HTMLラッパ自動構築手法の提案

Proposal of a Method for Automatic Construction of HTML Wrappers

澤 菜津美 ♡ 森嶋 厚行 ♢
杉本 重雄 ♢ 北川 博之 ♠

Natsumi SAWA Atsuyuki MORISHIMA
Shigeo SUGIMOTO Hiroyuki KITAGAWA

本稿では，HTML で記述された Web コンテンツをラッピングして XML データに変換するソフトウェアである HTML ラッパの構築支援手法について提案する．本アプローチの特徴は，(1)HTML ラッパ構築の問題を，XML データからの DTD 生成の問題の一般化としてモデル化していること，および (2) ラッパ選択のための基準として，DTD の適切性とラッピングの具体性の組合せを利用すること，である．本稿では，提案手法の説明及び実際の Web ページを用いた予備実験の結果を示す．

This paper proposes a method to support the construction of HTML wrappers, which are software modules to transform Web contents written in HTML into XML data. In our approach, (1) we model the problem of wrapper construction as a generalization of the problem of DTD extraction from XML data, and (2) we define a measure to find good wrappers as a combination of the appropriateness of DTDs and the concreteness of wrappers. This paper explains the proposed method and shows the result of applying the proposed method to actual Web pages.

1. はじめに

情報統合問題の鍵となる要素技術の一つとして情報源のラッピング技術がある．本稿では，特に HTML で記述された Web コンテンツをラッピングして XML データに変換するソフトウェアである HTML ラッパの構築支援手法について提案する．

提案手法の第一のポイントは，HTML ラッパ構築の問題を，XML データからの DTD 生成の問題の一般化としてモデル化する事である．まずは，このモデル化について説明する．図 1 は，XML データからの DTD 生成の問題を模式的に表したものである．この問題では，複数の XML データインスタンスが与えられると，それらを Valid とするような DTD_j を列挙し，何らかの基準でスコアをつける (例えば，XTRACT [3] では MDL コストという値を用いる．この MDL コストは，後述するように DTD の **適切性**を 簡潔さと精度の観点から表現したものである)．最後に，そのスコアの性質に応じて，スコアが最小もしくは最大のものを最も適切な DTD とみなし，解とする．これが，XML データからの DTD 生成の問題である．

次に，その一般化として HTML ラッパ生成の問題をモデル化したものについて説明する．

HTML ラッパ生成の問題. 本問題 (図 2) では，与えられるものは XML ではなく HTML データインスタンスの集合である．HTML ラッパは，HTML データから XML データへの変換関数 f_i として表せる．したがって，図中でもそのように記述している．図に示されるように，一般にはこのような変換関数の候補は複数ある．このとき，各 f_i 毎に XML インスタンスの集合が得られるが，DTD 抽出の問題で説明したように，それぞれに対して複数の DTD 候補 $\mathrm{DTD}_{i,j}$ が考えられる．それぞれの $\mathrm{DTD}_{i,j}$ には，対応する関数 $f_{i,j}$ が存在する[1]．次に，何らかの基準でスコアをつけ，$\mathrm{DTD}_{p,q}$ を選択する．最後に，この DTD によって表現される XML データを生成する関数 $f_{p,q}$ を HTML ラッパの解とする．

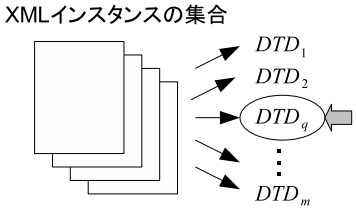

図 1 DTD 抽出の問題
Fig. 1 Problem of the DTD Extraction

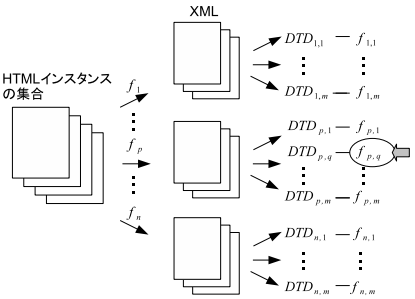

図 2 Wrapper 生成の問題
Fig. 2 Problem of the Wrapper Generation

提案手法の第二のポイントは，DTD 選択のための基準として，**DTD の適切性とラッピングの具体性を組み合わせる**ことである．用語の定義など詳細な説明は後述するが，ここでは直観的な理解のための例を示す．まず，下記の HTML データが存在するとする．各 LI 要素を一つの HTML インスタンスとすると，二つの HTML インスタンスが存在することになる．

```
<li>飯田，澤，森嶋，WISH</li>
<li>澤，森嶋，WISH2</li>
```

HTML ラッパ生成の問題において，この二つの HTML インスタンスが与えられたとする．このとき，カンマで区切られた各文字列をトークンと見なし，次のような 3 つのラッパ (変換関数)f_1,f_2,f_3 を考える．f_1 は各トークンに`<string>`というタグを付ける．f_2 は飯田，澤，森嶋に`<name>`というタグを付け，(英単語として辞書に載っている)WISH には`<word>`, WISH2 には`<string>`をつける．f_3 は飯田，澤，森嶋に`<name>`というタグを付け，WISH と WISH2 には`<string>`を付ける．

