# 半構造データにおける極大頻出タグ木パターンの発見について

宮原 哲浩 † 正代 隆義 ‡ 内田 智之 † 高橋 健一 † 上田 祐彰 †

† 広島市立大学 情報科学部
‡ 九州大学大学院 システム情報科学研究院

† 〒 731-3194 広島市安佐南区大塚東 3-4-1
Tel:082-830-1576, Email:miyahara@its.hiroshima-cu.ac.jp

**あらまし**　XML ファイルに代表される半構造データからの知識発見が注目を集めている．このような半構造データに頻出する木構造パターンを発見する方法を提案する．タグ木パターン (tag tree pattern) とは，XML ファイルに代表されるタグ付き半構造データの木構造パターンを表現するための，構造的な変数を含むデータ構造である．変数には，任意の木を代入することができ，辺ラベルはタグまたはキーワードである．半構造データにおける極大頻出タグ木パターンを発見する方法と，その実現について述べる．

**キーワード**　知識発見，Web マイニング，半構造データ，XML ファイル，タグ木パターン

# Discovery of Maximally Frequent Tag Tree Patterns in Semistructured Data

Tetsuhiro Miyahara† Takayoshi Shoudai‡ Tomoyuki Uchida† Kenichi Takahashi† Hiroaki Ueda†

†Faculty of Information Sciences, Hiroshima City University
‡Department of Informatics, Kyushu University

† 3-4-1, Ozuka-higashi, Asaminami-ku, Hiroshima 731-3194, Japan
Tel:082-830-1576, Email:miyahara@its.hiroshima-cu.ac.jp

**Abstract**　Many documents such as Web documents or XML files have no rigid structure. We propose a new method for discovering frequent tree structured patterns in such semistructured Web documents. We consider data mining problems of finding maximally frequent tag tree patterns in semistructured data such as Web documents. A tag tree pattern is an edge labeled tree which has structured variables. An edge label is a tag or a keyword in Web documents, and a variable can be substituted by arbitrary tree.

**Keywords**　knowledge discovery, Web mining, semistructured data, XML file, tag tree pattern

## 1 はじめに

インターネットの発展に伴い，Web 文書も急速に増大している．本研究の目的は，XML/SGML ファイルのような木構造を持つ Web 文書から知識を発見することである．このような Web 文書は，半構造データ (semistructured data) と呼ばれている [1]． 近年，半構造データからのデータマイニングやテキストマイニングが注目を集めている [4, 12, 13]．半構造 Web 文書から，意味がある知識を抽出するためには，まず，それらに頻出する木構造パターンを発見することが必要である．本稿では，半構造データにおける極大頻出タグ木パターンを発見する方法と，その実現について述べる．

本研究では，半構造データを表現するために，Object Exchange Model (OEM) [1] の変種を採用する．例として, 図 1に，XML ファイル *xml_sample* と，その OEM データである木 $o_1$ を図示する．OEM データを表現する木は，子に順序の無い根付き木で，辺ラベルがタグまたはキーワードであり，頂点ラベルを持たないものである．

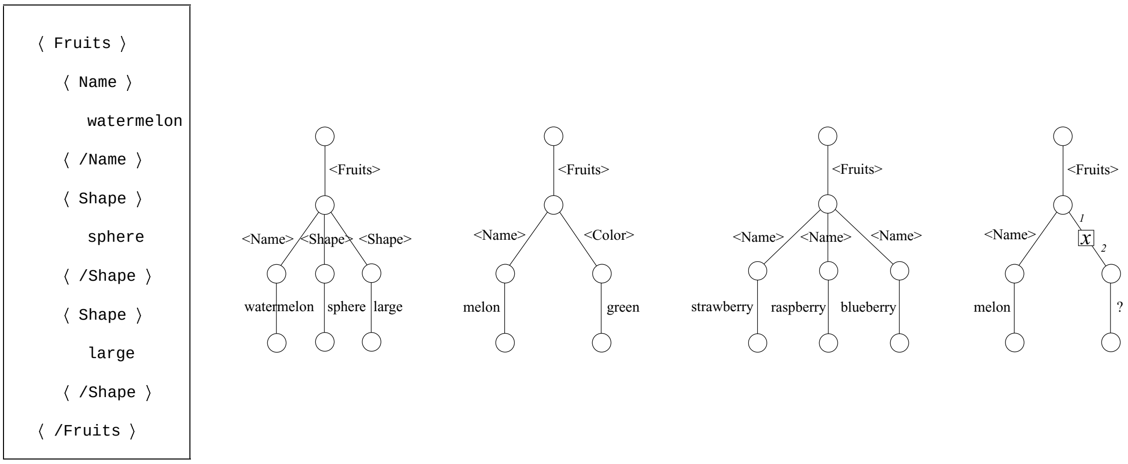
半構造データは，定まったスキーマを持っておらず，その構造は不規則で不完全である．半構造データのための知識表現として，the type of objects [9], tree-expression pattern [11], regular path expression [3] などが提案されている．我々は，木構造をした半構造データのパターンを表現するためのグラフパターンとして，**項木** (term tree) を提案している [6]．項木とは，構造的変数と木構造データからなるパターンである．項木は，任意の木を代入できる構造的変数を持ち，パターン全体のつながりを表現できる点で，半構造データのパターンを表現するための他の表現 [3, 9, 11] とは異なる．

