繰返し構造に基づいたWebページの構造化

南野朋之[†] 齋藤豪[††] 奥村学[†††]

World Wide Webは，急速に成長している巨大な情報源である．しかしながらWeb上の情報は，レイアウト記述言語で記述された，人が目で見て理解するための情報であるため，計算機で直接扱う際に困難がともなう．そこで本研究では，このようなWeb上の情報を人間が理解する構造に近い形で計算機が扱うことができるようにするために，HTML文書中に含まれる要素の繰返し構造に注目し，自動的な情報のセグメンテーション，構造化を行うことを目的とする．本論文では，まず完全一致ベースの繰返し構造によってWebページを構造化し，その後，構造化できなかった部分を類似度ベースの繰返し構造によって構造化する2段階手法を提案する．

Structuring Web Pages Based on Repetition of Elements

Tomoyuki Nanno,[†] Suguru Saito[††] and Manabu Okumura[†††]

The World Wide Web is a vast source of information accessible to computers, but most of its information is not easy to process by computer applications because Web pages are described in layout description languages, such as HTML. In this paper, we propose a method of automatically segmenting and structuring Web pages based on repetition of elements. Our system structures Web pages with a two-stage approach: first by detecting repetition structures based on "exact match" and then by detecting repetition structures based on "similarity."

1. はじめに

World Wide Webは，急速に成長している巨大な情報源である．しかしながらWeb上の情報は，レイアウト記述言語で記述された，人が目で見て理解するための情報であるため，計算機で直接扱う際には困難がともなう．なぜなら，HTML文書そのものには，情報の意味や構造を示すような記述が含まれていないためである．このような状況に対して，Amazon Web Service[1]，Google APIs[2]に代表されるWeb ServiceやRSS（RDF Site Summary）[3]のように，計算機がWeb上の情報を直接扱うことができる手段を提供するための様々な技術が登場している．

また，Semantic Web[4),5)]は，計算機が直接扱うことのできる情報を，情報に対する情報（メタデータ）として付加するための枠組みである．このようなメタデータが多くの情報に付加されれば，計算機はより知的な処理を行うことができるが，人手によるメタデータの付与は膨大なコストがかかるため，計算機による自動的なメタデータの付加が望まれている．

Webページの構造は，このようなメタデータの一種であり，計算機でWebページを処理する際に必須な情報の1つである．たとえば，「どこからどこまでが情報のひとまとまりか」を認識することは，Webページを扱う様々なアプリケーションにとって非常に重要である．また，構造に関する知識を利用できれば，構造を反映した順序でWebページを読み上げる音声ブラウザや，PDAや携帯電話など，ディスプレイの小さい端末向けにWebページを分割するシステムなどでの利用も考えられる．そこで，本研究ではこのようなメタデータの一種であると考えられるWebページの構造に注目し，Webページの自動的なセグメンテーション，構造化を目的とする．

HTML文書にはDOM（Document Object Model）構造[6]が含まれるが，Henzingerら[7]が述べているように，これは表示を制御するための構造であり，意図的に意味や構造を表現しているわけではない．それで

† 東京工業大学大学院総合理工学研究科
Interdisciplinary Graduate School of Science and Engineering, Tokyo Institute of Technology
†† 東京工業大学大学院情報理工学研究科計算工学専攻
Department of Computer Science, Graduate School of Information Science and Engineering, Tokyo Institute of Technology
††† 東京工業大学精密工学研究所
Precision and Intelligence Laboratory, Tokyo Institute of Technology

は，なぜ人は Web ページを見ると，このようなページのセグメントや構造を理解できるのか？ 我々は，Web ページが表示される際の “統一性” が Web ページの構造を理解する際の手がかりになっていると考える．たとえば，もし同じフォント属性によって記述される一連の項目があれば，それらは同一種の情報であると理解されるであろう．よって，我々はこれら表示上の統一性を獲得するために，繰返し構造を再帰的に検出することで，Web ページを構造化することを試みる．本論文では，まず完全一致ベースの繰返し構造によって Web ページを構造化し，その後，構造化できなかった部分を類似度ベースの繰返し構造によって構造化する 2 段階手法を提案する．

2. 関 連 研 究

Lim ら[8] は，HTML 構文木を Content Tree（CT）と呼ぶコンテンツ間の階層関係をとらえた意味構造木に変換する手法を提案している．彼らはまず，HTML 文書を Semistructured Data Tree（SDT）と呼ぶ内部構造に変換し，この SDT に対して，approximation と migration という 2 つの処理を適用することで CT へと変換する．migration の処理では，まず，heading タグ（h1..h6）を手がかりに Web ページをトップダウンに，section という単位に分割する．そして，この section 中に含まれるタグ以外の要素を含む部分木を cluster と呼び，これに対して，ヒューリスティックを使用し cluster の統合および活性処理をボトムアップに適用することで，ページ全体の構造化を行っている．

しかしながら彼らの提案する手法は，section を決定するために heading タグを使用している点や，統合処理において使用されるヒューリスティック（ex. h2 タグによって特徴づけられる cluster が h1 タグによって特徴づけられる cluster に統合される）を使用している点で，非常に適用範囲の狭いものになってしまっている．なぜなら，多くの Web ページにおいて，heading タグを用いている（さらには，正しい用法で用いている）ページは残念なことに少数であると考えられるからである．また，heading タグ以外によって示される section の境界を扱うことができないといった問題もある．これに対し，我々の提案する手法は，特定の HTML タグに依存せず，同じパターンが連続する部分をボトムアップにグルーピングする手法である．

また Lim らの手法では，活性処理において同じ親ノードに属する 2 つの cluster は，無条件で 1 つのクラスタとして扱われてしまう．我々の提案する手法では，同じ親ノードに含まれるクラスタでもクラスタ間の類似性を考慮し，類似している場合のみにグルーピングを行う．

