Vol.2009-MPS-75 No.20
2009/9/10

XMLスキーマで定義された型とXPath式との対応の解析手法

大野敦司[†1]　石原靖哲[†1]　藤原融[†1]

本稿では，XML スキーマで定義された型と XPath 部分式との対応を指定した場合に，指定した対応を満たしかつ，XPath 式を充足するような XML 文書が存在するかという問題を判定する手法を提案する．まず，XPath 式を有限木オートマトン変換する手法を与える．次に，得られた XPath 式の有限木オートマトンと XML スキーマを表す有限木オートマトンの積オートマトンを構成，解析することで，問題の判定を行う手法を与える．

Method of Analyzing the Correspondence Between Types Defined by an XML Schema and XPath Subexpressions

Atsushi Ohno,[†1] Yasunori Ishihara[†1]
and Toru Fujiwara[†1]

This paper proposes a method of deciding, given the correspondence between types defined by an XML schema and an XPath expression, whether or not there exists an XML document such that the document satisfies the XPath expression and the correspondence holds. In the proposed method, the given XPath expression is translated into a finite tree automaton. Then, the problem is decided by constructing and analyzing the intersection of tree automata of the XPath expression and a given XML schema.

†1 大阪大学大学院情報科学研究科
Graduate School of Information Science and Technology, Osaka University

1. まえがき

XML は構造化データを記述できるマークアップ言語である一方で，マークアップ言語を定義できるメタ言語でもあり，多くの環境で利用できることから盛んに用いられている．そのため，交換される XML 文書の構造はあらかじめ統一しておくと便利であることが多い．そこで，スキーマを用いて，XML 文書の各要素の型，すなわち，下位要素の出現順や出現回数を指定するのが一般的である．スキーマはしばしば有限木オートマトンで与えられる．一方，XML 文書の特定の要素を指定する問合せ言語として XPath[1] が広く使われている．XPath は，XML 文書を木構造に見立て，その根頂点からの経路を記述することで，XML 文書の特定の要素を指定する．XPath 式は，

部分式 “/” 部分式 “/” · · ·

という形になっており，各部分式は，軸，ノードテスト，述語から構成される．軸は探索すべき方向を，ノードテストは要素名を，そして述語はノードテストに条件を付加する．XPath はそれ自体が問合せ言語として使われるだけでなく，XQuery や XSLT などの実用的な問合せ言語の一部として使用されている．以上のことから，XML スキーマや XPath は XML データベースにおける重要な研究課題となっている．

XPath 式の関係の解析に関する研究としては，与えられたスキーマのもとでの XPath 式の充足可能性問題[2] や包含問題[3] を扱う研究が既に存在する．前者は，XPath 問合せ式 p とスキーマ S が与えられた場合に，XML 文書 t が S に従いかつ，t における p の答えが空でないような t が存在するかどうかを決定する問題であり，後者は，XPath 問合せ式 p_1 と p_2 が与えられた場合に，任意の (文書) 木 t について，p_1 の答えが p_2 の答えを含むかどうかを決定する問題である．しかし，これらの研究では，各 XPath 部分式によって指定され得る XML 文書の要素が，スキーマで定義されたどの型をもつのかの解析については検討していない．XPath 部分式 p' によって指定され得る XML 文書 t の要素が，スキーマ S の型 s によって定義されているとき，p' と s が対応するという．著者が知る限りでは，XML スキーマで定義された型と XPath 式との対応の解析に関する研究は存在しない．

本稿では，XML スキーマで定義された型と XPath 式との対応の解析手法を提案する．まず，XPath 式を有限木オートマトンに変換する手法を与える．変換は，部分式に対する有限木オートマトンを求めることから始まる．各部分式に対する有限木オートマトンへの変換は，軸別に変換方法を与える．次に各部分式に対する有限木オートマトンどうしの積オートマトンを構成する．XPath の構文定義に従って積オートマトンを構成していくことで，最

終的に XPath 式全体を有限木オートマトンに変換する．そして，得られた XPath 式の有限木オートマトンと XML スキーマを表す有限木オートマトンの積オートマトンを構成，解析することで，XPath 式と XML スキーマの型との対応を調べる．

以下，2 節では本稿で用いる定義を与える．3 節では，XPath 式を有限オートマトンに変換する手法を与え，4 節では対応の解析手法について述べる．最後に，まとめと今後の課題を述べる．

2. 諸 定 義

2.1 XML 文書

ここでは，文献 4) を参考に，本稿で考える XML 文書を定義する．

定義 1 (XML 文書) XML 文書はラベル付き順序木で表される．ノード v のラベルを $\lambda(v)$ と書く．ラベルは要素名に相当する．また，λ をノードの系列上の関数に拡張する．つまり，ノードの系列 $v_1 \cdots v_n$ について，$\lambda(v_1 \cdots v_n) = \lambda(v_1) \cdots \lambda(v_n)$ である．

以下では, ラベル付き順序木を単に木と呼ぶ.