このとき，f_1 の出力する XML データの $\mathrm{DTD}_{1,p}$ を `string*` とする．これは，本問題で出力しうる最も簡潔な DTD であろうが，抽象的過ぎて我々が欲しいものではない．次に，f_2 の出力する XML データの $\mathrm{DTD}_{2,q}$ を `(name, name|name, name, name) (word|string)` とする．これは最も具体的なタグ付け

♡ 学生会員 筑波大学大学院 図書館情報メディア研究科
sawa@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, sugimoto}@slis.tsukuba.ac.jp
♠ 正会員 筑波大学大学院 システム情報工学研究科
kitagawa@cs.tsukuba.ac.jp

[1] $f_{i,j}$ は次のようなものである．すなわち，ある f_i が与えられたとき，$f_{i,j}$ $(1 \leq j)$ は，定義域を図 2 にある HTML インスタンスの集合に限定した場合は $f_i = f_{i,j}$ となるが，定義域と値域自体は f_i のそれよりも広く，それぞれ異なるような関数群である．

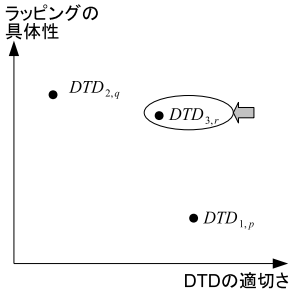

図 3 適切な HTML ラッパの選択
Fig. 3 Selection of Appropriate HTML Wrappers

が行われているが，DTD が複雑すぎる．最後に，f_3 の出力する XML データの $DTD_{3,r}$ として `(name)* string` がある．これは，具体性と簡潔な DTD の両方を持つため，$f_{3,r}$ を，HTML ラッパの解としたい．図 3 はこれを模式的に表したものである．

以上のように，本論文では，HTML ラッパ生成の問題を，XML データからの DTD 生成問題の一般化としてモデル化し，さらに，具体性と DTD の適切性を基準とすることを提案する．後述するように，DTD の適切性を表す MDL コストと，ラッピングの具体性を表す Specificity Value は，負の相関にあるため，これらのトレードオフを見つけることが重要である．我々の知る限り，HTML ラッパの生成の問題に関して，このようにアプローチした提案は存在しない．

関連研究. 既に，Web コンテンツをラッピングするための仕組みは数多く研究されてきた．XWRAP [6] は，HTML データから構造データを抽出するラッピング記述を，ユーザとの対話により半自動生成する．XWRAP による半自動生成の手法は，我々の手法と比較してより人手による干渉を必要とし，かつヒューリスティックベースである点が異なる．W4F [10] は Web 情報源のラッパを生成するためのツールキットである．そこでは Extraction Wizard と呼ばれるツールを提供しているが，これはあくまでもルールを書くためのヒントとなる情報を提供するものである．Gottlob と Koch はラッパ記述のために Monadic Datalog を使用することを提案している [11]．Monadic Datalog は強固な理論的基盤を与えており，我々の提案手法との関連は今後の研究課題である．

Arasu らの論文 [1] では，DB をバックエンドにした Web サイトの定型コンテンツから，テンプレートとデータを分離する手法を提案している．具体的には，複数の定型ページをサンプルとして比較を行うことにより，ページ生成に使われたテンプレートを推測し，データだけを抽出するものである．Crescenzi と Mecca[12] もこれと類似したアプローチをとる．これらの手法は定型のページを多量に持つような Web サイトからの構造抽出には向いているが，我々の想定する応用である半構造 Web コンテンツ管理のためのラッパ構築支援には向いていない．

言語処理の分野においては，特に自然言語のテキストを対象にタグ付けに関する研究が行われている [2][7]．これらと比較した場合，我々の手法は単にタグを付ける手法ではなく，論理的に何らかのデータ要素の集まりと考えられるデータベースライクな半構造データから共通の構造を抽出し，ラッパを自動生成しようとするものである．したがって，本手法を利用すれば，単に情報検索のための手がかりとなる情報を付与するだけでなく，ソフトウェアによる対象データの自動的な構造変換などが可能になる．また，これを実現するために，本手法はタグ付けとスキーマ (DTD) 抽出を連動させている点が，技術的な特徴であるといえる．

本稿の構成は次の通りである．まず，2 章で，本稿において HTML ラッパを記述するための言語として採用する Wraplet について説明する．Wraplet は我々が開発したシンプルなラッピング言語 [4][5] である．3 章で，DTD の適切さを示す尺度として採用する MDL コスト [3] について説明する．4 章で，ラッピングの具体性を表す尺度である Specificity Value (S-Value) について提案する．5 章で，以上を用いて HTML ラッパを生成するための仕組みを提案する．6 章では予備実験の結果を示す．7 章は，まとめと今後の課題である．