また，Web ページの構造に関する研究は wrapper 生成の分野で多くの研究が行われている．Laender ら[9] による，Web ページからのデータ抽出に関するサーベイ論文では，wrapper 生成ツールの分類を提案し，様々な視点から質的評価を行っている．wrapper は，構造化されていない Web ページから，構造化されたデータを抽出するプログラムである．本手法と wrapper 生成に関する手法との最も大きな違いは，wrapper は Web ページ中に存在する複数のレコードを認識し，各レコードから興味のあるデータを構造化されたデータとして抽出するのに対し，本論文で提案する手法は，Web ページ自体を構造化する手法であるという点である．よって多くの wrapper では，何を抽出すべきかを何らかの形でシステムに与えなければならず，また与えられた情報を使用して Web ページの構造を決定するのに対し，我々の提案手法はその必要がないため，単純な比較は困難である．そこで以下では，ユーザとのインタラクションや訓練事例を必要としない RoadRunner[10] や Embley ら[11] の研究との比較を行う．

RoadRunner[10] は，同じスクリプトで生成される Web ページの集合を比較し，ミスマッチを探すことで，レコードの単位を決定することができるツールである．このツールでは，レコードが入れ子構造を持つ場合や，ある部分にだけ含まれる例外なども扱うことが可能である．しかしながら，ミスマッチを元にレコード境界を決定するため，ユーザは同じスクリプトで生成された別のページを用意しなければならない．また，構造化される部分はページによって異なる動的な部分のみで，各ページで共通な部分などを構造化することができない．

また，Embley ら[11] が提案する手法は，Embley ら[12] によって示されたレコード境界発見手法を使用した wrapper 生成手法である．レコード境界の発見には，5 つのヒューリスティックを使用しているが，そのうちの 1 つは，人手によるオントロジの構築が不可欠である．また，構造化の対象となるのは，HTML Tag Tree 中の最も直下の子供の数の多いノードを root ノードとする部分木のみである．この制約により，抽出の対象となる一連のレコードがページ中に複数あるケースを扱うことができない．また彼らの手法では，個々のレコード中にさらなる構造が存在する場合，トップレベルの境界しか発見することができない．

これに対し，我々の手法は，レコード境界は，3.1 節

で述べる繰返し構造を検出することでページ中のあらゆる部分を構造化することが可能である．また，4.2 節で述べる繰返し構造の再帰的な検出や，4.3 節で述べる類似度ベースの手法を用いることで，入れ子構造や例外を扱うことも可能である．さらに我々の提案する手法は，そのページ内に含まれる情報だけしか使用しない．

table 理解システム[13),14)] は，Web ページ中の table から構造化された情報を抽出するシステムである．我々は，テーブルなど Web ページの一部分を対象にするのではなく，Web ページ全体を構造化することを試みる．また，上述したシステムでは，規則を生成するための素性を獲得する際や，テーブルの要素間の類似度を計算する際に言語情報を用いる．言語情報は非常に有効な手がかりとなりうるが，我々のシステムでは言語情報をいっさい用いないことにした．これは，あらゆる言語で記述された Web ページに適用できるシステムを構築するためである．

Chen ら[15)] は，小さい画面のデバイスで Web ページを閲覧するための手法を提案した．Web ページを分割するために，彼らは Yang ら[16)] によって提案された手法を用いて Web ページを構造化している．Yang ら[16)] は，Web ページを構成するオブジェクトの見た目の類似度に基づいてページ全体を構造化する手法を提案している．しかしながら，彼らは後述する「セパレータなしの繰返し構造」しか扱っていない．また，繰返し構造に複数の候補があった場合，ヒューリスティックに基づいて選択を行っている．それに対し我々は，後述する 2 種類の繰返し構造を考慮し，あらゆる可能性のある繰返し構造を検出し，全体の最適解を求めることによって Web ページを構造化する手法を提案する．

Yu ら[17)] は，Web ページのセグメンテーションを使用した効率的な情報収集に関する手法を提案している．彼らの手法は，トップダウンに荒いセグメンテーションを行う手法である．それに対し我々の手法は，最も細かい構造からボトムアップに全体の構造を組み立てる手法である．

3. 提案手法の概要

3.1 繰返し構造

図 1 に Yahoo!Japan のスクリーンショットの一部を示す．人はこれを見ると，“芸術と人文” と “ビジネスと経済” は同一種の情報であり，また “写真” と “建築” も同一種の情報であるということを瞬時に理解することができる．これは，Web ページの著者がそう

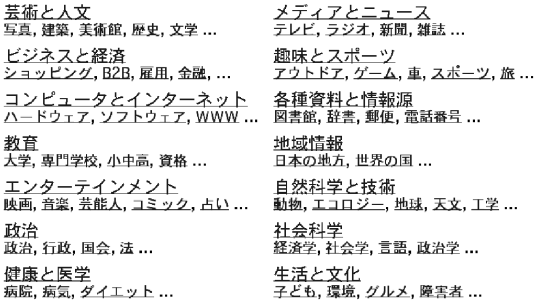

図 1 Yahoo!Japan のスクリーンショット
Fig. 1 Screenshot of a Yahoo!Japan page.

表 1 テーブルの例
Table 1 Example of table.

今日の天気	晴れのち雨
	降水確率 80%
明日の天気	晴れ時々くもり
	降水確率 50%

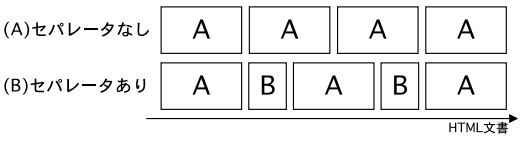

図 2 2 種類の繰返し構造
Fig. 2 Two types of repetition structures.

