A Dependency Parsing Method Based on the Hierarchical Structure in Japanese Language

Kazuki Ono

Graduate School of Culture and Information Scicence
Doshisha University
1-3 Tatara-Miyakodani, Kyotanabe, Kyoto, 610–0394, Japan
Email: ono@ilab.doshisha.ac.jp

Kenji Hatano

Faculty of Culture and Information Scicence
Doshisha University
1-3 Tatara-Miyakodani, Kyotanabe, Kyoto, 610–0394, Japan
Email: khatano@mail.doshisha.ac.jp

Abstract—**In this paper, we propose a new statistical dependency parsing method in Japanese language based on an extended hierarchical language model. Conventional Japanese statistical dependency parsers are primarily based on the bi-nominal dependency model between phrases, which has a limitation related to the order of phrases. Accordingly, the length of the phrase, which depends of dependency is limited by this limitation, the model is lacking. We propose a dependency model in Japanese language that considers the order of phrases based on the hierarchical Pitman-Yor process in order to overcome this limitation. Consequently, compared with conventional methods, our method can parse dependencies in long and complex Japanese sentences relatively well.**

Keywords-Dependency Parsing; Syntax Analysis; Syntax Tree Modeling

I. INTRODUCTION

Asian languages are based on case grammar and in many of them, sentences are composed in the order subject, object, and verb (SOV), giving rise to them being termed SOV languages. Japanese has the same characteristics; however, its phrase order is relatively unfettered.

In English, the syntactic function of each phrase is represented by phrase order, while in Japanese, postpositions represent the syntactic function of each phrase. Further, phrases that comprise one or more postpositions following a noun, which play a similar role to declension of nouns, have been devised and used to syntactically analyze Japanese sentences.

Hence, it may be said that syntactic analysis is equivalent to parsing dependencies between phrases. This is because the semantic rules of Asian languages are explained as relationships of phrases. As a result, conventional dependency parsing methods in such languages utilize the bi-nominal dependency model to decide whether a dependency relation exists between pairs of phrases or not [1]–[3].

In the syntactic analysis used for English, on the other hand, Tree Substitution Grammar (TSG) has attracted attention as a language model [4]. TSG helps to establish the highest accuracy of English parsing, including hierarchical structure, by learning the rewrite rules of any depth [5], [6].

The hierarchical structure is defined as a set of rewriting rules of Context Free Grammar (CFG). Syntax trees generated by TSG describe the order of phrases, whereas those generated by case grammar only represent relationships between phrases. In short, the syntax trees generated by TSG not only have the relationships between phrases but also the orders of phrases, so that they may facilitate precise extraction of dependencies between phrases.

Incidentally, the syntax trees from case grammar are generated based on the bi-nominal model [1], [2]. As a consequence, the order of phrases in these syntax trees do not include a hierarchical structure. They are instead subject to constraints related to the constant order of phrases. As a result, dependency parsing of Asian languages using the bi-nominal dependency model is not very successful. We believe that the accuracy of dependency parsing of Asian languages can be improved by applying TSG to them. Consequently, in this paper, we propose a dependency parsing method in Japanese language based on TSG that considers the syntax tree with context dependency.

Our method is characterized by the handling of exchangeable sequence of phrases in case grammar to utilize the extracted hierarchical structure using TSG. That is, we can generate a dependency parsing model by calculating the probabilities of integrating phrase dependencies from supervised training data with a hierarchical structure. Thus, it can be said that our approach is a novel dependency parsing method in Japanese language with hierarchical structure relations because of the way it handles the relationships of each of the phrase dependencies.

II. BASIC PROCESSES

In this section, we discuss the basic processes used in our research. These are, specifically, the Pitman-Yor process [7] and its extension called the Hierarchical Pitman-Yor process [8], which we use to generate a syntax tree model based on TSG [5].