2.2 XPath

ここでは，文献 2)，4) を参考に，本稿で対象とする XPath の構文および意味論を与える．

定義 2 (XPath 構文) Σ をアルファベットとする．XPath 式 p を以下のように定義する:

$p ::= \chi :: l \mid p/p \mid p \cup p \mid p[q]$,

$\chi ::= \cdot \mid \downarrow \mid \uparrow \mid \downarrow^+ \mid \uparrow^+ \mid \downarrow^* \mid \uparrow^* \mid \rightarrow^+ \mid \leftarrow^+ \mid \searrow \mid \swarrow$,

$q ::= p \mid q \wedge q \mid q \vee q$

ただし，$l \in \Sigma$ である．各 $\chi \in \{\cdot, \downarrow, \uparrow, \downarrow^+, \uparrow^+, \downarrow^*, \uparrow^*, \rightarrow^+, \leftarrow^+, \searrow, \swarrow\}$ を軸，q を述語という．

次に，XPath 式の位置を定義する．

定義 3 (XPath 式の位置) XPath 式 p の位置 α とは正整数の有限系列であり，p の部分式 $p|_\alpha$ を指定する．$p|_\alpha$ は以下のように定義される:

- $p|_\epsilon = p$
- $p|_\alpha = p'$ とすると，
  - $p' = p'_1/p'_2$ のとき，$p|_{\alpha\cdot 1} = p'_1$, $p|_{\alpha\cdot 2} = p'_2$
  - $p' = p'_1 \cup p'_2$ のとき，$p|_{\alpha\cdot 1} = p'_1$, $p|_{\alpha\cdot 2} = p'_2$
  - $p' = p''[q]$ のとき，$p|_{\alpha\cdot 1} = p''$, $p|_{\alpha\cdot 2} = q$
  - $p' = q_1 \wedge q_2$ のとき，$p|_{\alpha\cdot 1} = q_1$, $p|_{\alpha\cdot 2} = q_2$
  - $p' = q_1 \vee q_2$ のとき，$p|_{\alpha\cdot 1} = q_1$, $p|_{\alpha\cdot 2} = q_2$

定義 4 (XPath 意味論) 木 t が XPath 式 p の部分式 $p|_\alpha$ を充足するとは，ある θ, ρ について $(t, \theta, \rho, \alpha) \models p$ が成立することであると定義する．ここで θ, ρ は XPath 式の位置から，木 t のノード集合への写像であり，関係 $\models$ は以下のように定義される．直観的には，部分式がノードを指定する基準点となるノードをコンテクストノードといい，θ は XPath 式の位置から部分式のコンテクストノードへの写像，ρ は XPath 式の位置から部分式が指定するノードへの写像である．