理解されやすいように，同一種の情報を同じフォントサイズ，フォントカラーで記述しているためである．

次に，**表 1** にテーブルの例を考える．人は，これを見ると瞬時に 2 行ずつが情報のまとまりであることを理解することができる．もちろん言語的な情報や意味に関する情報を使えばこのようなセグメンテーションは可能であるが，言語情報が使用できなくても “降水確率” の左のセルの空白が規則正しく出現していることを検出できれば人間が理解する構造と同じ構造を得ることができる．

本研究では，このような Web ページのセグメントを，HTML 文書中に含まれる繰返し構造を再帰的に検出することによって獲得する手法を提案する．なお，本研究で考慮する繰返し構造のタイプは**図 2** に示す 2 つのタイプである．図 2 (A) をセパレータなしの繰返し構造と呼び，図 2 (B) をセパレータありの繰返し構造と呼ぶ．セパレータありの繰返し構造は**図 3** のような構造を得る際に使用される．繰返し構造の厳密な定義は，4.2.1 項で述べる．

3.2 構 造 化

前節で述べたように，繰返し構造を検出することで

メール - 住所録 - カレンダー - 挨拶状
(A) (B) (A) (B) (A) (B) (A)

図 3 セパレータありの繰返し構造の例
Fig. 3 Example of a repetition structure with separators.

芸術と人文 写真, 建築, 美術館, 歴史, 文学 ...	メディアとニュース テレビ, ラジオ, 新聞, 雑誌 ...
ビジネスと経済 ショッピング, B2B, 雇用, 金融, ...	趣味とスポーツ アウトドア, ゲーム, 車, スポーツ, 旅 ...
コンピュータとインターネット ハードウェア, ソフトウェア, WWW ...	各種資料と情報源 図書館, 辞書, 郵便, 電話番号 ...
教育 大学, 専門学校, 小中高, 資格 ...	地域情報 日本の地方, 世界の国 ...
エンターテインメント 映画, 音楽, 芸能人, コミック, 占い ...	自然科学と技術 動物, エコロジー, 地球, 天文, 工学 ...
政治 政治, 行政, 国会, 法 ...	社会科学 経済学, 社会学, 言語, 政治学 ...
健康と医学 病院, 病気, ダイエット ...	生活と文化 子ども, 環境, グルメ, 障害者 ...

図 4 繰返し構造
Fig. 4 Repetition structures.

芸術と人文 写真, 建築, 美術館, 歴史, 文学 ...	メディアとニュース テレビ, ラジオ, 新聞, 雑誌 ...
ビジネスと経済 ショッピング, B2B, 雇用, 金融, ...	趣味とスポーツ アウトドア, ゲーム, 車, スポーツ, 旅 ...
コンピュータとインターネット ハードウェア, ソフトウェア, WWW ...	各種資料と情報源 図書館, 辞書, 郵便, 電話番号 ...
教育 大学, 専門学校, 小中高, 資格 ...	地域情報 日本の地方, 世界の国 ...
エンターテインメント 映画, 音楽, 芸能人, コミック, 占い ...	自然科学と技術 動物, エコロジー, 地球, 天文, 工学 ...
政治 政治, 行政, 国会, 法 ...	社会科学 経済学, 社会学, 言語, 政治学 ...
健康と医学 病院, 病気, ダイエット ...	生活と文化 子ども, 環境, グルメ, 障害者 ...

図 5 構造化
Fig. 5 Structuring.

同一種の情報のグルーピングが可能になり，情報のセグメントを検出することができる．

しかしこのままでは，図 1 における，“芸術と人文” と “地域情報” はうまくグループ化することができない．なぜなら，“芸術と人文” はサブカテゴリを 5 つ持っているのに対し，“地域情報” は 2 つしかサブカテゴリを持っていないからである．

そこで，本研究ではまず，**図 4** に示すような最もプリミティブな繰返し構造を検出する．その後，その繰返し構造が存在する部分をトークンに置き換える．その際，繰返しの回数が異なるが，繰返しの基本単位が同じ繰返し構造は，同一のトークンで置き換える．すなわち，図 4 中で網掛けした部分は，繰返しの基本単位が同じため，含まれる要素数が異なってもその後の処理では同一視される．

トークンで置き換えられた Web ページに対して再び繰返し構造を検出することで，**図 5** で網掛けしたようなさらに大域的な構造を検出することができるようになる．

このように本研究では，最もプリミティブな構造を検出し，見つかった繰返し構造をつぎつぎとトークンに置き換え，再帰的に繰返し構造を発見することで，Web ページ上の情報をボトムアップに構造化する手法を提案する．

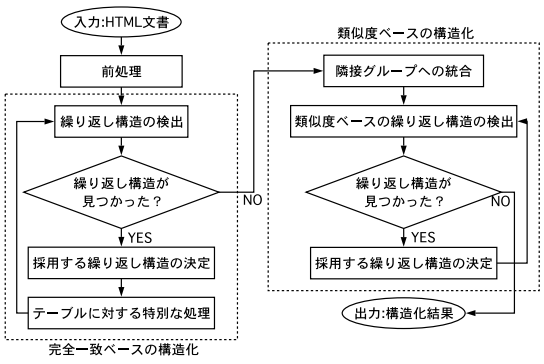

図 6 システムのフローチャート
Fig. 6 Flow chart of our system.

