データベースシステム 119−71
（1999. 7. 23）

ユーザ視点に即した仮想 WWW ページの動的生成による閲覧支援

品川 徳秀 † 北川 博之 ‡

近年の WWW の普及、発展に伴い、非常に多くの情報が利用可能となったが、その反面、必要な情報の発見、利用が難しくなっている。一般に、情報検索を行なうためには検索と閲覧の両プロセスが重要である事が知られている。本稿では、ユーザの視点に基づいた閲覧支援手法を提案する。本手法は、WWW ページからその論理構造を抽出し、それを用いてユーザの興味が反映された仮想的な WWW ページを生成、提供する。これにより、ユーザの視点から WWW を閲覧す事ができ、興味に関連する部分を発見しやすくなる。また、本手法を現在の WWW 環境での自然な実装方法を示す。

Browsing the WWW with Dynamic Generation of Virtual Pages Based on the User's Interest

Norihide Shinagawa † Hiroyuki Kitagawa ‡

Due to the recent advance of the WWW technology, a huge amount of information is available. However, it is difficult to find and utilize relevant information. Generally, querying and browsing are both indispensable to access to the required information. In this paper, we propose a scheme to support the browsing process of the information search based on the user's interest. In our scheme, we extract the logical structure of each WWW page and generate the virtual WWW page reflecting each user's interest. Our approach makes it possible to browse the WWW pages from his/her viewpoint and to identify required information more easily. We also show how such a scheme can be realized in a non-intrusive way in the current WWW environment.

1 はじめに

近年の WWW の普及、発展に伴い、非常に多くの情報が利用可能となったが、その反面、必要な情報の発見、利用が難しくなっている。このような問題に対し、様々な研究がなされている [1] [2] [3]。

一般に、WWW を探索して必要なページを特定するためには、検索と閲覧の両プロセスが必要である。ユーザはまず、検索エンジン等を利用して閲覧の候補を絞り込む。続いて、適当な候補ページを選択、閲覧し、必要ならリンクを辿ったり、前に戻って別の候補ページを選択したりする。このような検索と閲覧のプロセスが繰り返される。

ユーザの興味は検索プロセスには問合せ条件として明示的に表現されるのに対し、閲覧プロセスでは明確化されて利用されていない。通常、ユーザがいかなる興味を持っていようとも各ページの表示にその視点は反映されず、常に同じ様式で提供されている。しかし、WWW の閲覧では多くの未知のページを扱わねばならず、中には記述量の多いページも含まれる。このような状況において、それらの構造や内容を理解し、必要な記述のなされている部分を特定する事は決して容易な事ではない。一般に WWW ブラウザが提供する文字列検索機能ではこれを十分に支援する事は不可能であり、また、常に同じ表示しか得られないという状況は改善されない。

本稿では、ユーザの興味を考慮した閲覧支援手法を提案する。本手法は、ユーザの視点をユーザプロファイルとして明確化し、WWW ページの閲覧時にユーザプロファイルに基づいて動的に生成されたビューページを表示する。本質的には、ビューページはユーザプロファイルとして与えられたユーザの興味と詳細度に基づいた要約である。その生成の際、HTML の文書構造ではなく、文章の構成を表した論理木を抽出、利用する。各ページに対して自動的にビューページが生成、提供される事で、ユーザは視点によって異なった WWW のビューを与えられる事になる。また、本稿では、現在の WWW 環境における本手法の実装方法も示す。ユーザは特別なブラウザを使う事なく、現在利用している WWW ブラウザで本手法の提供する環境を利用可能である。

以下が本稿の構成である。まず、2 節で本手法の概要を示す。次に、3 節で論理木を導入し、その抽出方法を述べる。4 節では、ユーザプロファイルに基づいたビューページの生成方法について説明する。また、5 節では現在の WWW 閲覧環境における本手法の実現方法を述べる。6 節では関連研究について言及する。最後に、7 節で本稿のまとめを行なう。

