Using the Diff Command in Patent Documents

Masaki Murata and Hitoshi Isahara
Communications Research Laboratory
2-2-2 Hikaridai, Seika-cho, Soraku-gun, Kyoto 619-0289, Japan
{murata,isahara}@crl.go.jp

Abstract

We have been studying natural language processing using the Unix command Diff. We previously proposed ways to use Diff in natural language processing. In this contest, we tried handling three problems by using Diff. One is to extract rewriting rules. The second is to align the claims in a patent to the contents of the patent. The third is to extract differences of claims in a patent. We obtained interesting results in the three studies and showed the usefulness of Diff. These results would be useful for reading and writing patents.

Keywords: *Diff, Patent, Extraction of rewriting rules, Alignment of claims and embodiments in patents, Diff of claims*

1 Introduction

In the PATENT task of NTCIR-3, we participated in the optional task,[1] where the participants can perform any kind of research related to patents. We think that in a PATENT attempt, the optional task is very interesting, because we have already heard that some participants in previous contests wanted to make their studies as freely as they wanted. Various new ideas or new topics will come up in an optional task. These attempts would be novel and valuable. In the other contests, too, we hope that such attempts will be made.

In this contest, we made the following three studies for the optional task of PATENT.[2]

1. We extracted rewriting rules using data of patents.

2. We aligned the claim of a patent and its embodiment.

3. We extracted differences among plural claims in a patent.

The first two topics were given by organizers of PATENT as examples of the optional task. We consider these studies to be very interesting. The last topic is our idea. We sometimes write a patent, and had the experience of wanting to know the difference of claims. So, we did this study.

We have been studying natural language processing using the Unix command Diff [6, 8, 9]. We previously proposed ways to use Diff in natural language processing. The Diff command is very suitable for doing the above three studies. We have already extracted rewriting rules by using Diff in some research topics. For example, we used a pair of definition sentences having the same word entry in two different dictionaries and extracted the differences between them. These extracted differences can be used as synonym phrases because the definition sentences in the same entry have the same meaning. In another situation, we used aligned spoken-language and written-language texts and extracted the differences between them. These extracted differences can be used as rewriting rules transforming spoken-language sentences into written-language sentences or transforming written-language sentences into spoken-language sentences. Diff can also be used for alignment. Diff has a function of merging data like a DP-matching algorithm. So we can align two related texts by using Diff. In this study, we used this function for the alignment of a patent claim and its embodiment (working example). Finally, we used Diff for extracting the differences of patent claims. Extracting differences is an original function of Diff. Extracting differences between claims enables us to understand the claims of a patent more deeply.

2 Diff and Mdiff

In this section, we describe Diff. Diff is a software program of Unix systems that is used to compare files.[3] The program displays line-by-line differences between a pair of text files while retaining the order of the data. For example, suppose we have the following two files:

[1] We did not participate in the main task (retrieval task).

[2] Our proposals that we use Diff for the three methods had been already proposed by us in the web page (http://ntcirftp.nii.ac.jp/ntcpat/optional/proposal/crl.html).

[3] This section is from our paper [6, 9].

```
File 1:               File 2:
I                     I
go                    go
to                    to
school.               university.
```

When we give these data to Diff, the difference is displayed in the following way.

```
< school.
> university.
```

Diff has a *-D* option, which is very useful. When we use Diff with this option, common parts as well as differences are displayed. That is, files can be merged by using this option. However, the output of *diff -D* is in a form which is used for a C preprocessor such as “Ifdef” and this is difficult for people to read. Therefore, in this paper, we give the differences in the following way:

```
;===== begin =====
(The parts which only exist
in the first file)
;-----------------
(The parts which only exist
in the second file)
;=====  end  =====
```

where, “`;===== begin =====`” indicates the beginning of the differences, “`;=====  end  =====`” indicates the end of the differences, and “`;-----------------`” indicates the boundary between the two sets of data. In this paper, we refer to Diff in the case where the files are merged by using the `-D` option and the differences are displayed in the above form as Mdiff.