2. ラッピング言語 Wraplet

我々はこれまでに HTML で表現された Web コンテンツから構造データを抽出するためのラッピング言語 Wraplet を提案した [4][5]．Wraplet は，特に非定型 Web コンテンツから，構造データを抽出する事を考慮して設計されたものである．

説明のため，下記では，XML データを木の用語を用いて表現することがある．例えば XML 要素を**ノード**とよび，要素の入れ子関係を親ノード，子ノードなどで表現することがある．また，要素名をノードの**ラベル**と呼び，要素が直接含む CDATA をその要素 (ノード) の**値**と呼ぶ．

まず簡単な Wraplet 式の例を示す．図 4(a) のような果物在庫を表現する HTML ページがあり，ラッピングした結果の XML データが図 4(b) であるとする．この変換を行うための Wraplet 式は次のようになる．

```
在庫:/
   {果物:#li/
       [名前:_val(#anytext)\,, 数量:_val(#num)]
   }
```

Wraplet 式は，“ラベル:パターン/子ノードのための (一般には複数の)Wraplet 式” という入れ子構造で表現される．

ここで，**ラベル**は出力する構造データのノードラベルを指示する．例の場合では，在庫，果物，名前，数量がラベルである．**パターン**は，そのノードに対応する HTML データの部分を指定するのに使われる．これは，文字の正規表現や，あらかじめライブラリとして用意しているパターン部品を組み合わせて指定される．**パターン部品**は，よく使うパターンや複雑なパターンに名前を付けたものであり,「#パターン部品名」と表す．例の場合では，`#li` と `_val(#anytext)\,` と `_val(#num)` がパターンであり，そのうち`#li` と`#anytext` と`#num` がパターン部品である．`#li` は `li` 要素内の文字列にマッチし，`#num` は数字にマッチする．`#name` は，人名にマッチするためのパターン部品である．例えば，”Sawa, N.”など，氏名の間にカンマ等が入っている場合でも，適切に氏名を取得する．これらのパターン部品は，5 章で説明する HTML ラッパの生成支援において重要な働きをする．

また，パターンは，**特殊指示子**を含むことが出来る．上の例では`_val()` が特殊指示子である．これは，パターンのその部分にマッチした文字列を該当ノードの値とする指示子である．

在庫の子ノードの式は {...} でくくられているが，これは**繰返し構造**を表す．果物の子ノードの式は [...] でくくられているが，これは**列構造**を表す．

3. DTD の適切さを表す MDL コスト

MDL (Minimum Description Legnth) とは，論文 [8] で提案されている概念であり，DTD スキーマ推論システム XTRACT[3] で DTD を評価するために採用されている．本論文でも [3] の MDL コストを採用し，本章ではそこでの例を用いて説明する．直観的には，ある XML インスタンス集合に対する DTD の MDL コストとは，(A)DTD そのものが簡潔 (サイズが小さい) である事と，(B)DTD の精度が高い事 (入力のインスタンス集合に含まれないインスタンスに対してカバーし過ぎない) 時に，小さくなるような値である．次に説明するように，これらは DTD を表現するための情報量と，与えられたインスタンスをその DTD でエンコードしたときに必要な情報量の和で表現される．

MDL コストの考え方を示す例. XML インスタンスの集合 $I = \{ab, abab, ababab\}$ (ここで，a と b は XML 要素を表し，’ab’ 等は XML インスタンスを表す) に対して，その DTD 候補としては，

(a) 果物在庫リスト
```
<ul>
  <li>りんご,10</li>
  <li>みかん,20</li>
  <li>桃,30</li>
</ul>
```

(b) Wraplet 式でパースした結果
```
<在庫>
    <果物><名前>りんご</名前><数量>10</数量></果物>
    <果物><名前>みかん</名前><数量>20</数量></果物>
    <果物><名前>桃</名前><数量>30</数量></果物>
</在庫>
```

図 4 Wraplet 式の適用例
Fig. 4 Application of a Wraplet Expression

(1)$(a \mid b)*$, (2)$ab \mid abab \mid abababab$, (3)$(ab)*$, (4)$ab \mid ab(ab \mid abab)$ などがある．(1) は $aa...$ や $bb...$ などにもマッチしてしまい，I に出現するインスタンスの構造をうまく捉えていない．一方，(2) と (4) には I に出現するインスタンスの構造が正確に反映されているが，DTD 自体のサイズが大きく簡潔ではない．(3) は，DTD のサイズが小さく，かつインスタンスの特徴をとらえた DTD となっており，与えられたインスタンス集合に対する適切な DTD となる．

次に，MDL コストの算出方法について説明する．与えられた XML インスタンス集合と，それを Valid とするある DTD について，(A) その DTD 自体のサイズ，および，(B) その DTD で与えられた XML データインスタンス集合を表すのに必要な情報量，の合計を MDL コストの値とする．例として，上に述べた例のための MDL コスト計算例を以下に示す．計算方法の詳細は [3] にあるため省略する．