2 概要

本節では、本研究の提案する手法の概要を示す。図 1 に我々のアプローチを図示する。

本環境では、ユーザの視点はユーザプロファイルとして明確化される。ユーザプロファイルは、ユーザの興味を表現したキーワード群と表示の詳細度を制御する閾値から構成される。ユーザが WWW ページへアクセスすると、ユーザプロファイルに基づいたビューページが動

† 筑波大学 工学研究科
Doctoral Program in Engineering, University of Tsukuba
‡ 筑波大学 電子・情報工学系
Institute of Information Sciences and Electronics, University of Tsukuba

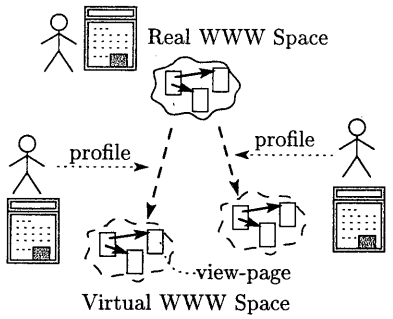

図 1: 本手法のアプローチ

的に生成され、ブラウザに表示される。このようなビューページの提供が暗黙的に行なわれる事で、ユーザは仮想的な WWW 空間を探索可能となる。各ビューページは本質的にはユーザの視点に基づいた要約となっており、その生成は次のように行なわれる。対象とする WWW ページは、HTML で記述されているものとする。

(1) 始めに、WWW ページ中から文書の論理構造を抽出し、本稿で扱う文書モデルである論理木としてモデル化する。論理木は HTML の一部のタグに着目して導出されるが、[2] で指摘されるように、一般に HTML のタグ階層が文書の論理的な階層性を反映しているとは限らない事に注意を要する。

(2) 論理木の各ノードとユーザプロファイル中で与えられるキーワード群との類似度を計算する。そして、類似度がユーザプロファイル中で与えられる閾値よりも低いノードをマークする。最後に、全ての上位ノードと内部ノードがマークされた部分木を取り除く事によってビューページを生成する。尚、その結果は正しい HTML 文書である必要がある。

(1) の詳細は 3 節で、(2) の詳細は 4 節で述べる。

3 論理木

本節では、HTML タグの分類を与え、それを用いた HTML 文書における論理木の導出方法について説明する。

3.1 HTML タグの分類

まず、HTML タグを以下の 4 グループに分類する。これは、3.2 小節で示す論理木の導出と、4 節で行なうプロファイルとの類似度の計算に用いられる。

a) 構造主体のタグ: このグループのタグは、主に見出しや段落、リスト等の文書構造を示すために用いられ、次のタグが該当する: H1, H2, H3, H4, H5, H6, P, BLOCKQUOTE, DIV, UL, OL, DL, TABLE。一般に、HTML 文書から論理構造を抽出するのは容易ではないが、このグループのタグはその手がかりを与える。この事から、これらを 3.2 節で説明する論理木の導出に利用する。

b) 表示主体のタグ: このグループのタグは、主に特殊な表示効果を得るために用いられ、次のタグが該当する: STRONG, EM, TT, I, U, B, BIG, SMALL, STRIKE, S, FONT, DL。これらのうち、幾つかは HTML 4.0 においては推奨されないタグであり、スタイルシートを用いるべきである事を注意しておく。これらは、文書の論理構造を示すものではなく、重要部分を強調するために用いられる事が多い。この事から、これらで囲まれた文書要素は論理木には影響を与えない。また、4.1 小節で部分文書の特徴ベクトルを求める際に、それらの文書要素にはタグ毎に定められた重みを与える。

c) メディア主体のタグ: このグループのタグは、画像やアプレットといったメディアオブジェクトを埋め込むために用いられ、次のタグが該当する: IMAGE, FORM, SCRIPT, APPLET, OBJECT, EMBED, MAP。これらで囲まれた文書要素は論理木の導出には影響を与えず、また、特徴ベクトルを求める際には無視される。

d) その他のタグ: 上記以外のタグは特別な効果を持たず、これらで囲まれた文書要素は通常の文字列として扱われる。即ち、論理木の導出に影響を与えず、特徴ベクトルを求める際にも特別な重みを持たない。

