Webページ中の階層的見出し構造の発見

真鍋　知博†　　田島　敬史††

†, †† 京都大学大学院情報学研究科　〒 606–8501 京都府京都市左京区吉田本町
E-mail: †manabe@dl.kuis.kyoto-u.ac.jp, ††tajima@i.kyoto-u.ac.jp

あらまし　多くの Web ページは，ある実体やその属性などに関する記述が互いに入れ子になった論理上の階層構造を持ち，各記述には人間にとっての視覚的見出しが付けられている．しかし，Web ページの DOM 木構造と論理上の階層構造は必ずしも一致せず，また，見出しは必ずしも見出しを表す HTML 要素で表現されていない．そこで本論文では，Web ページの内容の論理上の階層構造と，各部分の視覚的見出しを抽出するための手法を提案する．提案手法では，まず，ページ中のノードのうち，実体や属性の記述の切れ目を表示する「セパレータ」を検出し，各セパレータが切り分ける範囲を推定することにより，ページを階層的に切り分ける．次に，切り分けられた各記述から，人間にとっての視覚的見出しを検出する．セパレータや視覚的見出しの検出には，ルートからのパス上の要素名を並べたタグパスの情報などに基づき，ノードを表示スタイル毎に分別した結果を用いる．また，各セパレータが切り分ける範囲の推定には，各ノードから，同種のノードと出会わない範囲でどこまで DOM 木を遡れるかという情報などを用いる．見出しを用いる本手法は，類似構造の繰り返しを利用する従来手法に比べ，不規則な構造のページにも対応できる．

キーワード　情報抽出，論理構造抽出，閲覧支援

1. はじめに

Web の利用の拡大と Web 上の情報量の増大に伴い，効果的な Web ページの閲覧支援や，自動利用可能な形式での Web からの情報抽出が必要とされている．これらを実現するための既存研究では，テキストの文構造やページ間のリンク構造の理解が重視されているが，加えてページの DOM 木構造が意味する論理的構造の理解も重要である．Deep Web からの情報抽出，Web ページ中のリストや表の抽出，Web ページのセグメンテーションなどは，特に論理構造抽出と関係が深い分野である．

一般に，ある程度の長さを持つ文書は，内容が階層的な構成になっている．このことは，例えば書籍の章節構成などに表れている．Web ページにおいても同様で，その記述言語 HTML には，H1 から H6 までの 6 レベルの見出し要素が定められている．しかし一般のブラウザでは，これら見出し要素の階層構造は文字サイズで表現されるため，文字サイズを直接指定するタグによって見出しが表現されることもある．このように，見栄えの情報を分離できていないなどの理由により，Web ページの DOM 木構造から，内容の階層構造は自明ではない．そこで，階層構造の抽出のために様々な手法が試みられている．

実世界のモデリング手法として，実体とその属性という二種類の概念が広く用いられている．Web ページの論理的構造の抽出に関するこれまでの研究の多くも，Web ページの階層構造を，実体のリストと各実体の属性の入れ子によりモデリングしている．しかし，このようなモデリングには 2 つの弱点がある．

多くの既存研究における実体のリストとは，類似する論理構造を持った部分の繰り返し出現を意味する．しかし，文書の章の並びや，図 1 のような Web 検索結果アイテムの並びのように，人間にはリストとして認識されるものでも，各実体の内部の論理構造が不一致な場合もある．このような，各実体の内部の論理構造がまちまちなリストをも扱える既存手法は数少ない．

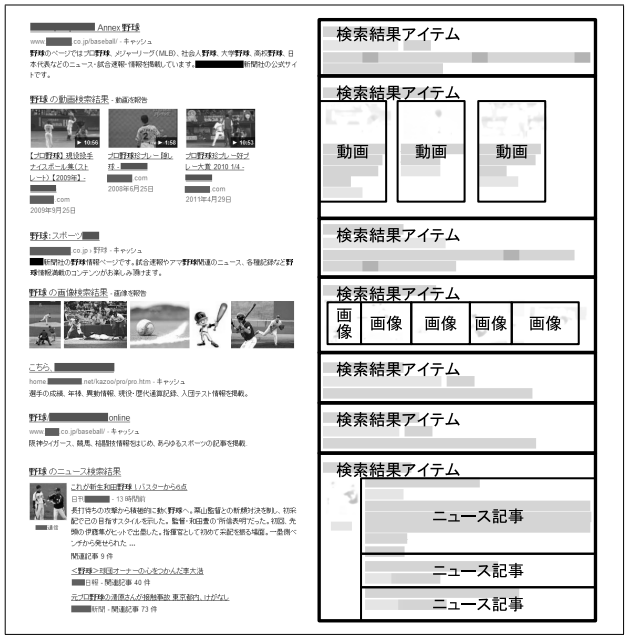

図 1　Web 検索結果ページの階層的な構造の例

また，実体と属性の概念を用いて階層構造を表現する場合，実体と属性の差の曖昧さの問題もある．例として，ある研究室のサイトメニューに,「メンバー」や「研究テーマ」が含まれる場合，これらは「メニュー項目」実体のリストであるとも，それぞれ「研究室」実体に共通する属性であるとも言える．実体と属性による階層構造の抽出のためには，二者の明確な区別が必要になるが，この点を深く考察している研究も少ない．

本研究では，人間による階層構造の認識は，同じ表示上のスタイルを持つ「見出し」の出現に基づくと考え，同スタイルの見出しを備える記述の列挙が，人間にとってのリストに相当す

る「並び」を形作ると考える．ここで，各記述が実体か属性値かは考慮しない．この基準により，並びの定義を厳密にすると共に，構造抽出の段階での実体と属性の区別の必要をなくす．また，並びの各記述の切れ目も，視覚的な区切りで表現されていると考える．視覚的見出し自体が区切りであることも多いが，そうでない場合もある．視覚的区切りは，各記述の内部の論理構造がまちまちな並びと，揃った並びの双方に備わっており，区切りによる階層構造の抽出は，これら両方に有効である．