A. The Pitman-Yor Process

The Pitman-Yor process is a non-parametric Bayesian model [7]. In natural language processing, it is used to generate an n-gram model. It is a stochastic process that generates an infinite discrete probability distribution G, and is denoted by $\mathrm{PY}(d, \theta, G_0)$. $\mathrm{PY}(d, \theta, G_0)$ has three parameters: d is a discount parameter with $0 \leq d \leq 1$, θ is a strength parameter that meets the condition $\theta \geq -d$, and G_0 is a base distribution over a probability space. In natural language processing, the probability space is usually generated from the probabilities of phrase occurrences.

When d is zero, $\mathrm{PY}(d, \theta, G_0)$ becomes the Dirichlet process, denoted $\mathrm{DP}(\theta, G_0)$. In short, because $\mathrm{DP}(\theta, G_0)$ can generate an infinite dimensional Dirichlet distribution, $\mathrm{PY}(d, \theta, G_0)$ can also support it. As outlined above, d is a concentration parameter for G_0; therefore, we can define the following equation:

$$G \sim \mathrm{PY}(d, \theta, G_0) \tag{1}$$

That is, $\mathrm{PY}(d, \theta, G_0)$ is an extended version of $\mathrm{DP}(\theta, G_0)$. Let W be a fixed and finite vocabulary of V phrases. The Pitman-Yor process generates for each phrase $w \in W$ a vector of phrase probabilities $G(w)$. $\mathrm{DP}(\theta, G_0)$ is approximated by Dirichlet distribution $Dir(\theta G_0(w_1), \ldots, \theta G_0(w_i), \ldots, \theta G_0(w_r))$ that expresses any division of the disintegration space for which the size of the phenomenon space is r in observing any phrase w_i as follows:

$$\mathrm{DP}(\theta, G_0) \sim Dir(\theta G_0(w_1), \ldots, \theta G_0(w_i), \ldots, \theta G_0(w_r)) \tag{2}$$

Here, $G_0(w_i)$ is an integral of the base distribution G_0, therefore it is equal to the sum of the probabilities of phrase occurrences in the disintegration space. In short, we can say that the Pitman-Yor process is a recursive stochastic process whose input is a set of phrases w_i and base distribution is G_0.

In general, these procedures for generating n-gram distribution drawn from G are often referred to as the Chinese restaurant process [9]. In the Chinese restaurant process, we fancifully imagine a restaurant with an infinite number of tables whose capacities are infinite. The existing distribution of customers (phrases) who come to the restaurant and the tables (discount strength) with one by one dish (phrase vocabulary) is based on the n-gram model, denoted by G, and the hypothesis to the tables elicits the base distribution G_0. The customers are compared to phrases in the Pitman-Yor process, such that a customer continues to sit at the same table if the customer coming to the restaurant is that same customer. However, if it is another customer that had not previously entered the restaurant, the customer sits down at a new table. Given this scenario, we can formulate the Pitman-Yor process as (3):

$$\mathrm{PY}(d, \theta, G_0) = \frac{c_k - d}{\theta + c_.} + \frac{\theta + dt}{\theta + c_.} p(w_k) \tag{3}$$

where t is the number of customers under the base distribution G_0, c_k is the number of species, $c.$ is the total number of customers, and $p(w_k)$ is the probability of a customer visiting the restaurant. Here, the parameters d and θ is in the Pitman-Yor process are non-parametric. Therefore, we have to generate the distribution approximated by the base distribution G_0 with these parameters using a Gibbs sampling algorithm, such as Markov Chain Monte-Carlo [9].

B. The Hierarchical Pitman-Yor Process

The Hierarchical Pitman-Yor process [8] is a stochastic process that is based on a hierarchical extension of the Pitman-Yor process. An n-gram language distribution over which the current phrase is given various context $\mathbf{u}$, consisting of up to $(n-1)$ phrases, can be described by the Hierarchical Pitman-Yor process. Consequently, an n-gram distribution G_u is generated by the Pitman-Yor process using the base distribution $G_{\pi(u)}$, which is generated in the given context $\mathbf{u}$ as follows:

$$G_{\mathbf{u}} \sim \mathrm{PD}(d_{\pi|\mathbf{u}|}, \theta_{\pi|\mathbf{u}|}, G_{\pi(\mathbf{u})}) \tag{4}$$

where strength and discount parameters are calculated on the basis of the length of context $\mathbf{u}$, $\pi(\mathbf{u})$ is the suffix of $\mathbf{u}$ consisting of all but the earliest phrase, and $G_{\pi(\mathbf{u})}$ is a vector of probabilities of its context.