- $(t, \theta, \rho, \alpha) \models (\cdot :: l)$
  $\theta(\alpha) = \rho(\alpha)$ かつ $\lambda(\rho(\alpha)) = l$.
- $(t, \theta, \rho, \alpha) \models (\downarrow :: l)$
  $\rho(\alpha)$ は $\theta(\alpha)$ の子であり，かつ $\lambda(\rho(\alpha)) = l$.
- $(t, \theta, \rho, \alpha) \models (\uparrow :: l)$
  $\rho(\alpha)$ は $\theta(\alpha)$ の親であり，かつ $\lambda(\rho(\alpha)) = l$.
- $(t, \theta, \rho, \alpha) \models (\downarrow^+ :: l)$
  $\rho(\alpha)$ は $\theta(\alpha)$ の子孫であり，かつ $\lambda(\rho(\alpha)) = l$.
- $(t, \theta, \rho, \alpha) \models (\uparrow^+ :: l)$
  $\rho(\alpha)$ は $\theta(\alpha)$ の祖先であり，かつ $\lambda(\rho(\alpha)) = l$.
- $(t, \theta, \rho, \alpha) \models (\downarrow^* :: l)$
  $\rho(\alpha)$ は $\theta(\alpha)$ 自身またはその子孫であり，かつ $\lambda(\rho(\alpha)) = l$.
- $(t, \theta, \rho, \alpha) \models (\uparrow^* :: l)$
  $\rho(\alpha)$ は $\theta(\alpha)$ 自身またはその祖先であり，かつ $\lambda(\rho(\alpha)) = l$.
- $(t, \theta, \rho, \alpha) \models (\rightarrow^+ :: l)$
  $\rho(\alpha)$ は $\theta(\alpha)$ の弟であり，かつ $\lambda(\rho(\alpha)) = l$.
- $(t, \theta, \rho, \alpha) \models (\leftarrow^+ :: l)$
  $\rho(\alpha)$ は $\theta(\alpha)$ の兄であり，かつ $\lambda(\rho(\alpha)) = l$.
- $(t, \theta, \rho, \alpha) \models (\searrow :: l)$
  $\rho(\alpha)$ は $\theta(\alpha)$ のある祖先 n の弟 n' 自身またはその子孫であり，かつ $\lambda(\rho(\alpha)) = l$.
- $(t, \theta, \rho, \alpha) \models (\swarrow :: l)$
  $\rho(\alpha)$ は $\theta(\alpha)$ のある祖先 n の兄 n' 自身またはその子孫であり，かつ $\lambda(\rho(\alpha)) = l$.
- $(t, \theta, \rho, \alpha) \models p_1/p_2$
  $(t, \theta, \rho, \alpha \cdot 1) \models p_1$ かつ $(t, \theta, \rho, \alpha \cdot 2) \models p_2$ かつ $\theta(\alpha) = \theta(\alpha \cdot 1)$ かつ $\rho(\alpha) = \rho(\alpha \cdot 2)$

かつ $\rho(\alpha\cdot 1)=\theta(\alpha\cdot 2)$.

- $(t,\theta,\rho,\alpha)\models(p_1\cup p_2)$
  $(t,\theta,\rho,\alpha\cdot 1)\models p_1$ かつ $\theta(\alpha)=\theta(\alpha\cdot 1)$ かつ $\rho(\alpha)=\rho(\alpha\cdot 1)$，または $(t,\theta,\rho,\alpha\cdot 2)\models p_2$ かつ $\theta(\alpha)=\theta(\alpha\cdot 2)$ かつ $\rho(\alpha)=\rho(\alpha\cdot 2)$.
- $(t,\theta,\rho,\alpha)\models(p[q])$
  $(t,\theta,\rho,\alpha\cdot 1)\models p$ かつ $(t,\theta,\rho,\alpha\cdot 2)\models q$ かつ $\theta(\alpha)=\theta(\alpha\cdot 1)$ かつ $\rho(\alpha)=\rho(\alpha\cdot 1)=\theta(\alpha\cdot 2)$.
- $(t,\theta,\rho,\alpha)\models(q_1\wedge q_2)$
  $(t,\theta,\rho,\alpha\cdot 1)\models q_1$ かつ $(t,\theta,\rho,\alpha\cdot 2)\models q_2$ かつ $\theta(\alpha)=\theta(\alpha\cdot 1)=\theta(\alpha\cdot 2)$.
- $(t,\theta,\rho,\alpha)\models(q_1\vee q_2)$
  $(t,\theta,\rho,\alpha\cdot 1)\models q_1$ かつ $\theta(\alpha)=\theta(\alpha\cdot 1)$，または $(t,\theta,\rho,\alpha\cdot 2)\models q_2$ かつ $\theta(\alpha)=\theta(\alpha\cdot 2)$.

2.3 有限木オートマトン

文献 5) を参考に，本稿で考える有限木オートマトン（正規木文法），解釈，受理の各定義，および有限木オートマトンの積演算を与える．

定義 5 (有限木オートマトン) 有限木オートマトンは以下のような四つ組 $TA=(N,\Sigma,s,P)$ である．

- N は状態の有限集合,
- Σ は要素名の有限集合,
- $s\in N$ は初期状態,
- P は $X\to a(reg)$ または $X\to Y$ の形をした遷移規則の有限集合.

