Semi-Automatic Documentation of an Implemented Linguistic Grammar Augmented with a Treebank

Chikara Hashimoto
Graduate School of Informatics, Kyoto University

Francis Bond and Takaaki Tanaka
Machine Translation Research Group, NTT Communication Science Laboratories

Melanie Siegel
Deutsches Zentrum für Luft- und Raumfahrt

Abstract. We have constructed a large scale and detailed database of lexical types in Japanese from a treebank that includes detailed linguistic information. The database helps treebank annotators and grammar developers to share precise knowledge about the grammatical status of words that constitute the treebank, allowing for consistent large scale treebanking and grammar development. In addition, it clarifies what lexical types are needed for precise Japanese NLP on the basis of the treebank. In this paper, we report on the motivation and methodology of the database construction.

Keywords: Documentation, Lexical Types, Linguistic Grammar, Treebank

1. Introduction

Treebanks constructed by a linguistically detailed grammar play an important role in various aspects of natural language processing (Bond et al., 2004b; Toutanova et al., 2005) A detailed grammar in turn is a fundamental component for **precise** natural language processing.

However, such a deep linguistic treebank and a grammar are difficult to keep consistent through development cycles. This is both because multiple people, often in different locations, participate in a development activity, and because deep linguistic treebanks and grammars are complicated by nature.

We have constructed a linguistically enriched treebank named 'Hinoki' (Bond et al., 2004a; Bond et al., 2007), which is based on the same framework as the Redwoods treebank (Oepen et al., 2002) and uses the Japanese grammar JACY (Siegel and Bender, 2002) to construct the treebank.[1] In the construction process, we have also encountered the problem just mentioned. We are aiming to resolve this problem, which we expect many other project groups that are constructing detailed linguistic treebanks have encountered. Our strategy is to take a "snapshot" of one important aspect of the treebank and grammar for each development cycle. To be more precise, we extract information about lexical items that are being used in treebanking from the treebank and grammar and convert it into a structured database (the lexical-type database[2]). Such a snapshot, the database, certainly helps treebank annotators and grammar developers to share precise and detailed knowledge of the treebank and grammar and thus to make them consistent throughout the development cycle.[3]

Lexical items whose information is included in the database are grouped together according to their grammatical behavior, and we will refer to each of the groups as a **lexical type** in the rest of the paper. Examples of lexical types will be described in §2.

The next section describes the framework of treebanking and motivates the lexical type database. The third section discusses what information the lexical type database should contain and shows how the database is created. The fourth section discusses the usefulness of the lexical type database for many purposes other than treebanking. An overview of related works follows in the fifth section. Finally, we conclude the paper with a discussion of our plans for future work.

2. Background to the Database

Our treebank is semi-automatically generated by a computational grammar. Each sentence is parsed and the intended reading chosen from the possible interpretations. In doing so, we find the grammar's flaws such as insufficient coverage and spurious ambiguities. The feedback allows us to refine the grammar. Currently this process is carried out by several people, distributed over four continents.

As is often the case with detailed linguistic treebanking, our grammar and treebank consist of very fine-grained linguistic information. For example, our grammar distinguishes several usages of the Japanese dative marker *ni*. The Japanese sentence (1) can represent the two meanings described in (1a) and (1b). Lexical type names for each usage of *ni* are written in `typewriter` font.[4]

(1)	hanasiai-wa	sinya-**ni**	itaru
	discussion-TOP	midnight-DAT	reach

a. "The discussion comes (to a conclusion) at midnight."
ni as `adv-p-lex-1`

b. "The discussion continues until midnight."
ni as `ga-wo-ni-p-lex`

The dative phrase, *sinya-ni* (midnight-DAT), can act as either an adjunct (1a)[5] or an object of *itaru* "reach" (1b). Clearly, these two usages of *ni* show differences in both syntax and semantics. Below is an example showing other usages of *ni*.

