

Master Thesis

**Extraction of Category Structure of  
Web Sites using User Operation  
History**

Supervisor    Professor Toru ISHIDA

Department of Social Informatics  
Graduate School of Informatics  
Kyoto University

Motoyoshi WADA

February 8, 2007

## ユーザの操作履歴に基づく Web サイトのカテゴリ構造抽出

和田 基良

### 内容梗概

インターネット上に存在する Web サイトにおいて、コンテンツがカテゴリ毎に分類されているケースは数多く存在する。ここでカテゴリとは、同じ性質を持つものが含まれる集合である。カテゴリは Web ページ内の領域として表現され、名称を表すラベルを持つものとする。カテゴリ同士は階層関係を持つこともある。

Web サイト全体に対するカテゴリ階層構造を提示する事が出来れば、Web サイト全体像の把握が容易になり、目的のページへのアクセスが容易になる。またブラウジングにコストがかかる状況においては、アクセスへの障壁が低くなると考えられる。また、機械可読にするために意味を表すタグを付与する Semantic Web が提唱されており、今後はそのような機械可読なカテゴリ情報を持った Web サイトが増加してくることが予想される。しかし、現時点でそのようなタグ付けがなされた Web サイトは非常に少ない。このことから、Web サイト中で表現されているカテゴリの親子関係を抽出し、それを基に Web サイト全体のカテゴリ階層構造を獲得するような仕組みが必要である。

しかし、Web サイトからのカテゴリ構造の自動取得には以下のような問題点が存在する。

**HTML によるカテゴリ記述の曖昧さ** HTML で記述されたカテゴリの階層は、人間には判別できても機械には理解不可能である。HTML タグは Web ページのプレゼンテーションを表してもセマンティクスを表現しないためである。そのため、カテゴリが表す領域の位置やカテゴリ同士の親子関係を機械が自動的に判別することは不可能である。

**カテゴリ表現方法の多様性** Web サイト上におけるカテゴリやカテゴリ間の親子関係は、箇条書きで列挙されたり、単語を列挙したりと、様々な表現形式で示される。カテゴリのプレゼンテーション方法に明確な決まりがないためである。そのため、カテゴリ抽出において多種多様な Web サイトに対応することが難しい。

以上の問題点の解決策として、本研究では演繹学習を用いた以下の手順からなるアプローチを提案する。

1. Web ページ中に出現するカテゴリの親子関係に対し，人手で訓練例を与える
2. カテゴリの親子関係に対する解釈を行う
3. 解釈を与えた関係を一般化する
4. Web ページ全体を解釈する
5. Web サイト全体のカテゴリ構造木を構築する

Web ページ中におけるカテゴリの出現箇所およびそれらの親子関係を人間が解釈し，システムに教示する．これによって，システムは様々な表現形式で形成された Web ページに合わせてカテゴリ関係を解釈できる．システムは領域理論を用いて，教示のあったカテゴリの親子関係に対する解釈を与える．また，解釈が与えられた概念を一般化することで，より多くのカテゴリ関係を獲得することができる．一般化されたカテゴリ関係に当てはまる箇所を Web ページ中から探索することで，まだ発見されていない子カテゴリを獲得できる．これらの手続きを，Web サイトを構成する Web ページ群に対してそれぞれ行う事により，最終的には Web サイト全体に対するカテゴリ階層構造が得られる．

カテゴリの解釈に人手を使うと同時に，演繹学習を用いることで，人手で訓練例を与える際のコストを削減できる．

本研究の貢献は以下の通りである．

カテゴリと HTML の対応付け 人手によって訓練例を与えることで，HTML によるプレゼンテーションとカテゴリの階層とを一般化して対応付けることが出来る．これによって，HTML によるカテゴリのプレゼンテーションに一貫性がある Web サイトにおいて，カテゴリ構造の抽出が可能になる．幅広い表現方法に対応可能 提案手法では，人手によってカテゴリ及びカテゴリ同士の親子関係を与えている．そのため，出現位置やタグ付けなどに関して，幅広い表現方法でカテゴリが記述された Web サイトに対応することが可能である．

本研究の応用として，視覚障害者の方がスクリーンリーダーを用いて Web サイトを閲覧するケースが考えられる．あらかじめサイトマップ等が用意されていない Web サイトを閲覧する場合，全体構造の把握が困難となってしまう．カテゴリ構造を提示することでこれらのユーザにも Web サイトへの容易なアクセス手段を提供することは，デジタルデバイドの解消といった観点からも社会的に必要である．

# Extraction of Category Structure of Web Sites using User Operation History

Motoyoshi WADA

## Abstract

There are a lot of cases where contents of web sites on the Internet are classified by category. In this paper, a category means a set where an element with the same character is included. A category is expressed as a domain in a web page. A category has a label that shows its name. Two or more categories might have a hierarchical relation.

If we can show the layered category structure of whole web sites, we can figure out the whole image of them and it becomes easy to access the target pages that we want to see. And in case browsing is costly, the layered category structure enables us to remove barriers to access them. The Semantic Web, method of giving a tag that shows the semantics, is advocated in order to keep web sites machine-readable. And web sites with such machine-readable category information are expected to increase in the future. However, at present, the number of the web sites tagged in this way is very few. Thus, a system that extracts the parent-child relationship of categories in websites and constructs the layered category structure is required.

However, it is difficult to extract categories from a website. The research issues are as follows.

**Ambiguity of categories described with HTML** Human beings can recognize the hierarchy of categories, but machines cannot. This is because HTML tags represent presentation of web pages, but they do not express semantics. Machines cannot distinguish the areas of categories and parent-child relation of categories automatically.

**Diversity of expression of categories** Categories and the parent-child relationship between two categories are expressed in various ways, because there is no concrete rule to describe categories. Thus, it is difficult to correspond to various web sites in extracting categories.

To solve these issues, this paper proposes the following method using deductive learning.

1. Give training examples of parent-child relations of categories manually
2. Construe parent-child relations of categories
3. Generalize relations for which constructions are given
4. Interpret whole web pages
5. Build the category structural tree of web sites entirely

People construe the parts of categories appearing in web pages, and the parent-child relationship of categories, and they teach the system. So, the system can construe the parent-child relationship of categories in various ways of expression. By generalizing the concept for which construction is given, we can get more parent-child relationships. We can acquire the child-categories by searching the part applicable to the generalized category relations. We can extract the layered category structure of a web site by applying these procedures to web pages which constitute a web site.

By using deductive learning, the cost of giving training examples manually can be reduced.

The contributions of this work are as follows:

**Correlation of categories with HTML** The system can correlate categories with the presentation of HTML by using training examples, and generalize that correlation. Thus, we are able to acquire the structure of categories from a website where presentations of categories have consistency.

**Application to a wide range of means of expression** In the proposed technique, a position of a category and a parent-child relationship of categories are given by a user. Therefore, it is possible to deal with the web site where categories are represented with a wide range of expressions about an appearing position, a tagging, etc.

The application of this study is to solve serious problem with browsing web sites. For example, people with visual impairment would use a screen reader while browsing web sites. If they browse web sites where site maps are not prepared beforehand, it is difficult to understand their overall structure. We can provide easy access to web sites by displaying the category structure. So, this research is socially important from the viewpoint of bridging the digital divide.

# Extraction of Category Structure of Web Sites using User Operation History

## Contents

<b>Chapter 1</b>	<b>Introduction</b>	<b>1</b>
<b>Chapter 2</b>	<b>Observation of Category Structure</b>	<b>7</b>
2.1	Features of Category Expression . . . . .	7
2.2	Class of Layered Structure . . . . .	8
<b>Chapter 3</b>	<b>Formalization</b>	<b>11</b>
3.1	Category Structural Tree . . . . .	12
3.2	Training Example . . . . .	13
3.3	Constraints . . . . .	14
3.4	Predicates . . . . .	14
3.5	Rules . . . . .	15
3.5.1	Given Rules . . . . .	16
3.5.2	Auto-Generated Rules . . . . .	17
<b>Chapter 4</b>	<b>Algorithm</b>	<b>20</b>
4.1	Relation to Deductive Learning . . . . .	20
4.2	Generation of Proof Tree . . . . .	22
4.3	Generalization of Relation . . . . .	25
4.4	Generation of Concept Description . . . . .	26
4.5	Interpretation of Web Pages . . . . .	27
4.6	Generation of Category Structural Tree . . . . .	29
<b>Chapter 5</b>	<b>Discussion</b>	<b>30</b>
5.1	Result of execution . . . . .	30
5.2	Applicability of proposed technique . . . . .	33
<b>Chapter 6</b>	<b>Related Works</b>	<b>36</b>
<b>Chapter 7</b>	<b>Application</b>	<b>38</b>
7.1	Screen Reader . . . . .	38
7.2	Mobile Terminal . . . . .	38

<b>Chapter 8</b>	<b>Conclusion</b>	<b>42</b>
	<b>Acknowledgments</b>	<b>45</b>
	<b>References</b>	<b>46</b>

# Chapter 1 Introduction

Today, there are many kinds of web sites on the Internet. Various contents are published in web sites, and there are a lot of classification ways. As a demarche of arranging contents, a lot of web sites classify the contents by the category.

In this paper, a category means a set where elements with the same character are included. A category is expressed as a domain in a web page. A category has a label that shows its name. Two or more categories might have a hierarchical relation. We call a superordinate category a “parent category”, and subordinate one a “child category”. An example of categories is shown in Figure 1. In this example, “main unit of personal computer” is a parent category, and “desktop personal computer” corresponds to a child category. These two categories have labels that represent their names.