When we give our earlier pair of files to Mdiff, we obtain the following result.

```
I
go
to
;===== begin =====
school.
;-----------------
university.
;=====  end  =====
```

“I go to” matches, while “school” and “university” are differences. The output of Mdiff is easy to examine and understand because, unlike Diff, it also displays the common parts.

We can reproduce the two original files from the output of Mdiff. When we take the common part and the upper part of the differences, we obtain the contents of the first file. When we take the common part and the lower part of the differences, we obtain the contents of the second file. We can reproduce all of the original data in this way.

Table 1. Examples of a claim and its embodiment

A claim sample	Its embodiment sample
Claim 1.	The
A	method
method	predicts
for	whether
predicting	data
whether	that
data	is
that	not
is	yet
not	known
yet	to
known	be
to	positive
be	or
positive	not

Since Mdiff only displays the common part of the data once, it is able to reduce the amount of data. Since it is possible to fully reproduce the original data from Mdiff’s output, we are able to say that Mdiff compresses the data while retaining the original information.

Since the output of Diff is difficult to read and the output of Mdiff contains all information output by Diff, we use Mdiff for our explanations in the following sections. In the following sections, let’s look at some actual examples of the application of Mdiff to patents.

3 Extraction of rewriting rules in patents

We have already confirmed that two different texts having the same meaning can be used to extract the rewriting rules (paraphrases) [5, 8]. In the paper [8], we used the spoken-language and written-language and extracted the difference between them. These extracted differences could be used as rewriting rules transforming spoken-language-like sentences into written-language-like sentences or transforming written-language-like sentences into spoken-language-like sentences. We performed the study on the transformation of written-language-like sentences into spoken-language by using the extracted transformation rules. In the paper [5], we used definition sentences of two different dictionaries and extracted the differences between them. These extracted differences could be used as synonym phrases. We performed the study on a sentence compression (summarization) and sentence polishing-up system using extracted synonym phrases [7].

Table 4. The differences between claims and embodiments (Top 15)

Freq.	Matching part in front	Claims	Embodiments	Matching part behind
65087	、	前記		給湯器
63369	気筒	（		Ｃ１～
42577	、	<RET>		タップホール
31058	Ｃ６	）		の
18566	バックプレート		１	に対する
17212	アンドチューブ式コンデンサ		２	、
14195	他方の隣接端がそれぞれ		、	シュー間隙調整アジャスター
13962	において		は	、
13753	制御量		）	を目標スロットル開度の
13674	外嵌し、この継ぎ輪		３	の内周面と各管部
12201	。<RET>	【請求項	（	２
11875	から		の	外側
11864	２	】<RET>	）	相対運動を確立するため
11046	継ぎ輪		４	外部とを連通する
9954	、	上記		第１の通路

In this contest, we made the following two kinds of experiments for extraction of rewriting rules from patents.

1. Extraction of rewriting rules by comparing a claim of a patent and its embodiment.
2. Extraction of rewriting rules by comparing an abstract of a patent and its abstract generated by JAPIO.

We describe these two experiments in the next two sections.

3.1 Comparison of a claim and its embodiment

We describe our method of extracting the differences between a claim and its embodiment. We first divide a claim and its embodiment into words by using the Japanese morphological analyzer ChaSen [3] and obtain the results shown in Table 1. In these samples, we used English ones. Next, we use Mdiff for these data and obtain the results shown in Table 2. From these results, we obtain the differences shown in Table 3.