このような構造化については，4.2 節で述べる．また，類似性を考慮した構造化については，4.3 節で述べる．

4. システム構成

図 6 にシステムのフローチャートを示す．我々のシステムは以下の 3 つのステップから構成される．

（1）前処理
（2）完全一致ベースの構造化
（3）類似度ベースの構造化

4.1 前 処 理

まずシステムは，入力 HTML 文書に対して HTML Tidy [18] を適用する．これは，HTML 文書を well-formed XML 文書にするためである．この処理により，開始タグと終了タグのバランスがとれていることが保証される．

次に，コメントやスクリプト部分などの不必要な要素を除去する．さらに，以下のタグはセグメントの境界を示さないケースが多いため，繰返し構造を検出する際に使用しない．

- `<br>,<font>,<b>,<i>,<s>,<tt>,<u>`

その後，テキストを修飾するインラインタグは，テキストの属性として保持するようにする．これは，以下の 2 つを同一のものとして扱うためである．

- `<small><a>text</a></small>`
- `<a><small>text</small></a>`

最後に，タグ以外のテキストをすべてトークン “(text)” で置き換える．なぜなら，我々のシステムではその部分にテキストが存在するという情報しか使

用しないためである．これは，言語非依存のシステムを構築することを目的としているためである．

4.2 完全一致ベースの構造化

このステップは，3 つのサブステップから構成される．

（2a）繰返し構造の発見

（2b）繰返し構造の最適な組合せの発見

（2c）テーブルに対する特別な処理

4.2.4 項に述べるように，以上 3 つのサブステップを上記（2a）の “繰返し構造の発見” の処理によって，新たな繰返し構造が発見されなくなるまで繰り返し行う．

4.2.1 繰返し構造の発見

このサブステップでは，システムは Web ページ全体から図 2 で示した 2 種類の繰返し構造を発見する．

以下に繰返し構造の定義を述べる．まず文書を，HTML タグ，“(text)” トークン，あるいは，4.2.2 項で述べるグループトークンからなるトークン列と考える．図 2 中の “A”，“B” は，このトークン列の部分トークン列のうち以下の条件を満たす部分トークン列を示す．

- “(text)” トークン，あるいはグループトークンを 1 つ以上含む．
- 開始タグと終了タグの対応が正しくとれている．
- 内部に繰返し構造を含まない．

図 2 (A) に示す “セパレータなしの繰返し構造” は，上記 3 つの条件を満たすまったく同じ部分トークン列が，隣接して現れる部分を指す．また，図 2 (B) に示す “セパレータありの繰返し構造” は，上記 3 つの条件を満たすまったく同じ部分トークン列が，間に以下の条件を満たす別のトークン列（以後，セパレータと呼ぶ）を挟んで隣接して現れる部分を指す．

- 開始タグと終了タグの対応が正しくとれている．
- 内部に繰返し構造を含まない．
- すべてのセパレータは，それぞれ等しくなければならない．
- さらに，“(text)” トークンを含む場合は，トークンに置き換える前の文字列も等しくなければならない．
- セパレータは `<a>` タグを含んではならない．

4.2.2 繰返し構造の最適な組合せの発見

前項の処理が終わると，Web ページの同じ範囲に対して複数の繰返し構造が発見されることがある．以下の例を考える（下記の “A”〜“D” は，4.2.1 項の “A”，“B” と同じ制約を満たす HTML 文書の部分トークン列を示す）．

A–B–A–B–A

この部分では，もし “B” がセパレータの制約を満たすのであれば，セパレータありの繰返し構造と考えられる．同時に，最後の “A” を余りと考えれば，“AB” が繰返し単位であるセパレータなしの繰返し構造であるとも考えられる．どちらの繰返し構造がもっともらしいかを決定するためには，それに続く部分の構造を考慮する必要がある．例として以下を考える．

（1）A–B–A–B–A–C–D–C

（2）A–B–A–B–A–C–A–C

（1）のケースでは，もし “B” と “D” がそれぞれセパレータの制約を満たすのであれば，“ABABA” と “CDC” の 2 つのセパレータありの繰返し構造があると考えるのが妥当である．それに対して，(2) のケースでは，(1) のケースと前半部分は同じであるにもかかわらず，もし “A” がセパレータの制約を満たさないのであれば，たとえ “B” がセパレータの制約を満たしたとしても，“ABAB” と”ACAC” の 2 つのセパレータなしの繰返し構造があると考えるのが妥当であろう．

このように，システムは周辺にある，あらゆる繰返し構造を考慮に入れて，最も良い繰返し構造の組合せを選択する必要がある．

我々のシステムは，最も細かい構造からボトムアップにページ全体の構造を組み立てるため，お互いに出現位置が重ならず，繰返し構造の繰返し回数の和が最大となる組合せを最も良い組合せであると考える．

この組合せを発見するために，システムはまず，以下の手順で有向非循環グラフ G 構築する．

(1) G に先頭ノード，終了ノードを加える．

(2) 発見された繰返し構造の集合を R とし，R 中から最も文書の先頭近くに出現する繰返し構造を 1 つ取り出し，G にノードとして追加する．

(3) 追加された繰返し構造と HTML 文書中での出現位置が重複しない G 中のノード間を辺で結ぶ（ただし，先頭ノードと終了ノードはどのノードとも重複しない）．

(4) “推移律的な辺”（たとえば $A \to B$，$B \to C$ の辺があった際の，$A \to C$ のような辺）を消去する（繰返し回数を最大にするというゴールに対して，明らかに不利であるため）．

(5) (2)〜(4) の処理を R が空になるまで繰り返し行う．

以上の処理によって構築されるグラフ G では，先頭ノードから末尾ノードへのパスに含まれるノードは，それぞれ出現位置が重複しないことが保証される．よって，G 中の各ノードに対して，そのノードが示す

表 2 縦に読むべきテーブル
Table 2 Table which should be read vertically.

A	B
C	D
C	D
C	D

表 3 レイアウト目的のテーブル
Table 3 Table which is used for layout purposes.