If a label of a category is tagged with an HTML link tag, the area of a category is a whole web page to which that link refers. An example is shown in

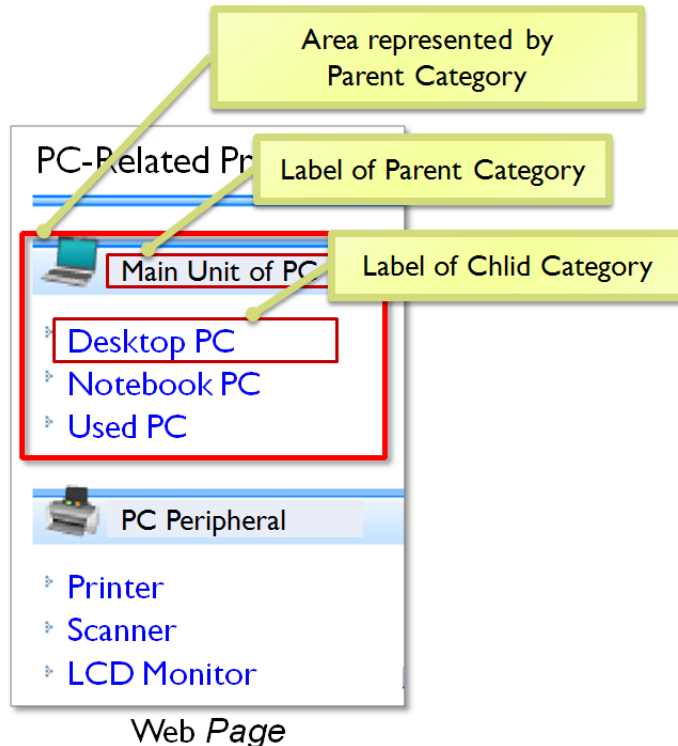


Figure 1: Example of a category in a web page



Figure 2: Area represented by a category with a link label

Figure 2. This figure shows that there is a label of the category “パソコン関連 (PC-Related products)” in the top page of a web site, and that label is a link to another web page. In this case, that linked web page represents the content described in the label of the link. In other words, the area of the category “パソコン関連” is the entire web page that is linked from the label.

However, the hierarchy of the category does not necessarily correspond to the structure of the link. For instance, on the one hand the web page and the category corresponds one to one, and they might be connected by a hyperlink mutually. On the other hand, there are cases where the label of a category is described as plain text. That is, a logical category structure does not necessarily correspond to the link structure. Therefore, if a category is acquired and a category structural tree is generated paying attention only to the link structure,



Figure 3: Example of a web site (Kakaku.com)

various problems will happen.

We will explain with the web site shown in Figure 3. This figure shows that the web page “パソコン関連” is linked from the top page of “kakaku.com<sup>1)</sup>”, a web site about the price of consumer electronics, personal computers, etc. We attempt to extract the parent-child relationship of the categories from such a web site. In figure 4, we try to generate a category structural tree by considering the link to another page as the label of the category. However, we cannot recognize categories with labels that are not tagged with anchor tags like “パソコン関連” and “パソコン周辺機器 (PC peripheral devices)”. Advertisements(広告) etc. included in the web page are recognized as categories because they are

<sup>1)</sup> <http://kakaku.com/>

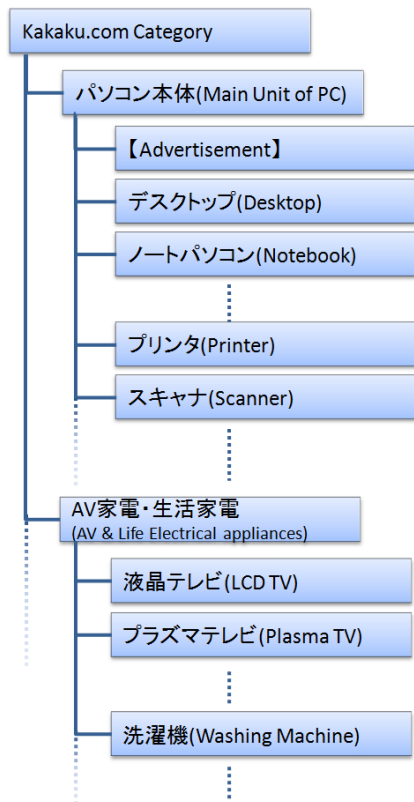


Figure 4: Category hierarchy generated only from link structure

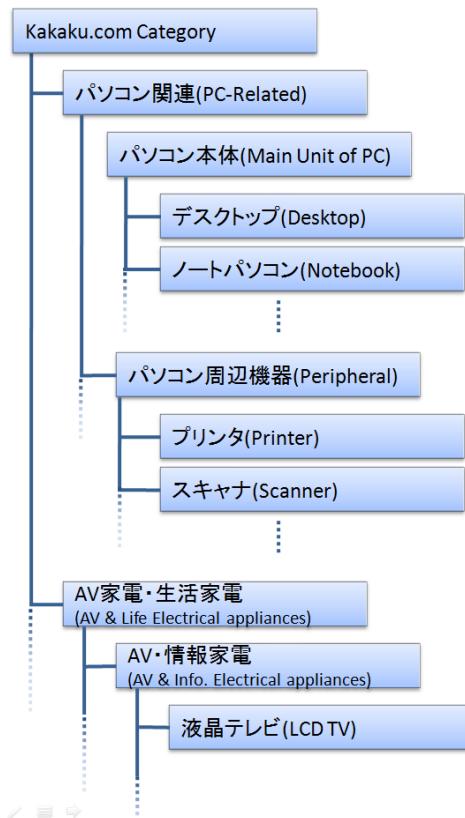


Figure 5: Category hierarchy in consideration of plain text label

tagged as a link. Thus, we can say that acquisition of the category-layered structure from the web site is a complicated problem.

Figure 5 is an example of paying attention to the category with regard to not only the link structure but also the label of the category described by a plain text. In order to achieve this, it is necessary to extract the category with the plain text label in the web page.

If the semantic structure of the whole website can be shown, understanding the global image of the website will be easy and the barrier to access will be lowered. When visually impaired persons browse a website, software known as a screen reader or a voice browser is used. A screen reader reads out a web page by a synthetic voice. In order for this software to work optimally, the producer of a website has to build a site with careful attention to accessibility.

Regrettably, however, not all the websites are built with careful attention to

accessibility by a screen reader. For example, the site map etc. is not prepared beforehand in a website where accessibility is not considered. When browsing such a website with a screen reader, getting a grip on the structure of the whole website will be difficult. From the viewpoint of dissolution of the digital divide, it is socially needed to provide these users with easy means of access websites.

The Semantic Web, method of giving a tag that shows the semantics, is advocated in order to keep web sites machine-readable. And web sites with such machine-readable category information are expected to increase in the future. However, at present, the number of the web sites tagged in this way is very few.

Consequently, a mechanism is required in which category relations are extracted from a web page and the category-layered structure of the whole website is gained from category relations. Previous works in this area are information extraction and web wrapper[1]. There are various kinds of web wrapper. For example, those which use the structure of a HTML tag[2] and those which perform inductive learning[3], etc.

However, problems still exist in automatic acquisition of the category structure from a website. The research issues are as follows.

**Ambiguity of categories described with HTML** Human beings can recognize the hierarchy of categories, but machines cannot. This is because HTML tags represent presentation of web pages but they do not express semantics. Machines cannot distinguish the areas of categories and parent-child relation of categories automatically.

**Diversity of expression of categories** Categories and the parent-child relationship between two categories are expressed in various ways, because there is no concrete rule to describe categories. Thus, it is difficult to correspond to various web sites in extracting categories.

To solve these issues, this paper proposes the following method using deductive learning.

1. Give training examples of parent-child relations of categories manually
2. Construe parent-child relations of categories
3. Generalize relations for which constructions are given

4. Interpret whole web pages
5. Build the category structural tree of web sites entirely

People construe the parts of categories appearing in web pages, and the parent-child relationship of categories, and they teach the system. So, the system can construe the parent-child relationship of categories in various ways of expression. By generalizing the concept for which construction is given, we can get more parent-child relationships. We can acquire the child-categories by searching the part applicable to the generalized category relations. We can extract the layered category structure of a web site by applying these procedures to web pages which constitute a web site.

By using deductive learning, the cost of giving training examples manually can be reduced.

This paper is organized as follows: Chapter 2 describes the observation of categories and category relation in web sites. In Chapter 3, formalization of category structures is presented. An algorithm for acquisition of category relation from a web site is given in Chapter 4. In Chapter 5, we discuss cases where we apply this proposed technique to actual web sites and the applicability of the proposal. Related works are described in Chapter 6. In Chapter 7, application of the category structural tree is presented, followed by the conclusion in Chapter 8.

## Chapter 2 Observation of Category Structure

We try to extract parent-child relations of categories from each web page which constitutes a website, and finally, acquire the category-layered structure of the whole website. So, if a parent-child relation of categories is generalized, it will be thought that a new relation can be gained from more parts.

Therefore, it is necessary to generalize category structure based on what kind of feature parent-child relations of categories are determined by, and to define a formal expression of a parent-child relationship of categories.

In advance of this generalization, this chapter describes the observation of the features of category expression which appear in a website, and the features of the category structure by a parent-child relationship of categories.

### 2.1 Features of Category Expression

HTML - hyper-text markup language - is the language which shows presentation. So it is not a language expressing semantics, strict restrictions or rules when expressing a category in a web page do not exist. Various patterns may exist in the mode of expression of a category.