We used this method for the Japanese patents and obtained many differences. We used about 100,000 patents for the experiments. We obtained 4,444,046 kinds of differences. We extracted regions from “【.*実施.*】” to “【(.*効果.*|.*説明.*)】” as embodiments from patents. We showed the differences having the highest 15 frequencies in Table 4.[4] <RET> is the code of carriage return. The other SGML tags were originally contained in the patent data. From Table 4, we learned many things. For example, we found that the index number of a component of a system (“１” in “バックプレート１” (back plate)) written in embodiments were often eliminated in claims. We found that “前記” (mentioned previously) and “上記” (mentioned above) written in claims were often eliminated in embodiments. We found that “、” (,), “は” (case-particle), and “の” (of) were supplemented in embodiments. Such interesting results could be extracted very easily by using Mdiff. This method can also obtain synonym phrases. We extracted synonym phrases shown in Table 6. We consider that sentences in embodiments are simpler and easier to understand than sentences in claims. Extracted synonym phrases as shown in Table 6 can be used to transform difficult sentences like claims into easier sentences like embodiments.

3.2 Comparison of an original abstract and JAPIO's abstract

Next, we used an original abstract and the corresponding JAPIO abstract to extract transformation rules. We used the same method as the previous section. Although a patent contains an abstract, JAPIO made a new abstract based on an original abstract by supplementing some explanations. We think that the extraction of transformation rules of an original abstract into a JAPIO abstract is interesting. If we can do so, we may be able to transform an original abstract into a JAPIO abstract automatically.

[4] Although, in this study, we used the frequency for obtaining good differences, we have already developed better methods using probabilities and used these methods for our previous papers [5, 8]. These methods need much time, so we did not use them in this study. Katoh et. al. used DP matching and frequencies of differences to extract the transformation rules for automatic summarization [2].

Table 5. The differences between orginal abstracts and JAPIO's abstracts (Top 15)

Freq.	Matching part in front	Original abst.	JAPIO's abst.	Matching part behind
331309	補強材で補強し	<BR>		た枕枠に、モミガラ、ソバガラ等
85681	。	<RET>	</P><RET>	
38422		【課題】	<P>	生ごみを乾式粉砕して
22309	２	）		で表される繰返し単位
17729	型開き面の相当直径	（		Ｄ
14055	支持台５，６が、	前記		軸を基準として互いに同じ側に配
9913	た	データ	データ	とを比較して砥石３の
9071	鉱物金網３５、鉄１０	によって	によつて	ミネラルイオン水に
8464	ら構成させ、	該		炭化物非形成金属層
8181	ミネラルイオン水に	す<BR>る	する	。このミネラルイオン水に
6276	編集された優先順位を基に		、	加工
5776	する	こと		。
3739	の内面処理を	、		好適に自動化すること
3719	＾	【課題】	<P>（Ｊ）	金属、
3690	ダイヤモンド砥粒		１	と砥粒を結合する結合

We used about 100,000 patents for the experiments. We obtained 870,145 kinds of differences. The results are shown in Table 5. From the results, we found that JAPIO transformed a small-font character "っ" into a normal-size character "つ" and "ー" in "データ" to "ー".[5] We found that 【課題】 (subject or theme) which is a mark describing the subject of a patent in an abstract was transformed into <P> which indicates a mark of a boundary of paragraphs. We found that <BR> inserted into an original abstract was eliminated in JAPIO's abstracts. We also found that "１" in "ダイヤモンド砥粒１" (Diamond abrasive 1) which is an index number of a component of a system was supplemented in JAPIO's abstracts. Our extracted differences contain many kinds of interesting transformation rules shown in Table 7. These rules would be very useful to transform an original abstract into a JAPIO abstract automatically.

4 Alignment of claims and embodiments in patents

This section describes alignment of claims and embodiments in patents. We had studied alignment of a written paper and its corresponding speech using Diff [6, 9]. In this section, we used the same idea. We describe the method in the case of patents.