例えば，図 1 中の各検索結果アイテムは，Web ページのタイトルや「野球の動画検索結果」等の同スタイルの見出しを備え，各アイテムの内部の論理構造はまちまちであるが，これらは並びを形作る．この例では，これらの視覚的見出し自体が並びの各要素を区切る視覚的区切りともなっている．同様に，2 つ目の検索結果アイテムはキャプチャを見出しとする 3 つの動画の並びを含み，4 つ目の検索結果アイテムはサムネイルを見出しとする 5 つの画像の並びを含んでおり，これらの入れ子関係にある並びがこのページの階層構造を構成している．

本論文では，この考察に基づき，視覚的区切りを表す「セパレータ」の検出により並びを発見する，Web ページの階層構造の抽出法を提案する．また，人間にとって実際にある記述のラベルとして機能する「視覚的見出し」は，セパレータとは異なる場合があるので，提案手法では，階層構造の抽出の後，あらためて各記述の視覚的見出しを検出する．

2. 関 連 研 究

前述の通り，木構造文書からの論理構造の自動抽出の需要は大きく，関連研究も多岐に亘る．具体的には，Deep Web からのバックエンド DB 中の関係の再現，Web ページ中のリストや表形式の情報の抽出，Web ページのセグメンテーション，XML コレクションからのスキーマ発見の分野がある．

これまでの論理構造抽出の手法は，主に局所的構造や特定の論理構造に注目している．例えば Deep Web からの情報抽出では，DB 中のある関係中の 1 つのレコードに関する詳細な情報を提供する「Detail page」の集合を入力とするため，入力の時点で内容が実体単位に分割されている手法が多い．また，リスト抽出や表抽出においては，ネストを含まない表やリストが主要なコンテンツである Web ページを想定する研究が多い．XML のスキーマ発見では構造の規則性が重視される一方，実体と属性や，見出しとそれが付いた記述といった論理構造は重視されないため，そのまま応用はできない．

Web ページからの階層構造の抽出手法として，Arasu ら [1] の，Detail page にネストされたリストの存在を踏まえた手法がある．例えば，EC サイトの Detail page はレビューのリストを含み，その項目数は商品によって異なる．このような場合，「Reviewer Name」や「Rating」などの属性名が，各ページに同数ずつ出現することを利用し，ネストされたリストを発見するものである．しかしこの手法は，全ての Detail page が単一のテンプレートにより生成されているという Deep Web モデルに特化している他，二階層から成る構造しか抽出できない．

リスト抽出の分野では，Zhai ら [5] の手法のように，各実体の内部の論理構造，ひいては木構造の類似を前提とし，部分木の照合によりリストを抽出する手法が多い．一方 Miao ら [2] は，ネステッドリストをも抽出するため，ルートから各要素へのパス上の要素名を書き並べたタグパス別のノード集合を，似たパターンで出現するタグパス同士が近くなるようクラスタリングし，ノードをリスト別に分ける手法を考案した．この手法により，例えば，論文の年度別リストから，年度のリストと論文のリストを抽出することができる．しかし，年度と論文の関連など，上下のリストの要素が成す階層構造を抽出できない．

Web ページの階層的なセグメンテーションを行う VIPS [3] は，まず木構造や HTML の要素の特徴を元に，ページ中の意味単位であるブロックを検出し，次にレンダリング後のブロック間のマージンの大小からブロックの論理上の距離を判断して論理構造を抽出する．これを再帰的に行えば，ページを階層的に分割できる．しかし，手法が HTML に関する事前知識に強く依存し，HTML の仕様への追従が必要で，他言語への応用もできない．また，レンダリングの処理負荷が生じ，結果がレンダリングエンジンにも依存する．そして，ブロックの入れ子関係の論理的意味や，ブロックのラベルについては論じていない．

ページ部分に対するラベリング手法としては，Mukherjee ら [4] が提案したものがある．しかしこの手法は，例えばニュースサイトであれば”Major Headline News”，”Category News” など，あらかじめ各部に手動でラベリングされた Web ページを入力として学習を行い，後から新たなページが与えられた際，その各ページ部分に該当するラベルを選ぶというものである．本手法では，ラベルもページ中から抽出することを目指す．

3. Web ページのモデル

本章では，本研究が抽出を目指す論理構造である，Web ページの階層構造について述べる．

3.1 見出しと並びによる階層構造

本研究では，Web ページの論理構造とは，同スタイルの見出しを備えた記述の並びと，記述の間の入れ子関係による階層構造であると考える．**見出し**は記述の一部であり，人間の目を引き記述への導入の役割を果たす，内容を端的に表した部分である．Web ページにおける見出しは，テキストなど実際に表示されるノードで，特に目立つよう配置される．その特性上，多くの見出しは記述の初めに配され，記述のその他の部分とは異なるスタイルを持つ．**スタイル**とは，テキストの文字サイズや色，下線の有無など，局所的な見栄えに関する情報である．

そして**並び**とは，同じスタイルの見出しを共通に持つ記述の列挙である．例えば図 1 は検索結果アイテムの並びであり，各結果アイテムの見出しは，Web ページのタイトルや「野球の動画検索結果」等の同スタイルのテキストである．リスト抽出などで，類似の実体の列挙をリストとする既存研究が多いが，この定義を一般の Web ページに適用すると，類似性の基準や，実体と属性の区別が曖昧な場合がある．本論文では，同スタイルの見出しを共通して持つなら，それらは並びであり，各記述が実体と属性どちらであるかは議論しない．