However, a loose common feature of expression of categories exists. Generally, the label of a child category appears under the label of a parent category, and a parent category and child categories have a one-to-many relationship. The names of categories in the same hierarchy have a synonymous mutual re-

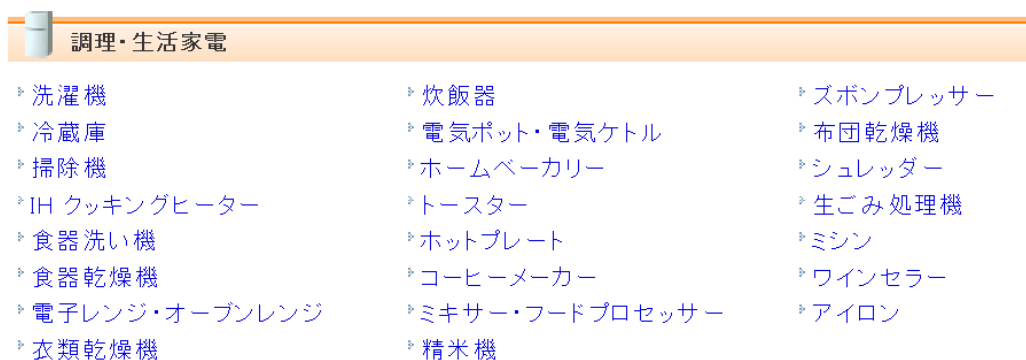


Figure 6: Example of category expression



Figure 7: Category expression using a label surrounded by an anchor / a link tag

lation in many cases. An example is shown in figure 6. Under the label of the parent category “調理・生活家電 (cooking / household appliances)”, there are the labels of two or more child categories called “洗濯機 (washing machine)”, “冷蔵庫 (refrigerator)”, and “掃除機 (vacuum cleaner)”. And the names of these child categories have a synonymous relationship respectively.

## 2.2 Class of Layered Structure

When expressing a layered structure like a parent-child relationship of categories with HTML, there are two ways: one is expression using a label surrounded by an anchor tag. The other is the expression that uses a label written in plain text.

### Category expression using a label surrounded by an anchor tag

Figure 7 is an example of a web page that has labels of categories surrounded by anchor/link tags. The whole page expresses the category “プリンタ (printer)”, and the label of the category “printer” links the area

すべて
  このカテゴリ
 
[ショップ検索](#)
[検索のヒント](#)

- PR [せっかく撮ったら写真にしよう！オンラインプリント料金比較！](#)  
 PR [1.2Mでホントにこの価格？魅力を増したADSLサービス](#)

 カメラ本体

- |           |                           |             |
|-----------|---------------------------|-------------|
| ▷ デジタルカメラ | ▷ デジタル一眼レフカメラ <b>NEW!</b> | ▷ ビデオカメラ    |
| ▷ 一眼レフカメラ | ▷ コンパクトカメラ                | ▷ インスタントカメラ |
| ▷ WEBカメラ  | ▷ MPEGムービー                |             |

 カメラアクセサリ

- |         |             |         |
|---------|-------------|---------|
| ▷ レンズ   | ▷ フラッシュ     | ▷ 防水ケース |
| ▷ 三脚・一脚 | ▷ カメラ バッテリー |         |

 カメラ関連製品

- |            |             |            |
|------------|-------------|------------|
| ▷ フォトストレージ | ▷ カードリーダー   | ▷ フィルムスキャナ |
| ▷ プリンター    | ▷ グラフィックソフト | ▷ 防湿庫      |
| ▷ 双眼鏡      |             |            |

Figure 8: Category expression using a plain text label

showing the category “プリンタメーカー (printer manufacturer)”. The category “プリンタメーカー (printer manufacturer)” is a child category of the category “プリンタ (printer)”.

### Category expression using a plain text label

Figure 8 shows an example of a web page that contains the child categories the label of which is expressed with plain text. A whole web page is an area showing the category “カメラ (camera)”. At the same time, this web page contains the labels of child categories, such as “カメラ本体 (main part of a camera)”, “カメラアクセサリ (camera accessories)”, and “カメラ関連製品 (camera-related products)”. All the labels of these child categories are expressed by plain text.

In this case, extraction of a label is more difficult than in the case of a label with a link tag, because there is little information with which to search the website for the position of the label. In this research, we employ supervised learning. A user shows the position of the category label in relation to the

system. This method can cope with a wider range of domains than that of using a dictionary to find the label.

Based on the above observation, a generalized formal expression which expresses the parent-child relationship between categories is defined in Chapter 3.

## Chapter 3 Formalization

In advance of the definition of the algorithm in Chapter 4, we will formalize the parent-child relationship of categories and the hierarchical structure of categories which appear in websites. In addition, definitions of the training example, predicate, and rule which are used by deductive learning are given.

From here onward, when processing a web page, we will not use the rendering result by a browser, but instead, the source described by HTML.

By way of explanation, the HTML source of figure 3 is simplified, translated in English and shown as figure 9.

```
<html>
  <head>
    :
    <title>PC-Related Product</title>
  </head>
  <body>
    :
    <a href="...">Advertisement</a>
    <div>Main Unit of PC</div>
    <a href="...">Desktop PC</a>
    <a href="...">Notebook PC</a>
    <a href="...">Used PC</a>
    <div>PC Peripheral Device</div>
    <a href="...">Printer</a>
    <a href="...">Scanner</a>
    :
  </body>
</html>
```

Figure 9: Simplified HTML source of figure 3

### 3.1 Category Structural Tree

We define the category hierarchical structure of a website of the object of processing as the tree structure. Hereafter, we call it a category structural tree. Each node in the category structural tree corresponds with the category which appears in a website, respectively.

The formal definition of the category structural tree is shown below. Node *node* in the category structure tree is defined as 4-tuples as follows.

$$node = (symbol, parent, label, area)$$

- *symbol* means a symbol used in the rule (mentioned later)
- *parent* means a pointer to the parent node.
- *label* means a character string indicating the name of a category and position information of a label of a category.

If *node* has no parents, variable *parent* is set to *null*. Furthermore, *label* is expressed by the following 5-tuple.

$$label = (text, URL, XPath, tag, pos)$$

- *text* means a character string showing the name of the category.
- *URL* means the URL of the web page in which the name of the category is described.
- *XPath* means the position of the tag which appears just before a category label. *XPath* is described with a notation of the XPath style.
- *tag* means an HTML tag by which the label of the category is surrounded.
- *pos* means a position of the category's label in an HTML source of the web page. It means the number of tags emerging before *tag* in the HTML source.

*area* means the restrictions for specifying the area of the category on the web page, and is defined as follows.

$$area = (URL, d_L, d_R)$$

- *URL* means a URL in which the area that the category expresses exists.

- $d_L, d_R$  means the character string used as a delimiter for pinpointing the area which the category expresses.

The category structural tree showing a hierarchical structure of categories is expressed as a set of *node* defined as mentioned above.

### 3.2 Training Example

In this proposed system, operation records and instructions by users are used as the training example for learning. In particular, a user selects a portion of a label of a parent category in the web page for processing, then shows that the category selected is the parent category to a system. The user also indicates to a system about a child category and a grandchild category by the same method.

The system acquires the training examples, namely, the portions showing the label of categories and parent-child relationships of categories presented by the user. The system stores them in each *node*.

According to a target of the user's selection, two kinds of instruction by the user exist. Namely, cases where a label written with plain text is chosen and cases where a label tagged by a link tag is chosen. When a plain-text label is chosen, what is necessary is just to record the appearance position of the category label and the name of the category described with it. On the other hand, if a link label is chosen, it is additionally necessary to store the information of a web page at the end of that link of the label, i.e. information about the area that the category shows.

Therefore, the training example by a user is formalized according to the selection target as follows.

$$Select_u = (label, rank)$$

or

$$Select_u = (label, rank, URL_{dest})$$

- *label* means information about the label of the range chosen by a user.
- *rank* means hierarchy level (a parent category, a child category, etc.) of a category.
- $URL_{dest}$  means the URL of a target web page of the link.

The former is an instruction when a category label is described with plain text, and the latter is an instruction when a category label is marked up with an anchor tag. When a label marked up with an anchor tag is selected, information about a target web page of the link -  $URL_{dest}$  - is also recorded.

### 3.3 Constraints

As restrictions which tie up two categories that have a parent-child relationship, we propose Syntactic Constraint and Semantic Constraint.

#### Syntactic Constraint

Syntactic Constraint is to be the restriction which is defined based on the structure of the HTML source: for example, if there are categories of the same hierarchy level, they are surrounded by a tag of the same kind. The label of the parent category exists in a higher part of the HTML source than the label of the child category. If a category is in the same hierarchy level as the category that is in the training example by a user, those two categories appear in the neighborhood with each other in the HTML source.

#### Semantic Constraint

Semantic Constraint refers to the restrictions using the semantic relation between words. If two or more categories are in the same hierarchical level, their names often have a mutual synonymous relation. Semantic Constraint is based on this feature.

Synonym search is used for checking whether Semantic Constraint is fulfilled. Specifically, we use a web Service which inputs a certain keyword and returns a set of the synonyms to the keyword. Google Sets<sup>1)</sup>, etc. exist as such services.

### 3.4 Predicates

In this subsection, predicates for describing rules are defined based on the restrictions mentioned in section 3.3.

Below,  $A$  and  $B$  shall express categories, unless otherwise indicated.