(2) Ken-wa yuka-o kirei-**ni** migaku
-TOP floor-ACC clean-DAT polish

a. "Ken polishes a floor clean."
(The floor is clean.)
ni as `naadj2adv-end-lex`

b. "Ken cleanly polishes a floor."
(His way of polishing the floor is clean.)
ni as `adv-p-lex-6`

The dative phrase, *kirei-ni* (clean-DAT), is used as an adjunct in both (2a) and (2b), but their usages and meanings are different. The usage in (2b) is an ordinary adverb that describes the manner of Ken's polishing the floor as clean, while in (2a) the dative phrase describes the resulting situation of the floor after polishing as clean. In addition, the *ni*s in (1) and (2) are different in that the former takes nouns as its complement while the latter takes adjectives. Thus, the four usages in (1a), (1b), (2a) and (2b) must be distinguished. In our terms, these *ni*s are said to belong to different lexical types.

However, as we augment the grammar with finer distinctions, the grammar becomes difficult to maintain, and so is the treebank. This makes unclear **(i)** what lexical types are assumed in a grammar and **(ii)** how differently they are used from each other. Our lexical type database helps to make clear **(i)** and **(ii)**.

3. Architecture of the Database

3.1. Content of the Database

To make it clear how each of the lexical types are used and distinguished, we include five kinds of information:

Type name & Linguistic discussion: To understand lexical types precisely, linguistic observations and analyses are a basic source of information. Firstly, the requirements for naming lexical-types are that they be short (so that they can be displayed in large trees) and easily distinguishable. Type names are not necessarily understandable for anyone but the developers, so it is useful to link them to more conventional names. For example `ga-wo-ni-p-lex` is a *Case Particle*. Next, the definition field contains a widely accepted definition statement of the lexical type. For example, `ga-wo-ni-p-lex` (1b) can be defined as "a particle that indicates that a noun it attaches to functions as an argument of a predicate." Thirdly, the criteria field provides users with means of investigating whether a given word belongs to the class. That is, it provides positive and negative usage examples. For example, `adv-p-lex-1` (1a) subcategorizes for nouns, while `adv-p-lex-6` (2b) subcategorizes for adjectives. Sentences like (1a) and (2b) that fit such criteria should also be treebanked so that they can be used to test that the grammar covers what it claims. This is especially important for regression testing after new development. Finally, the reference field points to representative papers or books dealing with the lexical type.

Exemplification: As we have constructed a treebank (Bond et al., 2004a; Bond et al., 2007), we can automatically extract relevant examples exhaustively. We give the database two kinds of examples: words, that are instances of the lexical types, and sentences, treebanked examples that contain the words. This link to the examples helps treebankers to check for consistency, and grammar developers to check that the lexical types are grounded in the corpus data.

Implementation: Grammar developers need to know the actual implementation of lexical types. TODOs or comments about the implementation are also helpful to ascertain the current status. Although this section is necessarily framework-dependent information, all project groups that are constructing detailed linguistic treebanks need to document this kind of information.

Links to "confusing" lexical types: For users to distinguish phonologically identical but syntactically or semantically distinct words, it is important to link confusing lexical types to one another within the database. For example, the four lexical types in (1) and (2) are connected with each other in terms of *ni*. That way, users can compare those words in detail and make a reliable decision when trying to disambiguate usage examples.[6]

Links to other dictionaries: This information helps us to compare our grammar's treatment with that of other dictionaries. This comparison would then facilitate understanding of lexical types and extension of the lexicon. We currently link lexical types of our grammar to those of ChaSen (Matsumoto et al., 2000), Juman (Kurohashi and

Nagao, 2003), ALT-J/E (Ikehara et al., 1991) and EDICT (Breen, 2004). For example, `ga-wo-ni-p-lex` is linked to ChaSen's particle-case_particle-general, Juman's case_particle, and ALT-J/E's adjunct-case_particle-noun/particle_suffix (Miyazaki et al., 1995). (EDICT concerns only content words. Thus, it does not contain the counterpart of JACY's `ga-wo-ni-p-lex`.) In general, JACY makes finer distinctions than ChaSen, Juman or EDICT, and has roughly the same level of granularity as ALT-J/E. In addition to these four Japanese lexicons, we link the lexical types to the GOLD linguistic ontolgy[7]. GOLD is not a lexicon. Rather it is an upper ontology for descriptive linguistics, providing a set of (possibly universal) linguistic notions. Hence, linking to GOLD helps to understand a implemented linguistics grammar and a treebank from a universal grammatical point of view. For example, `ga-wo-ni-p-lex` is linked to GOLD's Postposition.

Figure1 shows the contents for `ga-wo-ni-p-lex` that are rather simplified and translated into English for this paper.