我々は, グラフ構造データを受け取り，Formal Graph System を知識表現言語として仮説を生成する知識発見システム KD-FGS を提案している [8]．このシステムは，すべての入力データと無矛盾な仮説を見つけるものであり，完全なデータに対しては，正しく動くものである．しかしながら，不規則なデータや不完全なデータに対しては，これらのシステムは，自明で意味のない知識を出力する可能性がある．不規則で不完全な構造を持つ半構造データから得られる知識を表現するため，**タグ木パターン** (tag tree pattern) という特別な項木を提案する．例えば，図 1において，タグ木パターン $p$ は，OEM データ $o_1$ と $o_2$ にはマッチするが，OEM データ $o_3$ にはマッチしない．

本研究の目的は，正事例とみなされる木構造データから，これらのデータを特徴づける木構造パターンを抽出することである．よって，与えられた正事例としての木構造データを説明できる，過度に一般化されたパターンは，有用ではない．与えられた正事例を説明できる，最小に一般化されたパターンを見つけることが有用である．このようなパターンを **極大頻出タグ木パターン** (maximally frequent tag tree pattern) と呼ぶ．極大とは，タグ木パターンの頂点の個数が極大ということを表すものであり，その表現能力は極小である．まず，正事例とみなされる木構造データから，１つの極大頻出タグ木パターンを発見する多項式時間アルゴリズムについて述べる．我々は，子に順序が無い根付き木の標準的な表現を生成するアルゴリズム [2] を利用して，極大頻出タグ木パターンをすべて生成するアルゴリズムを提案している [7]．このアルゴリズムを使用することにより，意味がないタグ木パターンを排除することができ，有用なタグ木パターンを見落とすことがなくなる．この発見アルゴリズムを実現し，XML 文書に対して適用した実験結果についても述べる．

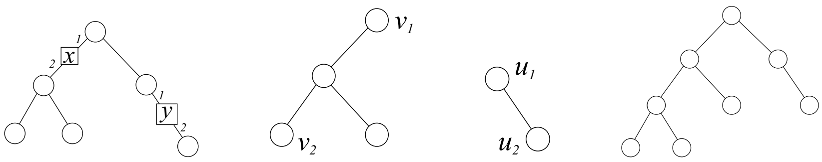
## 2 タグ木パターン

タグ木パターンとデータマイニングについて説明する．$T = (V_T, E_T)$ を頂点集合 $V_T$, 辺集合 $E_T$ を持つ，子に順序が無く, 辺ラベルを持つ根付き木 (以後, **木**) とする．$V_T$ における **変数**とは，$V_T$ の相異なる 2 つの頂点 $u, u'$ から成るリスト $[u, u']$ である．変数のラベルを，**変数ラベル**と呼ぶ．$\Lambda$ を辺ラベルの集合，$X$ を変数ラベルの集合とし，$\Lambda \cap X = \phi$ であるものとする．$V_g$ を頂点集合，$E_g$ を辺集合,$H_g$ を $V_g$ における変数の有限集合とする．ただし，任意の $[u, u'] \in H_g$ に対して，$[u', u] \notin H_g$ とする．$E'_g = \{\{u, v\} \mid [u, v] \in H_g\}$ とする．$V_g$ を頂点集合,$E_g \cup E'_g$ を辺集合とするグラフが木であるとき，3 つ組 $g = (V_g, E_g, H_g)$ を**根付き項木** (rooted term tree)(以後，**項木** (term tree)) という．項木は，すべての変数が相異なる変数ラベルを持つとき，**正則** (regular) であるという．変数を持たない項木を，**基礎項木** (ground term tree) といい，通常の木と同一視する．$f$ と $g$ を少なくとも 2 個の頂点を持つ項木とする．$\sigma = [u, u']$ を $g$ の相異なる 2 つの頂点から成るリストとする．$x := [g, \sigma]$ という形の表現を,$x$ に対する**束縛** (binding) と呼ぶ．束縛 $x := [g, \sigma]$ を $f = (V_f, E_f, H_f)$ に適用して得られる項木 $f\{x := [g, \sigma]\}$ は，次のように定義される．$e_1 = [v_1, v'_1], \ldots, e_m = [v_m, v'_m]$ を，変数ラベル $x$ を持つ $f$ 中の変数とする．$g_1, \ldots, g_m$ を $g$ の $m$ 個のコピーとし，$g_i$ の頂点 $u_i, u'_i$ は，$g$ の頂点 $u, u'$ に対応するとする．それぞれの変数 $e_i$ に対して，$H_f$ から変数 $e_i = [v_i, v'_i]$ を削除し，$e_i$ の頂点 $v_i, v'_i$ と $g_i$ の頂点 $u_i, u'_i$ をそれぞれ同一視することにより，$g_i$ を $f$ に追加する．$f$ の根を項木 $f\{x := [g, \sigma]\}$ の根とする．**代入** (substitution) $\theta$