3.2 論理木の導出

本小節では HTML 文書からの論理木の導出法について述べる。構造主体のタグは、この導出において重要な手がかりを与える。例えば、H1, ..., H6 は各レベルの節の見出しを与える。[2] 等で提案された HTML 文書の論理構造の導出法においても、同様の手がかりが利用されている。

論理木は次の 8 種のノードからなる: $\ll doc \gg$, $\ll desc \gg$, $\ll leading \gg$, $\ll trailing \gg$, $\ll packed \gg$, $\ll block \gg$, $\ll heading \gg$, $\ll paragraph \gg$。これらは、図 2 の規則に従って与えられた HTML 文書から導出される。

論理木の各ノードは、その階層レベルを示す属性 L を持つ。図 2 の規則では、各ノードの L の導出方法が "*attr*" で付記されている。H1 と対応するノードは $L = 1$、H2 と対応するノードは $L = 2$ であり、H3, ..., H6 も同様である。また、導出された論理木の全てノードは、"*cond*" で付記された条件を満たしていなければならない。これより、上位ノードは下位ノードよりも L の値が小さくなる。†

例えば、$\ll desc \gg$ の導出規則において "*cond*" は "$\forall c_1, c_2$ *which are not* $\ll leading \gg$ $\{c_1.L = c_2.L\}$"、"*attr*" は "$L := c.L$ *where* c *is not* $\ll leading \gg$" である。ここで、"*cond*" から $\ll leading \gg$ 以外の全ての子ノードの L の値は等しくなければならず、それ自身の L の値は "*attr*" から、子ノードのそれで与えられる。

図 3 で与えられる HTML 文書から、この規則に従って導出された論理木を図 4 に示す。点線は L の値が同じノード群を表している。

ここで、$\ll leading \gg$ ノードは $\ll desc \gg$ の最左ノードとして含まれるのみであるが、これは続く $\ll trailing \gg$ ノードの導入部や概要を与える事が多い事を注意しておく。この事は次節で利用される。

†実際には、BLOCKQUOTE と対応するノードは例外である。

$\ll doc \gg$ → <BODY> $\ll desc \gg$ </BODY>

attr : $L := 0$

$\ll desc \gg$ → $\ll leading \gg \ll trailing \gg$ +
| $\ll trailing \gg$ +
| $\ll paragraph \gg$ +

cond : $\forall c_1, c_2$ *which are not* $\ll leading \gg$ $\{ c_1.L = c_2.L \}$

attr : $L := c.L$ *where* c *is not* $\ll leading \gg$

$\ll leading \gg$ → $\ll block \gg$

cond : $L < child.L$

attr : $L := parent.L$

$\ll trailing \gg$ → <DIV> $\ll trailing \gg$ </DIV>
| $\ll packed \gg$

attr : $L := child.L$

$\ll packed \gg$ → $\ll heading \gg \ll block \gg$

cond : $L < \ll block \gg .L$

attr : $L := \ll heading \gg .L$

$\ll block \gg$ → <DIV> $\ll block \gg$ </DIV>
| $\ll desc \gg$

attr : $L :=$ *child.L*

$\ll heading \gg$ → <H1>*text*</H1>
| <H2>*text*</H2>
| <H3>*text*</H3>
| <H4>*text*</H4>
| <H5>*text*</H5>
| <H6>*text*</H6>

attr : $L :=$ *the level of the tag*

$\ll paragraph \gg$ → <BLOCKQUOTE> $\ll block \gg$ </BLOCKQUOTE>
| <DIV> $\ll block \gg$ </DIV>
| <P>*text*</P>
| <UL>*text*</UL>
| <OL>*text*</OL>
| <DL>*text*</DL>
| <TABLE>*text*</TABLE>
| *text−without−structure*

attr : $L :=$ if tag is DIV then $\ll block \gg .L$ else 7

注 1: "*text*" は任意の文字列を、"text-without-structure" は他の規則で受理されない文字列を受理する。即ち、"*text−without−structure*" は BODY, DIV, BLOCKQUOTE 文書要素内の構造主体のタグを含まない文字列を受理する。