ただし，$X,Y\in N$，$a\in\Sigma$ であり，reg は N 上の正規表現である．X を規則の左辺，Y, $a(reg)$ を右辺といい，a，reg をそれぞれこの規則のラベル，内容モデルという．
本稿ではスキーマを有限木オートマトン TA_s で表現する．スキーマの型は有限木オートマトンの状態で表現される．

定義 6 (解釈) 有限木オートマトン TA に対する木 t の解釈 I は，t 中の各ノード v から状態 $I(v)$ への以下を満たす写像である:

- v が t の根である場合，$I(v)$ は初期状態である．
- 各ノード v とその子ノード $v_1,\ldots,v_n$ について，TA 中の遷移規則 $X\to a(reg)$ が存在し，以下を満たす:
  - $I(v)=X$,
  - $\lambda(v)=a$,
  - $I(v_1)\ldots I(v_n)$ が reg にマッチする．

有限木オートマトン TA によって受理される木の集合を $TL(TA)$ と表す．有限木オートマトン TA が木 t を受理するとは，TA に対する t の解釈が存在することである．

定義 7 (有限木オートマトンの積演算) 有限木オートマトンを $TA_1=(N_1,\Sigma_1,s_1,P_1)$, $TA_2=(N_2,\Sigma_2,s_2,P_2)$ とする．一般性を失うことなく，$\Sigma_1=\Sigma_2=\Sigma$ と仮定できる．$TL(TA_1)$ と $TL(TA_2)$ の積を受理する有限木オートマトンを構成する．
N_1 上の正規表現 reg_1 と N_2 上の正規表現 reg_2 が与えられ，それらの積 $reg_1\oplus reg_2$ を構成する． $reg_1\oplus reg_2$ は $N_1\times N_2$ 上の正規表現である．$n_1^1n_1^2...n_1^i$ が reg_1 と一致し，かつ $n_2^1n_2^2...n_2^i$ が reg_2 と一致する場合に，$N_1\times N_2$ 中の状態の系列 $[n_1^1,n_2^1]\ [n_1^2,n_2^2]\ ...\ [n_1^i,n_2^i]$ は $reg_1\oplus reg_2$ と正確に一致する．
$TL(TA_1)$ と $TL(TA_2)$ の積を受理する有限木オートマトン構成の 4 ステップは以下の通り:

(1) reg_1 から reg_1'（$N_1\times N_2$ 上の正規表現）をつくる．これは N_1 中の各 n を $[n_1,n_2^1]\mid[n_1,n_2^2]\mid ...\mid[n_1,n_2^k]$ で置き換えることで行われる．ただし，$n_2^1,n_2^2,...,n_2^k$ は N_2 の列挙である．同様にして，reg_2 から reg_2'（$N_1\times N_2$ 上のもう一方の正規表現）をつくる．
(2) reg_1' と reg_2' から 2 つのオートマトンをつくる．
(3) これらの積オートマトンをつくる．
(4) この積オートマトンから正規表現，すなわち $reg_1\oplus reg_2$ をつくる．

積のオートマトンは $TA_3=(N_1\times N_2,\Sigma,s_1\times s_2,P_3)$ である．ただし，

$$\begin{aligned}P_3=\{&[n_1,n_2]\to a[reg_1\oplus reg_2]\mid n_1\to a[reg_1]\in P_1,\ n_2\to a[reg_2]\in P_2\}\\&\cup\{[n_1,n_2]\to[n_1,n_2']\mid n_1\in N_1,\ n_2\to n_2'\in P_2\}\\&\cup\{[n_1,n_2]\to[n_1',n_2]\mid n_1\to n_1'\in P_1,\ n_2\in N_2\}\end{aligned}$$

定義 8 (有限木オートマトンの和演算) 本稿では有限木オートマトンの和演算を次のように定義する．有限木オートマトンを $TA_1=(N_1,\Sigma_1,s_1,P_1)$，$TA_2=(N_2,\Sigma_2,s_2,P_2)$ とする．一般性を失うことなく，$\Sigma_1=\Sigma_2=\Sigma$，$N_1\cap N_2=\emptyset$ と仮定できる．$TL(TA_1)$ と $TL(TA_2)$ の和を受理する有限木オートマトンは $TA_3=(N_1\cup N_2\cup\{s_3\},\Sigma,s_3,P_3)$ である．ただし，s_3 は新しい初期状態であり，P_3 は以下で定義される．