---

<sup>1)</sup> <http://labs.google.com/sets/>

*CategoryHierarchy(A, B)*

$A$  and  $B$  are in a parent-child category relationship, that is,  $A$  is a parent category and  $B$  is a child category.

*SyntacticConstraint(A, B)*

$A$  and  $B$  are in a parent-child relationship in syntax.

*SemanticConstraint(A, B)*

$A$  and  $B$  are in a parent-child relationship semantically.

*Above(A, B)*

The label of  $A$  appears higher than that of  $B$  in the HTML source.

*SurroundedByTag(A, B)*

The label of  $A$  and  $B$  is marked up by some tag in the HTML source.

*SurroundedByXTag(A)*

The label of  $A$  is marked up by  $\langle X \rangle$  tag in the HTML source. It is a predicate used in the template mentioned later.

*Neighbor(A, B)*

The appearance position of the labels of  $A$  and  $B$  are close together in the HTML source.

*Synonym(A, B)*

The labels of  $A$  and  $B$  have a mutual synonymous relationship.

### 3.5 Rules

In this research, a rule is used as domain knowledge explaining a parent-child relationship of categories in web sites and web pages. A rule is used in order to express correspondence with a description of a parent-child relationship in a web page and a parent-child relationship itself.

Presupposed information about a domain is used for the rules for reasoning. The following are of the presupposed information about a web page which includes a parent-child relationship of categories.

- A label of a child category appears that of a parent.
- A parent category and child categories are in one-to-many relationship.
- Labels of categories in the same hierarchy are surrounded by the same tag.
- Names of child categories in the same hierarchy are in synonym relationship

each other.

Based on these, we create the rules to learn a goal concept. Rules are expressed by a Horn clause.

### 3.5.1 Given Rules

Rules and their explanations are shown in figure 10 and 11. These are prepared beforehand in the system side.

*CategoryHierarchy(A, B)*

$\leftarrow \text{SyntacticConstraint}(A, B) \vee \text{SemanticConstraint}(A, B)$

If satisfying a Syntactic Constraint or a Semantic Constraint,  $A$  and  $B$  are in a parent-child relationship of categories.

*SyntacticConstraint(A, B)*

$\leftarrow \text{Above}(A, B) \wedge \text{SurroundedByTag}(A, B) \wedge \text{Neighbor}(A, B)$

$A$  and  $B$  satisfy a Syntactic Constraint if the label of  $A$  and  $B$  are surrounded with some kind of tag, the labels of  $A$  and  $B$  are in nearby positions, and the label of  $A$  appears higher than that of  $B$ .

*SemanticConstraint(A, B)*

$\leftarrow \text{Synonym}(A, B)$

Semantic restrictions are satisfied if the labels of  $A$  and  $B$  have a synonymous relation with each other.

Figure 10: Given rules

*Above*( $A, B$ )

$\leftarrow pos(A) < pos(B)$

Category  $A$  appears above category  $B$  if  $pos(A)$  is lower than  $pos(B)$ .

*Neighbor*( $A, B$ )

$\leftarrow |pos(A) - pos(B)| \leq DISTANCE$

If the distance between the labels of  $A$  and  $B$  is less than  $DISTANCE$ , Category  $A$  and  $B$  are in the same neighborhood.  $DISTANCE$  is a constant indicating the threshold whether two labels are in the same neighborhood or not.

Figure 11: Given rules (continuation of figure 10)

### 3.5.2 Auto-Generated Rules

There are various kinds of HTML tags that can mark up a label of a category, and the number of rules containing the predicate about an HTML tag becomes huge. Therefore, it is useful not to prepare all rules beforehand, but to build based on a template.

For this reason, we use a method of generating a rule automatically based on a template. In this paper, we define a template as being that which has generalized or abstracted the part of the predicate which is contained in the rules. Rules are automatically generated by replacing the abstracted predicate with two or more concrete predicates.

For example, think about templates as follows:

$$\begin{aligned}
& \textit{SurroundedByTag}(A, B) \\
& \leftarrow \textit{TEMPLATE\_SurroundedByXTag}(A) \\
& \wedge \textit{TEMPLATE\_SurroundedByYTag}(B)
\end{aligned}$$

The predicate  $\textit{TEMPLATE\_SurroundedByXTag}(A)$  means that the category  $A$  is marked up by a tag  $\langle X \rangle$  in HTML. This template serves as a basis of the rule that expresses the tags by which category  $A$  and  $B$  are marked up. Materialization of a template is realized by replacing a generalized predicate with a concrete one. In this case,  $\textit{TEMPLATE\_SurroundedByXTag}(A)$  and  $\textit{TEMPLATE\_SurroundedByYTag}(B)$  that were generalized predicates are replaced with concrete predicates. Because they are candidates for replacement with concrete predicates when generating rules from a template, they are distinguished from other predicates by attaching the prefix  $\textit{TEMPLATE\_}$ .

Because there are many kinds of HTML tags that can mark up a label of category  $A$ , the number of combinations of tags, in other words, the number of the rules, becomes enormous. Supposing a markup is made only by two kinds of

$$\begin{aligned}
& \textit{SurroundedByTag}(A, B) \\
& \leftarrow \textit{SurroundedByPTag}(A) \wedge \textit{SurroundedByPTag}(B) \\
& \textit{SurroundedByTag}(A, B) \\
& \leftarrow \textit{SurroundedByPTag}(A) \wedge \textit{SurroundedByH1Tag}(B) \\
& \textit{SurroundedByTag}(A, B) \\
& \leftarrow \textit{SurroundedByH1Tag}(A) \wedge \textit{SurroundedByPTag}(B) \\
& \textit{SurroundedByTag}(A, B) \\
& \leftarrow \textit{SurroundedByH1Tag}(A) \wedge \textit{SurroundedByH1Tag}(B)
\end{aligned}$$

Figure 12: Rules automatically generated from a template

tag `<p>` and `<h1>` , four rules will be automatically generated from a template as in table 12.

A lot of rules are generated by this procedure. Generally, among the predicates that appear in one template, the number of rules generated will automatically become  $m^n$ , where  $m$  is the number of candidates for substitution and  $n$  is the number of the predicates that will be substituted.

However, in this case, the number of predicates to be substituted which appear in this template is at most two. In other words,  $n \leq 2$ . Moreover, rules used in the reasoning in deductive learning are only ones about HTML tags contained in the training example given by the user. That is, rules about HTML tags which were not concerned with the user's training examples are not used in fact. Therefore, problems such as execution speed of reasoning falling down do not arise.

## Chapter 4 Algorithm

In this research, acquisition of the category structural tree of a website is the final purpose. For that purpose, a procedure is required which learns a parent-child relationship of categories based on instructions of a user. In order to lessen the need for instructions by a user, deductive learning is used for acquisition of child categories in this research.

This chapter describes the algorithm of acquisition. The outline of the algorithm is as follows. First, training examples are generated based on instructions from a user. Based on training examples and rules, proof of the target concept known as a proof tree is generated. Then, the system searches the target web page for parent-child relationships that are not in instructions from the user. We get parent-child relationships of categories and a category structural tree. These operations are done for every web page in the website. Finally, a hierarchical category structure of the whole website based on parent-child relationships is gained.

Hereafter, we explain the details of the algorithm.

### 4.1 Relation to Deductive Learning

In this research, deductive learning is used for acquisition of child categories in a target web page.

Deductive learning is a kind of machine learning, and aims at learning the **target concept**. That is, the aim of learning is to be able to distinguish between the positive examples and the negative examples of the target concept when a certain concept is given to a learning system. The inputs of deductive learning are a target concept, a training example, domain theory, and operability criterion. The output of deductive learning is a concept description.

A **training example** is a positive example over the target concept, and is expressed with the form of logical expression.

A **domain theory** is the knowledge which is necessary when proving the target concept from the training example. We already have defined the domain theory as the rules in section 3.5.

Table 1: Correspondence between deductive learning and this research

<b>Deductive Learning</b>	<b>This Research</b>
target concept	$CategoryHierarchy(A, B)$ $B$ is a child category of $A$ .
training example	parent-child relationship
domain theory	rules described in section 3.5
operationality criterion	The main part of the target concept must be described by the fact clause.

A **concept description** is description of the target concept expressed by logical expression.

An **operationality** is a measure indicating how the problem solving is made efficient by the learned concept description. The conditions that the concept description should fulfill to raise operationality are called **operationality criterion**. In this research, the operationality criterion is “the main part of the target concept must be described by the fact clause”. A fact clause is main part of a Horn clause.

Deductive learning is supervised learning. A user who is a teacher gives training examples that are positive examples. Based on the training examples and the domain theory, the learning system derives the target concept. This is called **explanation**.

Based on the training examples and the domain theory, the data of the tree structure expressing how the target concept is proven is called a **proof tree**. Because this explanation is for the training examples, the learning system generalizes the obtained explanation so that it can also infer other inputs.

Deductive learning has the feature that it is possible to learn from a small number of the training examples, as compared with inductive learning which is also supervised learning.