MDL コストの計算例. ここでは，簡単化のため 1 文字に対して 1 コストかかると仮定して計算を行う．DTD(1)$(a \mid b)*$ の場合，まず，(A) は文字列の長さが 6 なので，コストは 6 となる．次に (B) は入力のインスタンスごとに求める．例えばインスタンスの一つ，ab を DTD(1) を使って表すには，まず，$(a \mid b)$ の繰返し回数 (この場合は 1) を指定するために 1 文字必要になる．さらに，繰返し毎に a と b のどちらを選択するか指定するため 2 文字必要となるので，コストの合計は 3 となる．同様に他のインスタンスについてもコストを求めると，$abab$ のコストは 5，$abababab$ のコストは 7 となる．よって，インスタンス集合 I に対する DTD(1) の MDL コストは (A) のコスト 6 と (B) のコストの合計 3+5+7=15 の合計値となるので，21 となる．同様に他の DTD の MDL コストを計算すると，DTD(2) は (A) のコスト 14+(B) のコスト合計 3=17，DTD(3) は (A) のコスト 5+(B) のコスト合計 3=8，DTD(4) は (A) のコスト 14+(B) のコスト合計 5=19 となる．したがって，DTD(3) が最も MDL コストが小さくなるので，最適な DTD として出力される．

4. ラッピングの具体性を表す Specificity Value

DTD の Specificity Value は，ラッピングの具体性を表す非負の実数であり，具体的であるほど，値が大きくなる．この値は，DTD に現れる要素型 (すなわち，各トークンに割当てられるタグの種類) によって決定する．この定義に進む前に，まずトークン型 t の Specificity Value $S\text{-}Value(t)$ を定義する．

定義 4.1. トークン型 t が与えられたとき，$S\text{-}Value(t)$ はそのトークン型の具体性を表す非負の実数であり，その値はトークン型 t のトークンインスタンス集合 I_t の包含関係に基づき，$I(t_1) \subseteq I(t_2)$ ならば $S\text{-}Value(t_1) > S\text{-}Value(t_2)$ となるよう定められたものである．□

例えば，図 5 はトークン型の S-Value を定義した例である．ノードはトークンの型であり，ノード間の線は包含関係を表す．上位にいくほど S-Value は小さくなり，下位にいくほど大きくなる．

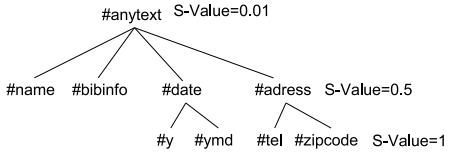

図 5 トークンの S-Value
Fig. 5 S-Values for Tokens

```
<ul>
  <li>N. Sawa, A. Mori. "Wraplet"</li>
  <li>T. Iida, N. Sawa, A. Mori. "WISH"</li>
</ul>
```

HTML ページ A

```
<ul>
  <li>N. Sawa and A. Mori. Wraplet.</li>
  <li>T. Iida, N. Sawa and A. Mori. WISH.</li>
  <li>Y. Mitsu and A. Mori. InfoSpace.</li>
</ul>
```

HTML ページ B

図 6 HTML ページの例
Fig. 6 HTML Page Examples

次に，DTD の Specificity Value を定義する．

定義 4.2. d を DTD としたとき，d の Specificity Value $S\text{-}Value(d)$ を次のように定義する．

$$S\text{-}Value(d) = \begin{cases} S\text{-}Value(d) & (\text{if } d \text{ is a token}) \\ \sum_i \frac{S\text{-}Value(d_i)}{m} & (\text{if } d \text{ is } d_1 \mid d_2 \mid ... \mid d_m) \\ \sum_i S\text{-}Value(d_i) & (\text{if } d \text{ is } d_1...d_m \text{ or } d_1*). \end{cases}$$

この定義から分かるように，$S\text{-}Value(d)$ は，DTD を構成するトークン型の種類とその構造によって決まる．

例. `#anytext` のトークン型を a, `#name` のトークン型を n とした時，(1)$a*$, (2)$(a|n)*$, (3)$anan$ の 3 つの DTD の S-Value を図 5 のトークンの S-Value を用いて求める事とする．(1) の S-Value は 0.01，(2) の S-Value は (0.01+0.5)/2 = 0.255，(3) の S-Value は 0.01+0.5+0.01+0.5=1.02 となる．したがって，ラッピングの具体度は，(3) が最も高く，(1) が最も低い．

5. HTML ラッパ生成支援手法

本章では，1 章で説明した考え方と 3, 4 章で説明した MDL, S-Value を用いて，Wraplet 式で記述された HTML ラッパを生成する手法を提案する．まず，5.1 節では，入力となる HTML のインスタンス集合と，トークン (ラッピングの結果，XML の要素となる最小単位) の分割が既知であると仮定した場合の，Wraplet 式生成手法の説明を行う．次に，5.2 節では，これらが既知でない場合でも，本手法が情報統合目的であるという性質を利用し，関係のある複数のページを入力として，HTML のインスタンス集合とトークンの同定を支援するための方法を提案する．