When we recursively place a stochastic process such as $G_{\pi(\pi(\mathbf{u}))}$ over $G_{\pi(\mathbf{u})}$ using (4), we can define a stochastic process that generates the n-gram distribution. This process is repeated until we get G_ϕ as the base distribution, the vector of probabilities over the current phrase given the empty context ϕ.

As described above, the structure of the stochastic process generated by the hierarchical Pitman-Yor process is expressed as a suffix tree with depth n.

Each node in the suffix tree corresponds to a context consisting of up to $(n-1)$ phrases, and each child corresponds to the addition of a different word to the beginning of the context. The Hierarchical Pitman-Yor process can generate an n-gram language model precisely as demonstrated in the language model based on Kneser–and–Ney smoothing [10].

III. Related Work

In this section, we first discuss a conventional dependency parsing method in Japanese language and its limitations. We then look at how TSG obtains highly precise English language parsing with a hierarchical structure, and examine its application to the parsing of dependencies in the Japanese language.

A. Dependency Parsing based with the Bi-nominal Model

The Bi-nominal approaches, which generate syntax trees, are the conventional approaches used to parse dependencies between phrases in Japanese language [1], [3]. They generate a syntax tree model whether there is a dependency in Japanese language between phrases or not based on features like part of speech, inflectional form, and so on, and conduct syntactic analysis using the generated syntax tree based on the following algorithm:

1) Check all the phrases in a sentence to ascertain whether they have a dependency with others located on the right side of the syntax tree.
2) Designate any phrase that has a dependency in Step 1 as the analysis result. At this time, the referrer of the dependency is excluded from the target of Step 1.

3) Repeat Steps 1 and 2 and keep the analysis results until all but the final phrase has a dependency.

Conventional methods based on the bi-nominal model do not consider the features of only the relationship between two phrases in a sentence. They misjudge some of the dependencies shown in Figure 1.

There is a Japanese sentence “トムはこの本をジムを見た女性に渡した” in Figure 1. This sentence contains a complex structure. The meaning of the sentence in English is “Tom gave this book to a woman who saw Jim.” in English. Originally, the phrase “本を (the book)” has to take the dependency depicted by the dashed line to another phrase “渡した (gave)”, because “the book” becomes the object of “gave”. However, sometimes the dependency from “本を (the book)” to “見た (saw)” as depicted by the solid line in Figure 1, occurs with the bi-nominal approaches. This is because the bi-nominal model cannot consider hierarchical structures. Thus, it is impossible to perform accurate parsing for a statement that has a complex structure.

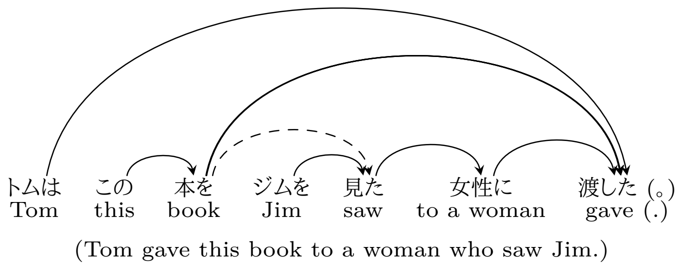

Figure 1. Example of parsing in Japanese language

Therefore, we propose a new method for dependency parsing in Japanese language that uses the hierarchical structure relation of phrases.

B. Tree Substitution Grammar

Tree Substitution Grammar (TSG) is an extension of Context Free Grammar (CFG) [5]. Both CFG and TSG have rewrite rules, called productions, which are used to construct syntax trees by replacing nonterminal symbols with elementary trees which is a part of the syntax tree.