Here, let's try to determine the parts of the embodiments to which each claim corresponds by using Mdiff. We suppose that the content is laid out in the same order in the claims and in the embodiments. In advance, we place symbols such as `<Claim 1>`, as shown in Figure 1, into the claims. This is so that we are easily able to recognize the claims of the patent.[6] By applying Mdiff to claims of a patent and its embodiments after both have been transformed so that each line has one word, we obtained the results shown in Figure 2. Next, we obtained the results shown in Figure 3 by eliminating the upper parts of the differences, i.e., those that correspond to the claims, and leaving symbols such as `<Claim 1>` in place. The symbols such as `<Claim 1>` are only inserted in the data of the embodiments. We are then able to recognize the claims of the patent that correspond with a given part of the embodiments.

`<Claim 1>`
(the contents of Claim 1)
`<Claim 2>`
(the contents of Claim 2)
`<Claim 3>`
(the contents of Claim 3)

Figure 1. Structure in the claims

To put this simply, we place information to indicate claim numbers in the data of embodiments. We then use the merging function of Mdiff to match the parts of the claims and embodiments. Then, by eliminating the contents of the claim, we are left with information on the claim numbers. We can easily align a claim with a corresponding embodiment by using Mdiff.

We used this method for the first thousand patents among the patents given from PATENT of NTCIR-3. Some patents were aligned roughly well. Some patents

[5] A small-font character is right in Japanese grammar. "ー" is more natural in Japanese character fonts. Use of a normal-size character for "っ" and "ー" for "ー" is JAPIO's special notation.

[6] In Japanese patents, we can use existing symbols such as "【請求項１】" instead of `<Claim 1>`.

Table 2. The results of Mdiff of a claim and its embodiment

The results of Mdiff
;===== begin =====
Claim 1.
A
;——————
The
;===== end =====
method
;===== begin =====
for
predicting
;——————
predicts
;===== end =====
whether
data
that
is
not
yet
known
to
be
positive

Table 3. The differences between a claim and its embodiment

A claim	Its embodiment
Claim 1. A	The
for predicting	predicts

were not aligned well at all. We did not examine the accuracy rates. One example of roughly good alignment is shown in Table 5. The claims of it are shown in Table 4. Words surrounded by brackets, “《” and “》”, in Table 5 appear in claims and embodiments. In Table 5, two marks for index numbers for claims, “【請求項１】” (Claim 1) and “【請求項２】” (Claim 2), were inserted. Paragraph 15 (【００１５】) describes Claim 2, so the result is roughly good. However, Paragraph 16 contains the content of Claim 1, not of Claim 2. So, the result is not so good. The reasons for producing bad results are as follows: The contents in embodiments are not in the same order as in the claims. The contents of a certain claim appear in several separate parts of embodiments. Mdiff can be used only when the contents of a certain claim appear in one part of embodiments and it appears in the same order as in the claim. There are many cases when our supposition is not true. We may have better used another method such as passage retrieval based on a bag of words.[7] However, our method of using Mdiff is a very simple method, and it can be used very easily. We can obtain even a few useful results, such as in Table 5, very easily by using Mdiff.

Table 6. The differences between claims and embodiments

Claims	Embodiments
備え (provide)	有し (have)
基づい (based on)	応じ (according to)
設け (provide)	取り付け (attach)

```
;===== begin =====
<Claim 1>
(content of the claim only)
;——————
(content of the embodiment only)
;===== end =====
(content of the claim and embodiment)
;===== begin =====
(content of the claim only)
;——————
(content of the embodiment only)
;===== end =====
(content of the claim and embodiment)
;===== begin =====
<Claim 2>
(content of the claim only)
;——————
(content of the embodiment only)
;===== end =====
```

Figure 2. Results of applying Mdiff to claims of a certain patent and embodiments of it

5 Extraction of differences between claims

We finally made experiments extracting differences among plural claims in a patent. This study is a new trial. We applied Mdiff to two claims among plural claims in a patent after both claims had been trans-

[7] For the alignment of the transcription and its corresponding paper, the technologies of information retrieval or similar methods were used [11, 1].