5.1 MDL と S-Value を用いた HTML ラッパの生成

例として，図 6 に示す簡単な論文リストの HTML ページ A が与えられたとする．このとき，タグとカンマで区切られた文字列 (`N. Sawa`, `A. Mori`, `WISH` 等) がトークンとして認識されていると仮定し，さらに，`<LI>`タグで区切られた各要素が，それぞれ独立した HTML インスタンスとして認識されていると仮定する．すなわち，提案手法に対する入力は，2 つの HTML インスタンスからなる集合となる．

このとき，Wraplet 式作成手順は以下の 3 ステップから構成される．

```
label1:/
   {label2:#li/
        [{name:_val(#name)},
          string:_val(#anytext)]
   }
```

図 7 DTD から求められる Wraplet 式
Fig. 7 Wraplet Expression Derived from a DTD

(ステップ 1) トークン型の割当て. まず，4 章で説明した，トークン型に関する S-Value を定義したものを用意しておく．ここに現れる各トークン型は，Wraplet のパターン部品に対応させ，パターンにマッチする文字列の包含関係によって，関係を決めておく．

次に，入力された HTML インスタンス集合に含まれる各トークンに対して，パターン部品にマッチするかどうかの判定を行う．もしマッチするならば，各トークンを，パターン部品に対応するトークン型に置き換える．このような置き換えのパターンは複数存在するので，それらを全て列挙する．一例として，`N. Sawa` や `A. Mori` などのトークンには，人の名前にマッチするパターン部品`#name` に対応するトークン型`<name/>`を割り当て，`Wraplet` や `WISH` などのトークンには，任意の文字列にマッチするパターン部品`#anytext` に対応するトークン型`<string/>`を割当てるとすると，HTML ページ A のインスタンス集合は，{`<name/><name/><string/>`, `<name/><name/><name/><string/>`} となる．

(ステップ 2) DTD 生成. 次に，ステップ 1 の結果に対して，XML データを対象とした DTD 生成手法を適用する．簡単化のため，以降では，各要素を下記のように 1 文字で表記する．`<name/>`に n, `<string/>`に s をそれぞれ割当てる事とする．その結果，DTD 生成手法の入力となるインスタンス集合は，$\{nns, nnns\}$ と表記できる．これらに対し，XTRACT[3] の手法を適用すると，DTD として下記が列挙される．

$$nns|nnns$$
$$n*s$$
$$nn(s|ns)$$

(ステップ 3) MDL コストと S-Value を用いた DTD および Wraplet 式の選択. ステップ 2 で出力された各 DTD について，MDL コストと S-Value を考慮したスコア付けを行う．MDL コストは，小さい方が DTD としてより適切であるという尺度であるため，具体的には MDL の逆数と S-Value の相乗平均をスコアとする．

$$Score(d, I) = \sqrt{\frac{1}{MDL(d, I)} \times S\text{-}Value(d)}$$

この $Score(d, I)$ が最も大きいような DTD を選択し，それに対応する Wraplet 式を，HTML ラッパの解とする．先程求めた HTML ページに対しては，DTD として $n*s$ が選択され，それに対応する Wraplet 式が解となる．

以上の手順により DTD が与えられると，Wraplet 式は次のように簡単に求まることに注意して欲しい．すなわち，DTD の構造をそのまま Wraplet 式の入れ子構造とし，トークン型を表す要素に合わせてパターン部品を入れる．$n*s$ に対応する Wraplet 式を図 7 に示す．

この例のように，出力される Wraplet 式に現れるラベルには，利用したパターン部品の名前，もしくは適当な名前 (label1) などが仮につけられる．ユーザは必要があればラベルを変更する．

5.2 インスタンス集合とトークンの同定

これまで，HTML のインスタンス集合と，それらに含まれるトークンが既に同定されていると仮定して話を進めてきた．本節では，これらの同定を支援する手法について提案する．ポイントは，本手法が情報源の統合利用のために利用されることを用いて，異なる HTML ページ同士の内容に重複があることを利用することである．特に，研究室と研究室の構成員の論文リストのような，片方の内容がもう一方の内容の部分集合になっている等，それぞれのコンテンツが表す集合間に重複がある時に，これらを同定するための手法を提案する．

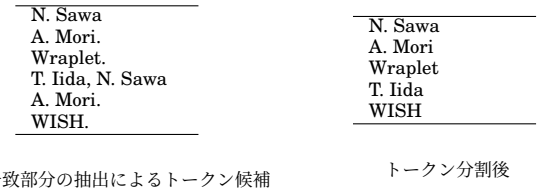

図 8 トークンの同定
Fig. 8 Identification of Tokens

具体的なアイデアは次の通りである．(1) 2 つの異なる HTML ページで重複する文字列をトークンの候補として選択する．さらに，これらの文字列に対して，パターン部品を順次適用することにより，より小さなトークンに分割する．(2) 重複部分を多く含む要素 (例えば ul) から最も外側の繰返し構造 (例えば最も外側の li 要素) を抽出し，ここに含まれる各繰返し要素をそれぞれ一つの HTML インスタンスと見なし，入力の HTML インスタンス集合を得る．