注 2: "*node.L*" は、*node* の L の値を表す。

図 2: 導出規則

```
<BODY>
    <H1> 1. 章の見出し </H1>
    <P> 章の導入 </P>
    <H2> 1.1. 節の見出し </H2>
    <P> 本文 1.1.1 </P>
    <P> 本文 1.1.2 <B> 重要語 </B> </P>
    <H2> 1.2. 節の見出し </H2>
    <P> 本文 2 </P>
    <H1> 2. 章の見出し </H1>
    <H2> 2.0.1 小節の見出し </H2>
    <TABLE> 表 </TABLE>
    <P> 表の説明 </P>
</BODY>
```

図 3: サンプル HTML 文書

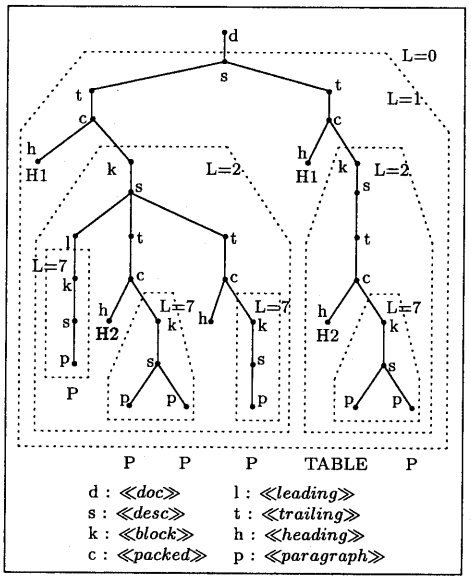

図 4: 抽出された論理木

4 ビューページの生成

本節では、論理木に基づいたビューページの構築法について述べる。本手法ではベクトル空間モデルを用いるが [4] [5]、3.1 節で導入した構造主体のタグと表示主体のタグを考慮する。

始めに、論理木の各ノードの特徴ベクトルを求める。次に、各ノードとユーザプロファイルとの類似度を算出し、それがユーザプロファイル中に与えられる閾値より低いノードをマークする。最後に、全ての上位ノードと内部ノードがマークされている部分木を取り除く事でビューページを構築する。

4.1 特徴ベクトルの導出

4.1.1 末端ノード

まず、末端ノードの初期特徴ベクトルを tf-idf 法に従って求める。idf 因子は、全末端ノードの集合を対象文書集合とみなして計算する。即ち、その文書内での語の分布に従った重みを与える事になる。

次に、表示主体のタグが重要部分を示すのに使われる事が多い事に着目する。これらのタグで囲まれた語について、表 1 でタグ毎に決められた重み因子をかける事によって重み付けを変更する。該当するタグが複数ある語については、最大の重み因子を用いる。

このような末端ノード n の特徴ベクトルは次で定式化される。ここで、$\beta_{n,t}$ は n に含まれ t が出現する文書

要素の重み因子のうち、最大のものである。

$$v_n = (tf_t \cdot idf_{n,t} \cdot \beta_{n,t})_{t \in \mathcal{T}} \quad (1)$$

タグ	$\beta_{n,t}$	意味
STRONG	5	強い強調
EM	3	強調
BIG	3	大きな文字
U	2	下線
B	2	ボールド
I	2	イタリック
DT	2	DL での定義語