Table 7. The differences between original abstracts and JAPIO's abstracts (Same meaning)

Freq.	Original	JAPIO
1752	提供する (provide)	得る (obtain)
1350	ことを目的とする (that is a goal)	
1135	前記 (mentioned previously)	上記 (mentioned above)
1129	され (is done)	し (do)
1011	該 (the corresponding)	この (this)
837	ことにある (that is)	
795		ようにする (so as to)
756		できるように (so that)
751	当該 (the corresponding)	
711	ことである (that is)	
692	ものである (that is)	
640	ことを特徴とする (that is a characteristic)	
579	し (do)	する。そして (do. And)

`<Claim 1>`

(content of the embodiment only)

(content of the claim and embodiment)

(content of the embodiment only)

`<Claim 2>`

(content of the embodiment only)

Figure 3. Results of insertion of information on claim numbers in embodiments

【請求項１】
刃部材の先端の刃部を凹凸に形成し波状刃とするとともに螺旋状に湾曲させせ、前記刃部材に取っ手を取り付けたことを特徴とする草取り鎌。
【請求項２】
取っ手の上部及び下部に滑り止め部を設けたことを特徴とする草取り鎌。

Figure 4. A sample of claims of a certain patent

formed so that each line has one word. We obtained results such as in Figures 6 and 7. Figure 6 is the result of claims in Figure 4 and Figure 7 is the result of claims in Figure 8. From this result, we can understand differences of claims very easily. For example, in Figure 6 we found that the differences are “上位装置” (host system) and “プリンタ” (printer). However, when the differences are more complicated, the results of Mdiff are difficult to see as shown in Figure 7. We found that Mdiff has trouble when differences are complicated.

To solve this problem, we developed two new methods of obtaining differences. These methods do not use the Diff command, so they are not suitable for our purpose of using Diff. However, these new methods are useful, so we show them here.

The first new method extracts all words from all claims of a patent other than the currently analyzed claim and specifies words in the currently analyzed claim that do not appear in other claims. The result is shown in Figure 9. The words that do not appear in other claims are surrounded by brackets, “《” and “》”. The result is easier to see than the result of Figure 7. From Figure 9, we very easily found that the characteristic of “【請求項２】” (Claim 2) is “上部及び下部の滑り止め部” (a device to prevent slipping at the upper and lower parts). If we understand the characteristic of Claim 2 is “滑り止め部” (a device to prevent slipping), we can extract embodiments corresponding to Claim 2 easily by extracting paragraphs containing this term “滑り止め部” (a device to prevent slipping). In the example of Figure 5, we can obtain the correct corresponding part, Paragraph 15, which contains the term “滑り止め部” (a device to prevent slipping). This method is very useful for extracting distinguishing characteristics and it can be also used for extracting the corresponding part of embodiments to a certain claim, i.e., aligning claims and embodiments in Section 4.

Next, we used the first new method for another case which has three claims such as in Figure 10. In this case, we obtained a result shown in Figure 10. In this result, we could not obtain the characteristics of Claim 2 and Claim 3, “滑り止め部”. To solve this problem, we made the second new method. This method extracts all words from all claims of a patent above the

【請求項１】

次に、本発明について図面を参照して説明する。図１は本発明である草取り鎌の正面図、図２は本発明である草取り鎌の背面図、図３は、本発明である草取り鎌の右側面である。

【０００７】本草取り鎌１は、図３に示すように、《刃部材》２《の》刃部２ｂは当該《先端の》一面が波状の波状《刃》５に形成されているとともに背面が平坦に形成されている刃部材２と、取っ手３から構成されている。

【０００８】刃部材２は、図１、図２及び図３に示すように、延長《部》２ａがあり取っ手３の約２倍程の長さがある。波状刃５の刃部２ｂは一方向に湾曲している。

【０００９】図４は本発明の草取り鎌の刃部の正面拡大図である。図に示すように、雑草《を》刈り取る刃部２ｂは、凸部５ａと凹部５ｂが交互《に》存在《し波状》《と》なっている。