In this research, a user as a teacher gives labels of categories and their hierarchy level in the target web page as training examples. The system applies the rules that are the domain theory to the training examples, and gives an expla-

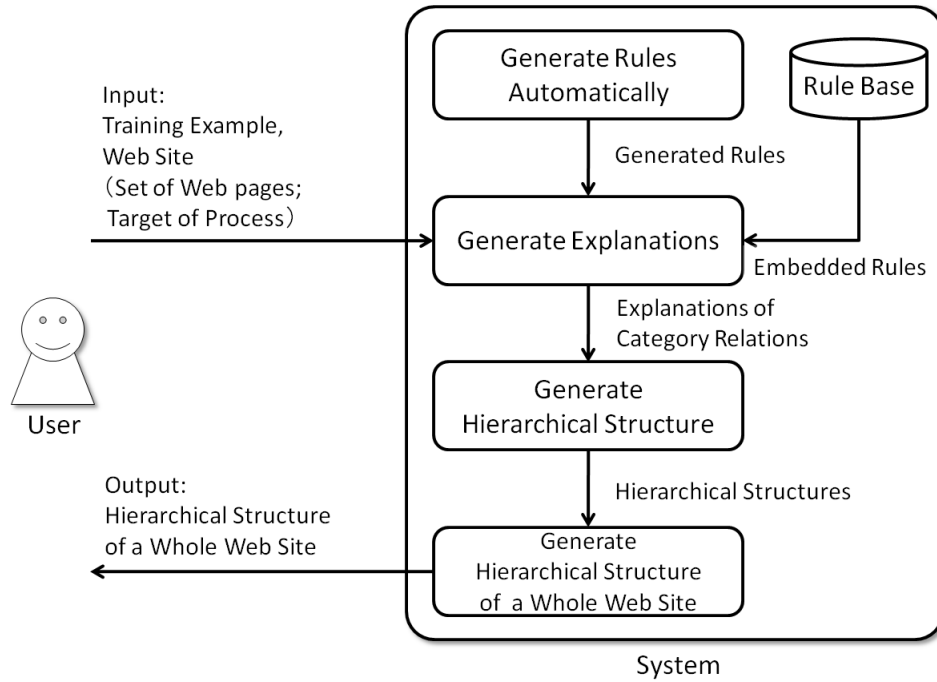


Figure 13: Overview of the system

nation. Then, the explanation is generalized using the rules. The generalized explanation is applied to the web page in order to search the portion to which the interpretation is not given for child categories.

Correspondence between deductive learning and this research is shown in table 1.

An overview of the proposed system is shown in Figure 13.

## 4.2 Generation of Proof Tree

Based on the rules as the domain theory and the training examples from a user, the proposed system forms a proof tree. The root of that tree is the target concept.

The proof tree gives proof of two categories instructed by a user in a parent-child relationship.

Before generating the proof tree, the training examples given by a user are changed into nodes of the category structural tree.

The hierarchical relationship of the three categories “PC-Related Products”, “Main Unit of PC”, and “Desktop PC” is taken up as a training example this time, and the symbols  $C_1$ ,  $C_2$  and  $C_3$  are given to these three categories respectively. The system searches for the area that nodes in the proof tree express, by means of looking for a similar part proposed in [2].

From the HTML source in figure 9,  $node_1, node_2, node_3$ , which represent categories “PC-Related Products”, “Main Unit of PC”, and “Desktop PC” respectively are expressed as follows:

$$\begin{aligned}
 node_1 &= (C_1, null, label_0, area_1) \\
 label_1 &= (“PC-Related Products”,  $URL_1$ , /html/title, title, 3) \\
 \\ \\
 node_2 &= (C_2, node_1, label_2, area_2) \\
 label_2 &= (“Main Unit of PC”,  $URL_2$ , /html/body/div, div, 7) \\
 \\ \\
 node_3 &= (C_3, node_2, label_3, area_3) \\
 label_3 &= (“Desktop PC”,  $URL_3$ , /html/body/a, a, 8)
 \end{aligned}$$

The tree structure from the training examples is shown in figure 14. A circle in the figure expresses a parent category, a child category, or a grandchild category. A node drawn with a solid line means a category contained in the training example, and is already known to the system. A node drawn with a dashed line means a category unknown for the system, because a training example of that category is not given by a user, although that category actually exists in the target web page.

Then, the system changes these nodes into predicates in order to be able to apply the rules. For instance, the label of the category “Main Unit of PC” is marked up by the `<div>` tag. Therefore, a predicate is generated as follows:

$$SurroundedByDivTag(C_2)$$

Similarly, since the label of the category “Desktop PC” is surrounded by the

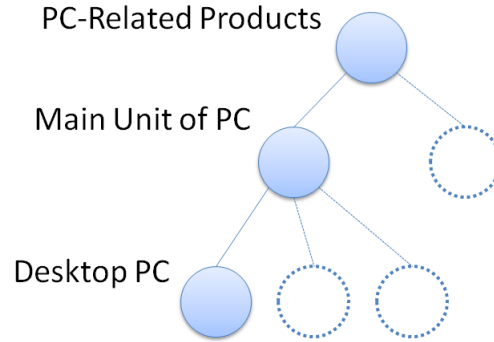


Figure 14: Tree structure representing the training examples

<a> tag, a predicate is created as below:

$$SurroundedByATag(C_3)$$

Moreover, from the appearance position of two labels of the categories, two predicates are generated as follows:

$$pos(C_2) < pos(C_3)$$

$$|pos(C_2) - pos(C_3)| < DISTANCE$$

The conjunction of these predicates is the training example in generating the explanation with the application of rules.

Based on the training examples for two categories, the rules are applied until the target concept is derived. There are two kinds of rules as mentioned in section 3.5: the fixed ones prepared for the system beforehand, and the ones automatically generated from templates. Both kinds of rules are used.

As a result, the proof tree as shown in figure 15 is obtained. Elements in the lowest row in that figure mean predicates included in the training example. The element in the highest position of the figure, *CategoryHierarchy*( $C_2, C_3$ ), means the target concept. An arrow expresses the direction of application of a rule. The figure shows that the rules are applied sequentially from the training example, and finally the target concept is proven.

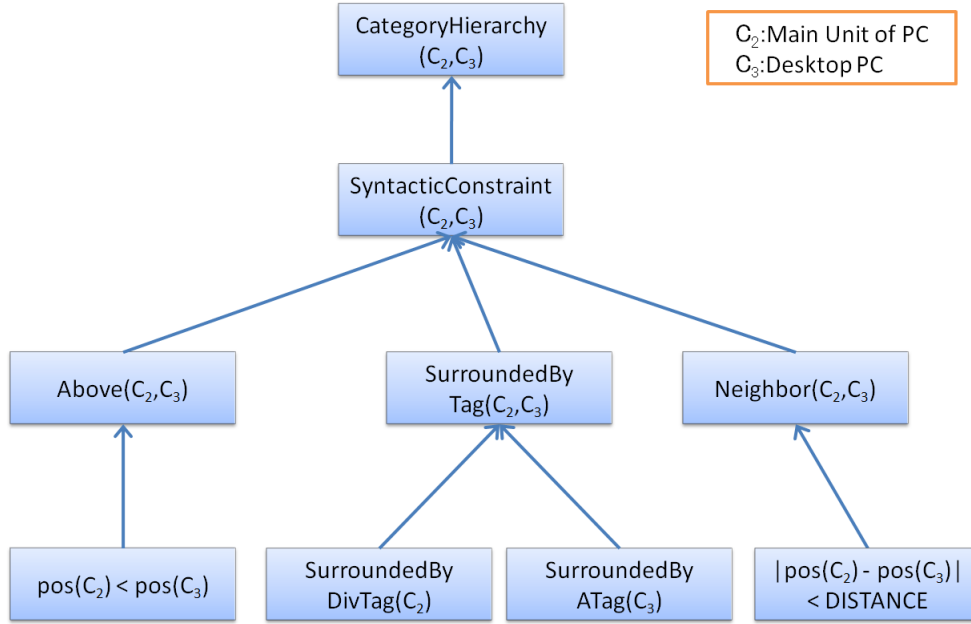


Figure 15: Proof tree

### 4.3 Generalization of Relation

The proof tree acquired in the previous section only provides proof of the parent-child relationship that is included in the training example. In this section, we generalize the explanation in order to obtain more relationships of categories.

Generalization of the obtained explanation is performed using the following procedure:

1. Reconstruct the explanation from the target concept by applying the rules backward.
2. Remove the subtree whose root fulfills the operability criterion from the explanation.
3. Unify arguments of predicates that compose rule clauses.

First, the rules used for generation of the explanation are applied backward from the target concept. Then, the arguments of predicates are arranged. The generalized proof tree is shown in figure 16.

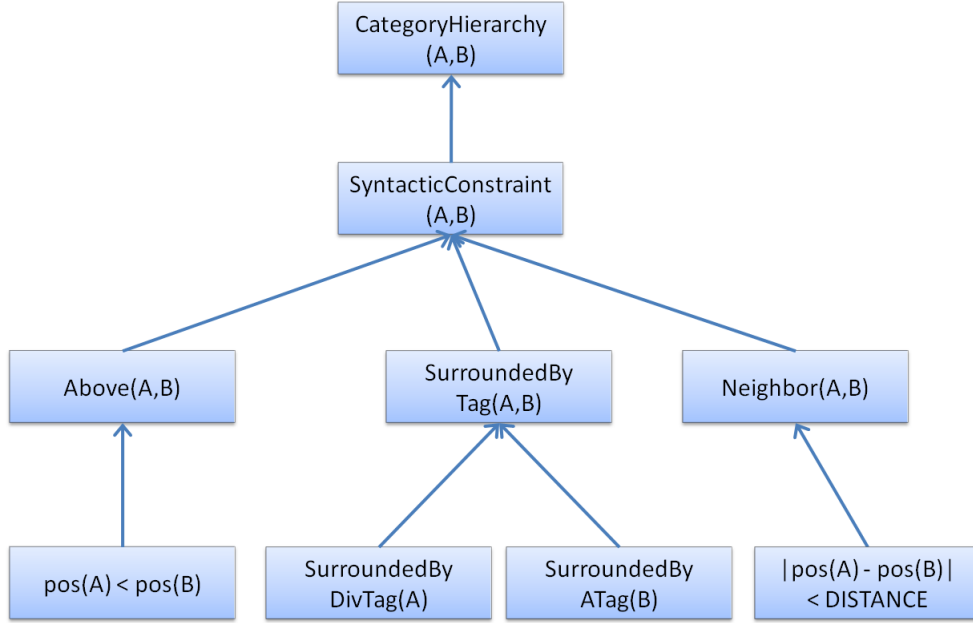


Figure 16: Generalized proof tree

#### 4.4 Generation of Concept Description

A concept description is generated from the generalized proof tree. The head of the concept description is the root of the generalized proof tree. In this case,  $CategoryHierarchy(A, B)$  corresponds to the head of concept description. The body of the concept description is a conjunction of the leaves of the generalized proof tree.