A	A
A	A
A	A
A	A

繰返し構造の繰返し回数をスコアとして割り振り，このスコアの和が最大となるパスを Viterbi アルゴリズム[19] によって発見することで，お互いに出現位置が重ならず，かつ繰返し回数が最大となる繰返し構造の組合せを発見することができる．

以上のような処理により，繰返し構造の組合せが決定された後，それぞれの繰返し構造が含まれる HTML 文書の該当箇所を “グループトークン” に置き換える．その際，繰返しの基本単位が同じ繰返し構造は同じ id を持ったグループトークンで，別のものは別の id を持ったグループトークンで置き換える．

4.2.3 テーブルに対する特別な処理

テーブルは表現能力が豊かなため，ある種のテーブルに対しては特別な処理が必要となる．

例として，**表 2** を考える．図中の “A”，“B”，“C”，“D” はそれぞれ各セル内のトークン列を示す．もし，このテーブルが縦に読むテーブルで，第 1 行が見出しを表す行であった場合，このままでは正しく構造化することができない．なぜなら，ここまでの手法では，隣接する要素をグループ化することで構造を構築するのに対し，この場合では，グループ化されるべき “A” と “C”，“B” と “D” が HTML 文書中で離れた位置に存在するためである．よって，システムはこのようなテーブルを転置し，グループ化されるべき要素が隣接するように HTML 文書を修正する．システムは，テーブルが以下の条件をすべて満たした場合，縦に読むべきテーブルであると判断する．

(1) 第 1 行が他の行のトークン列とは別のトークン列から成る．
(2) 転置した結果，すべての行が，行を繰返し単位とする繰返し構造に含まれる．

また，テーブルは，単に 2 段組のようなレイアウトを調整する目的で使用される場合も多い．例として，**表 3** を考える．もし，もしすべての “A” が同じトークン列であり，レイアウト上の理由でこのようなテーブルを用いて表現されている場合，本来すべての “A” は並列な要素として構造化されるべきである．しかしながら，ここまでの処理ではこのように正確に構造化することができない．なぜなら，各行の間には tr タグがあるため，“A” を繰返し単位とした繰返し構造は行をまたぐことができないからである．よって，このような場合，行を表すタグを除去することで正しく構造化することが可能となる．システムは，テーブルに含まれるすべてのセル内のトークン列が等しく，かつ以下 2 つの条件のいずれかを満たした場合，レイアウト目的のテーブルであると判断する．

(1) すべてのセル内のトークン列に，グループトークンが含まれる．
(2) すべてのセル内のトークン列に，a タグが含まれる．

これらの制約は，テキストのみから構成されるレイアウト目的でないテーブルとレイアウト目的のテーブルを区別するために導入される．

なお，これらの処理は入れ子になったテーブルにも再帰的に適用される．

4.2.4 構造の組み上げ

以上の処理を新たな繰返し構造が発見されなくなるまで繰返し行う．繰返し構造の発見された部分がグループトークンに置き換えられているため，処理が進むにつれ，より大きな繰返し構造が発見され，Web ページがボトムアップに構造化されていく．

4.3 類似度ベースの構造化

前節までの処理では，繰返し構造の定義が “完全マッチ” と厳しいため，Web ページ全体を構造化することができない場合が存在する．本節では，前節までの処理を適用した文書に対して，さらに以下の 2 つのサブステップを適用することでこの問題を解決する．

（3a）隣接グループへの統合
（3b）類似度を用いた繰返し構造の検出

このように，我々のシステムは前節までに述べた “完全マッチベース” の手法と本節で述べる “類似度ベース” の手法の 2 段階によって構造化を行う．このような 2 段階手法をとる理由は，以下の点を考慮したためである．

- 厳密な定義のみでは，Web ページの一部分しか構造化することができない．
- 最初から類似度を導入すると，誤った繰返し構造が検出されてしまう．

よって，本研究では，まず厳しい制約を課した条件で構造化を行い，厳密な繰返し構造の基本単位を抽出

マッチング対象

$a =$ [(text), ,(text),]

$b =$ [(text), ,<img>,]

マッチング結果

a:	(text)		(text)	*	
b:	(text)		*	<img>	

図 7 DP マッチングの例
Fig. 7 Example of DP matching.

した後に，類似度を考慮することで洩れた部分の構造化を行っていくという手法を選択する．

類似度の計算には，DP（Dynamic Programming）マッチング[20] を使用する．DP マッチングの例を，**図 7** に示す．また，使用する漸化式を式 (1) に示す．

$$g(i,j) = \min \begin{cases} g(i-1,j) + d(a_{i-1}, *), \\ g(i-1,j-1) + d(a_{i-1}, b_{j-1}), \\ g(i,j-1) + d(*, b_{j-1}). \end{cases} \quad (1)$$

a_n，b_n はそれぞれ，比較対象となる HTML 文書の部分に含まれる n 番目の要素を示す．図 7 の例では，a_0 は “(text)” を示し，b_2 は “<img>” を示す．

式 (1) 中のコスト d を式 (2) に示す．

$$d(\alpha,\beta) = \begin{cases} 0 & \text{if } \alpha = \beta \\ 1 & \text{if } \alpha \neq \beta \ and \ (\alpha = * \ or \ \beta = *) \\ \infty & \text{if } \alpha \neq \beta \ and \ \alpha \neq * \ and \ \beta \neq *. \end{cases} \quad (2)$$

“*” は対応する要素がないことを意味する．この結果を利用した類似度を式 (3) により定義する．

$$Sim(a,b) = \frac{M}{M + \max(A,B).} \quad (3)$$

M はマッチした箇所 $[a_i, b_j]$ の数，A，B はそれぞれ $[a_i, *]$，$[*, b_j]$ の数を表す．この類似度は直感的には順序を考慮したときに全体の何割が一致しているかを示している．また，HTML タグの多くは，開始タグと終了タグのペアで用いられるため，より使用されるタグが一致する場合に，類似度は高くなる傾向にある．

図 7 の例では，M は 3，A と B はそれぞれ 1 となる．よって，類似度は以下のように計算される．

$$Sim(a,b) = \frac{3}{3 + \max(1,1)} = 0.75 \quad (4)$$

この値が閾値以上であれば，システムはそれらを類似していると判断する．現在の実装では，閾値は 0.5 に設定している．この理由は，**図 8** に示すような，本来 “(text)” であるべき部分が “<img> (text)” になるようなケース（“NEW!” という画像が “TV” の前に

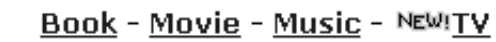

図 8 繰り返し構造に含まれない例
Fig. 8 Example of the element that should be contained in a repetition structure.