`ga-wo-ni-p-lex`

Linguistic Discussion

LINGUITIC NAME: Case Particle

DEFINITION: It attaches to a noun and indicates what grammatical relation the noun takes on in relation to a predicate.

POSITIVE		NEGATIVE	
sinya-**ni**	itaru	kirei-**ni**	migaku
midnight-**DAT**	reach	clean-**DAT**	polish

LITERATURE: Shigeru Miyagawa. *Structure and Case Marking in Japanese.* Academic Press. 1989.

Exemplification

"ni": kazari-**ni** naru (accessory-**DAT** become)
'(That) can be used as an accessory.'

Implementation

```
ga-wo-ni-p-lex := case-p-lex &
[SYNSEM.LOCAL.CAT.VAL.COMPS <[LOCAL.CAT.HEAD noun_head]>]
```

TODO

The dative subject of stative predicates is not recognized.

Links

CHASEN: particle-case_particle-general
JUMAN: case_particle
ALT-J/E: adjunct-case_particle-noun/particle_suffix
GOLD: Postposition

Figure 1. (Simplified) Database Contents for `ga-wo-ni-p-lex`

3.2. Method of Database Construction

The next question is how to construct such a database. Fully manual construction of the database is unrealistic, since there are about 300 lexical types and more than 30,000 words in our grammar. In addition, we assume that we will refer to the database each time we annotate parser outputs to build the treebank and that we develop the grammar based on the treebanking result. Thus the database construction process must be quick enough. Thus, our method of construction for the lexical type database is semi-automatic.

3.2.1. *Component Databases*

To understand the construction process, description of the four component databases that feed the lexical type database is in order. **The grammar database** contains the actual implementation of the grammar. **The lexicon database** gives us mappings between words in the grammar, their orthography, and their lexical types. **The treebank database** stores all treebank information, including syntactic derivations, words, and the lexical type for each word. We also use **the other lexicon databases** that are compiled from other sources, such as ChaSen, Juman, EDICT and GOLD.

3.2.2. *Automatic Construction*

Next we move on to describe the automatic construction. Firstly, we collect all lexical types assumed in the grammar from the grammar database.

Secondly, we extract words that belong to a given lexical type and sentences that contains the words from the treebank database.

Thirdly, implementation information except for TODOs is extracted from the grammar database.

Fourthly, in order to establish "confusing" lexical type links, we collect from the lexicon database homonyms of a word that users enter as a query. To be more precise, the lexicon database presents all the words with the same orthography as the query but belonging to different lexical types. These lexical types are then linked to each other as "confusing" in terms of the query word.

Fifthly, we construct links between our lexical types and POS's of other lexicons such as ChaSen from the other lexicon databases. To do this, we prepare an interface (a mapping table) between our lexical type system and the other lexicon's POS system. As this is a finite mapping it could be made manually, but we semi-automate its construction (except for the mapping of GOLD). The similarity between types in the two lexicons is calculated as the Dice coefficient, where $|W(L_A)|$ is the number of words W in lexical type L:

$$\mathrm{sim}(L_A, L_B) = \frac{2 \times |(W(L_A \cap L_B)|}{|W(L_A)| + |W(L_B)|} \tag{1}$$

The Dice coefficient was chosen because of its generality and ease of calculation. Any pair where $\mathrm{sim}(L_A, L_B)$ is above a threshold should potentially be mapped.

3.2.3. *Manual Construction*
Linguistic discussion and implementation TODOs have to be entered manually. Linguistic discussion is especially difficult to collect exhaustively since the task requires an extensive background in linguistics. We have several linguists in our group, and our achievements in this task owe much to them. We prepared a web-based interface for the linguists in different locations to enter linguistic discussion.

The on-line documentation is designed to complement the full grammar documentation (Siegel, 2006). The grammar documentation gives a top down view of the grammar, giving the overall motivation for the analyses. The lexical-type documentation gives bottom up documentation. It can easily be updated along with the grammar.

Writing implementation TODOs also requires expertise in grammar development and linguistic background. But grammar developers usually take notes on what remains to be done for each lexical type anyway, so this is a relatively simple task.