The TSG production replaces a nonterminal symbol with an elementary tree whose depth is greater than one while the CFG production does the same with an elementary tree whose depth is exactly one. As a result, the TSG production has a hierarchical structure based on any context. TSG is 4-tuple and is defined as $G = \{T, N, S, R\}$, where T is a set of terminal symbols, N is a set of nonterminal symbols, $S(\in N)$ is a set of the distinguished root nonterminals, and R is a set of productions. In general, the syntax tree of an English sentence is described as a tree structure because of its phrase-structure rule.

Here, the root node is labeled with a nonterminal symbol, and its leaf nodes are labeled with either terminal symbols or nonterminal symbols. In short, the syntax tree of the input sentence is constructed by recursively replacing the nonterminal symbols with the elementary trees.

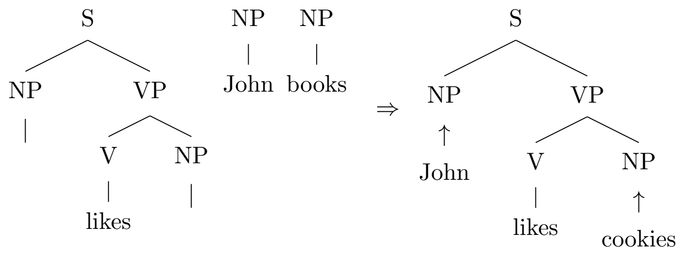

Figure 2. A syntax tree comprising three elementary trees

Figure 2 shows an example of parsing from the syntax tree, which is labeled on the left. For example, the S $\rightarrow$ (NP (VP (V like) NP)) production rewrites the nonterminal symbol S with the fragment (S NP (VP (V likes) NP)). This syntax tree has two NPs as its nonterminal symbols, and the production rewrites these nonterminal symbols with the fragments (NP John) and (NP cookies). In short, a derivation creates a tree by starting with the root symbol and rewriting (substituting) it with an elementary tree, then continuing to rewrite frontier nonterminals with elementary trees until there are no remaining nonterminals.

In TSG, the replacement of nonterminal symbols with elementary trees is performed in an arbitrary manner; however, we can calculate the probabilities of the rewriting rules in the syntax tree. Probabilistic Tree Substitution Grammar (PTSG) is an extension of TSG whose productions have their probabilities $P(e|c)$ where the elementary tree e replaces a nonterminal symbol c.. $P(e|c)$ is statistically calculated on the basis of training data. The distribution of the PTSG production G_c can be generated by the Hierarchical Pitman-Yor process [8] as follows:

$$G_c \sim \mathrm{PY}(d_c, \theta_c, G_{\pi(c)}) \quad (5)$$

where d_c and θ_c are hyper-parameters in the Hierarchical Pitman-Yor process when we give the nonterminal symbol c, and $G_{\pi(c)}$ is a distribution over the infinite dimensional distribution of the elementary tree with c.. To generate e, we now draw e_1 from G_ϕ giving us an elementary tree with nonterminal symbol $c_1, \ldots, c_m$, and then draw $e_2, \ldots, e_m$ in turn from base measure $G_{\pi(c_1)}, \ldots, G_{\pi(c_m)}$. We continue in this fashion until a full tree is generated.

However, a problem associated with the segmentation of elementary trees arises when PTSG is derived. Gibbs sampling, in which random variables are repeatedly sampled conditioned on the current values of all other random variables in the model is usually used to solve this problem. Figure 3 shows an example of segmentation of a syntax tree.

In the procedures described above, we can perform dependency parsing based on hierarchical structure. This method, the state-of-the-art in English dependency parsing [6], is known as Symbol Refined TSG (SR-TSG). SR-TSG applies *symbol refinement* techniques [11], [12] to parse English sentences. These techniques are used in parsing methods that have no hierarchical structure, such as CFG. Thus, although not directly related to TSG, the symbol refined syntax trees constructed by SR-TSG use a state-of-the-art method.