*xml_sample* $o_1$ $o_2$ $o_3$ $p$

図 1: XML ファイル *xml_sample* とその OEM データとしての木 $o_1$ ． OEM データ $o_1$ と $o_2$ にはマッチするが, $o_3$ にはマッチしない，タグ木パターン $p$.




$t$ $t_1$ $t_2$ $t\theta$

図 2: 基礎項木 $t_1$, $t_2$. 代入 $\theta = \{x := [t_1, [v_1, v_2]], y := [t_2, [u_1, u_2]]\}$ を正則項木 $t$ に適用して得られる代入例 $t\theta$.

とは，束縛の有限集合 $\{x_1 := [g_1, \sigma_1], \cdots, x_n := [g_n, \sigma_n]\}$ のことである．ここで， $x_i$ は， $X$ の相異なる変数ラベルとする．代入 $\theta$ を項木 $f$ に適用して得られる項木 (**代入例** (instance) という) $f\theta$ とは， $f$ に対して $\theta$ 中の束縛を同時に適用して得られる項木のことである．

図 2において，正則項木，代入，代入例の例をあげる．変数をその要素へ線が延びている箱で表現し，要素の順番はその線に付加された数字が示すものとする．変数ラベルを，この箱の中に表示する．

**タグ木パターン** (tag tree pattern). $\Lambda_{Tag}$ と $\Lambda_{KW}$ を無限個の語から成る言語とし, $\Lambda_{Tag} \cap \Lambda_{KW} = \emptyset$ であるとする． $\Lambda_{Tag}$,$\Lambda_{KW}$ の語をそれぞれ，**タグ** (tag), **キーワード** (keyword) と呼ぶ．**タグ木パターン** (tag tree pattern) とは，辺ラベルがタグかキーワードか特別な記号 “?” であるような，正則項木のことである．変数を持たないタグ木パターンを，**基礎タグ木パターン** (ground tag tree pattern) という．タグ木パターンの辺 $\{v, v'\}$ と，木の辺 $\{u, u'\}$ について，次の条件 (1)-(3) を満たすときに， $\{v, v'\}$ が $\{u, u'\}$ と**マッチする** (match) という． (1) $\{v, v'\}$ の辺ラベルがタグであれば， $\{u, u'\}$ の辺ラベルも同じタグである．または， $\{u, u'\}$ の辺ラベルは，タグの間の同一関係の下で，同じとみなされるタグである． (2) $\{v, v'\}$ の辺ラベルがキーワードであれば， $\{u, u'\}$ の辺ラベルもキーワードであり,$\{v, v'\}$ の辺ラベルは $\{u, u'\}$ の辺ラベルの部分文字列である． (3) $\{v, v'\}$ の辺ラベルが “?” であれば， $\{u, u'\}$ の辺ラベルは任意でよい．

基礎タグ木パターン $\pi = (V_\pi, E_\pi, \emptyset)$ が，木 $T = (V_T, E_T)$ に**マッチする**とは，次の条件 (1)-(3) を満たすような $V_\pi$ から $V_T$ への全単射 $\varphi$ が存在するときにいう． (1) $\pi$ の根は， $\varphi$ により $T$ の根に対応する． (2) $\{v, v'\} \in E_\pi$ と $\{\varphi(v), \varphi(v')\} \in E_T$ が同値である． (3) 任意の $\{v, v'\} \in E_\pi$ について， $\{v, v'\}$ は $\{\varphi(v), \varphi(v')\}$