表 1: 表示主体のタグに与えられた重み因子 β_n

4.1.2 非末端ノード

非末端ノードの特徴ベクトルは末端ノードの特徴ベクトルの集約として与える。一般に、タイトルや見出し、概要や導入等は本文の重要語を多く含む [5] [6]。論理木において、見出しは *≪heading≫* ノードに対応する。また、概要や導入は 3.2 小節で言及したように *≪leading≫* として現れる事が多い。

この事を考慮し、非末端ノード n の特徴ベクトルは子ノードの特徴ベクトルの荷重和として次式で与える。

$$v_n = \sharp C \cdot \frac{\Sigma_c \alpha_c \cdot v_c}{\Sigma_c \alpha_c} \quad (2)$$

ここで、c は n の子ノード、$\sharp C$ は子ノードの総数である。係数 α_c は表 2 で与える。

ノード	α_c	ノード	α_c
≪heading≫ (H1)	15	*≪desc≫*	1
≪heading≫ (H2)	11	*≪leading≫*	5
≪heading≫ (H3)	8	*≪trailing≫*	1
≪heading≫ (H4)	6	*≪packed≫*	1
≪heading≫ (H5)	4	*≪block≫*	1
≪heading≫ (H6)	3	*≪paragraph≫*	1

表 2: 子ノードの重み係数 α_c

4.2 ビューページの生成

ビューページを生成するため、まず、ユーザプロファイルに適合する部分を決定する。まず、論理木の各ノードはその特徴ベクトルと、ユーザプロファイルとして与えられたキーワード群の特徴ベクトルとの類似度が計算される。各ノードと対応するテキストの長さは様々であり、これを考慮して類似度の算出には典型的なコサイン測度でなく C-pivot 測度を用いる (付録を参照)。次に、全ての上位ノードと内部ノードの類似度が、ユーザプロファイルで与えられた閾値より低い部分木を取り除く。これによって得られる結果は正しい HTML 文書でなければならない事に注意する。

ビューページの提供によって、文書構造の把握を妨げるべきではない。特に見出しはその重要な手がかりであるから、元のページに含まれる全ての見出しをビューページに含めるものとする。

また、除去された部分を示す目印が与えられるべきと考えられ、更に、単純な除去をしてしまうと期待されない結果になる事がある。例えば、"text <P> paragraph </P> text" から P 文書要素を除去した結果、"text text" となってしまい、前後の二つの "text" が連結されてしまう。これらから、実際には除去せずに "<DIV> (snip) </DIV>" で差し替える事にする。

この手続きは以下で与えられる。これは任意の正しい HTML 文書に対して、正しい HTML 文書を生成する。

1. 各ノードの類似度を計算する。
2. 類似度が閾値より低いノードをマークする。但し、*≪doc≫*、*≪heading≫* ノードは除く。これは、BODY 文書要素は必須である事と、H1, ..., H6 文書要素は重要であるためである。
3. 上位ノードと内部ノードが全てマークされた極大な部分木を特定する。
4. 特定された部分木に対応する HTML 文書要素の列を "<DIV> (snip) </DIV>" で置き換える。
5. ビューページを出力する。

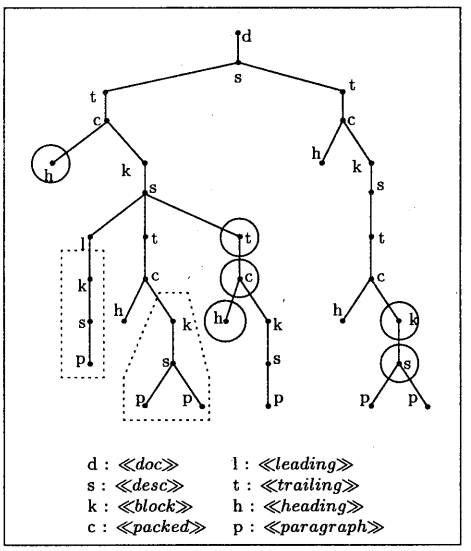

図 5: ビューページ生成例

```
<BODY>
   <H1> 1. 章の見出し </H1>
   <DIV> (snip) </DIV>
   <H2> 1.1. 節の見出し </H2>
   <DIV> (snip) </DIV>
   <H2> 1.2. 節の見出し </H2>
   <P> 本文 2 </P>
   <H1> 2. 章の見出し </H1>
   <H2> 2.0.1 小節の見出し </H2>
   <TABLE> 表 </TABLE>
   <P> 表の説明 </P>
</BODY>
```