各記述は，内部に更に記述やその並びを含み，**階層構造**を形

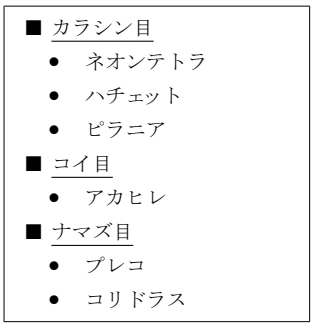

図 2　一般の Web ページの例

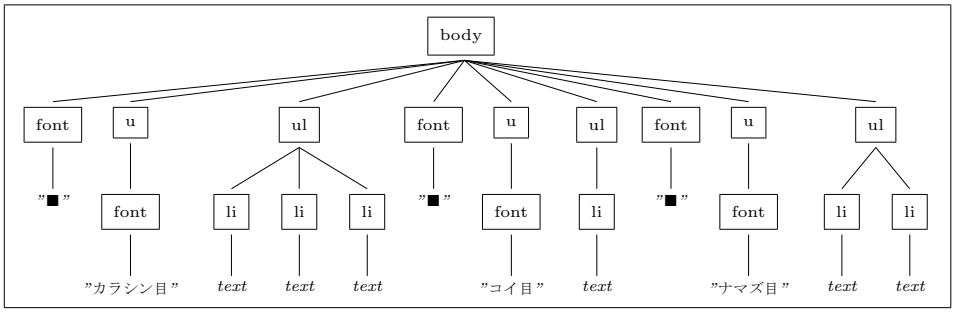

図 3　図 2 のページの木構造

作ることができる．例えば図 1 では，2 つ目の検索結果アイテムはキャプチャを見出しとする動画の並びを含む．同様に，4 つ目の検索結果アイテムはサムネイルを見出しとする画像の並びを，7 つ目の検索結果アイテムはタイトルを見出しとするニュース記事の並びを含む．このような構造は，実体と属性を任意の入れ子関係にできるネステッドリレーションに似るが，実体と属性を記述として一般化したものであるとも言える．

階層構造は，多くの Web ページをモデリングできる．例えば長い文書は章タイトルを見出しとする章の並びで，各章は節タイトルを見出しとする節の並びを含む．Weblog は記事タイトルを見出しとする記事の並びで，各記事はコメントやトラックバックの並びを含む．ポータルトップは大小のカテゴリ名を見出しとするカテゴリと，それ自体を見出しとする種々のコンテンツへのリンクアンカーの成す階層構造でモデリングできる．

3.2　記述を代表するノードと断片ノード

本節では，前節の階層構造と DOM 木構造の対応を述べる．

前節の通り，階層構造の単位は見出しを備えた記述である．DOM 木構造においても，ある記述全体を子孫に持ち，他の記述の部分を子孫に持たないノードが存在すれば．その**記述を代表するノード**と呼ぶ．図 2 の熱帯魚の分類に関するページの例とその木構造図 3 で言えば，LI 要素は，ネオンテトラ，ハチェットなどの各種に関する記述を代表する．

しかし同図で，カラシン目，コイ目などの各目に関する記述を過不足なく指し示すことは，一つのノードによっては不可能で，font 要素，u 要素，ul 要素という 3 つのノードが必要である．このように，一つの記述が連続した兄弟関係にある複数のノードに亘っている場合を，記述を代表するノードが存在しない場合と呼び，それらのノードのそれぞれを**断片ノード**と呼ぶ．

記述を代表するノードは，同じ記述に属する兄弟の並びの長さが 1 である特殊な断片ノードである．つまり本研究では，あらゆる記述は，連続した兄弟関係にある複数のノードで過不足なく指し示せると考える．この仮定が不適な場合として，ある記述が木構造上で連続でない Web ページなども考えることができる．しかし本研究では，実際にはこのような木構造を持つページは非常に少ないと考え，考慮しない．

4.　諸概念の定義とその意味

本章では，実際に Web ページの内容を階層構造に切り分けるために，本研究が導入する諸概念を説明する．

4.1　セパレータと視覚的見出し

記述は一つの意味単位であり，記述を代表するノードの不在は，論理構造と DOM 木構造の乖離の理由の一つである．前述の通り人間は，このような場合でも見出しに基づいて Web ページを階層構造に分割でき，本研究でも見出しに基づいて機械的にページの分割を行う．例えば，図 1 の例では，各検索結果アイテムのタイトルの出現を開始の合図として，各検索結果アイテムを切り分けることができる．

また，図 2 のページとその木構造図 3 の例では，カラシン目，コイ目などの各目に関する記述は，「■」というテキストの出現を開始の合図として切り分けられる．しかし，記述の内容を端的に表す部分，という見出しの原義から言えば，人間にとっての見出しはカラシン目，コイ目などの，目名を表すテキストであり，各記述のラベルとしてはこちらが適当である．このように，記述の機械的な切り分けに利用できる区切り表示と，実際に人間が利用する見出しは，一致するとは限らない．

そこで，まずページの内容の機械的な切り分けに利用できる区切り表示を**セパレータ**と呼ぶことにする．またセパレータは，それが切り分ける記述の範囲の情報を持つので，これを**係り先**と呼ぶことにする．あるセパレータの係り先に別のセパレータが含まれるなら，前者のセパレータが切り分ける記述は，後者のセパレータが切り分ける記述を含む．本論文が提案する手法では，セパレータを発見し，その係り先を求めることの繰り返しによって，ページを階層構造に分割する．

一方本研究では，実際に人間が利用し，ラベルとなる見出しを**視覚的見出し**と呼ぶ．本論文が提案する手法では，セパレータが切り分けた各記述について，改めて人間にとっての見出しである可能性が高い部分を検出し，その部分は記述の視覚的見出しであるとする．全ての記述には視覚的見出しがつくと考え，セパレータそのものが視覚的見出しである記述もある．検出した視覚的見出を，元の記述のラベルとして割り当てるラベリングは，抽出した情報の有用性を高める．

4.2　遡るという操作と巣

本節では，セパレータやその係り先，視覚的見出しの検出のために重要な，同じスタイルのノードと出会わない範囲で DOM 木を**遡る**という操作について述べる．

遡る操作の入力は同スタイルの全ノードの配列であり，出力もノードの配列である．ただし後述するが，提案手法においては同スタイルのノードは木構造上でも必ず同じ深さにあるもの