とマッチする．タグ木パターン $\pi$ が木 $T$ に**マッチする**とは，ある代入 $\theta$ があって，$\pi\theta$ が基礎タグ木パターンであり，$\pi\theta$ が $T$ とマッチするときにいう．タグ木パターン $\pi$ の**言語** $L(\pi)$ は，$L(\pi) = \{木T \mid \piがTとマッチする\}$ と定義され，$\pi$ の表現能力を表すものである．

**データマイニング.**
**半構造データの集合** $\mathcal{D} = \{T_1, T_2, \ldots, T_m\}$ とは，木の有限集合である．タグ木パターン $\pi$ の $\mathcal{D}$ に関するマッチ数 $match_{\mathcal{D}}(\pi)$ とは，$\pi$ がマッチするような 木$T_i \in \mathcal{D}$ $(1 \leq i \leq m)$ の個数のことである．$\pi$ の $\mathcal{D}$ に関する**頻度** (frequency) $supp_{\mathcal{D}}(\pi)$ を，$supp_{\mathcal{D}}(\pi) = match_{\mathcal{D}}(\pi)/m$ と定義する．$\sigma$ を $0 \leq \sigma \leq 1$ であるような実数とする．タグ木パターン $\pi$ が $\mathcal{D}$ に関して $\sigma$ **頻出** ($\sigma$-frequent) であるとは，$supp_{\mathcal{D}}(\pi) \geq \sigma$ であるときにいう．$\Pi(L)$ は，そのすべての辺ラベルが $L$ に属するようなタグ木パターン全体の集合を表すとする．$Tag$ を $\Lambda_{Tag}$ の有限部分集合とし，$KW$ を $\Lambda_{KW}$ の有限部分集合とする．

タグ木パターン $\pi \in \Pi(Tag \cup KW \cup \{?\})$ が $\mathcal{D}$ に関して**極大** $\sigma$ **頻出** (maximally $\sigma$-frequent) であるとは，次の (1) 及び (2) が成り立つときにいう．(1) $\pi$ は，$\sigma$ 頻出である．(2) 任意のタグ木パターン $\pi' \in \Pi(Tag \cup KW \cup \{?\})$ に対して，$L(\pi') \subsetneq L(\pi)$ であれば，$\pi'$ は $\sigma$ 頻出ではない．図 1において，例として，半構造データの集合 $\{o_1, o_2, o_3\}$ に関して，極大 $\frac{2}{3}$ 頻出なタグ木パターン $p \in \Pi(\{\langle\mathrm{Fruits}\rangle, \langle\mathrm{Name}\rangle, \langle\mathrm{Shape}\rangle\} \cup \{\mathrm{melon}\} \cup \{?\})$ を図示する．$p$ は $o_1$ と $o_2$ にはマッチするが，$o_3$ にはマッチしない．

## 3　極大頻出タグ木パターンを 1 つ見つける問題

本節では，次のデータマイニング問題を考える．

**極大頻出タグ木パターンを 1 つ見つける問題.**
**入力:**　半構造データ (正事例) の集合 $\mathcal{D}$，閾値 $\sigma(0 \leq \sigma \leq 1)$，タグの有限集合 $Tag$，キーワードの有限集合 $KW$．
**問題:** $\Pi(Tag \cup KW \cup \{?\})$ 中のタグ木パターンであって,$\mathcal{D}$ に関して極大 $\sigma$ 頻出であるものを，1 つ見つけよ．

正則項木に対する多項式時間マッチングアルゴリズム [5] を拡張することにより．　タグ木パターン $\pi$ の $\mathcal{D}$ に関するマッチ数 $match_{\mathcal{D}}(\pi)$ を，多項式時間で計算することができる．極大頻出タグ木パターンを 1 つ見つける問題は，多項式時間で解くことができる．すなわち，図 3のアルゴリズム MF-TTP が，極大頻出タグ木パターンを 1 つ見つける多項式時間アルゴリズムである．このアルゴリズムは，辺ラベルが無限にある場合の正則項木の極小言語を計算する多項式時間アルゴリズム [10] を拡張したものである．

## 4　極大頻出タグ木パターンをすべて生成する問題

本節では，次のデータマイニング問題を考える．

**極大頻出タグ木パターンをすべて生成する問題.**
**入力:**　半構造データ (正事例) の集合 $\mathcal{D}$，閾値 $\sigma(0 \leq \sigma \leq 1)$，タグの有限集合 $Tag$，キーワードの有限集合 $KW$．
**問題:** $\Pi(Tag \cup KW \cup \{?\})$ 中のタグ木パターンであって,$\mathcal{D}$ に関して極大 $\sigma$ 頻出であるものを，すべて生成せよ．

我々は，子に順序が無い根付き木の標準的な表現を生成するアルゴリズム [2] を利用して，この問題を解くアリゴリズム，すなわち，極大 $\sigma$ 頻出タグ木パターンをすべて生成するアルゴリズムを提案している [7].