$$P_3=P_1\cup P_2\cup\{s_3\to s_1\}\cup\{s_3\to s_2\}$$

2.4 対応の判定問題

ここでは本稿で考える問題を定義する．

定義 9 (対応の判定問題) スキーマを TA_s, XPath 式を p, p の位置の系列を $\alpha_1,\ldots,\alpha_k$,

TA_s 中の型の系列を $n_{s_1}, \ldots, n_{s_k}$ とすると，本稿で考える問題は以下が成り立つかどうかを判定する問題である．

- $\exists t \in TL(TA_s),\ \exists I,\ \exists \theta,\ \exists \rho$
  - $(t, \theta, \rho, \epsilon) \models p$ かつ，
  - $\forall i (1 \leq i \leq k),\ (t, \theta, \rho, \alpha_i) \models p|_{\alpha_i} \ \wedge \ I(\rho(\alpha_i)) = n_{s_i}$

ただし，I は t の解釈である．

3. XPath 式から有限木オートマトンへの変換

ここでは XPath 式に対する有限木オートマトンを構成する手法を与える．本稿で提案する変換手法では，定義 2 の XPath 式の構文定義に従って以下のように変換を進める．まず $\chi :: l$ を有限木オートマトンに変換する．この変換は各軸 χ に対して用意する有限木オートマトンを用いて行う．用意された有限木オートマトンには，入力状態と出力状態があり，それぞれ $\chi :: l$ に対するコンテクストノード，$\chi :: l$ が指定するノードを出力する状態である．

次に XPath 式の構文に従って，ボトムアップ的に有限木オートマトンを構成していく．この際，有限木オートマトンの出力状態と入力状態を考慮する必要がある．

3.1 各軸に対する有限木オートマトン

ここでは $\chi :: l$ という形の部分式に対する有限木オートマトンを与える．部分式に出現する軸によって，各部分式に対する有限木オートマトンが異なる．以下では，$A \to \sigma(\cdots)$ という記述は $\{A \to \sigma(\cdots) \mid \sigma \in \Sigma\}$ を表すものとする．

- $\cdot :: l$ に対する有限木オートマトン
  - 初期状態：B
  - $A \to \sigma(A^*)$
  - $B \to \sigma(A^*\ (B|C)\ A^*)$
  - $B \to C$
  - $C \to D$
  - $D \to l(A^*)$
- $\downarrow :: l$ に対する有限木オートマトン
  - 初期状態：B
  - $A \to \sigma(A^*)$
  - $B \to \sigma(A^*\ (B|C)\ A^*)$
  - $B \to C$
  - $C \to \sigma(A^*\ D\ A^*)$
  - $D \to l(A^*)$
- $\downarrow^+ :: l$ に対する有限木オートマトン
  - 初期状態：B_1
  - $A \to \sigma(A^*)$
  - $B_1 \to \sigma(A^*\ (B_1|C)\ A^*)$
  - $B_1 \to C$
  - $C \to \sigma(A^*\ (B_2|D)\ A^*)$
  - $B_2 \to \sigma(A^*\ (B_2|D)\ A^*)$
  - $D \to l(A^*)$
- $\downarrow^* :: l$ に対する有限木オートマトン
  - 初期状態：B_1
  - $A \to \sigma(A^*)$
  - $B_1 \to \sigma(A^*\ (B_1|C)\ A^*)$
  - $B_1 \to C$
  - $C \to \sigma(A^*\ (B_2|D)\ A^*)$
  - $C \to D$
  - $B_2 \to \sigma(A^*\ (B_2|D)\ A^*)$
  - $D \to l(A^*)$
- $\rightarrow^+ :: l$ に対する有限木オートマトン
  - 初期状態：B_1
  - $A \to \sigma(A^*)$
  - $B_1 \to \sigma(A^*\ (B_1|B_2)\ A^*)$
  - $B_1 \to B_2$
  - $B_2 \to \sigma(A^*\ C\ A^*\ D\ A^*)$
  - $C \to \sigma(A^*)$
  - $D \to l(A^*)$
- $\searrow :: l$ に対する有限木オートマトン
  - 初期状態：B_1
  - $A \to \sigma(A^*)$
  - $B_1 \to \sigma(A^*\ (B_1|B_2)\ A^*)$