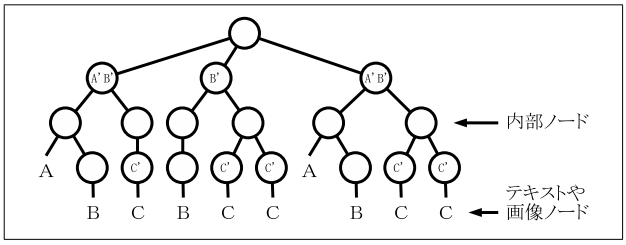

図 4　同スタイルのノードと出会わない範囲で遡る操作の例

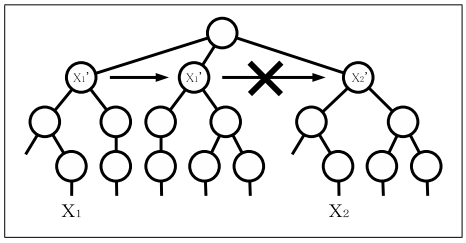

図 5　セパレータの巣から兄弟関係を辿って係り先を求める操作の例

とする．遡る操作とは,「配列が含むノードを一斉にその親ノードで置き換える」という操作を，配列の要素のノードがユニークでなくなる直前まで繰り返すという操作である．今後，あるノード集合を表す配列に関して遡る操作により得られた配列が表す集合を**巣集合**，入力の配列中の各ノードが遡った先のノード (出力の配列中で同インデックスにあるノード) を**巣**と呼ぶ．ただし，あるスタイルのノードがページ中に 1 つしか存在しない場合，その巣はドキュメントルートノードである．

HTML には，スタイルを変化させるための，見栄えを表すタグが定められている．記述を代表するノードの不在の他，これらのタグによるノードの木構造上の深さの変化も，論理構造と DOM 木構造の乖離の理由の一つである．遡るとは，直感的には，あるスタイルのノードが一個しか含まれない範囲を求めることであり，求まる範囲が巣である．このような意味を持つ巣は，元のノードの論理構造上の深さを示し，元のノードを含む記述を代表するノードや，その断片ノードであることが多い．このように遡る操作は，論理構造上の深さと DOM 木構造上の深さの乖離を打ち消す作用を持つ．

図 4 に例を示す．各葉ノードが遡る操作の対象とするノード，英字はそのスタイルを表す．また英字にプライムを付けて巣ノードを表した．A については 2 つ上のノードまで遡れるが，もう 1 つ遡るとすると 2 つの A が重なってしまう．C については 1 つ上のノードまで遡れるが，もう 1 つ遡るとすると左から 2 つ目と 3 つ目の C が重なってしまう．木構造からは論理上の階層構造は不明であったが，遡る操作によって，A と B から成る記述の並びが，C から成る記述を含んでいるという構造が明らかになった．

4.3　辿るという操作と係り先

前項によれば，あるノードの巣は，そのノードを含む記述を代表するノードや，その断片ノードである．つまり，少なくともセパレータの巣は，その基本的な係り先である．巣が断片ノードである場合を考えると，そこから兄弟関係を辿り係り先を決定する必要がある．節 3.1 の通り，見出しは記述の始めの方に配置されやすいため，巣から兄弟関係にあるノードを文書順で順方向に辿ったノードを係り先に加える．この際，兄弟関係の終端に達した場合はもちろん，以下のいずれかのノードに達した場合も，それを係り先に含めず，辿る操作を終了する．

- 注目するセパレータと同スタイルのノードの巣
- 注目するセパレータを含んでいる，より上位の記述の範囲外にあるノード

前者を係り先に含めないのは，該当するノードが同じ並びを形作る次のセパレータの基本的な係り先だからであり，後者を係り先に含めないのは，階層構造において，より上位の記述の範囲を，それが含む下位の記述の範囲が逸脱することはないためである．辿る操作の例を図 5 に示す．この例では，セパレータ X_1 の巣から辿る操作によって係り先を求め，同じスタイルの次のセパレータ X_2 の巣への到達により操作を終了している．

- 金賞『泡沫の輝き』
- 銀賞『CASE』
- 『レンタル』

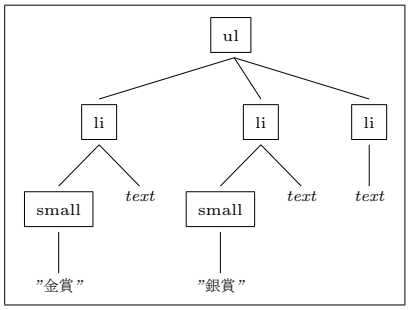

図 6　オプショナルな属性を含むページとその木構造

4.4　オプショナルな部分

図 4 中には，A の巣であるが B の巣でないノードは存在せず，B の巣であるが A の巣でないノードは存在する．もし A がセパレータであったとすれば，A の係り先に含まれる B のうち，A と巣を共有するものと，そうでないものが存在することになるが，このような木構造を持つ Web ページは少ない．通常，このような構造は，**オプショナルな部分**，つまりある並びの全ての記述が共通に持つわけではない，同じスタイルを持つ部分の存在を示唆している．見出しは，ある記述が独立してある並びに参加するために必須であり，オプショナルな部分は見出しではありえない．そこで，あるノード集合 X の巣集合が，他のノード集合 Y の巣集合の真部分集合ならば，X はセパレータや視覚的見出しではありえないと考えられる．

具体例を示す．図 6 は，映像作品に関する並びを含むページと，その木構造の例である．ここで小さい文字による「受賞」属性値はオプショナルであり，大きい文字による「作品名」属性値が見出しである．ここで，前者に関する巣集合は，一部の LI 要素から成るのに対して，後者に関する巣集合は，全ての LI 要素から成る．よって巣集合の包含関係に関する条件を満たし,「受賞」属性値は見出しでないと正しく推定される．

