Information Extraction from Specifications on the World Wide Web

Atsushi Fukumoto and Tsutomu Endo

Department of Artificial Intelligence, Kyushu Institute of Technology
Iizuka, Fukuoka 820-8502 JAPAN
{a_fuku, endo}@pluto.ai.kyutech.ac.jp

Kazutaka Shimada

Department of Computer Science and Intelligent Systems, Oita University
Dannoharu, Oita 870-1192 JAPAN
shimada@csis.oita-u.ac.jp

Summary

The World Wide Web is the most important service on the Internet. The capability of only retrieving information is insufficient because there are many different kinds of information on the Web. Systems to help use retrieved information effectively are needed for users. This paper proposes a method to integrate computer specifications retrieved from multiple Web sites, to extract features of each computer based on integrated information, and to present a computer suitable for user's request. The specifications written in HTML are converted into normal form called table structure. The quantitative features such as speed, capacity and dimensions are extracted by comparison with mean or mode of all sample data, and the qualitative ones such as kind of processor and graphics chip are extracted by using knowledge provided manually. Comparing features with user's request such as mobile, graphics oriented, or network-ready machine determines the recommended computer. Experimental results show the effectiveness of our method.

Keywords: information extraction, information integration, specifications, HTML, WWW, tabular representation.

1 Introduction

The World Wide Web (WWW) is the most important service on the Internet. Search engines and portal sites help users find useful online information sources. However, there are many different kinds of information on the Web, and only the capability of information retrieval is insufficient for users. What they need now are systems to help use the retrieved information effectively. For example, consider product information about personal computer. Information of this sort is obtained from Web sites of a lot of computer makers. It is difficult for users except some experts to select a suitable computer for their own purpose from this information. The reasons are as follows: (1) each Web site provides its own products, and does not contain comparison with other maker's products, (2) Web pages of each site have various styles, and it is not easy to compare with each other, and (3) association of user's request with specifications of each product and evaluation of specifications require technical knowledge. To satisfy a user's request, a Web-based system must integrate the information from the various sites into a single, coherent whole. Unfortunately, integrating information from diverse sources is very hard when information is presented in simple structure [1].

We are now developing a system which summarizes the specifications from multiple Web sites, focusing on personal computer products [2]. The final goal of our research is to integrate data from multiple sites of personal computer maker, to extract various features of each computer from integrated information, and to propose a computer suitable for user's request. This paper proposes a method to integrate specifications presented in tabular representation and to extract features of each personal computer based on integrated information.

2 Architecture for Web-based Information Extraction

Figure 1 shows a process flow of Web-based information extraction. Using ordinary search engines, Web pages about personal computer are retrieved from multiple sites. The search keys are words such as "personal computer" and "PC", and URLs of typical computer makers such as www.ibm.com and www.fujitsu.co.jp. These pages contain text, tables, images, audio, and video. In general, the specifications of the computer are presented in tabular form as shown in Table 1. In order to determine what part of the HTML-based pages presents the specifications, several rules of thumb based on our observation of HTML pages are used:

- The page of which the title contains key words such as "SPECIFICATIONS" or "SPEC".
- A paragraph which exists in HTML tags (<TABLE> and </TABLE>. It contains keywords such as "model", "processor", and "memory".

The identified specifications are then represented in normal form called table structure. It is a set of simple list consisting of model name of computer, attribute, and value. The conversion and integration of tables are discussed in detail in Section 3. Next step is feature extraction from the table structure. The features mean the noticeable or outstanding attributes of each computer in comparison with others. This is discussed in detail in Section 4. Finally, scoring these features according to user's request such as budget PCs, mobile PCs, high performance and versatility, or high graphics performance determines the recommended personal computer.

3 Normal Forms of Specifications

Specifications are expressed in the form of a two-dimensional table. Generally, the first column corresponds to the attribute of PC, and the cell in the *i*-th row shows the name of the *i*-th attribute. The second or more columns correspond to a model of PCs, and the cell in the *i*-th row shows the value of the *i*-th attribute. The role of the column and row is sometimes transposed each other. This is decided by whether the name of key attributes such as processor and memory is written in the first column or the first row.