- $B_1 \rightarrow B_2$
- $B_2 \rightarrow \sigma(A^*\ (B_3|C)\ A^*\ (B_4|D)\ A^*)$
- $B_3 \rightarrow \sigma(A^*\ (B_3|C)\ A^*)$
- $B_4 \rightarrow \sigma(A^*\ (B_4|D)\ A^*)$
- $C \rightarrow \sigma(A^*)$
- $D \rightarrow l(A^*)$

各軸に対する有限木オートマトンの状態のうち，C は部分式のコンテクストノードに対応する状態であり，D は部分式が指定するノードに対応する状態である．これらの有限オートマトンの入力状態集合は $\{C\}$ であり，出力状態集合は $\{D\}$ である．

XPath の軸の中で，↑，$\uparrow^+$，$\uparrow^*$，←，↙ のことを reverse step と呼び，それぞれ ↓，$\downarrow^+$，$\downarrow^*$，→，↘ とは逆向きにノードを辿る軸である．reverse step を含む部分式を有限木オートマトンに変換するには，ここで与えた各軸に対する有限木オートマトンの規則のうち，C と D の規則を入れ替えることで対応可能である．

3.2 XPath 式の有限木オートマトンの構成法

ここでは XPath 式の有限木オートマトンの構成法を与える．

p_1/p_2，$p[q]$，$p_1 \cup p_2$，$q_1 \wedge q_2$，$q_1 \vee q_2$ それぞれの場合について XPath 式の有限木オートマトンの構成法を与える．p の有限木オートマトン TA_p の状態集合を N_p，入力状態集合を $NI_p \subset N_p$，出力状態集合を $NO_p \subset N_p$ と表す．

有限木オートマトンの積演算は定義 7 に従い，和演算は定義 8 に従う．

3.2.1 p_1/p_2 の場合

p_1，p_2 はそれぞれ有限木オートマトン TA_{p_1}，TA_{p_2} に変換済みであるとする．TA_{p_1} の出力状態と，TA_{p_2} の入力状態以外の状態のペアや，TA_{p_1} の出力状態以外の状態と，TA_{p_2} の入力状態のペアが，積を取ったオートマトンの状態に現れないように積を取る．すなわち，積を取ったオートマトン TA_{p_1/p_2} の状態集合は，

$$N_{p_1/p_2} = \{[n_{p_1}, n_{p_2}] \mid (n_{p_1} \in NO_{p_1},\ n_{p_2} \in NI_{p_2}) \vee (n_{p_1} \in (N_{p_1} - NO_{p_1}),\ n_{p_2} \in (N_{p_2} - NI_{p_2}))\}$$

となる．また，入力状態集合は，

$$NI_{p_1/p_2} = \{[nI_{p_1}, n_{p_2}] \mid nI_{p_1} \in NI_{p_1},\ n_{p_2} \in (N_{p_2} - NI_{p_2})\}$$

出力状態集合は，

$$NO_{p_1/p_2} = \{[n_{p_1}, nO_{p_2}] \mid n_{p_1} \in (N_{p_1} - NO_{p_1}),\ nO_{p_2} \in NO_{p_2}\}$$

となる．

3.2.2 $p[q]$ の場合

p，q はそれぞれ有限木オートマトン TA_p，TA_q に変換済みであるとする．TA_p の出力状態と，TA_q の入力状態以外の状態のペアや，TA_p の出力状態以外の状態と，TA_q の入力状態のペアが，積を取ったオートマトンの状態に現れないように積を取る．すなわち，積を取ったオートマトン $TA_{p[q]}$ の状態集合は，

$$N_{p[q]} = \{[n_p, n_q] \mid (n_p \in NO_p,\ n_q \in NI_q) \vee (n_p \in (N_p - NO_p),\ n_q \in (N_q - NI_q))\}$$

となる．また，入力状態集合は，

$$NI_{p[q]} = \{[nI_p, n_q] \mid nI_p \in NI_p,\ n_q \in (N_q - NI_q)\}$$

出力状態集合は，

$$NO_{p[q]} = \{[nO_p, n_q] \mid nO_p \in NO_p,\ n_q \in NI_q\}$$

となる．

3.2.3 $p_1 \cup p_2$ の場合

p_1，p_2 はそれぞれ有限木オートマトン TA_{p_1}，TA_{p_2} に変換済みであるとする．TA_{p_1}，TA_{p_2} の和オートマトン $TA_{p_1 \cup p_2}$ を構成する．新しい初期状態を n_{ini} とする．和を取ったオートマトン $TA_{p_1 \cup p_2}$ の状態集合は，

