ブラウザでの表示位置に着目したWebコンテンツの書換えによる応用処理支援

袖山 広輝 †　森嶋 厚行 ‡

筑波大学 図書館情報専門学群 †　筑波大学大学院 図書館情報メディア研究科 ‡

1. はじめに

今日，公開されたWebコンテンツの様々な応用処理が広く行われている．例えば，収集したWebページからのコーパスの作成[1]などがある．このような処理を行う際，Webコンテンツに対して前処理が必要な場合がある．例えば，コーパス作成ではそのコーパスの例文として適切でない部分（広告など）を発見し，その部分を除外する処理などが必要である．また，PC用のブラウザに単純に表示するだけでなく，PDA等の小さなディスプレイに表示する必要がある場合などでは，小さなディスプレイでのブラウジングを支援するために，ページを表示する範囲をどのようにグルーピングするかといったことが問題となる．

これまで，このような前処理に関しては様々な研究が行われてきた[2)~4)]．これらの研究では特定の種類の役割発見等に焦点が当てられている．しかし，実際の応用では，様々な種類の前処理が必要になることが考えられる．

そこで，本研究ではそのような前処理の実装を支援するためのライブラリを提案する．特に，本研究ではWebコンテンツの構成要素の役割発見と構成要素のグルーピングに焦点を当て，それを支援する事を考える．本ライブラリの特徴は，これらの前処理はHTMLソースをただ解析するだけでなく，ブラウザ上でのHTML要素の表示位置情報を利用すれば容易になるケースがあることに着目したことである．本稿で提案するライブラリを利用すれば，HTML要素に対し図1のようなページタイトルやメニューといったWebページの各部分の役割を発見するためのプログラムの容易な記述を支援できる．

関連研究. ブラウザ上の表示結果を利用したWebページ応用処理支援としては次のような既存研究が存在する．(1) 鶴田ら[2)]のWebページ上の主要領域部分を把握する手法．(2) Dengら[3)]，渡井ら[4)]のWebページを構成するHTML要素のグルーピングを行う手法．本研究は特定の前処理に焦点を当てたものではなく，Webコンテンツを用いた応用の前処理に必要となる，様々なWeb構成要素の役割発見やグルーピング処理を容易に行うことを目的としたライブラリの開発である．

2. 前処理支援のためのライブラリ

本章では，本稿で提案するライブラリについて説明する．まずライブラリの機能について説明し，次にライブラリの実現方法について説明する．

2.1 ライブラリの機能

ライブラリの機能には次の3つがある．(1) HTML要素のブラウザ上での表示位置の取得とそれらの比較に関する機能　(2) 役割を表すタグの追加や木の再構成のための機能　(3) その他前処理のプログラムの記述を支援する機能

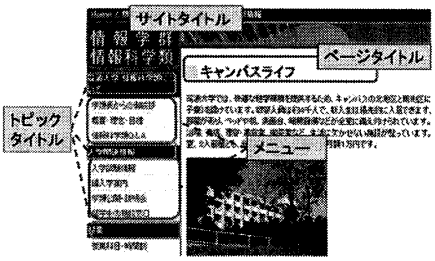

図1　組み込み関数で発見する役割[1]

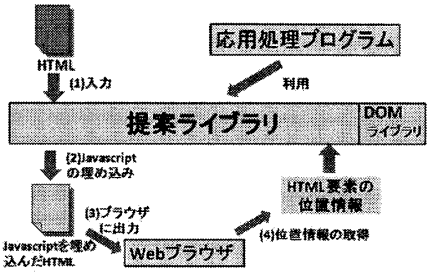

図2　ライブラリの構造

ライブラリの機能の一部を表1に示す．表中において，EはWebページpを構成するHTML要素の集合，$e, e_i, e_j \in E$はpに含まれるHTML要素，$edges$はエッジの集合である．

2.2 ライブラリ実現方法

ライブラリの実現方法について説明する．ライブラリの構造を図2に示す．まず，入力されたHTMLデータにJavascriptを埋め込む．このJavascriptを実行することにより，ブラウザ上での各HTML要素の位置情報を入手する．ここで取得する位置情報とは，ブラウザのウィンドウの左上端を原点した，HTML要素の上辺下辺のy座標と右辺左辺のx座標である．さらに，取得した位置情報を基にHTML要素の辺の長さ，面積を計算する．次に，ライブラリ本体がそこで得られた情報と，DOM等の他のライブラリから入手できる情報を用いた各種関数を提供する．

3. ライブラリを用いた応用処理支援

本ライブラリはWebコンテンツ応用処理のための前処理の実装を支援するためのものであるが，それらを用いた実装が既に用意された組込みの機能も用意する．本章では，役割発見とグルーピング機能の実装を用いてライブラリの利用例について説明する．特定の言語に依存せず説明するために，実装例は擬似コードで示す．

3.1 役割発見の実装例

役割発見の実装例として，ここではサイトタイトルの発見関数について説明する．サイトタイトルを発見するための関数 `siteTitleScore(p, e)` を図3に示す．関数

Supporting Web Content Applications by Page Rewriting Based on Display Positions on Browsers
Hiroki SODEYAMA† and Atsuyuki MORISHIMA‡ Sch. of Library and Information Science, Univ. of Tsukuba.† Grad. Sch. of Library, Information and Media Studies, Univ. of Tsukuba.‡

[1] http://www.coins.tsukuba.ac.jp/05_campuslife.html

表 1　ライブラリが提供する基本関数 (一部)