4.5　説　明　文

前節で，見出しではないノード集合の存在について述べた．このようなノード集合としては，他に説明文の集合が考えられる．**説明文**とは，主部と述部を備え文構造で物事を説明する文である．説明文は表現力が高く，例外的な物事を記述できるため，木構造で任意の箇所に出現する．また，まずプレーンテキ

ストを書き，各部をマークアップするという手順で製作された文書では，マークアップされずに残る部分であるため，スタイルによる分別が難しい．

一方，説明文は，セパレータや視覚的見出しとなることが稀である．そこで，説明文を表すテキストノードの集合を検出し，見出しの候補から除去する．説明文の検出には，テキストの長さや，句読点を含むという特徴の利用が考えられる．

5. 手　　法

本章では，これまでの観察に基づき，Web ページから論理上の階層構造を抽出し，各部分から視覚的見出しを検出してラベリングを行うための手法を提案する．入力は，HTML による単一の文書とする．本手法の大部分はマークアップ言語一般に適用可能であるが，一部言語に固有の特徴も利用するため，今回は対象を Web ページ記述言語として最も普及している HTML に限定する．本手法で利用する HTML 固有の特徴として，IMG 要素は画像を表すという事前知識と，CSS のプロパティがある．単一の Web ページを入力とするのは，一般に，周辺に同じスキーマを持たないページも存在するためである．

本手法の流れは以下の通り．

（1） Web ページ中のレンダリング時に表示されるノードを，スタイル毎にノードの集合に分別する (節 5.1)

（2） 得られたスタイル別集合の巣集合を計算し，スタイル別集合のうち見出しとなり得ないものを除去する (節 5.2)

（3） スタイル別集合を，上位の見出しであると思われる順に取り出してセパレータを決定し，その係り先を検出することの繰り返しにより，Web ページの階層構造を抽出する (節 5.3)

（4） 各部分の視覚的見出しを決定する (節 5.4)

5.1 ノードの分別

本節では，前著 [6] に基づき，Web ページ中のノードを，スタイル毎に分別する手法を述べる．まず，ノードのタグパスによる分別を基本とする．次に，テキストノードについては，直接スタイルを表す CSS のプロパティの情報を含める．最後に，意味的に繋がっている文を木構造上で切るノードを検出し，一つの文を一つのテキストノードにまとめる処理を行うという順序で，ノードのスタイルによる分別を行う．

5.1.1 対象とするノード

まず，見出しであり得るノードを選択する．HTML 文書中で，実際にセパレータや視覚的見出しとして使用され得るのは，レンダリング時に表示されるノードであり，実データの観察によれば多くは以下のものである．

- 空白でない文字を含むテキストノード
- src 属性値を持つ IMG 要素

提案手法では，これらのノードについてのみセパレータや視覚的見出しである可能性があるとし，同じ並びを形作るセパレータや視覚的見出しを集めるための，スタイルによる分別を行う．

5.1.2 タ グ パ ス

次に，タグパスによってノードを分別する．タグパスとは，ルートから注目するノードまでのパスの，パス上の要素名を描き並べた表現である．見栄えを表すタグが定められている HTML では，スタイル毎の分別にはタグパスの情報が必要である．例えば，図 3 中のテキストノードの持つタグパスは，`/body/font/text()`，`/body/u/font/text()`，`/body/ul/li/text()` の 3 種類があり，これはスタイル毎の分別となっている．

5.1.3 CSS のプロパティ

次に，CSS の情報によって更にノードを分別する．前著 [6] では，CSS の情報を取得するため，セレクタとして多用される class 属性値に注目した．しかし class 属性値は，要素をユニークに識別する目的で使われることもあり，スタイルの情報として扱うには不適なページもあった．そこで今回は，ノードの CSS の各プロパティを直接利用する．人間がスタイルの同等性を判断するのに用いるプロパティとして，以下が考えられる．

- font-size(サイズ)
- font-style(傾き)
- font-weight(太さ)
- text-decoration(傍線)
- color(文字色)
- background-color(背景色)

テキストノードについては，タグパスに加え親ノードが持つこれらのプロパティで分別することで，スタイルをより正確に反映した分別ができる．プロパティは見栄えの情報ではあるが配置の情報ではないため，ページを実際にレンダリングせず取得でき，処理負荷やレンダリングエンジンへの依存は小さい．

5.1.4 文を意味的に切らないノードの検出

最後に，文を意味的に切らないノードを検出する．木構造文書には，意味的に繋がっているが，内部ノードの挿入によりテキストノードとしては分かれた文が含まれることがある．例えば図 1 の例で，各ページのタイトルやスニペット中で，クエリと一致し太字で強調された部分である．提案手法では，このようなノードを検出し，一つの文を一つのテキストノードとして扱うことが，見出しや説明文の正しい検出のために必要である．

具体的には，文書中に，テキストノードに挟まれ，子孫にテキストノードを持つ内部ノードが存在する場合，その内部ノードとスタイルが同じ全てのノードについて文を意味的に切らないと判断する．文を意味的に切らないノードは，その内部テキストから作られたテキストノードと置換され，前後のテキストノードがあればそれらとまとめられたものとして扱う．この手法は前著 [6] と同様であり，詳細は割愛する．

5.2 見出しとなり得ないノード集合の除去

前節で示した基準により，セパレータや視覚的見出しでありうるノードの**スタイル別集合**が得られた．次に，スタイル別集合のうち，セパレータや視覚的見出しとなり得ないものを除去する．見出しとなり得ないスタイル別集合には，説明文の集合とオプショナルな部分の集合がある．オプショナルな部分の集合の検出には，各スタイル集合の巣集合の計算も必要である．

5.2.1 説明文の集合の除去

まず，スタイル別集合のうち説明文の集合を検出し除去する．説明文の除去のために，今回は，言語依存であるが精度が高いと考えられる日本語の句点(注1)の検出を利用する．句点が含まれるテキストノードを説明文であると判断し，説明文の割合が 1/2 を超えるスタイル別集合を，見出しの候補から除去する．