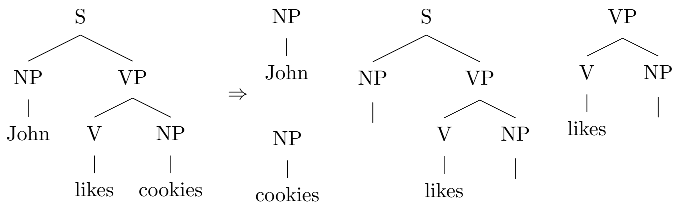

Figure 3. Elementary trees based on PTSG

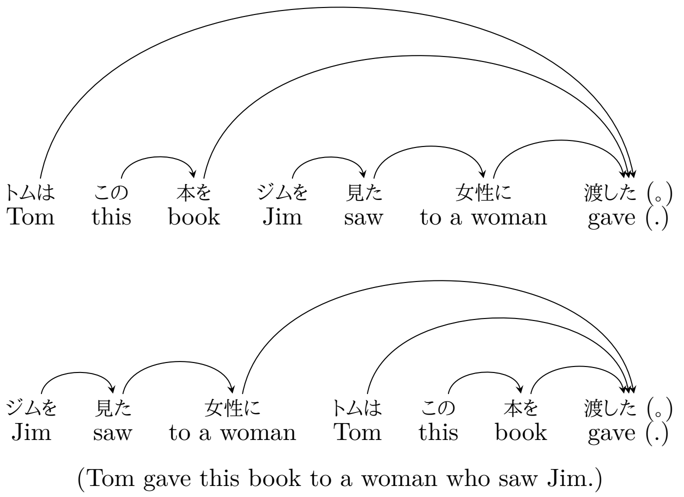

Figure 4. Two Japanese sentences that have the same structure

However, TSG cannot support dependency parsing in Japanese language because its syntax tree cannot have a tree structure as postpositions in the Japanese language represent the syntactic function of each phrase. For example, there are two sentences in Figure 4. The phrase order is different in the two sentences, while each phrase has the same dependencies. Therefore, it is impossible to conduct dependency in Japanese language parsing in TSG which is applied to the language syntax tree to express it as a tree structure. As a result, we propose a novel dependency parsing method in Japanese language that considers both the weak context dependency and the order of phrases using an extension of TSG.

IV. Hierarchical Dependency Parsing

In this section, we propose a method in Japanese language for parsing dependencies with weak contexts.

A. Syntax Tree Construction

In order to consider dependencies between phrases, we generate a syntax tree based on the n-gram model made from the occurrence of dependencies. We can calculate the probabilities of the referrer phrases subject to occurrence of the destruction phrase. For example, in the dependency between phrase "トムは"(Tom) and "渡した"(gave) referrer phrase "トムは"(Tom) depends on the referenced phrase "渡した"(gave). Then, occurrence probability of the dependency between phrase "トムは"(Tom) and "渡した"(gave) is calculated as the

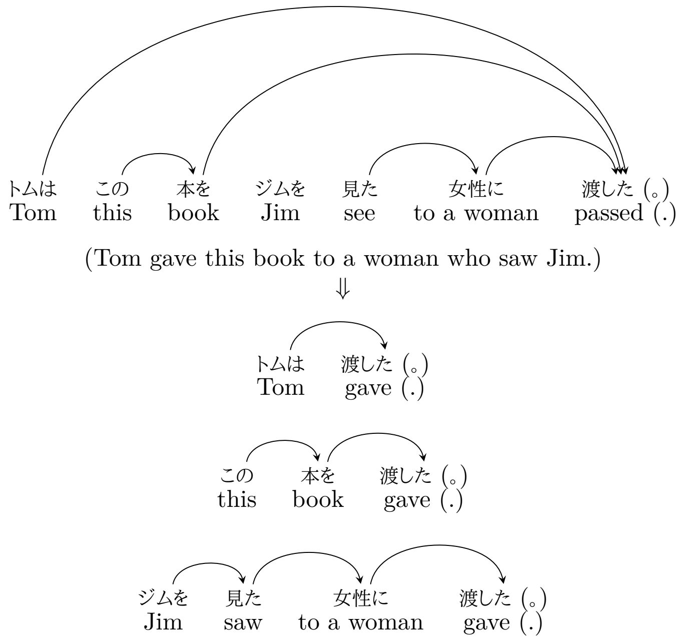

Figure 5. Hierarchical dependencies in the Japanese language

conditional probability $P_D(\text{トムは} \mid \text{渡した})P_D(\text{Tom}|\text{gave})$ that is the *bi*-gram model used as a prior probability $P_D(\text{渡した})(P_D(\text{gave}))$.