このアルゴリズムを実現してプロトタイプシステムを作成し，XML 文書に対して適用した実験結果について述べる．　実現は GCL2.2 で行い，SUN Ultra-10 clock 333 MHz 上で実験した．図 1における *xml_sample* と同様な形式をした，衣服の販売に関するサンプル XML 文書から実験に使用したサンプルファイルを作成した．このサンプルファイルは，32 個の木構造データから成る．木の頂点の個数は最大で 58 である．深さの最大値は 3 であり，頂点の子の最大数は 6 である．図 4で示されている実験について説明する．"<Quarter>" と "<Description>" をタグとして，"Summer" と "Shirt" をキーワードとして，プロトタイプシステムに与えた．システムは，与えられた閾値 $\sigma$ に対して，サンプルデータに関して極大 $\sigma$ 頻出なタグ木パターンをすべて生成した．

図 4 の実験結果を，実験 Exp. 1 の最後の列を例にとって，説明する．仮説であるタグ木パターンの頂点の個数の最大値 ("max # of vertices in TTPs") を指定できるようにしており，この場合は 7 である．最小頻度を表す閾値 $\sigma$ は 0.3 である．最大で 7 個の頂点から成る極大 $\sigma$ 頻出タグ木パターンの総数 ("# of max freq TTPs") は 34 である．実行時間は，6162 秒である．この実験 Exp. 1 で見つかった極大頻出タグ木パターンの 1 つを図 4に示す．実験 Exp. 2 は，最小頻度を

```
Algorithm MF-TTP;
begin
  π := Basic-Tree(𝒟);
  foreach variable [u, v] ∈ H_π do begin
    let π' be a tag tree pattern which is obtained from π by replacing variable [u, v]
    with an edge labeled with "?";
    if π' is σ-frequent w.r.t. 𝒟 then π := π'
  end;
  foreach edge {u, v} ∈ E_π with an edge label "?" do begin
    foreach edge label w ∈ Tag ∪ KW do begin
      let π' be a tag tree pattern which is obtained from π by replacing "?" with w;
      if π' is σ-frequent w.r.t. 𝒟 then begin π := π'; break end
    end
  end;
  return π
end;
```

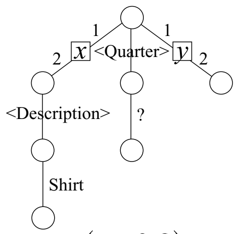
```
Procedure Basic-Tree(𝒟);
begin // Each variable is assumed to be labeled with a distinct variable label.
  d := 0; π := ({r}, ∅, ∅);
  π := breadth-expansion(r, π, 𝒟);
  max-depth :=the depth of π; d := d + 1;
  while d ≤ max-depth − 1 do begin
    v :=a vertex at depth d which is not yet visited;
    π :=breadth-expansion(v, π, 𝒟);
    while there exists a sibling of v which is not yet visited do begin
      Let v' be a sibling of v which is not yet visited;
      π :=breadth-expansion(v', π, 𝒟)
    end;
    d := d + 1
  end;
  return π
end;
```

```
Procedure breadth-expansion(v, π, 𝒟);
begin
  π' :=depth-expansion(v, π, 𝒟);
  while π ≠ π' do begin
    π := π';
    π' :=depth-expansion(v, π, 𝒟)
  end;
  return π
end;
```

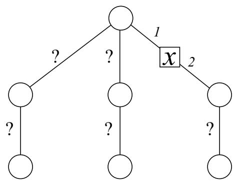
```
Procedure depth-expansion(v, π, 𝒟);
begin
  Let π be (V_π, ∅, H_π);
  Let v' be a new vertex and [v, v'] a new variable;
  π' := (V_π ∪ {v'}, ∅, H_π ∪ {[v, v']});
  while π' is σ-frequent w.r.t. 𝒟 do begin
    π := π'; v := v';
    Let v' be a new vertex and [v, v'] a new variable;
    π' := (V_π ∪ {v'}, ∅, H_π ∪ {[v, v']});
  end;
  return π
end;
```