5.2.2 巣の計算，オプショナルな部分の集合の除去

(注1)：Unicode 0x3002，0xFF0E，0xFF61

Algorithm 1 巣の計算

```
1: A ← 入力ノード集合を表す配列
2: m ← A の要素数
3: if m == 1 then
4:     return ドキュメントノードだけを要素とする配列
5: end if
6: while A が同じノードを 2 つ以上含まない do
7:     Result ← A のコピー
8:     for i = 0 to (m − 1) do
9:         A[i] ← A[i] の親ノード
10:    end for
11: end while
12: return Result
```

続いて，オプショナルな部分の集合の除去のため，各スタイル別集合の巣集合を計算する．各スタイル別集合の巣集合は，Algorithm 1 によって求められる．入力の配列 A でインデックス i に配されていたノードの巣は，出力の配列 $Result$ でインデックス i に配されているノードである．タグパスによる分別により，A 中のノードの木構造上の深さは等しいので，遅くとも A の要素が全てドキュメントルートノードになった時点で，このアルゴリズムはループを抜け停止する．

全てのスタイル別集合の巣集合の計算の後，オプショナルな部分に関する考察に基づき，2 つのスタイル別集合の全ての組み合わせをチェックし，巣集合が他方の巣集合の真部分集合であるスタイル別集合を見出しの候補から除去する．

5.3 セパレータとその係り先の繰り返し検出

本手法では，ページの内容をトップダウンに，上位の記述の並びから下位の記述の並びへ切り分ける．人間が Web ページの論理構造を理解する際には，まず大まかな構造を掴み，続いて細かな構造を見るという手順を踏んでいると思われるので，本手法でもトップダウンな切り分けを採用する．

本節では，スタイル別集合のうち前節の手法により選別された**見出し候補の集合**を，巣集合の木構造上の上下関係に基づいて整列し，上位の並びを切り分けるセパレータから順に採用して係り先を決定することで，トップダウンにページの内容を記述の階層構造に分割する手法を述べる．

5.3.1 ノード集合の整列

トップダウンの切り分けのために，まず見出し候補の集合を，より上位のセパレータらしい順に整列する．節 4.2 での考察の通り，セパレータの巣は，その切り分ける記述 e を代表するノード，またはその断片ノードに相当するため，巣の木構造上の深さは，記述の階層構造上の深さに相当する．そこで，巣ノードの木構造上の深さがより浅いセパレータこそ，より上位のセパレータらしいと考えられる．巣集合を求めるアルゴリズムによれば，ある巣集合が含む全てのノードについて，木構造上の深さは等しいので，それをもって巣集合全体の深さとし，これが小さい順に見出し候補の集合を整列する．

巣集合の深さが等しい見出し候補の集合が複数存在する場合もある．節 3.1 の通り，見出しは記述の先頭に配置されやすく，文書順で先に出現するノードこそ，より上位のセパレータらしい．そこで，巣集合の深さが等しい見出し候補の集合の間では，それら各々について，その集合中のノードの中で文書順が最小のものを選び，これらのノードの文書順の昇順に整列する．

Algorithm 2 ページの内容の，記述の階層構造への分割

```
1: for i = 0 to A_result の最大の添字 do
2:     S ← A_result[i]
3:     for all k such that H_e のキー do
4:         E ← H_e[k]
5:         if E が空 or S の要素を 2 つ以上含む E 中の記述が存在しない then
6:             continue
7:         end if
8:         for all e in E do
9:             for all s such that e の範囲中の S の要素 do
10:                新しい記述 e_new を作る
11:                e_new の範囲は s の係り先であるとする
12:                H_e[i] に e_new をプッシュ
13:            end for
14:        end for
15:    end for
16: end for
```

5.3.2 ページの内容の分割

本項では，実際に Web ページの内容を記述の階層構造に分割する手法を述べる．初めは，ページ全体を一つの記述であるとみなし，その範囲はドキュメントルートノードであるとする．並びの定義より，前項の方針に沿って整列した見出し候補の集合を順番に取り出したとき (これを S とする)，その要素を複数含む記述が一つでも存在すれば，その記述と同じ並びに属する各記述は，S の要素をセパレータとして切り分けられる下位の記述の並びを含むと考え，S の各要素の係り先を新たな記述として切り分ける．その要素を複数含む記述が一つも存在しなければ，S による切り分けは行わない．全ての見出し候補の集合についてこれを繰り返せば，ページを階層構造に分割できる．

詳しい分割の手順は，以下のようになる．

（1） 最上位の見出し候補の集合を取り出し S とする

（2） これまでに得られた記述のうち，S の要素を複数含むもの全てについて，その記述の属する並びは，S の要素をセパレータとする下位の並びを含むと判定し，以下の操作を行う

（a） 注目する上位の並びを E とする

（b） 全ての E 中の記述 e と，e が含む全ての S の要素 s について，s の係り先を新たな記述として切り分ける

（3） 見出し候補の集合が残っていれば，初めに戻る

疑似コードによる表現は Algorithm 2 の通りである．ここで，入力の配列 A_{result} とは，整列された見出し候補の集合の配列である．また，H_e は，見出し候補の集合を一意に識別するキー (アルゴリズム中では A_{result} の添字を利用) から，その集合の要素をセパレータとして既に切られた記述の配列へのハッシュである．初期状態の H_e には，ページ全体の記述を表す特別なキーからドキュメントルートノードのみの配列への対が入っており，他のキーが指す配列の内容は空であるとする．

5.4 視覚的見出しの検出

本節では，各記述の視覚的見出しの検出について述べる．

まず前提として，視覚的見出しは全ての記述に付くはずなので，視覚的見出しを検出したい並び A と見出し候補の集合 S について，A 中の各記述に，S の要素のノードが 1 つずつ含まれる場合にのみ，S を A の**視覚的見出し候補の集合**として扱う．少なくとも A を切り分けるのに利用したセパレータ集合は条件を満たすので，記述には必ず視覚的見出しがつく．