$$N_{p_1 \cup p_2} = N_{p_1} \cup N_{p_2} \cup \{n_{ini}\}$$

となる．また，入力状態集合は，

$$NI_{p_1 \cup p_2} = \{n_{ini}\}$$

出力状態集合は，

$$NO_{p_1 \cup p_2} = NO_{p_1} \cup NO_{p_2}$$

となる．

3.2.4 $q_1 \wedge q_2$ の場合

q_1，q_2 はそれぞれ有限木オートマトン TA_{q_1}，TA_{q_2} に変換済みであるとする．TA_{q_1}，TA_{q_2} の積オートマトン $TA_{q_1 \wedge q_2}$ を構成する．$TA_{q_1 \wedge q_2}$ の状態集合は

$$N_{q_1 \wedge q_2} = \{[n_{q_1}, n_{q_2}] \mid n_{q_1} \in N_{q_1},\ n_{q_2} \in N_{q_2})\}$$

となる．また，入力状態集合は，

$$NI_{q_1 \wedge q_2} = \{[n_{q_1}, n_{q_2}] \mid (n_{q_1} \in NI_{q_1},\ n_{q_2} \in N_{q_2}) \cup (n_{q_1} \in N_{q_1},\ n_{q_2} \in NI_{q_2})\}$$

出力状態集合は，

$$NO_{q_1 \wedge q_2} = \{[n_{q_1}, n_{q_2}] \mid (n_{q_1} \in NO_{q_1},\ n_{q_2} \in N_{q_2}) \cup (n_{q_1} \in N_{q_1},\ n_{q_2} \in NO_{q_2})\}$$

となる．

3.2.5 $q_1 \vee q_2$ の場合

q_1, q_2 はそれぞれ有限木オートマトン TA_{q_1}, TA_{q_2} に変換済みであるとする．TA_{q_1}, TA_{q_2} の和オートマトン $TA_{q_1 \cup q_2}$ を構成する．新しい初期状態を n_{ini} とする．和を取ったオートマトン $TA_{q_1 \cup q_2}$ の状態集合は，

$$N_{q_1 \cup q_2} = N_{q_1} \cup N_{q_2} \cup \{n_{ini}\}$$

となる．また，入力状態集合は，

$$NI_{q_1 \cup q_2} = \{n_{ini}\}$$

出力状態集合は，

$$NO_{q_1 \cup q_2} = NO_{q_1} \ \cup \ NO_{q_2}$$

となる．

4. スキーマで定義された型と XPath 部分式との対応の解析

ここでは，スキーマで定義された型と XPath 部分式との対応の解析手法を与える．まず，XPath 式 p から得た有限木オートマトン TA_p と，スキーマを表す有限木オートマトン TA_s の積オートマトン $TA_{p \cap s}$ を構成する．次に $TA_{p \cap s}$ を解析することで，本稿で考える定義 9 の問題の真偽を判定する．以下の条件を満たすときかつそのときのみ問題は真と判定される．

- ある $t \in TL(TA_{p \cap s})$, ある I, t のあるノード $v_1, \ldots, v_k$ について
  - $I(v_i)(1 \leq i \leq k)$ が $p|_{\alpha_i}$ に対して構成された有限木オートマトンのある出力状態および n_{s_i} の両方を成分として含む

ただし，I は $TA_{p \cap s}$ に対する t の解釈である．上記の条件の判定は以下のようにして行える．$TA_{p \cap s}$ の各状態 $n_{p \cap s}$ から同時に到達可能な状態の極大集合をすべて求める．得た状態の極大集合に，各部分式の出力状態と指定された型に対応する状態が対となっている状態のすべてが含まれているかどうかを判定する．このような極大集合が存在すれば，上記の条件は真と判定できる．

以下に，上記のような条件が成立するならば，問題が真と判定でき，存在しないならば偽と判定できることを簡単に説明する．上記の条件が成り立つことと，問題定義 9 の数式が成り立つことが同値であることを示せばよい．