図 6: ビューページの HTML 文書

図 5 にビューページの生成を図示する。円は閾値以上の類似度を持つノードを示しており、また、点線で囲まれた部分木は置き換えの対象を示している。これから生成された HTML 文書は図 6 である。

5 プロトタイプシステムの概要

我々は本手法をプロトタイプシステムとして実装を行なった。その際の基本要請として、可能な限りユーザの選択肢を狭めず、現在利用されている WWW 環境をそのまま利用できる事があった。本システムは次の三つのモジュールから構成されており、図 7 のアーキテクチャによってこの要請を実現している。実線は要求やメソッド起動の、点線は応答や返値の流れを示している。

- コントローラ
- ブラウザ
- 閲覧支援エンジン

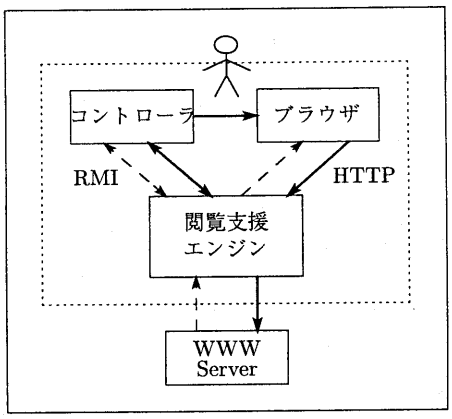

図 7: システムアーキテクチャ

ユーザは Netscape Communicator や Internet Explorer 等の一般的に利用されている WWW ブラウザで本環境を利用可能である。**コントローラ**はその WWW ブラウザ上の Java アプレットとして実行される。ユーザプロファイルの入力等を担当し、付加機能として閲覧中のページの論理木と各ノードの類似度等の視覚的な表示機能も併せ持つ。また、WWW ブラウザのウィンドウを自動起動し、**ブラウザ**とする。ユーザは各ページの閲覧を行なうためにこのブラウザのウィンドウを利用する。その表示は、ユーザプロファイルの変更によってビューページが変化すると、コントローラによって自動更新される。一方、**閲覧支援エンジン**は独立した Java アプリケーションであり、ビューページの生成、提供する機能を持つ。これは、WWW ブラウザからはマルチユーザ対応の WWW プロキシサーバとして利用される。また、ビューページの生成は必要に応じて抑制できるようになっている。

利用画面例を図 8 に示す。左奥のウィンドウがコントローラであり、右手前のウィンドウがブラウザである。これは、ユーザプロファイルを $\{extended, link, group, hub\}$ として XLL の仕様書を表示した結果である。

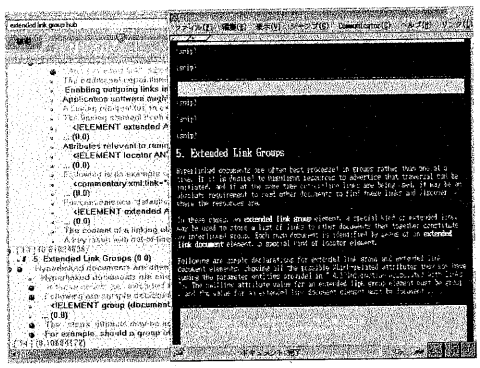

図 8: 利用画面例

これらは次のように連携動作する。まず、閲覧支援エンジンはあらかじめ実行されており、コントローラアプレットがそのホスト上に置かれている必要がある。この時、ユーザがコントローラアプレットを実行すると、ブラウザが起動され、コントローラが閲覧支援エンジンへ登録される。これにより、接続が確立する。起動が完了した後、ユーザはユーザプロファイルを設定する。これは閲覧支援エンジンに通知され、閲覧支援エンジンはこのユーザに対してビューページの提供する準備ができた事になる。