After the database is first constructed, how is it put to use and updated in the treebanking cycles? Figure 2 illustrates this. Each time the

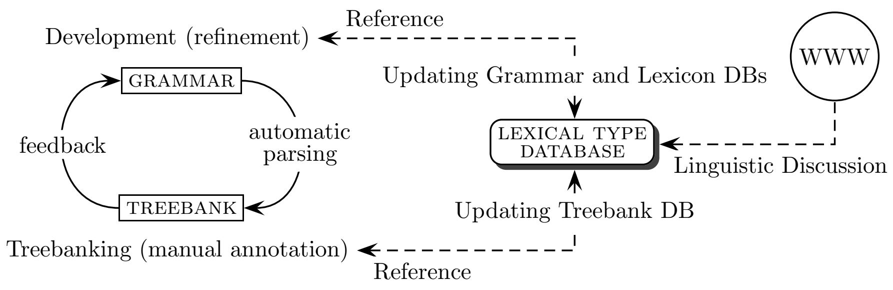

Figure 2. Database Construction Intergrated with Treebanking Cycles

grammar is revised based on treebank annotation feedback, grammar developers consult the database to see the current status of the grammar. After finishing the revision, the grammar and lexicon DBs are updated, as are the corresponding fields of the lexical type database. Each time the treebank is annotated, annotators can consult the database to make sure the chosen parse is correct. Following annotation, the treebank DB is updated, and so is the lexical type database. In parallel to this, collaborators (linguists) continue to enter relevant linguistic discussions via the WWW.

3.3. Related Work

Tsuchiya et al. (2005) have been constructing a database that summarizes multiword functional expressions in Japanese. That describes each expression's linguistic behavior, usage and examples in depth. Notable differences between their database and ours are that their database is mostly constructed manually while ours is constructed semi-automatically.

Hypertextual Grammar development (Dini and Mazzini, 1997) attempted a similar task, but focused on documenting the grammar, not on linking it to a dynamic treebank.

4. Lexical Type Database as a General Linguistic Resource

In this section, we speculate some of the ways the database can benefit people other than treebank annotators and grammar developers.

One way is by serving as a link to other Japanese lexical resources. Currently, in Japanese NLP, various lexical resources have been developed, but their intercorrespondences are not always clear. These lexical resources often play complementary roles, so synthesizing them seamlessly will make a Japanese lexicon with the widest and deepest knowledge ever. Among our plans is to realize this by means of the lexical type database. That is, the lexical type database can act as a "hub" that links those lexical resources together.

Related to this, the link to the linguistic notions assumed in GOLD could also act as a "interlingual hub" that connects already-connected lexical resources of several languages. Although GOLD's linguistic notions are a bit coarse, they would tell us inter-lingual correspondences between lexical types of different languages.

We expect to create successively better approximations of the Japanese language, as long as our grammar describes Japanese syntax and semantics precisely. Consequently, the database would be of use to anyone who needs an accurate description of Japanese. Japanese language teachers can use its detailed descriptions of word usages, the links to other words, and the real examples from the treebank to show for students subtle differences among words that look the same but are grammatically different. Lexicographers can take advantage of its comprehensiveness and the real examples to compile a dictionary that contains full linguistic explanations.

5. Future Work

We would like to link the grammar to other Japanese lexicon projects, in particular the Japanese FrameNet Project (Ohara et al., 2004) and LCS (Takeuchi et al., 2006).

Although this paper deals with a lexical type database of Japanese, the importance of such a database holds for any large scale deep grammar. We use the tools from the DELPH-IN collaboration(`http://www.delph-in.net/`) and plan to make our tool available for groups working with other languages. In particular, we plan to construct a lexical type database for the Redwoods treebank.

Acknowledgements

We would like to thank the other members of NTT Natural Language Group, Dan Flickinger, Stephen Oepen, and Jason Katz-Brown for their stimulating discussion.

Notes

[1] Currently, the Hinoki treebank contains about 121,000 sentences (about 10 words per sentence).

[2] `http://wiki.delph-in.net/moin/JacyLexTypes`

[3] We think we also need another snapshot, that of the grammar rules and principles being used. In this paper, however, we do not deal with it.

[4] These are actual names of the lexical types implemented in our grammar and might not be understandable to people in general.

[5] The object, *a conclusion*, is expressed by a phonologically null pronoun.

[6] Note that this information is not explicitly stored in the database. Rather, it is dynamically compiled from the database together with a lexicon database, when triggered by a user query. User queries are words like *ni*.