【００１０】図５は本発明である草取り鎌の刃部の拡大図である。刃部２ｂを構成《する》凸部５ａの先端はやや左方向《に》傾いている。これは、雑草をより引っ掛けて刈り取り易くするためである。

【００１１】図６は本発明である草取り鎌の刃部の《湾曲》状態を示した一部拡大図である。図に示すように《、》《刃》部２ｂの延長部２ａより刃部２ｂの先端２ｃは垂直線６からより湾曲している。

【００１２】図７は、図中のＡ－Ａ線《に》沿っ《た》断面図である。刃部２ｂの上面７は傾斜し、凸部５ａの先端５ｃは尖っている。そして、刃部２ｂ自体が湾曲《する》とともに螺旋している。

【００１３】図８は、本発明である《草取り鎌》の他の実施例の正面図、図９は本発明である草取り鎌の他の実施例の背面図、図１０は本発明である草取り鎌の他の実施例の右側面図、図１１は、本発明である草取り鎌の他の実施例の一部拡大図である。

【請求項２】

【００１４】本例の草取り鎌１ａは、刃部材２の延長部２ａが短いとともに刃部２ｂの部分がやや大きく形成してある。

【００１５】また、《取っ手》３が長く、握り部３ｂ《の》上《に》、握り部３ｂの径よりやや大きい径の上滑り止め部３ａを設けるとともに、下にも同様に握り部３ｂより大径の下《滑り止め部》３ｃｂ《を設け》てある。

【００１６】図１０に示すように、本例の草取り鎌１ａの刃部２ｂも図１から図７までに示し《た》草取り鎌１と同様に螺旋状に湾曲している。

【００１７】このように、先端部が螺旋状に湾曲させる《こと》により、芝生等に生えている雑草《を》根こそぎ取り除くことが容易《と》なる《。》

Figure 5. Actual results of insertion of information on claim numbers in embodiments

```
前記プリンタシステムは
;===== begin =====
上位装置
;___________
プリンタ
;===== end =====
を有することを特徴とする請求項１６記載のプリンタシステムの制御方法。
```

Figure 6. A result of Mdiff of claims (Case 1)

```
;===== begin =====
刃部材
;___________
取っ手
;===== end =====
の
;===== begin =====
先端の刃
;___________
上部及び下部に滑り止め
;===== end =====
部を
;===== begin =====
凹凸に形成し波状刃とするとともに螺旋状に湾曲させ、前記刃部材に取っ手を取り付け
;___________
設け
;===== end =====
たことを特徴とする草取り鎌。
```

Figure 7. A result of Mdiff of claims (Case 2)

currently analyzed claim and specifies words in the currently analyzed claim that are not appearing in the above claims. The result is shown in Figure 11. In this case, we could extract the characteristics of Claim 2 and Claim 3, "滑り止め部", in Claim 2. With this method, we can find new terms as distinguishing characteristics. This method is also useful.

Although these two new methods did not use Diff commands, they used an idea of extracting differences in Diff commands. We think that these methods have interesting aspects as Diff has. In the future work, we would like to use these two methods for other applications of natural language processing.[8]

[8] For example, we are now thinking of using the second new method to examine the problems of *new information* and *old information* in anaphora resolution [10, 4].

【請求項１７】前記プリンタシステムは上位装置を有することを特徴とする請求項１６記載のプリンタシステムの制御方法。

【請求項１８】前記プリンタシステムはプリンタを有することを特徴とする請求項１６記載のプリンタシステムの制御方法。

Figure 8. The claims of Case 2

【請求項１】《刃部材》の《先端》の《刃》部を《凹凸》に《形成し波状刃》とする《とともに螺旋状》に《湾曲させ、前記刃部材》に取っ手を《取り付け》たことを特徴とする草取り鎌。