次に、ユーザはブラウザにおいて通常の WWW 閲覧時と同様の操作を行なう。この HTTP 要求は閲覧支援エンジンで処理され、登録されたユーザプロファイルに基づいたビューページが提供される。この時、リンクを辿る等の操作も通常通り行なう事ができ、これに対しても同様にビューページが提供される。この表示ページの変更は、閲覧支援エンジンで HTTP 要求を処理した後にコントローラへ通知され、論理木の表示が更新される。

また、ユーザは必要に応じてコントローラでユーザプロファイルの変更を行なう。特に、閾値の調整は論理木の視覚的表示と照らし合わせながらインタラクティブに行なわれる。このユーザプロファイルの変更内容は閲覧支援エンジンへ通知、更新される。続いて、コントローラはブラウザに対してリロードを強制する事によって HTTP 要求を発行させるが、これに対して閲覧支援エンジンは新しいユーザプロファイルに基づいたビューページを提供する。

このようにしてコントローラとブラウザの表示内容は自動的に同期が取られ、ビューページ間の移動に際しても特別意識する事なく本環境を利用できる。また、必要に応じてビューページの生成を抑制する事で、検索エンジンでの検索結果はそのまま得て、各ページの閲覧にのみビューページを利用する、といった使い方も可能である。

6 関連研究

本手法では、ビューページはユーザプロファイルに基づいたHTML文書の要約として生成している。自動要約生成に関する研究は多く、[4] [7] 等がある。しかし、それらのほとんどはユーザの興味対象を考慮しない。Tombrosは検索条件を考慮した自動要約生成手法を提案した [8]。しかし、それは検索プロセスの視点であり、検索結果からの選択を支援する目的での利用を想定しており、また、そこではHTML文書としての正しさは考慮されていない。本研究は閲覧プロセスの支援を主目的としている。

閲覧プロセスでの要約の利用では、ページの取得時に動的かつ暗黙的にそれが生成されるべきである。三池らは検索結果の個々の文書に対して自動的に要約を生成するIRシステムを開発した [9]。しかし、その要約にはユーザの視点は考慮されていない。

他の閲覧プロセスの支援に関する研究として、リンクのナビゲーション支援がある。WebWatcher [3] はその一つであり、ユーザのナビゲーションパターンからユーザの興味を学習し、ページ中のどのリンクを辿るべきかを示唆するシステムである。このような方法を本手法と組み合わせる事は有効であると考えられる。

また、本手法は文書の部分的な類似度を測るという点で部分文書検索 [5] [10] [11] と関連するが、部分文書検索では閲覧プロセスでの利用には明示的に言及してはいない。また、それらでは主に文や段落、固定長のブロック等の単純な構造を元に検索を行なうが、本手法では文書の論理構造を考慮している。

7 まとめ

本稿では、ユーザプロファイルに基づいたHTMLページを動的に提供する事によるWWW閲覧環境の支援手法を提案した。本手法は、文書中のユーザの興味に関連する部分を特定し、WWWページの内容を把握する事を容易にする。これを行なうため、本稿ではHTML文書から論理木を抽出し、ユーザプロファイルに基づいた要約をビューページとして提供する。また、我々はプロトタイプシステムを構築し、本手法が現在のWWW閲覧環境で自然に実装可能である事を示した。

課題として、本手法の有効性の実験評価、詳細な性能評価、プロトタイプシステムの改良等が挙げられる。また、本手法のXML文書への適用も重要な課題である。

参考文献

[1] P. Atzeni, A. Mendelzon and G. Mecca (eds.). *The World Wide Web and Databases*, Lecture Notes in Computer Science 1590, Springer-Verlag, 1998.