The middle hypothesis, which appears in the middle of the proof tree, is proved by the rules. Therefore, improvement in efficiency is realized by removing the middle hypothesis.

The concept description obtained from figure 16 is as follows.

$$\begin{aligned}
 CategoryHierarchy(A, B) \quad \leftarrow \quad & (pos(A) < pos(B)) \\
 & \wedge SurroundedByDivTag(A) \\
 & \wedge SurroundedByATag(B) \\
 & \wedge (|pos(A) - pos(B)| < DISTANCE)
 \end{aligned}$$

As compared with the case where reasoning is done only by the rules, the

---

**Algorithm 1** GenerateCategoryTree: Generate the category structural tree

---

*GenerateCategoryTree*(*node*)INPUT: *node*

OUTPUT: a set of nodes that are in the tree.

**if** *child*(*node*) = *null* **then**    **return** *node***end if***C*  $\leftarrow$  *SearchChildren*(*node*)**for all** *c* in *C* **do**    *N*  $\leftarrow$  *N*  $\cup$  *GenerateCategoryTree*(*c*)**end for****return** *N*  $\cup$  *node*

---

concept description scales up the operatinality. That is, the problem-solving performance improves.

## 4.5 Interpretation of Web Pages

Based on the generalized explanation, the system reasons about categories in the portion which is not interpreted in a web page.

The purpose is to discover unknown child categories of a certain category. Intrepretation is carried out using the following procedures.

The procedure for building a category structural tree is shown in algorithm1. The function *GenerateCategoryTree* initially inputs *node<sub>parent</sub>*, which represets the parent category. If a node *node<sub>parent</sub>* which represents a parent category as an initial input is given to the function *GenerateCategoryTree*, this function searches for the unknown child category which is not included in the training example and generates the tree structural data of categories. The function *child*(*node*) returns a set of the child nodes of *node*.

*SearchChildren*(*node*) is a function that searches child nodes of *node* and returns them, as shown in algorithm?? precisely. The system searches for the child categories using rules that are employed when the proof tree is generated from training examples.

---

**Algorithm 2** SearchChildren: Search for child nodes from a web page

---

$SearchChildren(node_{parent})$

INPUT: parent node  $node_{parent}$

OUTPUT: a set of child nodes  $N$

$N_{candidate} \leftarrow FindSameTag(node.area.URL, node.label.tag)$

**for all**  $node_{candidate}$  in  $N_{candidate}$  **do**

**if**  $chkConceptDescription(node_{candidate})$  **then**

$N \leftarrow N \cup node_{candidate}$

**end if**

**end for**

**return**  $N$

---

$FindSameTag(URL, tag)$  is a function that search the web page located with  $URL$  for the area that is marked up with  $tag$ .  $FindSameTag$  returns a set of  $node$ .  $chkConceptDescription(node)$  is a function that discriminates whether  $node$  fulfills the Concept Description mentioned in section 4.4. This function returns boolean value.

Thereby, the system can find child categories that are not in the training examples from the target web page. Figure 17 shows the acquired child categories.

If the proposed system cannot reach the target concept with the rules, Semantic Constraint is used secondarily. For instance, if there is a web site that

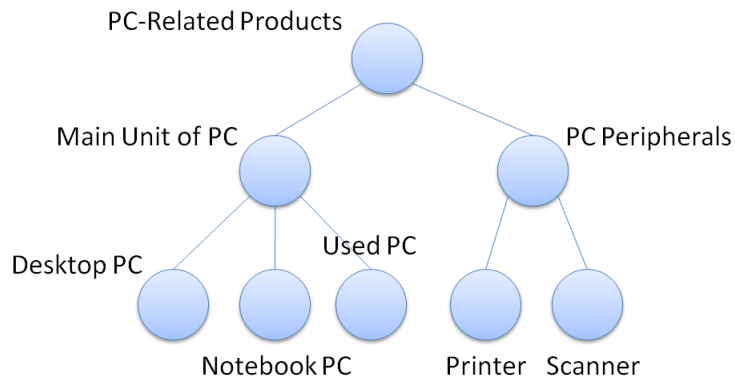


Figure 17: Child categories acquired from a web page

does not fulfill the condition “a label of the parent category exists in the upper part of the HTML source than the label of the child category”, the rule about the predicate *Above* does not fire. Thereby, the proposed system cannot prove the target concept with the domain theory in stock, and fails to create the explanation. In this case, rules about Semantic Constraint are added to the domain theory in order to be able to generate the proof tree. For example, the rule “If there is a label and is the synonym relationship with the label of the child category which is contained in the training example, that label also means the child category.” is added.

#### **4.6 Generation of Category Structural Tree**

The proposed system applies the procedure mentioned above to each web page. Finally, the category structural tree of the whole website which is the aggregate of web pages is obtained.

## Chapter 5 Discussion

In this chapter, we discuss the method proposed in this paper. As examples, we focus on Kakaku.com and Yahoo! News<sup>1)</sup> that is a website about news.

We also discuss the applicability of this method.

### 5.1 Result of execution

In this section, we introduce the examples of application of this technique to websites that have category structure, and give consideration to the execution result.

The screenshot shows the homepage of Kakaku.com. At the top, there is a navigation bar with the site logo '価格.com' and links for '新規ID登録', 'ログイン', and 'モバイル ご利用案内'. Below this is a search bar with the text '製品検索' and a '検索' button. A large banner for 'Endeavor NJ5000Pro' is visible. On the right side, there is a '新着・お知らせ' section with several news items, including '【パソコン関連】USBワンセグチューナー特集!' and '【生活家電】花粉対策2007 お役立ち家電特集!'. Below the search bar is a 'カテゴリー一覧' (Category List) section with various categories like 'パソコン関連', 'プロバイダ', '自動車・バイク', etc. At the bottom, there are promotional banners for '光ファイバー 最大級のキャッシュバック' and '新製品ニュース'.

Figure 18: Top page of kakaku.com

<sup>1)</sup> <http://headlines.yahoo.co.jp/hl>



Figure 19: Example of searching categories in the same hierarchy

## Kakaku.com

In Kakaku.com, wide range of price information of products and services is listed. These contents are classified into categories. Screenshot of the top page is shown in figure 18. We apply the proposed method to this website.

Sometimes it is difficult to set a value of the constant *DISTANCE*. Based on the hypothesis that labels of categories in the same hierarchy emerge nearby mutually, this constant is prepared to discover labels of categories they are in the same hierarchy. The value of *DISTANCE* is set before the deductive learning. However, distances between labels are all different. Therefore the optimal value of it depends on the websites. There is possibility that this has an influence on the learning.

For instance, in figure 19, a whole page represents the category “デスクトップパソコン (Desktop PC)”. It has child categories like “HP” and “NEC”. These child categories have labels that are surrounded by anchor tags.

Now the category “HP” is selected by a user and becomes a training example as a child category. If the value of *DISTANCE* is too small, the proposed system cannot find the category “富士通 (Fujitsu)”.

## 竹中工務店、「みなし配当」281億円源泉課税漏れ

2月7日(水)3時4分配信 [読売新聞](#)

大手ゼネコン「竹中工務店」（大阪市中央区、非上場）が、社員の持ち株会から自社株を譲り受け、相殺した貸付金の一部約281億円は同会へのみなし配当にあたるとして、大阪国税局から源泉所得税の課税漏れを指摘されたことがわかった。[\[記事全文\]](#)

- ▶ [死亡児童の写真をネット掲載の教諭、7日にも逮捕へ](#) ([読売新聞](#)) - 3時10分
- ▶ [<山崎パン>不二家と資本提携へ 3月初めにも交渉](#) ([毎日新聞](#)) - 3時4分

Figure 20: Top page of Yahoo! News

### Yahoo! News

Figure 20 shows the top page of Yahoo! News. In this website, contents are classified into the categories of “ニュース (News)”, “トピックス (Topics)” and “写真 (Photographs)”. Each category has child categories like “主要 (Main news)” and “国内 (Domestic news)”.

We consider extracting category structural tree from this site. The category structural tree of this site is shown in figure 21.

In this tree, the category “北方領土 (Northern Territories)” and “News A” belong to the category “主要 (Main news)”. In other words, the parent category of “北方領土 (Northern Territories)” is the category “主要 (Main news)”. However, they also belong to the category “政治 (political news)”. The same category appears twice in the same tree.

Generally, if there is a category that has more than one parent category, representing the category structure with a tree structure has redundancy. Thus, our method cannot represent such a category structure efficiently. In order to deal with it, we have to use another data structure.