一般に，視覚的見出しには記述を一意に識別する属性値が採

用されるので，視覚的見出しの集合を選ぶ第一の基準として，内容のユニークネスが利用できる．また，視覚的見出しには目立つという特徴もあるが，この特徴は周囲のノードとの関係により発揮され定式化が難しいため，今回は比較的明らかな基準である文字サイズのみを利用する．これらによって差がつかない場合，セパレータの特定のための整列と同様，各々の集合中で文書順が最小のノードを選び，その文書順の昇順に整列する．

視覚的見出し候補の集合の整列の基準を以下に整理する．

（1） テキストノードの集合について，内部のテキストがユニークなものや，IMG 要素の集合について，src 属性値がユニークなものを上位とする．

（2） ユニークネスに差がない場合，文書順で最初のノードの親ノードの font-size プロパティの降順に整列する．

（3） 他の基準に差がない場合,「集合中の全ノードの文書順のうち最小のもの」の昇順に整列する．

これらの基準によって最初に整列された視覚的見出し候補の集合を，注目する並び A の視覚的見出しの集合として採用する．

6. 評　　価

本章では，実際に多くの Web ページに提案手法を適用し，抽出された階層構造と人間が判断した階層構造を比較する．抽出された階層構造の評価基準は確立されておらず，ユーザ評価によっている研究 [3] も存在するが，本論文ではセパレータやその係り先，視覚的見出しの検出に注目した定量的な評価を行う．

6.1 評 価 基 準

以下の 4 つの基準により評価を行う．

- セパレータの検出の適合率，再現率
- 検出に成功したセパレータの係り先の特定の成功率
- 特定に成功した係り先の視覚的見出しの特定の成功率

評価は，Web ページから提案手法によって抽出された階層構造を，人間が記述を切り分け視覚的見出しを特定した階層構造と比較することによって行う．評価は初期状態で表示されるノードについてのみ行い，ポップアップや折り畳み式のメニュー，ドロップダウンボックスなどは評価に含めていない．

セパレータに関しては，並びの全ての記述が 1 つずつ備える同スタイルのノードを正解とする．ただし，そのようなノードを 1 つの記述について複数特定した場合，文書順で 2 つ目以降は誤りとする．正解のセパレータについては次に，係り先と実際の記述の範囲との一致を判定する．ただしこの際，記述の切れ目に配置され，どちらの記述にも含まれないノード，例えば HR 要素による罫線や区切り記号として使われるテキストノードについては，どちらの記述に含まれても成功とする．正解のセパレータの係り先の特定にも成功して得られた記述については，視覚的見出しの特定に成功しているかどうかを判定する．

6.2 データセット

様々な論理構造を持つ Web ページに対し，手法の有用性を評価するため，8 つのドメイン毎に 10 前後の Web ページを収集し，合計 79 ページを得た．この際，例えばポータルサイトであればディレクトリ型検索エンジン上のリストから収集し，検索結果ページであればクエリは excite サーチストリーム(注2)からランダムに取得するなど，偏りのないページが集まるよう配慮した．また，実体の一覧ページは前著 [6] と同様に,「岩石 一覧」をクエリとする Web 検索結果より，実際に実体の一覧のページであったもののうち上位 10 件を採用した．

表 1　ドメイン毎の結果及び総合的な結果

ドメイン名	N	$Precision_S$	$Recall_S$	$Accuracy_T$	$Accuracy_C$
ポータルトップ	8	0.95	0.93	0.96	0.93
Web 検索結果	9	0.85	0.91	0.92	0.86
百科事典の項目	12	0.84	0.86	0.87	0.88
Wiki	10	0.81	0.84	0.89	0.75
商品検索結果	11	0.77	0.90	0.91	0.94
Weblog	10	0.53	0.86	0.82	0.94
実体の一覧	10	0.47	0.80	0.74	0.96
掲示板	9	0.38	0.86	0.94	0.82
合計	79	0.70	0.87	0.88	0.89

N：ページ数

$Precision_S$：セパレータの検出の適合率

$Recall_S$：セパレータの検出の再現率

$Accuracy_T$：検出に成功したセパレータの係り先の特定の成功率

$Accuracy_C$：特定に成功した係り先の視覚的見出しの特定の成功率

6.3 結　　果

収集した 79 ページに関する評価の結果を表 1 に示す．各基準に関して，ドメイン毎のページ間平均を適合率の降順に示し，全ページ間平均を別に示した．

6.4 考　　察

セパレータとして利用できるノードの検出に関して，適合率のカテゴリ毎の差が大きく，再現率は全体的に高い．合計でも，適合率に対して再現率が高く，正解の階層構造に比べて並びを検出しすぎる傾向がある．ある記述が同スタイルのノードを複数含めば，それらは並びを形作ると見なす提案手法は，ページによっては記述を分割しすぎると言える．例えば，挿絵のある文章で，画像をセパレータとしてしまう場合がある．

また，本文の記述量の関係もある．本文とは，Web ページ中のヘッダやサイドバーなどを除く主要コンテンツである．カテゴリ毎に見ると，Weblog や掲示板など，本文の記述量が多く，内容が口語的なページで適合率が低い．この原因は本文の大部分である説明文の検出の失敗である．口語文では，句読点の欠落や，文中の強制改行がみられるが，これらの特徴は句点を含むテキストノードの割合を下げ，説明文の検出を失敗させる．その結果，各テキストノードが独立した記述とされてしまう．

また，実体の一覧のページでの適合率が低いのは，行が実体，列が属性に相当し，内容のほとんどのテキストが同スタイルの，フラットな表の存在によるところが大きい．これは各行と各列が共に並びとして直交しており，各列の並びの各記述は木構造上で連続でないという，提案手法が想定していない構造である．

一方，本文と呼べる部分が存在しないポータルトップページでは，セパレータの検出の適合率は高い．観察によれば，メニューやサイドバー等に対し，提案手法は強力に働く．モチ