[2] N. Ashish and C. A. Knoblock. Wrapper Generation for Semi-Structured Internet Sources, *ACM SIGMOD Records*, Vol.26, No.4, pp.8-15, December 1997.

[3] R. Armstrong, D. Freitag, T. Joachims and T. Mitchell. WebWatcher: A Learning Apprentice for the World Wide Web, *AAAI Spring Symposium on Information Gathering from Heterogeneous, Distributed Environments*, Stanford, 1995.

[4] G. Salton. *Automatic Text Processing: The Transformation, Analysis, and Retrieval of Information by Computer*, Addison-Wesley, Reading, MA, 1989. z

[5] G. Salton, J. Allan and C. Buckley, Approaches to Passage Retrieval in Full Text Information Systems, *Proc. of ACM SIGIR'93*, pp.49-58, Pittsburgh, 1993.

[6] R. Willkinson. Effective Retrieval of Structured Documents. *Proc. of ACM SIGIR'94*, pp.311-317, Dublin, 1994.

[7] C. D. Paris. Constructing Literature Abstracts by Computer: Techniques and Prospects, *Information Processing and Management*, Vol.26, No.1, pp.171-186.

[8] A. Tombros and M. Sanderson. Advantage of Query Biased Summarization in Information Retrieval, *Proc. of ACM SIGIR'98*, pp.2-10, Melbourne, 1998.

[9] S. Miike, E. Itoh, K. Ono and K. Sumita. A Full-Text Retrieval System with a Dynamic Abstract Generation Function, *Proc. of ACM SIGIR'94*, pp.152-161, Dublin, 1994.

[10] M. Kaszkiel and J. Zobel. Passage Retrieval Revisited, *Proc. of ACM SIGIR'97*, pp.21-29, Philadelphia, 1997.

[11] J. Zobel, A. Moffat, R. Wilkinson and R. Sacks-Davis. Efficient Retrieval of Partial Documents, *Information Processing and Management*, Vol.31, No.3, pp.361-377, 1995.

[12] D. K. Harman. Overview of the first TREC Text Retrieval Conference, *Proc. of TREC-1*, pp.1-20, Washington, November 1992. National Institute of Standards Social Publication 500-207.

[13] A. Singhal, C. Buckley and M. Mitra. Pivoted Document Length Normalization, *Proc. of ACM SIGIR'96*, pp.21-29, Zurich, August 1996.

付録: C-Pivot 測度

tf·idf 法を用いたベクトル空間モデルでは、文書をベクトル表現し、そのコサイン測度により文書間の類似度が測られる。しかし、単純なコサイン測度では小さな文書が比較的高い類似度を示してしまう傾向がある事が知られている [12]。この問題に対し、Singhal らは文書長を考慮した **C-pivot** 測度を提案した [13]。

文書数 N、語彙集合 $\mathcal{T}$ の文書集合 $\mathcal{D}$ について、文書 d と問合せ v_q の類似度は次で与える。

$$
\begin{aligned}
sim(v_d, v_q) &= V_d \circ V_q \\
V_d &= \frac{v_d}{(1-S)\cdot U_d + S \cdot U_d} \\
V_q &= \frac{v_q}{\|v_q\|} \\
v_x &= (tf_{x,t} \cdot idf_t)_{t \in \mathcal{T}} \quad (x = d, q) \\
idf_t &= \log(N/n_t + 1),
\end{aligned}
$$

ここで、$d \in \mathcal{D}$ で、U_d は d 中の語の種類の数、S は $[0,1]$ の傾斜定数、$tf_{x,t}$ は語の出現回数、n_t は t を含む $\mathcal{D}$ 中の文書数である。尚、S は $\mathcal{D}$ に対して経験的に決定する必要がある。