上記の条件が成り立つと仮定する．$TA_{p \cap s}$ は XPath 式の有限木オートマトン表現 TA_p とスキーマの有限木オートマトン表現 TA_s の積オートマトンであるから，$t \in TL(TA_{p \cap s})$ であるような t に対して以下が成り立つ．

$$t \in TL(TA_s) \wedge t \in TL(TA_p)$$

XPath 式 p から TA_p への変換が正しいと仮定すると $t \in TL(TA_p)$ より，t は XPath 式 p を充足するので以下が成り立つ．

$$\exists\theta,\ \exists\rho,\ (t, \theta, \rho, \epsilon) \models p$$

$TA_{p \cap s}$ は TA_p と TA_s の積オートマトンなので，$TA_{p \cap s}$ に対する木 t の任意の解釈 I について，TA_p に対する t の解釈 I_p と TA_s に対する t の解釈 I_s が存在して，任意の t のノード v について $I(v) = [I_p(v), I_s(v)]$ である．したがって上記の条件を満たすような I について以下が成り立つ．

$$\forall i(1 \leq i \leq k),\ (t, \theta, \rho, \alpha_i) \models p|_{\alpha_i} \ \wedge \ I_s(\rho(\alpha_i)) = n_{s_i}$$

以上から，上記の条件が成立すれば問題定義 9 の数式を満たすことが分かる．

問題定義 9 の数式が満たされるならば上記の条件が成立することを示す．問題定義 9 の数式が満たされている仮定から以下が満たされている．

- $\exists t \in TL(TA_s),\ \exists I_s,\ \exists\theta,\ \exists\rho$
  - $(t, \theta, \rho, \epsilon) \models p$ かつ，
  - $\forall i(1 \leq i \leq k),\ (t, \theta, \rho, \alpha_i) \models p|_{\alpha_i} \ \wedge \ I_s(\rho(\alpha_i)) = n_{s_i}$

$(t, \theta, \rho, \epsilon) \models p$ より，$t \in TL(TA_p)$ であり，TA_p に対する t の解釈 I_p が存在する．さらに $\forall i(1 \leq i \leq k),\ (t, \theta, \rho, \alpha_i) \models p|_{\alpha_i} \ \wedge \ I_s(\rho(\alpha_i)) = n_{s_i}$ であるから，各 i について $p|_{\alpha_i}$ の出力状態と n_{s_i} に対応する t のノード v_i が存在する．ここで $TA_{p \cap s}$ は XPath 式の有限木オートマトン表現 TA_p とスキーマの有限木オートマトン表現 TA_s の積オートマトンであるから，$t \in TL(TA_p)$ かつ $t \in TL(TA_s)$ である t について $t \in TL(TA_{p \cap s})$ である．$TA_{p \cap s}$ に対する t の解釈 I として $I(v_i) = [I_p(v_i), I_s(v_i)]$ を満たす I を考えると，その I について上記の条件が成り立つ．以上から，問題定義 9 の数式が満たされるならば上記の条件が成立することについて示せた．

5. ま と め

本稿では，XPath 式を有限木オートマトン変換する手法，有限木オートマトンとスキーマを表す有限木オートマトンの積オートマトンを解析することで，XPath 部分式に対してスキーマの型が指定されている場合に，スキーマに従う木が XPath 式を充足しかつ，XPath 部分式が指定し得る XML 文書の各要素がスキーマの指定された型によって定義されているかどうかを判定する手法を与えた．今後，本稿で提案した手法のより形式的証明を与え，計算複雑性について考察する予定である．

Vol.2009-MPS-75 No.20
2009/9/10

参 考 文 献

1) Clark, J., DeRose, S. et al.: XML Path Language (XPath) Version 1.0, *Available on: http://www.w3.org/TR/xpath* (1999).
2) Benedikt, M., Fan, W. and Geerts, F.: XPath Satisfiability in the Presence of DTDs, *Journal of the ACM*, Vol.55, No.2, pp.25–36 (2008).
3) Miklau, G. and Suciu, D.: Containment and Equivalence for a Fragment of XPath, *Journal of the ACM*, Vol.51, No.1, pp.2–45 (2004).
4) Ishihara, Y., Morimoto, T., Shimizu, S., Hashimoto, K. and Fujiwara, T.: A Tractable Subclass of DTDs for XPath Satisfiability with Sibling Axes, *In: Proceedings of the 12th International Symposium on Database Programming Languages*, pp. 68–83 (2009).
5) Murata, M., Lee, D., Mani, M. and Kawaguchi, K.: Taxonomy of XML Schema Languages Using Formal Language Theory, *ACM Transactions on Internet Technology*, Vol.5, No.4, pp.660–704 (2005).