(注2)：http://www.excite.co.jp/search_stream/

図 7　マージンと見出しの検出が相補的である例

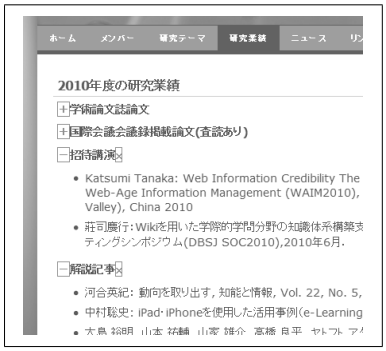

図 8　任意の記述を折り畳めるブラウザ

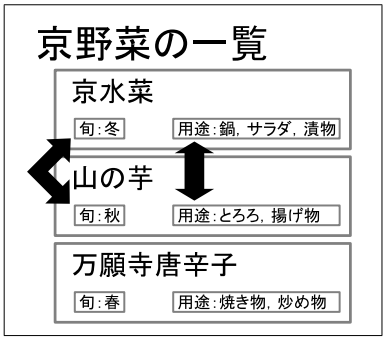

図 9　階層構造上のいとこの比較による属性の発見

ベーションである各記述の内部の論理構造がまちまちな並びの正しい分割も，本文に章節構造のある百科事典・Wiki や，ペインの並びであるポータルトップで確認できた．

係り先の特定については，全体的に成功率が高く，本研究が想定する記述の木構造上のモデル，遡る操作や辿る操作の有用性が示された．しかし，手書きのページが多いと思われる実体の一覧ページでは，成功率が低下している．手書きで作成されるページには，しばしば同スタイルでタグパスが異なるテキストが含まれる．提案手法の，手書きのページへの対応性は低い．

視覚的見出しの検出についても，全体的に成功率が高い．しかし実データにおいては，ある実体の名前と画像が併記されるなど，視覚的見出しの定義が曖昧な場合があり，考察を深めることが必要である．また，明らかな失敗の例として，LI 要素の先頭に付される印が，画像であり視覚的見出しである場合が散見された．HTML 文書については，LI 要素の点についても，セパレータや視覚的見出しとなる可能性を考慮すべきである．

7. おわりに

本論文では，Web ページの階層構造を抽出するための概念，具体的な手法，手法の評価について述べた．

今後の課題としては，Web ページ中で最も重要な本文に関する，セパレータの検出精度向上がある．実験では，ページの内容のうち，特に本文部分についてセパレータの適合率が落ちた．そこで，Weblog やニュース記事からの本文抽出の手法により，本文を他の部分と区別し，本文の特徴に沿って扱う手法があり得る．具体的には，本文としては小さすぎる記述への分割を閾値によって避ける方法や，見出し候補の集合の整列の時点で，文字サイズなど視覚的な情報を含める方法が考えられる．

他の課題として，表構造への対応がある．こちらは Web ページからの表形式情報の抽出手法が応用できる．表抽出の分野では，既に表全体の検出，テーブルヘッダの検出などが課題となっており，これらの手法を組み合わせ対応できる可能性が高い．

また，VIPS [3] のようにマージンを用いる手法と，見出しを用いる提案手法の併用も考えられる．図 7 に，マージンによる切り分けが有効な例と，見出しによる切り分けが有効な例を並べて示した．この例のように，2 種の手法は相補的関係にあり，併用することでより強力に階層構造を抽出できる可能性がある．

また，応用の考案と，それに沿った手法の発展も有効である．例えば，現在考えているアプリケーションとして，ユーザが任意の記述を折り畳めるブラウザ (図 8) があり，小画面の端末での Web ページの表示などに有用である．この応用においては，小さいサイズの記述はそもそも畳めるようにする必要がないので，分割が進みすぎることは問題となりにくい，

情報の利用において，ある実体に関する記述の一部を，同じ実体集合の他の実体に関する記述の一部と比較したい場合は多い．属性とは，比較への要求に基づく，記述の一部の間の対応付けであると言える．そこで，提案手法の知識発見や情報抽出への応用のために，提案手法により抽出された記述の階層構造について，実体と属性の二階層から成る構造を発見することが有用である．そのためには，図 9 のように，階層構造におけるいとこの関係にある記述を比較すればよい．対応付けの判断基準としては，前著 [6] で考察した通り，記述の均等な出現や，記述に前置されて冗長に出現する属性名などが利用できる．

その他の応用として，視覚的見出しの階層構造の抽出による Web ページの目次の自動生成とナビゲーションや，Web ページの論理構造を自動抽出し直感的な編集を可能にする編集システム，クエリ中の語の階層構造上の位置に基づいた Web ページの適合判定やランキングが挙げられる．

謝　辞

本研究の一部は科研費 (23650048) の助成を受けたものである。

文　献

[1] A. Arasu and H. Garcia-Molina: “Extracting structured data from web pages”, Proceedings of the 2003 ACM SIGMOD international conference on Management of data, pp. 337–348 (2003).

[2] G. Miao, J. Tatemura, W.-P. Hsiung, A. Sawires and L. E. Moser: “Extracting data records from the web using tag path clustering”, WWW, pp. 981–990 (2009).

[3] D. Cai, S. Yu, J.-R. Wen and W.-Y. Ma: “Vips: a vision-based page segmentation algorithm”, Technical Report MSR–TR–2003–79, Microsoft Technical Report (2005).

[4] S. Mukherjee, I. V. Ramakrishnan and A. Singh: “Bootstrapping semantic annotations for content-rich html documents”, ICDE, pp. 583–593 (2005).

[5] Y. Zhai and B. Liu: “Web data extraction based on partial tree alignment”, WWW, pp. 76–85 (2005).

[6] 真鍋, 田島 : “Web ページ中のノード間の論理的関係の発見”, 第 3 回データ工学と情報マネジメントに関するフォーラム (DEIM2011), pp. D2–5 (2011).