The serious problem in these tables is that the style of description in each cell is not standardized as follows: (1) the kind and name of attributes are not standardized, (2)

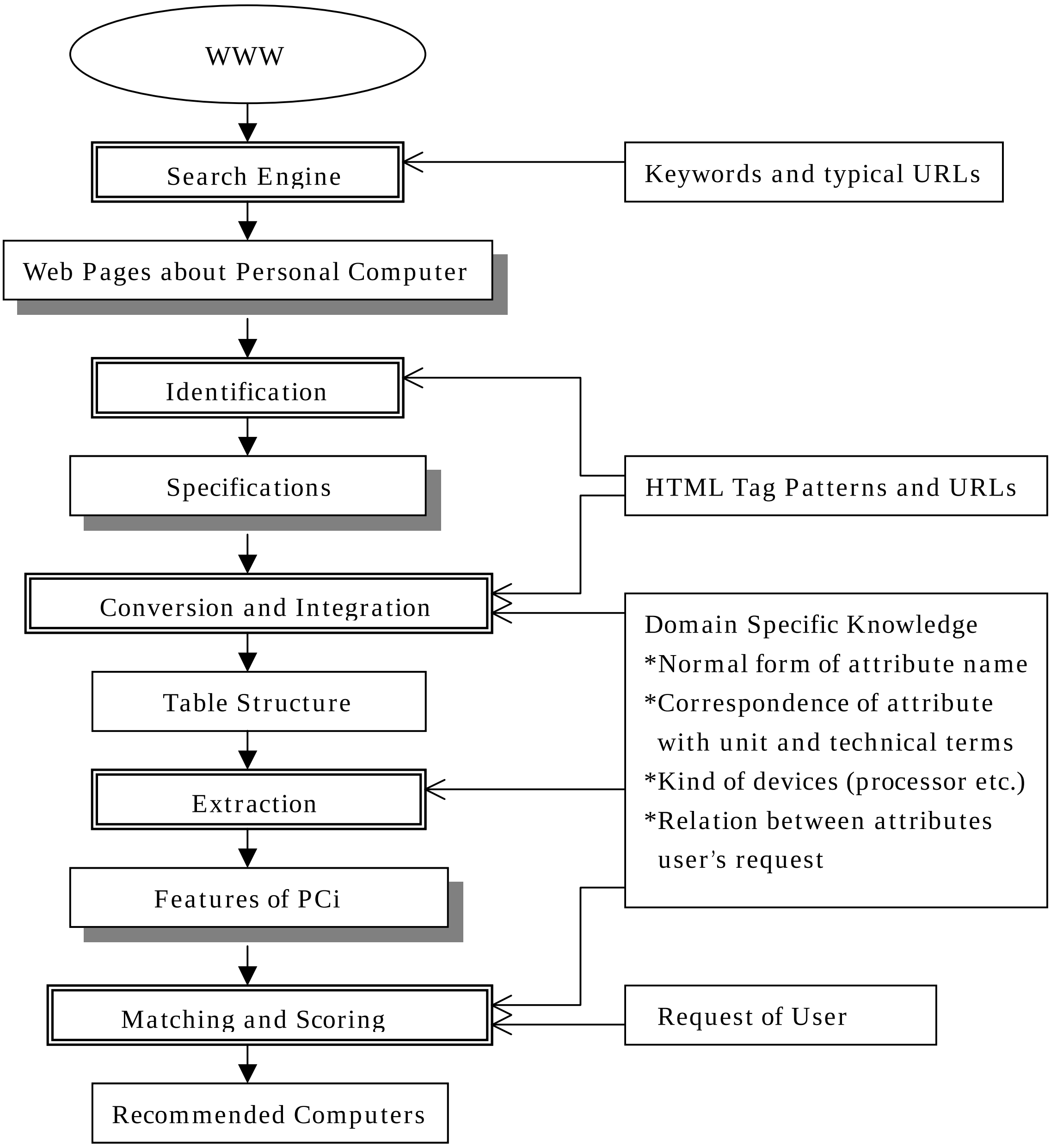
WWW
Search Engine
Keywords and typical URLs
Web Pages about Personal Computer
Identification
Specifications
HTML Tag Patterns and URLs
Conversion and Integration
Domain Specific Knowledge
*Normal form of attribute name
*Correspondence of attribute with unit and technical terms
*Kind of devices (processor etc.)
*Relation between attributes user's request
Table Structure
Extraction
Features of PCi
Matching and Scoring
Request of User
Recommended Computers