At present, we cannot construct an n-gram model that reflects the occurrence of phrases if n is small. Conversely, the size of the n-gram model is large if n is large.

In order to solve this problem, we use Variable-order Pitman-Yor Language Model (VPYLM) [13], an extension of the hierarchical Pitman-Yor process [8]. This technique can help to treat n as as any variable in the n-gram model. Using VPYLM, we can generate a syntax tree model considering the number of phrases when we calculate the probabilities of the referrer phrase as being the root node of the elementary tree. For example, Figure 5 shows three elementary trees comprising the end of a sentence. In short, we can get three hierarchical dependency trees and can calculate the probabilities $P_D(\cdot| \text{渡した})$ of the end of a sentence being the root node. Thus, we can generate a precise syntax tree with weak context dependency.

B. Hierarchical Dependency Parsing Algorithm

Dependencies in Japanese language have the following limitations:

- Japanese is a head-final language. Except for the rightmost one, thus, each segment modifies exactly one segment among the segments appearing to its right.
- Dependencies do not cross one another.

Since the syntax trees constructed by the method described in Figure 6 (a), each of whose root node is the phrase at the end of the sentence, are hierarchical, we parse the sentence in bottom-up fashion using an algorithm called CYK [14] which is based on depth-first search. The phrase at the end of a sentence gets the dependency from the one immediately before it, so we regard the phrase at the end of the sentence and the one immediately before it as the weak context dependency and find another dependency in the sentence.

In Figure 6 (b), for example, the phrase “渡した” is the phrase at the end of the sentence, so that we can get the dependency from “女性に”. As a result, we can calculate the probability in the calculated probability of the dependency P_D(女性に | 渡した)(P_D(to an woman|gave)). In the same fashion, we can calculate different probabilities of dependencies P_D(見た | 女性に渡した)(P_D(saw|gave to an woman)) and P_D(見た | 渡した)(P_D(saw|gave)), if we assume that there is a dependency between “女性に” and “渡した”. Presently, we compare P_D(見た | 女性に渡した)(P_D(saw|gave to an woman)) with P_D(見た | 渡した)(P_D(saw|gave)), and then, extract the dependency between the phrases. If P_D(見た | 女性に渡した)(P_D(saw|gave to an woman)) is large compared with P_D(見た | 渡した)P_D(saw|gave), we decide that there is a dependency between “見た” and “女性に渡した”. These processes continue to work until we get to the phrase at the beginning of the sentence. In the end, everything for which the relationships are adjudged to be dependencies form the syntax tree of the sentence.

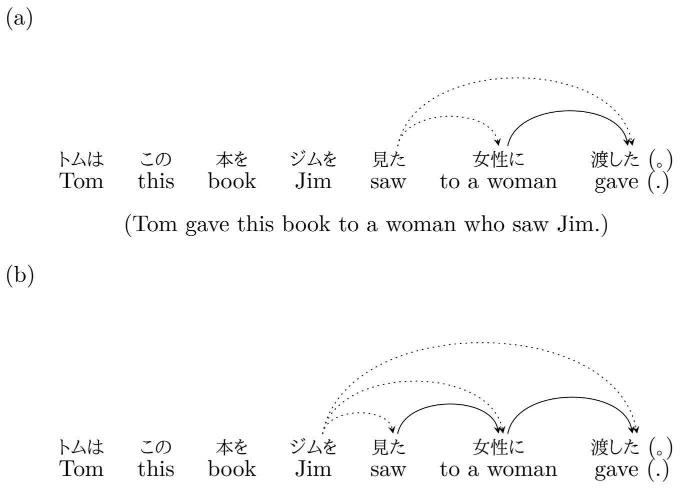

Figure 6. An example of dependency parsing processes using the hierarchical dependency model in Japanese language