この問題に関しては，本稿では例を用いてアイデアのみ説明する．

例. ここでは，図 6 に示したような二つの論文リストを表す HTML を例とする．これらに含まれる論文集合には $A \subseteq B$ という包含関係があると言うことが分かっているとする．したがって，これらを統合的に管理するために，それぞれの HTML ページに対してラッピングを行いたい．

これらが与えられたとき，本手法では，まず，これら二つのページの最長一致部分文字列をトークン候補として切り出す．その結果が図 8 左である．次に，これらに対して，パターン部品を適用し，マッチした部分をより詳細なトークンとして順次切り出していく．T.Iida 等は`#name` にマッチするため，分割されて図 8 右のようになる．

次に，包含関係があるということから，それぞれの HTML ページは，何らかの要素の集合をエンコーディングしていることが分かる．そこで，これらの HTML ページより，繰返し構造の抽出をおこない，その繰返しの各要素の集合を，HTML インスタンスの集合として抽出する．上記例において，一致部分全体を含む最小の要素 (この場合は ul) の中の最も外側の繰返し構造は`<li>*`である．したがって，各 li 要素を HTML インスタンスとする． □

以上の例で示すように，本手法は，複数のページの内容間に重複関係が存在することを前提に，トークンとインスタンス集合の同定を行う．ポイントは，例えば包含関係はリレーショナルデータモデルにおける外部キー制約に相当している事などの理由によって，コンテンツの重複関係が，複数の情報源を統合するための手がかりとして利用しやすいことである．

次章では，そのような重複関係が存在するような実在の HTML ページを対象に予備実験を行った結果を示す．

6. 予備実験

予備実験として，実際の Web ページを用いて提案手法による Wraplet 式で記述された HTML ラッパの生成を行い，適切な Wraplet 式が選択できるかどうか検証する．

ラッピングの対象として，筑波大学森嶋研究室の教員の論文リストが掲載された Web ページを選んだ．手順は次の通りである．(1) まず，そのページと複数の学生論文リストの間に包含関係が存在する事を利用して，HTML インスタンス集合の抽出とトークンの同定を行った．そこでは，論文リストを含む HTML ページのう

```
<li>三森祐一郎，森嶋厚行
「データベース統合のための作業手順作成支援システムの開発」
(Development of a Support System for Constructing Database Integration Workflows)
<br/>
第 69 回情報処理学会全国大会講演論文集 (第 1 分冊), pp. 493-494, 東京, 2007 年 3 月.
</li>
<li>石川憲一，森嶋厚行，田島敬史
「大規模ドキュメント空間統合管理システムの提案」
(Proposal of an Integrated Document Space Management System)<br/>
日本データベース学会 Letters, Vol. 5, No. 2, pp. 89-92, 2006 年 9 月, 日本データベース学会.
</li>
<li>
澤 菜津美，森嶋 厚行，飯田 敏成，杉本 重雄，北川 博之
「コンテンツ一貫性制約を用いた Web サイト管理手法の提案」
(Proposal of a Web-Site Management Method Using Content Integrity Constraints )
<br/>
電子情報通信学会第 18 回データ工学ワークショップ (DEWS2007), 7 pages, 広島 元宇品, 2007 年 3 月.
</li>
<li>澤菜津美，森嶋厚行，杉本重雄，北川博之
「非定型 Web コンテンツ管理のための軽量ラッピング言語」
(A Lightweight Wrapping Language for the Management of
Non-Template-Based Web Contents)<br/>
情報処理学会研究報告 Vol.2007, No.65 (2007-DBS-143), pp.527-532.
電子情報通信学会技術研究報告 Vol.107, No.131, pp.527-532, 仙台, 2007 年 7 月.
</li>
```

図 9 実験で使用した論文リストの一部
Fig. 9 Part of the Publication List Used in the Experiment

ち，<LI>タグで区切られた各要素が，それぞれ独立した HTML インスタンスとして認識された．

(2) HTML インスタンスのうち，構造が異なる論文を 22 個 (図 9 に抜粋) を選択した．その理由は，構造が同じ論文は提案アルゴリズムの出力に影響を与えないからである．

(3) 次に，タグとカンマを境界としてトークン分割を行い，Wraplet のパターン部品を利用して，各 HTML インスタンス集合のトークン型の発見を行った．

(4) (3) で発見されたトークン型の組合わせが，ラッピング f_i を表す．今回の実験では，32 通りの組合せになった．この中では，f_{32} が具体性が最も高いもの，f_1 が具体性が最も低いものである．