関数	説明
positionScore(e, p)	p を構成する全ての HTML 要素のうち下辺の座標で降順に並べた時の e の順位を返す
existsInHeadOf(e, p)	e が p の上部に位置するか否かに応じて値 ($MaxInt$〜0.1) を返す
existsInLeftSideOf(e, p)	e が p の中央よりも左に位置するか否かに応じて値 (1 or 0.1) を返す
existsInCenterOf(e, p)	e が p の横中央をまたがって位置しているか否かに応じて値 (1 or 0.1) を返す
existsInUpperLeftOf(e, p)	e が p の右最上部付近に位置するか否かに応じて値 (1 or 0.1) を返す
existsIn(e_i, e_j)	$e_i \neq e_j$ であり，e_j が e_i の内側に位置するか否かに応じて (1 or 0.1) を返す
hasOnlyOneAnchor(e)	e 内に含まれるリンク数が 1 か否かに応じて値 (1 or 0.1) を返す
tagScore(e, $table$)	e のタグ名とタグ名と値の対応表である $table$ に応じて，e のタグ名と対応した $table$ 上の値を返す
hasTheSameElementIn(e, E)	e と同じタグ名かつ同じ大きさの HTML 要素が E 内に存在するか否かに応じて値 (0 or 1) を返す
mergeSameElements(E)	E に存在する上下左右の座標がそれぞれすべて等しい HTML 要素同士を 1 つにまとめた HTML 要素の集合 E' を返す
createEdge(e_i, e_j)	e_i から e_j を結ぶエッジを返す
deleteTransitiveEdges(p, $edges$)	$edges$ に含まれるエッジのうち，他のエッジから推移で得られるエッジ (ex. $A \rightarrow B \wedge B \rightarrow C$ のエッジが存在する際の $A \rightarrow C$) を削除
treeByDisplayPositions(p)	表示上の入れ子関係から p を構成する全ての HTML 要素をグルーピングした結果を木として返す

siteTitleScore(p, e) は入力として，Web ページ p と p の中の HTML 要素 e を受け取り，e がサイトタイトルである可能性を示すスコアを返す．この関数は，表 1 に示したライブラリの関数 existsInHeadOf，existsInLeftSideOf，existsInCenterOf，existsInUpperLeftOf，tagScore，hasOnlyOneAnchor，hasTheSameElementIn の結果を乗算するだけの簡単な仕組みで実装されている．ユーザは，この関数の結果を利用して，ある閾値以上のスコアを返す HTML 要素がサイトタイトルである可能性が高いという情報を出力する．これはライブラリの別の関数を用いて，元の HTML ソースを書き換える方法か独立したファイルに出力する方法で行う．

3.2　表示位置を利用した HTML 要素のグルーピング機能の実装

次に，表示位置を利用した HTML 要素のグルーピング機能の実装について説明する．本機能では，Web ページを構成する HTML 要素を，表示位置情報に基づいてグルーピングし，その構造に基づいた木を作成する．表示位置に基づくグルーピングは一般に DOM 木の構造とは異なる場合があり，HTML に含まれる論理構造を把握するために役立つことがある．

```
1. function siteTitleScore(e, p) {
2.   E = p.allElements();
3.   table t={(h1,20), (h2, 15), ..., } ; //タグの重み付け
4.   float score = existsInHeadOf(e, p)*existsInLeftSideOf(e, p)
5.           *existInCenterOf(e, p)*existsInUpperLeftOf(e, p)
6.           *tagScore(e, t)*hasOnlyOneAnchor(e)
7.           *hasTheSameElementIn(e, E);
8.   return score;
9. }
```

図 3　サイトタイトルの発見の実装

```
1. function treeByDisplayPositions(p) {
2.     E = p.allElements();
3.     Set(edge) edges = {};  // エッジの集合
4.     E = mergeTheSameElements(E);
5.     for each e1 in E {
6.         for each e2 in E {
7.             if (existsIn(e1, e2) == 1)
8.                 edges.add(createEdge(e1, e2));
9.         }
10.    }
11.    deleteTransitiveEdges(p, edges);
12.    return new Tree(E, edges);
13. }
```

図 4　表示位置による HTML 要素のグルーピングの実装例

図 4 にその実装を示す．まず，入力された Web ページ p を構成する HTML 要素の集合 E を計算する (2 行目)．次に，表示位置が同じ HTML 要素同士を 1 ノードとして扱うために mergeTheSameElements を呼び出す (4 行目)．続いて，$e1 \in E$ と $e2 \in E$ において，表示範囲が入れ子になっている (existsIn($e1$, $e2$) == 1) が真) のとき，$e1$ から $e2$ を結ぶエッジを作成し，エッジの集合に加える (7〜8 行目)．最後に，他のエッジから推移で求まるエッジを削除し，以上の結果から作られた木を出力する (11〜12 行目)．

4. ま と め

本稿では，Web コンテンツに対する応用処理支援を目的とし，ブラウザ上での HTML 要素の表示位置情報を利用して，前処理の容易な実装を支援するためのライブラリを提案した．また，本ライブラリを利用した Web ページの各部分の役割発見や HTML 要素の表示位置からのグルーピングといった前処理実装の例を示した．

謝　辞

本研究の一部は科学研究費補助金若手研究 (B)(#20700076) による．

参 考 文 献

1) 関口洋一，山本和英．Web コーパスの提案．情報処理学会研究報告 情報学基礎研究会報告，2003(98)，p.123-130.
2) 鶴田雅信，増山繁．未知のサイトに含まれる Web ページからの主要部分抽出手法．言語処理学会第 14 回年次大会発表論文集，2008.
3) Deng. Cai, et al, VIPS: a vision-based page segmentation algorithm. Microsoft Technical Report MSR-TR-2003-79, 2003.
4) 渡井康行ほか．レンダリング情報に基づく Web ページの視覚的構造抽出．電子情報通信学会総合大会講演論文集 情報システム (2)，2005，p.219.