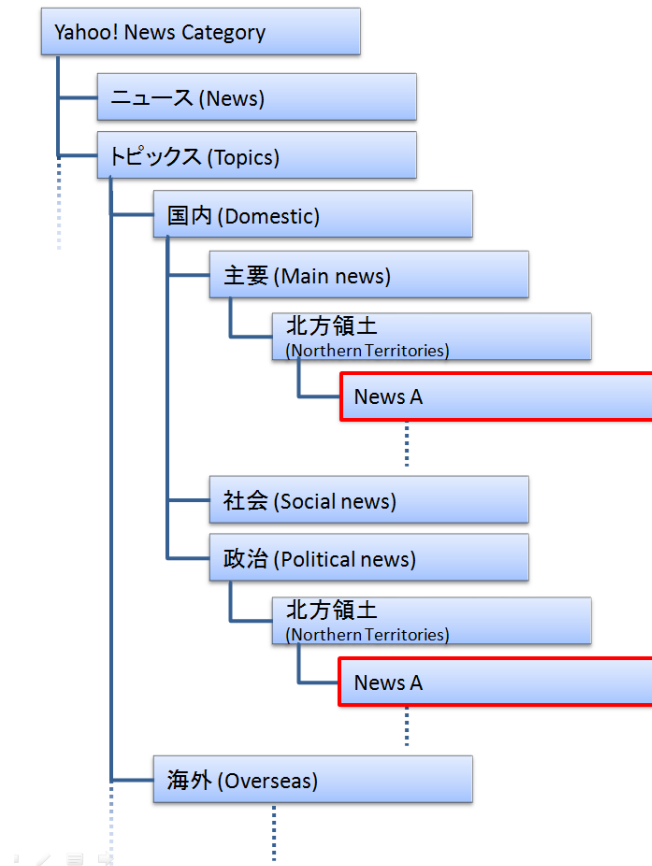


Figure 21: Category structural tree of Yahoo! News

## 5.2 Applicability of proposed technique

For comparison with inductive-learning-based method, the deductive learning has the feature that it is possible to learn from a small number of the training examples. Therefore the proposed technique can reduce the cost of generating training examples by a user. Specifically in this research, instructions by a user have an important role for extraction of categories. Hence, it is significant to decrease the load of a user.

The researchers which use training examples of a user to find ambiguous contents are accomplished by the others, too. The system proposed in this paper deals with websites that have a category structure, and use of training examples is limited to acquire the category structure.

From the supposition about the category expressions given in Chapter 3, the proposed system is applicable to websites which have following characteristics.

### **Described by HTML**

We realize the extraction of the category structure by relating the category and the HTML source which expresses the presentation with the hand in this research. Therefore, learning and extraction are possible when website is described by HTML. However, there are websites where the technology like Adobe Flash is used for expression of categories. In such a case, it is not possible for the proposed system to extract categories.

### **Consistency of tagging**

In this research, tags that mark up labels of categories that are contained in training examples are used for creating an explanation. After generalization of the explanation, the system searches the web page for the areas that are surrounded by the same tag. If such areas are found, these areas are compared with the concept description. Thus, if there are labels of categories that are in the same hierarchical level, and they are marked up by the same tag, these labels can be discovered from the web page because they comply with the generalized explanation.

On the other hand, if a tag which marks up a label of the category in training examples is different from tags of other categories, the proposed system cannot extract category structures.

### **Consistency of URL**

At the sites that generates web pages dynamically, there is possibility that the URL changes every time when accessed. Therefore, links in the category structural tree may not work correctly.

### **Satisfaction of precondition**

In this research, wide ways of category expression are supported. However, as for some expressions, we set preconditions. The supposition about the position of the label is the one.

From physical relationship of the two category labels that are contained in the training examples, the rule which has predicate *Above* on the head fires. Thus, we succeed in the derivation of the target concept. That is, the system can adapt the generalized explanation to child categories that are not included in the training examples, and we realize extraction.

However, in cases where a label of a parent category appears lower than that of a child category in an HTML source, the rule does not fire because the precondition is not satisfied. We think that the number of such a website is few. On the other hand, when existing, we fail to extract the parent-child category relations.

The proposed method does not depend on the presentation of an HTML. Thus, if assumptions about the parent-child relationships and rules are fulfilled, the system can extract categories from web sites without regard to domains or kind of tags. Moreover, the system does not require the knowledge base about domains of categories since categories are given by users.

However, there are limitations from assumptions based on the page structure.

If the feature of markup is same, a category that a user did not intend may be extracted. For instance, if categories are marked up with header tags and advertisements are also surrounded by header tags, advertisements will be extracted as categories. In cases where categories are not represented with HTML tags, the system cannot find categories, because rules in this research are based on tags. Furthermore, if two categories are in the same hierarchy and these parents are different, users have to give training examples twice. In this case, giving training examples by users is redundant.

## Chapter 6 Related Works

The formalization in Chapter 3 and the technique of information extraction in Chapter 4 play a very important role in this research.

There are a lot of studies regarding information extraction from web pages as follows.

[1], [2], [3] and [4] extract data from HTML.

In [1], wrapper is generated using inductive learning with training examples. On the assumed web page where a lot of records of same form, requesting the delimiter that becomes a delimitation of the attribute and the record based on the training example extracts requested information. However, cost to create training examples is high, since inductive learning requires many complex training examples.

A system known as IEPAD is proposed in [2]. IEPAD automatically discovers extraction rules from web pages based on the patterns of the HTML tag. IEPAD searches strings that are most repeated in HTML. Therefore IEPAD can extract without training examples.

Inductive learning and delimiter-based extraction patterns are used in [3].

Power browser, the proposed system in [4], can provide a tree structure of a website mainly for the purpose of web access with a mobile terminal. An algorithm of Power Browser is based on the syntactical structure of a website. It does not take the semantic hierarchical structure into consideration.

In [5], [6] and [7], information extraction from the table of HTML is described.

The method proposed in [5] does not require prior knowledge. Once interpretations of table structures are given by users, table structures are automatically generalized. Formal representation is given and this method can cope with various relations.

In [6], the method of acquiring the hierarchical structure from tables of HTML is proposed. However, the proposed method interprets according to the table structure which was assumed in advance. Thus applicability is limited.

Information extraction with wrapper learning from the table is proposed in

[7]. When training examples are given, rules are acquired using tags of HTML and contents of tags.

## Chapter 7 Application

By using the proposed method, a category structural tree can be extracted from a target website which has categories in parent-child relationships. In this chapter, we propose the application of category structural trees.

### 7.1 Screen Reader

When people who have difficulty in seeing characters and pictures on a screen browse websites, they use software known as a screen reader or a voice browser. These read out text information on a web page with speech synthesis technology.

In order for this software to work optimally, the producer of a website has to build a site with careful attention to accessibility beforehand. In Japan, the requirements for improvements in accessibility are standardized by JIS X 8341. Therefore, it is necessary to create websites that are easy to use for all people including users of a screen reader.

Regrettably, however, not all websites are built with careful attention to accessibility, and they are not necessarily easy to browse with a screen reader. Moreover, even if websites are made in consideration of accessibility, it is still difficult to get a grip on their structure with only voice information.

People who are able to see operate the proposed system and create training examples of categories of the target website. Then the category structural tree of the target website can be gained. By having a screen reader read out this category structural tree, users of the screen readers can understand the structure of the whole website quickly and access to the target contents becomes easy. That is, it is possible to reduce time and operation costs to access the target part of the website and to raise accessibility, as compared with cases where the whole website is read out.

### 7.2 Mobile Terminal

Against the background of the spread of mobile terminals with an Internet connectivity function, such as cellular phones and PHSs, the opportunity to access to the Internet with a mobile terminal has been increasing. In Japan, the num-

ber of Internet access service contractors by cellular phone exceeds 82 million as of January, 2007. Furthermore, in recent years, there are mobile terminals which have a full browser that can display websites designed for personal computers.

On the other hand, a designer of websites assumes that these sites are browsed with PCs and designs for them; excluding cases where a special page for mobile terminals is provided. Therefore, when browsing websites with a mobile terminal, some difficulties exist.

With mobile terminals, the size of the screen is small, and the standard of screen resolution is only  $320 \times 240$  dots. The cost per unit traffic is high, and transmission speed is slow. The input interface is restricted and poor.

The markup language that most mobile terminals can interpret is a subset of the usual HTML. For this reason, the browser for mobile terminals may find it impossible to interpret and show web pages designed for PCs.

Figure 22 is an example of the image when displaying a web site that is designed for computers with a mobile terminal. This example is created by the

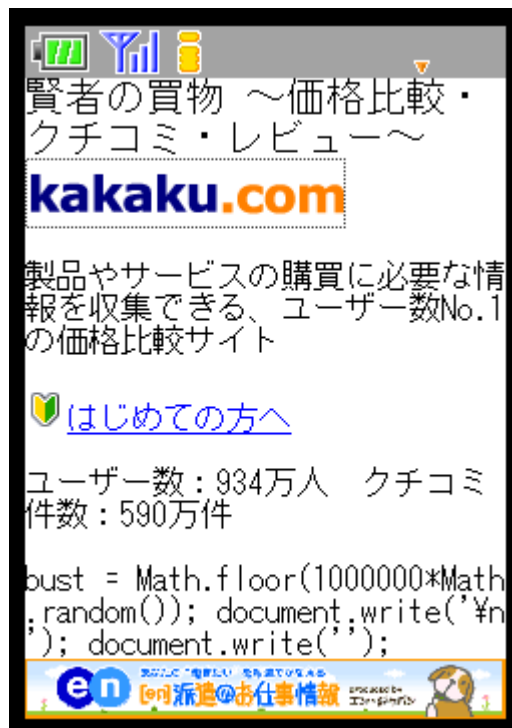


Figure 22: Browser of a mobile phone showing a website designed for PCs



Figure 23: Category structural tree on a mobile browser

simulator of the browser on the mobile phone<sup>1)</sup>.