[7] `http://www.linguistics-ontology.org/gold.html`

References

Bond, F., S. Fujita, C. Hashimoto, S. Nariyama, E. Nichols, A. Ohtani, T. Tanaka, and S. Amano: 2004a, 'The Hinoki Treebank — Toward Text Understanding'.

In: *Proceedings of the 5th International Workshop on Linguistically Interpreted Corpora (LINC-04)*. Geneva, pp. 7–10.

Bond, F., S. Fujita, and T. Tanaka: 2007, 'The Hinoki Syntactic and Semantic Treebank of Japanese'. *Language Resources and Evaluation*. (this volume).

Bond, F., E. Nichols, S. Fujita, and T. Tanaka: 2004b, 'Acquiring an Ontology for a Fundamental Vocabulary'. In: *20th International Conference on Computational Linguistics (COLING-2004)*. Geneva, pp. 1319–1325.

Breen, J. W.: 2004, 'JMDict: a Japanese-Mulitlingual Dictionary'. In: *Coling 2004 Workshop on Multilingual Linguistic Resources*. Geneva, pp. 71–78.

Dini, L. and G. Mazzini: 1997, 'Hypertextual Grammar Development'. In: *Computational Environments for Grammar Development and Linguistic Engineering*. Madrid, pp. 24–29.

Ikehara, S., S. Shirai, A. Yokoo, and H. Nakaiwa: 1991, 'Toward an MT System without Pre-Editing – Effects of New Methods in **ALT-J/E**–'. In: *Third Machine Translation Summit: MT Summit III*. Washington DC, pp. 101–106. (`http://xxx.lanl.gov/abs/cmp-lg/9510008`).

Kurohashi, S. and M. Nagao: 2003, 'Building a Japanese Parsed Corpus — While Improving the Parsing System'. Chapt. 14, pp. 249–260.

Matsumoto, Y., A. Kitauchi, T. Yamashita, Y. Hirano, H. Matsuda, K. Takaoka, and M. Asahara: 2000, 'Morphological Analysis System ChaSen version 2.2.1 Manual'. Nara Institute of Science and Technology.

Miyazaki, M., S. Shirai, and S. Ikehara: 1995, 'gengo katēsetsu-ni motozuku nihongo hinshi-no taikēka-to sono kōyō [A Japanese Syntactic Category System Based on the Constructive Process Theory and its Use]'. *Journal of Natural Language Processing* **2**(3), 3–25. (in Japanese).

Oepen, S., D. Flickinger, K. Toutanova, and C. D. Manning: 2002, 'LinGO Redwoods: A Rich and Dynamic Treebank for HPSG'. In: *Proceedings of The First Workshop on Treebanks and Linguistic Theories*. Sozopol, Bulgaria, pp. 139–149.

Ohara, K. H., S. Fujii, T. Ohori, R. Suzuki, H. Saito, and S. Ishizaki: 2004, 'The Japanese FrameNet Project: An introduction'. In: *Proceedings of the LREC-2004 Satellite Workshop Building Lexical Resources from Semantically Annotated Corpora*. pp. 9–11.

Siegel, M.: 2006, 'JACY, A Grammar for Annotating Syntax, Semantics and Pragmatics of Written and Spoken Japanese for NLP Application Purposes'. ms.

Siegel, M. and E. M. Bender: 2002, 'Efficient Deep Processing of Japanese'. In: *Proceedings of the 3rd Workshop on Asian Language Resources and International Standardization*. Taipei, Taiwan.

Takeuchi, K., K. Inui, and A. Fujita: 2006, 'Description of Syntactic and Semantic Characteristics of Japanese Verbs based on Lexical Conceptual Structure'. In: *Lexicon Forum*, Vol. 2. Hituzi Syobou, pp. 85–120. (in Japanese).

Toutanova, K., C. D. Manning, D. Flickinger, and S. Oepen: 2005, 'Stochastic HPSG Parse Disambiguation using the Redwoods Corpus'. *Research on Language and Computation* **3**(1), 83–105.

Tsuchiya, M., T. Utsuro, S. Matsuyoshi, S. Sato, and S. Nakagawa: 2005, 'A Corpus for Classifying Usages of Japanese Compound Functional Expressions'. In: *Proceedings of Pacific Association for Computational Linguistics 2005*. Tokyo, Japan.