j$ であるので, 任意の $\sigma, \sigma' \in \Sigma$ について, $\sigma \neq \sigma'$ ならば $\mu_{P_1}(\sigma) \cap \mu_{P_1}(\sigma') = \emptyset$ は成り立つ. また, 9 で新たに付け加えた任意の頂点のラベルを σ' とすると, $\sigma' \in \Sigma'$ である. $\Sigma' \cap \Sigma = \emptyset$ かつ, $\Sigma' \cap \Delta = \emptyset$ より P_1 はラベル重複しない. 同様の理由で P_2 もラベル重複しない.

定理 2: 任意のスキーマ $\mathcal{S}_1, \mathcal{S}_2$ と無矛盾な集合 EQ に対して, $\mathcal{S}_1$ と $\mathcal{S}_2$ のよい統合スキーマ $\mathcal{S}$ が存在する.

[証明]

補題 2, 3 より, 任意のスキーマ $\mathcal{S}_1, \mathcal{S}_2$ と無矛盾な集合 EQ に対して, 統合用木変換器 P_1, P_2 を求めることができる. よって, $\mathcal{S}', \mathcal{S}''$ を $\{P_1(s) \mid s \in T(\mathcal{S}_1)\} \subseteq T(\mathcal{S}')$, $\{P_2(s) \mid s \in T(\mathcal{S}_2)\} \subseteq T(\mathcal{S}'')$ を満たすスキーマとすると, 定理 1 より, $\mathcal{S}'$ は $\mathcal{S}_1$ を包摂し, $\mathcal{S}''$ は $\mathcal{S}_2$ を包摂する. また, $T(\mathcal{S}') \cup T(\mathcal{S}'') \subseteq T(\mathcal{S})$ であるようなスキーマを $\mathcal{S}$ とすると, $\mathcal{S}$ は $\mathcal{S}_1$ と $\mathcal{S}_2$ を包摂するので, $\mathcal{S}$ は $\mathcal{S}_1$ と $\mathcal{S}_2$ の統合スキーマである.

また, $\mathcal{S}_1, \mathcal{S}_2$ 間で同じ意味の頂点を指定する問合せの組の集合 EQ の要素である問合せ q_{1k}, q_{2k} は, それに対応する EQ' の要素である問合せ $q'_{1k} = q'_{2k}$ が $\mathcal{S}$ に対する同じ意味の問合せになっている. $q'_{1k} = q'_{2k}$ は根頂点から始まる連接である. したがって, この統合スキーマ $\mathcal{S}$ はよい統合スキーマである.

6 あとがき

本稿では, 複数のスキーマを意味的に含む統合スキーマを定義し, さらに, 統合後のスキーマに対する問合せ式が統合前のスキーマに対する同じ意味の問合せ式と比べて複雑にならないことを “よい統合スキーマ” の条件として定義した. そして, 問合せ式のクラスを正規表現とその部分集合とした上で, その統合スキーマが “よい統合スキーマ” となるための十分条件を提案した.

今後の課題としては, その “よい統合スキーマ” のうち, そのスキーマをもつ木の集合が極小となるスキーマをみつけるアルゴリズムの検討が挙げられる. また, 問合せ式のクラスを正規表現以外のクラスに設定した場合の, “よい統合スキーマ” が存在するための条件の考察などが挙げられる.

参考文献

[1] S. Abiteboul, “Querying semi-structured data,” Proc. 6th ICDT, LNCS 1186, pp.1–18, 1997.

[2] S. Maneth, F. Neven, “Structured document transformations based on XML,” Proc. 7th DBPL, LNCS 1949, pp.80–98, 2000.

[3] C. Batini, M. Lenzerini, S.B. Navathe, “A comparative analysis of methodologies for database schema integration,” ACM Computing Surveys, 10, 4, pp.323–364, 1986.

[4] R. Behrens, “A grammar based model for XML schema integration,” Proc. 17th BNCOD, pp.172–190, 2000.

[5] S. Cluet, C. Delobel, J. Simèon, K. Smaga, “Your mediators need data conversion,” Proc. SIGMOD Conf., pp.177–188, 1998.

[6] A.Y. Halevy, “Answering queries using views: A survey,” VLDB Journal, 10, pp.270–294, 2001.

[7] E. Jeong, C.N. Hsu, “Induction of integrated view for XML data with heterogeneous DTDs,” Proc. 2001 ACM CIKM, pp.151–158, 2001.

[8] G.M. Kuper, J. Simèon, “Subsumption for XML types,” Proc. 8th ICDT, LNCS 1973, pp.331–345, 2001.

[9] M. Lenzerini, “Data integration: A theoretical perspective,” Proc. 21st PODS, pp.233–246, 2002.

[10] P. McBrien, A. Poulovassilis, “A semantic approach to integrating XML and structured data sources,” Proc. 13th CAiSE, LNCS 2068, pp.330–345, 2001.