V. EXPERIMENTAL EVALUATION

To evaluate our method which considers hierarchical structure in Japanese language, we compared it with a conventional one using the Kyoto University Text Corpus [15]. The Kyoto University Text Corpus is a Japanese text corpus for handling the dependencies of morpheme and segment in Japanese language, and is commonly used to evaluate the effectiveness of Japanese language morphological analysis and dependency parsing in Japanese language. We chose CaboCha [1] as the conventional method, because it is said that CaboCha is the most effective method for parsing dependencies in Japanese language. CaboCha itself also used the Kyoto University Text Corpus for its evaluation.

As stated in Section IV, our method utilizes training data to generate syntax trees and parses dependencies in Japanese language using CaboCha and our method; therefore, we calculated the accuracies of dependencies parsed by our method.

A. Experimental Procedures

Although the Kyoto University Text Corpus comprises newspaper articles for five days, we utilized only one day’s articles (1100 sentences) of dependencies parsed in Japanese language by our method. This is because it takes more time to get our syntax trees, so that we have to save the processing time.

In our procedure, generating syntax trees, we extract dependencies in Japanese language between phrases, next, construct syntax trees focusing on the phrases at the end of sentences that are assigned to the root nodes of the syntax trees. We then apply the hierarchical Pitman-Yor process and estimate parameters θ using the Gibbs iteration.

Finally, we can obtain dependencies in Japanese language using our extracted syntax tree model. In this procedure, we first divide the sentences into segments after performing Japanese language morphological analysis, and check the segment sequences against our syntax trees. This procedure helps to parse dependencies in Japanese language while continuing to extract the dependencies between phrases from the phrase at the end of an input sentence.

B. Experimental Results

We also performed the same processes using CaboCha based on bi-nominal models [1]. To compare our method with CaboCha, we utilized the following measurement denoted by R that is frequently used to evaluate the accuracies of dependencies:

$$R = \frac{Y}{X} \tag{6}$$

where X denotes the number of dependencies in the test data, and Y denotes the number of dependencies extracted by each parser (our method and CaboCha). Of course, dependencies extracted by the each parser are contained in the test data, so that R is usually termed recall in the information retrieval research field [16]. In order to calculate R using each parser, we randomly selected 200 sentences from newspaper articles from other days that were not used to construct syntax trees. That is to say, we focused on the efficacy of the dependencies extracted by each method. Table I indicates that our method

TABLE I. EXPERIMENTAL RESULTS 1

	CaboCha	our method
R	0.871	0.769

is, unfortunately, inferior to CaboCha from the viewpoint of effectiveness. This is because our method utilizes syntax trees not only using a small quantity features for parsing dependencies in Japanese language. For which, CaboCha parses dependencies in Japanese language with various features (word, lexical form of word, POS tag, and inflectional form), so that the accuracies of parsing dependencies in Japanese language are effective using CaboCha. Therefore, the precision of our method may be improved still more.

However, our method is ideal for accurately extracting dependencies in Japanese language in long sentences whose

TABLE II. EXPERIMENTAL RESULTS 2

	Our method	CaboCha
complex and long sentences	0.700	0.550

syntax tree has a complex structure. That is, we also have to evaluate sentences from which CaboCha cannot extract dependencies in Japanese language. Therefore, we randomly selected 20 long sentences (10 percent of the test data in previous experiment) with relatively complex structures.

Table II shows that our method was able to extract dependencies accurately from 14 out of the 20 sentences while CaboCha was only able to do this for eleven. In short, our method was better able to parse dependencies in Japanese language from the long and complex sentences than CaboCha. The reason for this is that it is still possible for the parser to interpret the meaning of the postposition in some cases, even if the syntax tree model does not have a hierarchical structure. However, the syntax trees constructed by our method contain not only the relationships between phrases but also the hierarchical structure of each phrase. Consequently, our method can restrain failed analysis of the complex Japanese sentence in by conventionally method.