これらの詳細を説明する．簡単化のため，以下では生成される DTD 中での各トークン型は，英文字 1 文字だけを用いて表現している．f_{32} は名前には#name(n と略記．以下同様．) を割当て，書誌情報 (Vol. 5, No. 2, pp. 89-92 など) には#bibinfo(b) を割当て，日付 (2007 年 3 月など) には#date(d) を割当て，論文誌名として,「電子情報通信学会技術研究報告」には#ieice(s),「情報処理学会研究報告」には#ipsj(h) を割当て，タイトルやその他の情報には#anytext(a) を割当てている．このように f_{32} は最もトークン型の種類が多く具体的なことから，S-Value が最も高くなる．f_{31} から f_2 は#name，#bibinfo，#date，#ieice，#ipsj に#anytext(a) を割当てた場合の組合せである．f_1 は全てのトークンに#anytext を割当てているため，最も S-Value は低くなる．

(5) 以上 32 個のトークン型の割当てに対し，DTD 生成を行い，各 DTD について MDL コストと S-Value を用いてスコア付けを行った．

6.1 実験結果

本実験において，X 軸を MDL コストの逆数，Y 軸を S-Value として $DTD_{i,j}$ のスコアの分布を調べた結果を図 11 に示す．図中における菱形が，自動生成された DTD の各スコアを表す．また，図 10 に，生成された 88 個の DTD の内，スコアの高い順にランキングした結果から上位 10 位と下位 10 位を示す．結果として，f_{25} によって生成される XML インスタンス集合に対する $DTD_{25,3}$ が最も高いスコアを持つため，これに対応するラッピング $f_{25,3}$ が，これらのうち最も適切なラッピングとして出力された．

6.2 考察

今回最も高いスコアを持つ $DTD_{25,3}$ は，名前 (n) の繰返し，タイトル (a) の繰返し，書誌情報 (b) またはその他の情報 (a) の繰返しの列を表す DTD となっており，DTD の簡潔性とラッピングの具体性の両方が高いものが選ばれたといえる．その他の上位の DTD を見ても，似たような構造を持った DTD が上位に選ばれている．また，実験で用いたデータに対して，人手で適切と考えるラッピングに対応する DTD を作成したが，本実験ではこれは 3 位にランクされた．$DTD_{1,1}$ は，DTD の簡潔性はランキング中で最も高いが，ラッピングの具体性が低いため，スコアが低くなっている．以上より，本実験においては提案するスコア付け手法が適切であったと考えられる．

rank	id	DTD	$\frac{1}{MDL}$	S-Value	$Score$
1	$DTD_{25,3}$	$n*a*(b\|a)*$	0.0022	0.7650	0.0410
2	$DTD_{17,1}$	$n*a*$	0.0032	0.5100	0.0406
3	$DTD_{29,2}$	$n*a*(b\|a\|d)*$	0.0019	0.8467	0.0405
4	$DTD_{25,6}$	$n*a*(b\|a)*a*$	0.0021	0.7750	0.0401
5	$DTD_{18,2}$	$n*(a\|s)*$	0.0021	0.7550	0.0400
6	$DTD_{19,2}$	$n*(a\|h)*$	0.0021	0.7550	0.0400
7	$DTD_{21,1}$	$n*(a\|d)*$	0.0021	0.7550	0.0400
8	$DTD_{25,9}$	$n*(a\|b)*$	0.0021	0.7550	0.0400
9	$DTD_{25,4}$	$n*a*(b*\|a)*a*$	0.0021	0.7750	0.0399
10	$DTD_{18,1}$	$n*(a*\|s)*$	0.0021	0.7550	0.0398
		(中略)			
79	$DTD_{14,1}$	$(a\|b\|s\|d)*$	0.0014	0.3775	0.0234
80	$DTD_{15,1}$	$(a\|h\|b\|d)*$	0.0014	0.3775	0.0234
81	$DTD_{10,2}$	$(a*\|(b\|s)*)*$	0.0021	0.2550	0.0232
82	$DTD_{16,1}$	$(a\|h\|b\|s\|d)*$	0.0013	0.4020	0.0228
83	$DTD_{10,5}$	$(a*\|b\|s\|a)*$	0.0020	0.2550	0.0228
84	$DTD_{11,3}$	$(a*\|h\|b\|a)*$	0.0020	0.2550	0.0224
85	$DTD_{11,1}$	$(a*\|h\|b*\|a)*$	0.0019	0.2550	0.0220
86	$DTD_{9,7}$	$(a*\|b\|a)*$	0.0022	0.1733	0.0195
87	$DTD_{9,5}$	$(a*\|b*\|a)*$	0.0020	0.1733	0.0188
88	$DTD_{1,1}$	$a*$	0.0050	0.0100	0.0070