Because there is a part displayed incorrectly and the area currently displayed on the screen is narrow, comprehension of the whole image of the website is difficult. It costs time and money to access the target category with a browser of a mobile terminal. By using a full browser, we can avoid the phenomenon of incorrect rendering results. However, it is still difficult to get a grip on the whole image of the website.

As a solution to such a problem, showing a category structural tree supports those users who browse websites with mobile terminals. Instead of accessing websites designed for computers directly with a mobile browser, the proposed system shows the category structural tree to the users. An image is shown as figure 23.

Because the category structural tree has links to each category of the target webpage, users can access the webpage they intend to see directly.

With the display of only a category structural tree, users can get an overview

<sup>1)</sup> <http://www.nttdocomo.co.jp/service/imode/make/content/html/tool2.html>

of the whole website on the small screen of a mobile terminal, and the number of clicks for scrolling can be reduced. Traffic also decreases as compared with the case where the website is displayed as it is. These points are the benefits for users of mobile terminals.

## Chapter 8 Conclusion

There were problems in automatic acquisition of category structures from web-sites. The research issues were as follows:

**Ambiguity of categories described with HTML** Human beings can recognize the hierarchy of categories, but machines cannot. This is because HTML tags represent presentation of web pages, but they do not express semantics. Machines cannot distinguish the areas of categories and parent-child relation of categories automatically.

**Diversity of expression of categories** Categories and the parent-child relationship between two categories are expressed in various ways, because there is no concrete rule to describe categories. Thus, it is difficult to correspond to various web sites in extracting categories.

To solve these issues, this paper proposed the method using deductive learning.

People construe the appearance parts of categories and the parent-child relationship of categories in a web page, and they teach the system. So the system can construe the parent-child relationship of categories in a various ways of expression. As generalizing the concept which construction is given, we can get more parent-child relationships. The system acquires the child-categories by searching the part applicable to the generalized category relations. Finally, we can extract the layered category structure of a web site by applying these procedures to web pages which constitute a web site.

The contributions of this work are as follows:

**Correlation of categories with HTML** The system can correlate categories with the presentation of HTML by using training examples, and generalize that correlation. Thus, we are able to acquire the structure of categories from a website where presentations of categories have consistency.

**Application to a wide range of means of expression** In the proposed technique, a position of a category and a parent-child relationship of categories are given by a user. Therefore, it is possible to deal with the web site where categories are represented with a wide range of expressions about an

appearing position, a tagging, etc.

In this proposed method, training examples from a user are used in order to acquire categories and parent-child relationships of categories. Therefore, we succeed in extraction of categories and category hierarchy from a website that has ambiguity in category description. We also use deductive learning for the discovery of child categories and that reduces the hand of a user. The deductive learning has the feature that it is possible to learn from a small number of the training examples as compared with the inductive learning which is also the supervised learning.

The Semantic Web, the method of giving a tag that shows the semantics is advocated in order to keep web sites machine-readable. And the Web sites with such machine-readable category information will be expected to increase in the future. If the Semantic Web is fully realized and can express category structures that the human being intended with machine-readable notation, to present the category hierarchical structure becomes easy substantially. On the other hand, at present, the number of the web sites tagged like that is very few. There are much websites that are described only in HTML. Therefore, at present, it is possible to say that the technique of this research is valid.

The point which we should evaluate about this research is still left. Performance evaluation of accuracy and speed of execution are the topic of our further study.

As mentioned above, there is possibility that a wrong hierarchical structure is generated depending on characteristic of a target website and the way of giving training examples by a user. In this research, a means of the correction of the category hierarchical structure is not available currently. Therefore it remains as one of the key issues to be clarified. As a solution of this problem, to prepare the means of the correction of the category hierarchical structure is possible.

The application of this study is to support users in the serious situation to browse web sites. For example, there are cases where people with visual impairment would use a screen reader while browsing web sites and cases people access the Internet with mobile terminals. If they browse the web sites where

site maps are not prepared beforehand, it is difficult to understand a whole structure of them. We can provide easy access and the overview of websites by displaying the category structure.

Therefore, we conclude that this research is socially important and significant from the viewpoint of bridging the digital divide and promotion of the ubiquitous environment.

## Acknowledgments

I would like to express my sincere gratitude to Professor Toru Ishida and Associate Professor Shigeo Matsubara at Graduate School of Informatics, Kyoto University, for invaluable advice and discussion.

I am also grateful to my research adviser, Associate Professor Mizuho Iwaihara and Associate Professor Keishi Tajima at Graduate School of Informatics, Kyoto University.

I would like to express special thanks to all the members of Ishida Laboratory at Kyoto University for their support.

## References

- [1] Kushmerick, N.: *Wrapper induction for information extraction*, PhD Thesis, University of Washington (1997). Chairperson-Daniel S. Weld.
- [2] Chang, C.-H. and Lui, S.-C.: IEPAD: information extraction based on pattern discovery, *WWW '01: Proceedings of the 10th international conference on World Wide Web*, New York, NY, USA, ACM Press, pp. 681–688 (2001).
- [3] Muslea, I., Minton, S. and Knoblock, C. A.: Hierarchical Wrapper Induction for Semistructured Information Sources, *Autonomous Agents and Multi-Agent Systems*, Vol. 4, No. 1-2, pp. 93–114 (2001).
- [4] Buyukkokten, O., Garcia-Molina, H., Paepcke, A. and Winograd, T.: Power browser: efficient Web browsing for PDAs, *CHI '00: Proceedings of the SIGCHI conference on Human factors in computing systems*, New York, NY, USA, ACM Press, pp. 430–437 (2000).
- [5] Tanaka, M. and Ishida, T.: Ontology Extraction from Tables on the Web, *saint*, Vol. 0, pp. 284–290 (2006).
- [6] Lim, S. and Ng, Y.: An Automated Approach for Retrieving Hierarchical Data from HTML Tables, *8th International Conference on Information and Knowledge Management*, pp. 466–474 (1999).
- [7] Cohen, W., Hurst, M. and Jensen, L.: A Flexible Learning System for Wrapping Tables and Lists in HTML Documents, *11th World Wide Web Conference*, pp. 232–241 (2002).
- [8] 山田誠二: 認知科学の発展, Vol. Vol.4, 講談社 (1991).
- [9] 田仲正弘, 石田亨: 表構造の一般化に基づくオントロジの獲得, 情報処理学会論文誌, Vol. 47, No. 5, pp. 1530–1537 (2006).
- [10] Anupam, V., Breitbart, Y., Freire, J. and Kumar, B.: Personalizing the Web Using Site Descriptions, *DEXA '99: Proceedings of the 10th International Workshop on Database & Expert Systems Applications*, Washington, DC, USA, IEEE Computer Society, p. 732 (1999).
- [11] Ashish, N. and Knoblock, C.: Wrapper Generation for Semi-Structured Internet Source, *ACM SIGMOD Records*, Vol. 26–4, pp. 8–15 (1997).

- [12] Baluja, S.: Browsing on small screens: recasting web-page segmentation into an efficient machine learning framework, *WWW '06: Proceedings of the 15th international conference on World Wide Web*, New York, NY, USA, ACM Press, pp. 33–42 (2006).
- [13] Bauer, M., Dengler, D., Paul, G. and Meyer, M.: Programming by example: programming by demonstration for information agents, *Commun. ACM*, Vol. 43, No. 3, pp. 98–103 (2000).
- [14] Cimiano, P., Handschuh, S. and Staab, S.: Towards the Self-Annotating Web., *13th World Wide Web Conference*, pp. 462–471 (2004).
- [15] Cohen, W. and Fan, W.: Learning Page-independent Heuristics for Extracting Data from Web Pages, *8th World Wide Web Conference*, pp. 1641–1652 (1999).
- [16] Cypher, A.: EAGER: programming repetitive tasks by example, *CHI '91: Proceedings of the SIGCHI conference on Human factors in computing systems*, New York, NY, USA, ACM Press, pp. 33–39 (1991).
- [17] Freitag, D.: Machine Learning for Information Extraction in Informal Domains, *Machine Learning*, Vol. 39, No. 2, pp. 169–202 (2000).
- [18] Hsu, C.: Initial results on wrapping semistructured web pages with finite-state transducers and contextual rules, *Workshop on AI and Information Integration*, pp. 66–73 (1998).
- [19] Mukherjee, S., Yang, G., Tan, W. and Ramakrishnan, I. V.: Automatic Discovery of Semantic Structures in HTML Documents, *ICDAR '03: Proceedings of the Seventh International Conference on Document Analysis and Recognition*, Washington, DC, USA, IEEE Computer Society, p. 245 (2003).
- [20] Soderland, S.: Learning Information Extraction Rules for Semi-Structured and Free Text, *Machine Learning*, Vol. 34, No. 1, pp. 233–272 (1999).
- [21] Wang, J. and Lochovsky, F. H.: Data extraction and label assignment for web databases, *WWW '03: Proceedings of the 12th international conference on World Wide Web*, New York, NY, USA, ACM Press, pp. 187–196 (2003).