Figure 1. Process Flow of Web-based Information Extraction.

Table 1. An Example of Specifications.

Model name	Kyutech PC-XC600	Kyutech PC-XP800
Microprocessor	Celeron Processor 600MHz	Pentium III Processor 800MHz
Main memory	64MB SDRAM exp. to 256MB	128MB SDRAM exp. to 512MB
Display	12.1” SVGA TFT	15” Super XGA TFT
Hard drive	5GB Ultra ATA	20GB Ultra ATA
Interface	2x USB, IEEE1394	3x USB, IEEE1394, Ethernet
Graphics card	Rage Mobility 3D	Rage Mobility-M4 3D
Dimensions	330 x 261 x 38.5 mm	377 x 290 x 42 mm
Battery life	3.0 hours	2.1 hours
Pre installed OS	Microsoft Windows 98 Second Edition	

some cell contains two or more values, (3) some attribute is divided into subcategories, and (4) two or more cells which contain the same value are unified.

3.1 Definition of Table Structure

To solve above problems, we define a normal form called table structure and provide domain specific knowledge manually. The table structure is a set of simple ternary list:

(Nam Atr Val)

where Nam, Atr, and Val are the model name, attribute name, and value respectively. Nam and Atr are sometimes represented by list.

On the other hand, the knowledge is the following: normal form of attribute name, correspondence of attribute with the unit and the relevant technical terms such as PCI and USB, kind of microprocessor (Pentium III, Celeron) and order of performance, relation between attribute and user’s request. This knowledge is also used for feature extraction.

3.2 Algorithm for Conversion

An algorithm to convert HTML-based specifications as shown in Figure 2(b) into table structure is as follows:

[1] Decompose a unified cell in the first column using HTML tag “ROWSPAN=*n*”, and make “*n*” cells by unifying the neighboring items. For example, “Memory” cell in Figure 2(a) is decomposed into three “Memory” cells and each cell is combined with “Standard”, “Maximum”, and “VRAM” respectively.

[2] Decompose a unified cell in any row using tag “COLSPAN”, and insert the common value into new cells. For example, “256MB” and “4MB” cells are also decomposed into two cells respectively, and same value is inserted into new cells.

Model name		PC1	PC2
Microprocessor		600MHz	700MHz
Memory	Standard	64MB	128MB
	Maximum	256MB	
	VRAM	4MB	

(a) Tabular Representation

```
<TABLE BORDER=”1”>
<TR><TD COLSPAN=”2”>Model name</TD>
        <TD>PC1</TD>
        <TD>PC2</TD> </TR>
<TR><TD COLSPAN=”2”>Microprocessor</TD>
        <TD>600MHz</TD>
        <TD>700MHz</TD> </TR>
<TR><TD ROWSPAN=”3”>Memory</TD>
        <TD>Standard</TD>
        <TD>64MB</TD>
        <TD>128MB</TD></TR>
<TR><TD>Maximum</TD>
        <TD COLSPAN=”2>256MB</TD></TR>
<TR><TD>VRAM</TD>
          <TD COLSPAN=”2”>4MB</TD></TR>
</TABLE>
```

(b) HTML Data

(PC1 Microprocessor 600MHz)
(PC1 (Memory Standard) 64MB)
(PC1 (Memory Maximum) 256MB)
(PC1 (Memory VRAM) 4MB)
(PC2 Microprocessor 700MHz)
(PC2 (Memory Standard) 128MB)
(PC2 (Memory Maximum) 256MB)
(PC2 (Memory VRAM) 4MB)

(c) Table Structure

Figure 2. An Example of Table.