図 3: MF-TTP: 極大 $\sigma$ 頻出タグ木パターンを 1 つ見つけるアルゴリズム

| Experiment (frequency $\sigma$) | Exp.1 ($\sigma$=0.3) | | | | | | Exp.2 ($\sigma$=0.5) | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| max # of vertices in TTPs | 2 | 3 | 4 | 5 | 6 | 7 | 2 | 3 | 4 | 5 | 6 | 7 |
| # of max freq TTPs | 1 | 2 | 4 | 9 | 15 | 34 | 1 | 2 | 2 | 4 | 3 | 5 |
| run time (secs) | 7 | 32 | 159 | 630 | 1948 | 6162 | 9 | 39 | 107 | 312 | 627 | 2721 |




($\sigma$=0.3)




($\sigma$=0.5)

図 4: 極大 $\sigma$ 頻出タグ木パターンをすべて生成する実験．実験で得られた極大 $\sigma$ 頻出タグ木パターン．

表す閾値 $\sigma$ だけを 0.5 にしたものである．この実験で見つかった極大頻出タグ木パターンの 1 つを図 4に示す．

## 5 おわりに

XML ファイルのような半構造 Web 文書からの知識発見のため，半構造データにおける極大頻出タグ木パターンを発見する方法と，その実現について述べた．本研究の一部は，広島市立大学特定研究費 (0066,0070) の助成による．

## 参考文献

[1] S. Abiteboul, P. Buneman, and D. Suciu. *Data on the Web: From Relations to Semistructured Data and XML.* Morgan Kaufmann, 2000.

[2] T. Beyer and S. Hedetniemi. Constant time generation of rooted trees. *SIAM J. Comput.*, 9:706–712, 1980.

[3] M. Fernandez and D. Suciu. Optimizing regular path expressions using graph schemas. *Proc. Intl. Conf. on Data Engineering (ICDE-98)*, pages 14–23, 1998.

[4] R. Fujino, H. Arimura, and S. Arikawa. Discovering unordered and ordered phrase association patterns for text mining. *Proc. PAKDD-2000, Springer-Verlag, LNAI 1805*, pages 281–293, 2000.

[5] T. Miyahara, T. Shoudai, T. Uchida, T. Kuboyama, K. Takahashi, and H. Ueda. Discovering new knowledge from graph data using inductive logic programming. *Proc. ILP-99, Springer-Verlag, LNAI 1634*, pages 222–233, 1999.

[6] T. Miyahara, T. Shoudai, T. Uchida, K. Takahashi, and H. Ueda. Polynomial time matching algorithms for tree-like structured patterns in knowledge discovery. *Proc. PAKDD-2000, Springer-Verlag, LNAI 1805*, pages 5–16, 2000.

[7] T. Miyahara, T. Shoudai, T. Uchida, K. Takahashi, and H. Ueda. Discovery of frequent tree structuted patterns in semistructured web documents. *Proc. PAKDD-2001, Springer-Verlag, LNAI 2035*, pages 47–52, 2001.

[8] T. Miyahara, T. Uchida, T. Shoudai, T. Kuboyama, K. Takahashi, and H. Ueda. Discovering knowledge from graph structured data by using refutably inductive inference of formal graph systems. *IEICE Trans. Inf. Syst.*, E84-D(1):48–56, 2001.

[9] S. Nestorov, S. Abiteboul, and R. Motwani. Extracting schema from semistructured data. *Proc. ACM SIGMOD Conf.*, pages 295–306, 1998.

[10] T. Shoudai, T. Uchida, and T. Miyahara. Polynomial time algorithms for finding unordered tree patterns with internal variables. *Proc. FCT-2001, Springer-Verlag, LNCS (to appear)*, 2001.

[11] K. Wang and H. Liu. Discovering structural association of semistructured data. *IEEE Trans. Knowledge and Data Engineering*, 12:353–371, 2000.

[12] 安部 潤一郎, 藤野 亮一, 下薗 真一, 有村 博紀, 有川 節夫. テキストデータからの高速データマイニング – 探索的文書ブラウジングとウェブデータへの応用 –. 人工知能学会誌, 15:618–628, 2000.

[13] 鷲尾 隆. 構造化データに関するマイニング技術の変遷と展望. *2000* 年度人工知能学会全国大会論文集, pages (93)–(96), 2000.