【請求項２】取っ手の《上部及び下部》に《滑り止め》部を《設け》たことを特徴とする草取り鎌。

Figure 9. A result of New method 1

【請求項１】《刃部材》の《先端》の《刃》部を《凹凸》に《形成し波状刃》とする《とともに螺旋状》に《湾曲させ、前記刃部材》に取っ手を《取り付け》たことを特徴とする草取り鎌。

【請求項２】取っ手の上部に滑り止め部を設けたことを特徴とする草取り鎌。

【請求項３】取っ手の上部《及び下部》に滑り止め部を設けたことを特徴とする草取り鎌。

Figure 10. A result of New Method 1 with another case

6 Conclusion

In the PATENT task of NTCIR-3, we participated in the optional task, where the participants can perform any kind of research relating to patents, and we made the following three kinds of studies.

1. We extracted rewriting rules using data of patents.
2. We aligned a patent claim and its embodiment.
3. We extracted differences among plural claims in a patent.

We had shown many examples in the previous papers [6, 8, 9]. In the current study, we could also show more examples of a Diff command and show its usefulness and effectiveness. We hope that Diff will be applied to an ever-widening range of studies.

In Section 5, we showed two new methods not using Diff. These methods would also be interesting. In future work, we would like to use them for many other applications of natural language processing.

【請求項１】《刃部材の先端の刃部を凹凸に形成し波状刃とするとともに螺旋状に湾曲させ、前記刃部材に取っ手を取り付けたことを特徴とする草取り鎌。》

【請求項２】取っ手の《上部》に《滑り止め》部を《設け》たことを特徴とする草取り鎌。

【請求項３】取っ手の上部《及び下部》に滑り止め部を設けたことを特徴とする草取り鎌。

Figure 11. A result of New Method 2

References

[1] A. Fujii, K. Itou, T. Akiba, and T. Ishikawa. A structure-based method for speech summarization. *Proceedings of the Spontaneous Speech Science and Technology Workshop*, 2001. (in Japanese).

[2] N. Katoh and N. Uratani. A new approach to acquiring linguistic knowledge for locally summarizing Japanese news sentences. *Journal of Natural Language Processing*, 6(7), 1999.

[3] Y. Matsumoto, A. Kitauchi, T. Yamashita, Y. Hirano, H. Matsuda, and M. Asahara. Japanese morphological analysis system ChaSen version 2.0 manual 2nd edition. 1999.

[4] M. Murata. Anaphora resolution in Japanese sentences using surface expressions and examples. *Doctoral dissertation, Kyoto University*, 1996.

[5] M. Murata and H. Isahara. Automatic paraphrase acquisition based on matching of two texts with the same meaning. *Information Processing Society of Japan, WGNL 142-18*, 2001. (in Japanese).

[6] M. Murata and H. Isahara. Nlp using diff. *IPSJ-WGNL 2001-NL-144*, 2001. (in Japanese).

[7] M. Murata and H. Isahara. Universal model for paraphrasing — using transformation based on a defined criteria —. In *NLPRS'2001 Workshop on Automatic Paraphrasing: Theories and Applications*, 2001.

[8] M. Murata and H. Isahara. Automatic extraction of differences between spoken and written languages, and automatic translation from the written to the spoken language. In *LERC 2002*, 2002.

[9] M. Murata and H. Isahara. NLP using DIFF — use of convenient tool for detecting differences, MDIFF —. *Journal of Natural Language Processing*, 9(2), 2002. (in Japanese).

[10] M. Murata and M. Nagao. Determination of referential property and number of nouns in Japanese sentences for machine translation into English. In *Proceedings of the 5th TMI*, pages 218–225, 1993.

[11] K. Uchimoto, C. Nobata, K. Ohta, M. Murata, Q. Ma, and H. Isahara. Segmenting the transcription of a talk by aligning the transcription with its corresponding paper. *7th Annual Meeting of the Association for Natural Language Processing*, pages 317–321, 2001. (in Japanese).