[2] Let c(*i, j*) denote cell in the *i*-th row, *j*-th column and (Nam_k, Atr_k, Val_k)$_k$ denote the *k*-th list in table structure. Set *k* = 1. For each *j* (1 < *j*), do the following substeps:

[2.1] Set Nam_k = c(1, *j*) (i.e., the model name of PC is set to Nam).

[2.2] For each *i*(1 < *i*),

(1) Set Atr_k = c(*i*, 1). (i.e., the *i*-th attribute

name is set to Atr).

(2) Set Val_k = c(*i, j*). (i.e., the value of the *i*-th attribute is set to Val).

(3) Set $k = k + 1$.

In the above set operation, if the text that exists between the tag <TD> and </TD> consists of several words, represent it in a list form of words.

[3] For each list, do the following substeps using appropriate knowledge:

[3.1] Transform a word in a list into the normal form.

Ex. Monitor, Screen → Display.
10/100BaseT → Ethernet.

[3.2] If the element Val contains two or more values, divide the list into several lists.

Ex. (PC-1 slots (5 PCI 1 AGP)) → (PC-1 (slots PCI) 5) and (PC-1 (slots AGP) 1).
(PC-1 CPU (Celeron 500MHz)) →
(PC-1 (CPU Processor) Celeron) and
(PC-1 (CPU Clock) 500MHz).

[3.3] Parse the particular notations and rewrite them into normal forms.

Ex. (PC-1 (bays [available]) 3[2]) → (PC-1 bays 3) and (PC-1 (bays available) 2)
(PC-1 Memory (64MB Standard)) →
(PC-1 (Memory Standard) 64MB).

Figure 2(c) shows an example of table structure converted from the HTML data (b).

4 Use of Specifications

Since the specifications contain a lot of attributes and values about PCs, it is not clear which are the important data. So we try to extract the features, that is, the attribute names and values that characterize each PC.

4.1 Feature Extraction

The attribute is classified into two categories: quantitative and qualitative. The typical example is listed in Table 2.

Table 2. Classification of Attributes.

Quantitative Attribute	(CPU Clock): MHz, GHz (Memory Standard): MB (Memory Maximum): MB (Memory VRAM): MB (Display Size): inch (14.1") (Display Resolution): dot Hard-drive: GB (Interface USB): number PC-card: number Dimensions: mm, cm Weight: kg, lbs. Battery-life: hours Price: Yen, $
Qualitative Attribute	(CPU Processor) Graphics (Drive Floppy) (Drive CD-ROM) Pre-installed-OS

The quantitative features are attributes whose value is maximum or greater than the standard value of the attribute. In some attributes such as dimensions, weight, or price, the value is minimum or smaller than the standard value. However, the attributes that have the same value in all PCs are rejected. The standard value is the mean or mode of all sample data. The attributes using mean are clock, price, dimensions, weight, power and so on. The attributes using mode are memory capacity, resolution, display size and so on. The following preprocessing is performed before computing mean or mode:

- The unit of value is standardized: (mm, cm), (MHz, GHz), and (KB, MB, GB).
- The maximum or minimum value is selected from ranging data: (38-40mm).

The qualitative features are extracted as follows:

[1] Collect all lists with the qualitative attribute and discard the attribute with same value.

[2] If the attribute does not have order, give point m_1 to only the lists with the attribute.

[3] If the attribute has order, give point (1-

m_2) to the lists with the attribute according to the order provided in domain knowledge (e.g., Celeron < Duron < Pentium III).

[4] Extract as the qualitative feature the attributes whose points exceed the given threshold value.