図 10 スコア表 (一部のみ抜粋)
Fig. 10 Part of Score Table

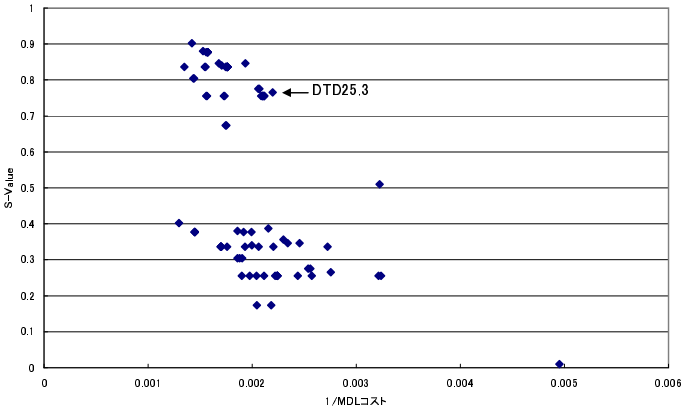

図 11 スコア分布
Fig. 11 Distribution of Scores

7. まとめと今後の課題

本稿では，特に HTML で記述された Web コンテンツをラッピングして XML データに変換するソフトウェアである HTML ラッパの構築支援手法について提案した．提案手法のポイントは，(1)HTML ラッパ構築の問題を，XML データからの DTD 生成の問題の一般化としてモデル化したこと，および，(2)DTD 選択のための基準として，DTD の適切性とラッピングの具体性を組み合わせたことである．今後の課題としては，より多くの Web ページに対する提案手法の有効性の検証や，提案手法の改良などが考えられる．

[謝辞]

ゼミなどでコメントいただきました筑波大学大学院図書館情報メディア研究科の阪口哲男准教授，永森光晴講師に感謝致します．本研究の一部は科学研究費補助金特定領域研究 (#19024006)，基盤研究 B(#19300081)，若手研究 B(#20800076)，科学技術振興機構戦略的創造研究推進事業 CREST「自律連合型基盤システムの構築」による．

[文献]

[1] Arvind Arasu, Hector Garcia-Molina. Extracting Structured Data from Web Pages. ACM SIGMOD International Conference on Management of Data, pp.337-348, 2003.

[2] Andrew M. Finch, Ezra Black, Young-Sook Hwang, Eiichiro Sumita: Using Lexical Dependency and Ontological Knowledge to Improve a Detailed Syntactic and Semantic Tagger of English. ACL 2006

[3] Minos Garofalakis, Aristides Gionis, Rajeev Rastogi, S. Seshadri and Kyuseok Shim. XTRACT: A System for Extracting Document Type Descriptors from XML Documents. Proc. of the 2000 ACM SIGMOD international conference on Management of data, pp. 165–176, 2000.

[4] 澤菜津美, 森嶋厚行, 杉本重雄, 北川博之, バックエンド DB を持たない Web コンテンツ管理のためのラッピング言語. 日本データベース学会 Letters, Vol.6, No.2, pp.69–72, 2007 年 9 月.

[5] Natsumi Sawa, Atsuyuki Morishima, Shigeo Sugimoto, and Hiroyuki Kitagawa. Wraplet: Wrapping Your Web Contents with a Lightweight Language. Proc. of IEEE The Third International Conference on Signal-Image Technology and Internet-based Systems (SITIS' 2007), 8 pages, December 2007.

[6] L. Liu, C. Pu, and W. Han. XWRAP: An XML-enabled wrapper construction system for web information sources. International Conference on Data Engineering (ICDE), pp. 611-621, 2000.

[7] Lawrence H. Reeve, Hyoil Han: Survey of semantic annotation platforms. SAC 2005: 1634-1638

[8] J. Rissanen. Modeling by shortest data description. Automatica, vol. 14, pp. 465-471, 1978.

[9] A. Mood, F. Graybill, D. Boes. Introduction to the theory of statistics. McGraw-Hill, 1974.

[10] Arnaud Sahuguet, Fabien Azavant: Building intelligent Web applications using lightweight wrappers. Data Knowl. Eng. 36(3): 283–316 (2001)

[11] Georg Gottlob, Christoph Koch: Logic-based Web Information Extraction. SIGMOD Record 33(2): 87–94 (2004)

[12] V. Crescenzi, G. Mecca: Automatic information extraction from large websites. J. ACM 51(5): 731–779 (2004)

澤 菜津美　Natsumi SAWA
2008 年　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修了.

森嶋 厚行　Atsuyuki MORISHIMA
筑波大学大学院図書館情報メディア研究科/知的コミュニティ基盤研究センター准教授．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
筑波大学大学院図書館情報メディア研究科/知的コミュニティ基盤研究センター教授．京都大学大学院工学研究科情報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, 日本データベース学会各正会員.

北川 博之　Hiroyuki KITAGAWA
筑波大学大学院システム情報工学研究科/計算科学研究センター教授．1980 年東京大学大学院理学系研究科修了．理学博士．ACM, IEEE-CS，情報処理学会，電子情報通信学会，日本ソフトウェア科学会，日本データベース学会各正会員.