VI. CONCLUSION

In this paper, we proposed a method for parsing dependencies in Japanese language using novel syntax trees based hierarchical structure, that are constructed by analyzing the relationships between segment sequences. By considering the relationships between segment sequences, our method was able to solve a phrase order problem. In the near future, we plan to utilize not only the order of words but also the result of syntactic and case structure analysis to improve the accuracies of dependencies in Japanese language. We also plan to conduct experimental evaluations using all the data in the Kyoto University Text Corpus.

ACKNOWLEDGMENT

We would like to give special thanks to all who commented on this work at the IPSJ Kansai Workshop 2013 for their assistance in identifying the weaknesses in our work and also for giving suggestions on how to improve it. This work was partly supported by a Japan Society for the Promotion of Science (JSPS) Grant-in-Aid for Scientific Research (KAKENHI), Grant Number 25540150.

REFERENCES

[1] T. Kudo and Y. Matsumoto, “Japanese Dependency Analysis using Cascaded Chunking,” in Proceedings of Conference on Natural Language Learning, 2002, pp. 63–69.

[2] Y. Cheng, M. Asahara, and Y. Matsumoto, “Machine learning-based dependency analyzer for chinese,” Journal of Chinese Language and Computing, vol. 15, no. 1, 2005, pp. 13–24.

[3] M. Sassano, “Linear-time dependency analysis for japanese,” in Association for Computational Linguistics, 2004.

[4] M. Post and D. Gildea, “Weight pushing and binarization for fixed-grammar parsing,” in Proceedings of the 11th International Conference on Parsing Technologies, 2009, pp. 89–98.

[5] P. Blunsom and T. Cohn, “Unsupervised induction of tree substitution grammars for dependency parsing,” in Proceedings of Conference on Empirical Methods in Natural Language Processing, 2010, pp. 1204–1213.

[6] H. Shindo, Y. Miyao, A. Fujino, and M. Nagata, “Statistical Parsing with Probabilistic Symbol-Refined Tree Substitution Grammars,” in Proceedings of International Joint Conference on Artificial Intelligence, 2013, pp. 3082–2086.

[7] J. Pitman and M. Yor, “The Two-Parameter Poisson-Dirichlet Distribution Derived from a Stable Subordinator,” The Annals of Probability, vol. 25, no. 2, 1997, pp. 855–900.

[8] Y. W. Teh, “A hierarchical Bayesian language model based on Pitman-Yor processes,” in Proceedings of Association for Computational Linguistics, 2006, pp. 985–992.

[9] H. Ishwaran and L. F. James, “Generalized weighted Chinese restaurant processes for species sampling mixture models,” Statistica Sinica, vol. 13, 2003, pp. 1211–1235.

[10] R. Kneser and H. Ney, “Improved backing-off for M-gram language modeling,” in Proceedings of International Conference on Acoustics, Speech, and Signal Processing, 1995, pp. 181–184.

[11] M. Collins, “Head-Driven Statistical Models for Natural Language Parsing,” Association for Computational Linguistics, vol. 29, no. 4, 2003.

[12] T. Matsuzaki, Y. Miyao, and J. Tsujii, “Probabilistic CFG with Latent Annotations,” in Proceedings of Association for Computational Linguistics, 2005, pp. 75–82.

[13] D. Mochihashi and E. Sumita, “The infinite markov model,” in Proceedings of Annual Conference on Neural Information Processing Systems, 2007, pp. 1017–1024.

[14] N. Chomsky, “Three models for the description of language,” IRE Transactions on Information Theory IT-2, vol. 2, no. 3, 1956, pp. 113–124.

[15] S. Kurohashi and M. Nagao, “Building a Japanese Parsed Corpus while Improving the Parsing System,” in Proceedings of Language Resources and Evaluation Conference, pp. 719–724.

[16] R. Baeza-Yates and B. Ribeiro-Neto, Modern Information Retrieval: The Concepts and Technology behind Search (ACM Press Books), 2nd ed. Addison-Wesley Professional, Feb. 2011.