加えられている）を類似していると判定すべきワーストケースと考えたためである．また，比較対象となる部分トークン列がグループトークンを含む場合は，繰返し単位を再帰的に展開した後，類似度の計算を行う．

4.3.1 隣接グループへの統合

ここでは，ある繰返し構造に含まれるべき要素が，なんらかの要素が欠けている，あるいは加えられているために，うまくグループ化できなかった場合の処理について述べる．

例として図 8 を考える．“Book”，“Movies”，“Music”，“TV” は同一の繰返し構造に含まれるべきである．しかしながら，“NEW!” という画像が “TV” の前に加えられているため，最初の 3 つのみを繰返し構造としてしまう．

このようなケースに対してシステムは，繰返し構造と判定された部分とそれに隣接する部分を比較し，式 (3) によって類似していると判断されれば，繰返し構造の一部として統合を行う．以下に統合処理の手順を示す．

(1) 注目するグループトークンに隣接する部分トークン列のうち，以下の条件をすべて満たす部分トークン列のうち最長のものを t とする．
- “(text)” トークン，あるいはグループトークンを 1 つ以上含む．
- 開始タグと終了タグの対応が正しくとれている．

(2) 注目するグループトークンが示す繰返し構造の繰返し単位と t との類似度を式 (3) により計算する．

(3) 以上 (1)，(2) の処理をあらゆるグループトークンに対して行い，類似度が閾値以上の組合せに対して，類似度の高い順に統合処理を行う．

ただし，隣接する部分の構造がすでに組み上げられているケースが存在するため，下位のグループトークンが構造化されていない部分に隣接するケースに対しても類似度の計算を行う．例として，**図 9** を考える．図中の “B+” は，B を基本単位とする繰返し構造を示し，“C” を繰返し構造からはみ出た要素と想定する．このケースにおいて，“C” は，“AB+” と類似している可能性とともに，“B+” と類似している可能性も考慮される．

表 4 結果
Table 4 Result.

カテゴリ	セグメンテーション						構造化					
	0-1	1-2	2-3	3-4	4-5	平均	0-1	1-2	2-3	3-4	4-5	平均
ポータル	0%	0%	0%	30%	70%	4.43	0%	0%	20%	50%	30%	3.40
トップページ	0%	0%	0%	0%	100%	4.63	0%	0%	0%	10%	90%	4.43
サイトマップ	0%	0%	0%	0%	100%	4.87	0%	0%	0%	40%	60%	3.97
機械生成	0%	0%	0%	30%	70%	4.30	0%	10%	20%	20%	50%	3.53
CSS 利用	0%	0%	0%	10%	90%	4.33	0%	0%	0%	50%	50%	3.83
研究室	0%	0%	0%	20%	80%	4.40	0%	0%	30%	60%	10%	3.13
ランダム	0%	0%	0%	40%	60%	4.20	0%	0%	20%	40%	40%	3.50
すべて（70 ページ）	0%	0%	0%	19%	81%	4.45	0%	1%	13%	39%	47%	3.67

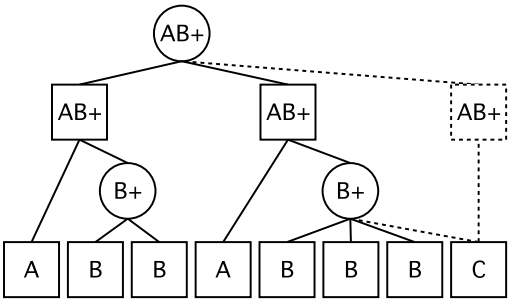

図 9 隣接グループへの統合
Fig. 9 Integration into the group.

4.3.2 類似度を考慮した繰返し構造の発見

4.3.1 項の処理では，グループトークンとその周囲の要素のみを対象としていたのに対し，本ステップでは繰返し構造の定義に類似度を導入し，より大域的な部分の構造化を行うことを目的とする．

以下に類似度ベースの繰返し構造の検出手順を述べる．

(1) 前項までの処理が適用された文書に対して，開始タグと終了タグの対応が正しくとれているトークン列をすべて列挙する．

(2) これらのトークン列のうち，出現位置が隣接し，かつ，隣接するトークン列間の類似度が閾値以上となるトークン列の集合を繰返し構造と見なす．

以上の処理により発見された繰返し構造の集合に対して 4.2.2 項で述べた手法を適用し，最適な組合せを発見する．この処理も 3.2 節の処理と同様，新たな繰返し構造が発見されなくなるまで繰り返し適用する．

5. 評　　価

5.1 評価データ

評価のために我々は 70 ページの Web ページをランダムに収集した．その際，以下の 7 カテゴリを設定した．このようなカテゴリを設定した理由は，たとえば CGI で機械生成されたページは人手により記述されたページよりも規則正しく記述されていたり，ポータルサイトなどには非常に多くの情報が詰め込まれていたりするなど，HTML 文書にも様々なものがあり，この違いによるシステムの出力の相違を明らかにするためである．

- ポータルサイト
- 企業のトップページ
- サイトマップ
- CGI などによって機械生成されるページ
- CSS（Cascading Style Sheets）を使用しているページ
- 我々の研究室のページ
- Yahoo!Japan のランダムリンクから取得したページ