We have developed a program that transforms these features into Japanese sentences using four kinds of sentence frame: heading, subheading, data description, and data comparison [2]. It fills the slots with the extracted features and generates sentences.

4.2 Matching Score to User's Request

Finding PCs relevant to a user's request from specifications has become important. Typical requests are shown in Table 3. We introduce a weight and a score. The former captures the strength of the relationship between attribute and request. The weight is provided in domain specific knowledge in advance. The high value is given to attributes shown in Table 3. The latter reflects the assessment of how well each PC matches the request. The matching process is as follows:

[1] Give the normalized point (0-m_3) to each quantitative attribute according to the difference between each value and the standard. Qualitative features have been given point in feature extraction process.

[2] If the concrete attribute and the value are contained in a user's request (e.g., "PCs with very long battery life", or "PCs under $1000"), select the lists that satisfy the conditions from table structure.

[3] For each PC in selected list, compute

$$score\,(c,r) = \frac{\sum_{i=1}^{n} w(i,r) \times pt(i,c)}{\sum_{i=1}^{n} w(i,r)}$$

where c, r and i are PC, request, and index of attribute respectively. $w(i, r)$ is the weight of attribute i in the request r. $pt(i, c)$ is the points given to the attribute i of the PC c.

[4] Select the PCs whose score exceeds the given threshold value, and return them as the recommended computer in descending order of matching score.

5 Experimental Results

To test our method, we wrote a Perl script that performs table conversion, feature extraction, and matching against user's request. We extracted 93 specifications from the 5 Web sites. 79.5% of them were correctly converted into table structure. The reasons why the conversion failed are as follows: (1) there are rows in which text for explaining the structure of table is written, (2) text for giving the meaning of technical terms is written in a cell, and (3) the TABLE tags are used in a cell. It is necessary to remove these expressions before table conversion.

Table 3. Examples of User's Request.

Request	Attributes to be considered
PCs with high performance and versatility	CPU, Memory, Display, Hard-drive, Interface
PCs with high graphics performance	Display, Graphics, CPU, Memory
Budget PC, Affordable PC	Price, Software, Memory, CPU
PCs for practical use	CPU, Hard-drive, Price, Interface, Software
PCs with high expandability	Memory, Slots, PC-card, Interface
Multimedia PC	DVD-ROM, CPU, Display, Memory, Interface
Mobile PC, PCs that are light and compact	Battery-life, Dimensions, Weight

Table 4. An Example of Feature Extraction from Specifications.

Model name	PC-X1	PC-Y1	PC-Z1
CPU: Clock	**800MHz**	700MHz	700MHz
Memory: Std	64MB	64MB	64MB
Memory: Max	128MB	128MB	**256MB**
Hard Drive	**15GB**	**15GB**	10GB
Diskette Drive	External	External	**Internal**
Display: Size	11.3” TFT	**12.1” TFT**	10.4” TFT
Display: Res.	**1024 x 768 dot**	800 x 600 dot	800 x 600 dot
Memory: VRAM	**2.5MB**	2MB	2MB
PC Card	1 Type II	1 Type II	1 Type II
Interface	1 USB, **1 IrDA**, **LAN**	1 USB, **1 IrDA**	**2 USB**
Dimensions	**270mm x 215mm x 29mm**	270mm x 224mm x 33.7mm	**257mm x 223mm x 29mm**
Weight	**1.6 kg**	1.98kg	**1.5kg**
Key Size	17mm	**18mm**	17mm
Battery Life: Std	1.8 hours	**2.5 hours**	1.5 hours
Battery life: Max	**11 hours**	5 hours	7 hours

We also carried out the experiment of feature extraction using table structure converted from 19 specifications. Table 4 shows an example, where PC-X1, PC-Y1 and PC-Z1 are produced by different company. The extracted features are typed in boldface. In order to evaluate whether the feature extraction algorithm is good or not, we compared the result with a review in a personal computer magazine. The three kinds of computer in Table 4 are reviewed as follows:

- PC-X1: the basic performance is high, the performance of display is high, the mobility is fairly good, and it is fairly easy to use.
- PC-Y1: it is very easy to use.
- PC-Z1: the mobility and extensibility are very high.