5.2 セグメンテーション，構造化に対する評価

評価は，我々のシステムが出力した結果が人間の理解する構造と対応しているかどうかに焦点を合わせて行った．3 人の被験者には，出力された結果とオリジナルのページを注意深く見比べてもらい，セグメンテーション，構造化それぞれに対して，5 段階で主観評価を行ってもらった．また，同時にシステムの出力が誤っている部分を指摘してもらった．

表 4 にセグメンテーション，構造化それぞれに対する評価の結果を示す．パーセント表示されている部分は，それぞれのカテゴリに属する各 10 ページに対して，3 人の被験者のスコア（5 が最も良い）の平均がどのように分布しているかを示している．また，平均の列は 3 人の被験者が付与したスコアの 10 ページでの平均を示す．

5.3 2 段階手法の有効性

本論文で提案した，“完全一致ベース” と “類似度ベース” の繰返し構造を段階的に検出する手法の有効性を調査するために，70 ページに含まれるグループトークンがどちらの手法によって生成されたかを調査

表 5 2 段階手法の有効性（1）
Table 5 Validity of the two-stage strategy (1).

完全一致ベース	767
類似度ベース	260

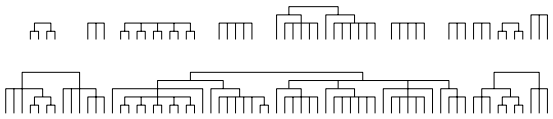

図 10 2 段階手法の有効性（2）
Fig. 10 Validity of the two-stage strategy (2).

した．**表 5** に結果を示す．また，70 ページすべてにおいて，類似度ベースの手法により構造化された部分が存在した．

また，**図 10** に，平均的なケース（完全一致ベース：21，類似度ベース：5）における，それぞれの手法による構造化の例を示す．図 10 の上段は，4.2 節までの処理を適用した段階での構造を示し，下段は，さらに 4.3 節の処理を適用した結果の構造を示す．完全一致ベースの処理のみでは，局所的にしか構造化できていないのに対し，類似度ベースの処理を後処理として用いることで，ページ全体が構造化されている．なお，図 10 に示した例は，3 人の被験者により，セグメンテーションは 5.00，構造化も 4.67 という評価を得ている．

5.4 考　　察

本節では，被験者によって指摘されたエラーの分析結果について述べ，現状のシステムの問題点について考察する．

5.4.1 繰返し構造の検出に使用するタグの種類

“ポータル”，“トップページ”，“サイトマップ” は他のカテゴリに比べて結果が良い．これは，繰返し構造を検出する際に使用したタグに原因があることが分かった．我々の現在のシステムは <a> タグを手がかりとして使用している．しかしながら，<a> タグを使用すべきでないケースも存在する．以下の例（A）（B）を考える．ただし，下線の引かれた部分はリンクがあるとする．

（A）昨日，ここ と ここ へ行った．

（B）映画 - 音楽 - テレビ

<a> タグに注目したことにより，(A）のケースでは 3 つのセグメントに分割されてしまうが，本来このようにセグメンテーションされるべきでない．他方，(B）のケースでは，<a> タグに注目することで構造化に成功している．そして，(B）のケースはこれら結果のよいカテゴリによく表れ，(A）のようなケースがあまり表れなかったため，実験結果に偏りが生じたと考えられる．

以上のような点を考えると，繰返し構造を検出する際に使用するタグを，Web ページあるいは Web ページの部分に依存して，動的に決定することができれば，より良い結果が得られると考える．

5.4.2 類似度の定義

我々のシステムでは，類似度を DP マッチングを使用して計算している．しかしながら，現在の類似度の定義では “マッチした” もしくは “マッチしなかった” という情報しか使用していない．

以下の 3 つの例を考える．

（A）リンク付きテキスト

（B）リンクなしテキスト

（C）リンクなし画像

現在の類似度の定義では，これら 3 つの要素はすべて “別なもの” としてのみ扱われる．しかしながら，(A）と（B）の間の類似度は，(B）と（C）の間の類似度よりも高いと考えるのが自然である．

このように，要素間の類似性を考慮した方法で類似度を計算すると結果が改善される可能性がある．また，現在の閾値は，経験的に決定されているが，適切な決定方法を考慮することによりより良い結果が得られる可能性がある．

5.4.3 使用しなかった情報の使用

現在のシステムでは，フォントの色，リンクの種類，画像の大きさなどの情報をいっさい使用していない．しかしながら，これらの情報がなければ正確に構造化することのできないケースがいくつか存在した．

また，我々のシステムでは言語情報をいっさい使用していない．そのため，あらゆるページに対して適用可能であるが，もし言語情報を使用すれば，セグメンテーション，構造化の精度のさらなる向上が可能であったと考えられる．たとえば，4.2.3 項で述べたテーブルに関する処理では，セル内の文字列の類似性（たとえば，表層の一致や数字，アルファベットなどの文字クラスの一致など）を考慮することで，より正確な構造化が可能であると考えられる．

また，別の問題として，我々はタグ以外の要素がセグメント境界になりうることを想定していなかった．しかしながら，括弧，ハイフンなどはセグメント境界を示すことも多い．今後はこのようなセグメント境界になりうる文字に関する情報も使用すれば結果を改善することができると考える．

6. おわりに

本論文では，Web ページを自動的にセグメンテーション，構造化する手法を提案した．

従来研究では，構造化の対象が Web ページの一部のみであったり，粒度の荒い構造しか扱われていなかったりする．それに対し我々は，あらゆる Web ページに適用でき，またより細かい部分から Web ページ全体の構造をボトムアップに構築することのできるシステムを目標とした．この目標を達成するために，まずは言語情報を用いずに構造化を行った．

また，Web ページに含まれる構造の例外に対して頑健なシステムを構築するために，完全一致ベースと類似度ベースの 2 段階手法を提案し，この手法の有効性を示した．

我々の現在のシステムにはまだ問題があるが，主観実験の結果は多くのページに対して，人間の理解する構造に対応する構造を出力できていることを示している．

今後の課題として，まず 5.4 節で述べた課題を考慮し，システムを改善することがあげられる．また，次の目標として，構造化されたセグメントに対して，タイプなどのメタデータを付与することを計画している．

さらなる課題として，Web ページの構造に関する知識を使用する何らかのアプリケーションを使用して，システムの評価を行うことがあげられる．たとえば，構造に関する情報を加味した読み上げ順序で Web ページを読み上げる音声ブラウザや，携帯電話や PDA など，画面の小さなデバイス向けに Web ページを分割するシステムなどを考えている．このような評価には，Chen ら[15] の研究が参考になると考えている．

謝辞 担当編集委員，査読者の方々からは貴重なコメントをいただきました．感謝いたします．

参考文献

1) Amazon.com: Amazon.com web services 2.0. URL:http://associates.amazon.com/exec/panama/associates/ntg/browse/-/106766%2/
2) Google: Google web apis. URL: http://www. google.com/apis/
3) Libby, D.: RDF Site Summary (RSS) 0.9 official DTD (1999). http://my.netscape.com/publish/formats/rss-0.9.dtd
4) The World Wide Web Consortium: Semantic Web (2001). URL: http://www.w3.org/2001/sw/
5) Berners-Lee, T., Hendler, J. and Lassila, O.: The Semantic Web, *Scientific American* (2001).
6) The World Wide Web Consortium: Document object model. URL: http://www.w3.org/DOM/
7) Henzinger, M.R., Motwani, R. and Silverstein, C.: Challenges in Web Search Engine, *Proc. International Joint Conference on Artificial Intelligence*, pp.1573–1579 (2003).
8) Lim, S. and Ng, Y.-K.: Converting the Syntactic Structures of Hierarchical Data to Their Semantic Structures, chapter 24, *Information Organization and Databases*, pp.343–355, Kluwer Academic Publishers (2001).
9) Laender, H.F., Ribeiro-Neto, B.A., da Silva, A.S. and Teixeira, J.S.: A Brief Survey of Web Data Extraction Tools, *ACM SIGMOD Record*, Vol.31, No.2, pp.84–93 (2002).
10) Crescenzi, V., Mecca, G. and Merialdo, P.: RoadRunner: Towards Automatic Data Extraction from Large Web Sites, *27th International Conference on Very Large Data Bases*, pp.109–118 (2001).
11) Embley, D., Campbell, D., Jiang, Y., Liddle, S., Lonsdale, D., Ng, Y.-K. and Smith, R.: Conceptual-model-based data extraction from multiplerecord web pages, *Data & Knowledge Engineering*, Vol.31, No.3, pp.227–251 (1999).
12) Embley, D.W., Jiang, Y.S. and Ng, Y.-K.: Record-Boundary Discovery in Web Documents, *1999 International Conference on Management of Data* (*SIGMOD'99*), pp.467–478 (1999).
13) Chen, H.-H., Tsai, S.-C. and Tsai, J.-H.: Mining tables from large scale html texts, *Proc. 18th International Conference on Computational Linguistics*, Vol.1, pp.166–172 (2000).
14) Wang, Y. and Hu, J.: A machine learning based approach for table detection on the web, *Proc. 11th International World Wide Web Conference*, pp.242–250 (2002).
15) Chen, Y., Ma, W.-Y. and Zhang, H.-J.: Detecting web page structure for adaptive viewing on small form factor devices, *Proc. 12th International World Wide Web Conference*, pp.225–233 (2003).
16) Yang, Y. and Zhang, H.: HTML page analysis based on visual cues, *Proc. 6th International Conference on Document Analysis and Recognition*, pp.859–864 (2001).
17) Yu, S., Cai, D., Wen, J.-R. and Ma, W.-Y.: Improving pseudo-relevance feedback in web information retrieval using web page segmentation, *Proc. 12th International World Wide Web Conference*, pp.11–18 (2003).

18) Raggett, D.: Clean up your web pages with html tidy. URL: http://www.w3.org/People/Raggett/tidy/
19) 長尾 真，佐藤理史（編）：岩波講座ソフトウェア科学（15）自然言語処理，岩波書店 (1996).
20) 上坂吉則，尾関和彦：パターン認識と学習のアルゴリズム，文一総合出版 (1990).

(平成 15 年 12 月 8 日受付)
(平成 16 年 7 月 1 日採録)

南野 朋之

1977 年生．2003 年東京工業大学大学院総合理工学研究科知能システム科学専攻修士課程修了．現在東京工業大学大学院総合理工学研究科知能システム科学専攻博士後期課程在学．Web マイニング，自然言語処理に関する研究に従事．

齋藤 豪（正会員）

1971 年生．1999 年東京工業大学大学院情報理工学研究科計算工学専攻博士後期課程修了．1999 年より東京工業大学大学院情報理工学研究科非常勤研究員．1999 年より東京工業大学精密工学研究所助手．2004 年より東京工業大学情報理工学研究科計算工学専攻助教授．コンピュータグラフィックス，画像処理，キャラクターエージェントに関する研究に従事．工学博士．ACM SIGGRAPH, IEICE，芸術科学会各会員．

奥村 学（正会員）

1962 年生．1989 年東京工業大学大学院情報理工学研究科計算工学専攻博士後期課程修了．1989 年より東京工業大学大学院情報理工学研究科助手．1992 年より 2000 年北陸先端科学技術大学院大学助教授．1997 年より 1998 年トロント大学客員助教授．2000 年より東京工業大学精密工学研究所助教授．自然言語処理，自動テキスト要約，コンピュータによる語学学習支援，テキストデータマイニングに関する研究に従事．工学博士．JSAI, AAAI，ACL，JCSS 各会員．