The speed of microprocessor and the capacity of hard drive are a measure of basic performance, the resolution and the capacity of VRAM are a measure of the performance of display, the weight and dimensions are a measure of mobility, and the maximum capacity of memory and the interface are a measure of extensibility. The key size is one of a measure of usability. From this correspondence, it is considered that the valid features are extracted in our method. The similar results are obtained from other specifications.

Finally, we carried out the experiment of weighting and matching against a user’s request using table structure converted from 38 specifications about notebook PC. We took 4 requests: (A) PC with high performance, (B) PC for practical use, (C) affordable PC and (D) mobile PC. We compared the result with recommended PCs appeared in the magazine “Nikkei Best PC”. For the requests C and D, same results as the article were obtained. The best PCs for the requests A and B in the article are included in the top five ranking of our results.

6 Discussion and Conclusion

In this paper, we have presented a method of information extraction from specifications on the Web, focusing on personal computer products. The conversion ratio of HTML-based specifications into table structure is 79.5%. On the other hand, in information extraction from the table structure, satisfactory results were obtained. The normalization using table structure seems to be effective.

Here we discuss what kind of domains our framework of table structure can be applied to. The table structure is a set of ternary list (object, attribute, value), and is very simple. It can be applied to the domain of which specifications are represented in ternary list. For example, the domain of household electric appliance, mobile station, automobile and so on is considered. The problem is knowledge acquisition about the domain and specifications. This knowledge must be manually provided to the system and updated whenever the information sources change. There are two possible solutions to the problem: use of some learning mechanism and partition of knowledge into domain independent and domain specific one.

Some web sites already have services comparing many products on WWW. Most of the sites are constructed manually. We have proposed a method to construct the similar contents to the Web pages of these sites. The idea that product information extracted from Web sites are integrated is also suggested [9]. Specifications expressed in the form of a table play an important role in product information extraction. The characteristics of our approach are as follows:

- Introduction of table structure to normalize the variety of description and presentation of the conversion algorithm of HTML-based specifications into it.
- Feature extraction from table structure and sentence generation from features.
- Finding products relevant to a user's request from specifications.
- Multimedia summarization based on integration of sentences and images [5].

Future studies will be undertaken in the following areas: (1) application of more complex domains, (2) evaluation of users, (3) introduction of machine learning, (4) integration of not only specifications but also text and images, (5) cooperative processing with search engine.

References

[1] W. Cohen, The Whirl approach to information integration, IEEE Intelligent SYSTEMS, Vol.13, No.5, pp.20-24, 1998.

[2] K. Shimada and T. Endo, Sentence generation from table structure of extracted important data, Technical Report of IEICE, TL99-29, pp.25-31, 1999 (in Japanese).

[3] P. Martine and P. Eklund, Knowledge retrieval and the World Wide Web, IEEE Intelligent SYSTEMS, Vol.15, No.3, pp.18-25, 2000.

[4] A. Kawai et al., Automatic extraction of relational information using document structure and tabular forms, Trans. of IEICE, Vol.J81-DII, No.7, pp.1609-1620, 1998 (in Japanese).

[5] K. Shimada, T. Ito and T. Endo, Classification of images using their neighboring sentences, Proc. of PACLING2001, 2001.

[6] R. Belew, Finding out about: a cognitive perspective on search engine technology and the WWW, Cambridge University Press, 2000.

[7] S. Sekine, Information extraction from texts, IPSJ Magazine, Vol.40, No.4, 1999 (in Japanese).

[8] Y. Ichimura et al., Text mining: case studies, Journal of Japanese Society for Artificial Intelligence, Vol.16, No.2, 2001 (in Japanese).

[9] R. Doorenbos et al., A scalable comparison-shopping agent for the World Wide Web, Proc. of the first International Conference on Autonomous Agents, 1997.