A COMPUTATIONAL TREATMENT OF V-V COMPOUNDS IN JAPANESE

A DISSERTATION
SUBMITTED TO THE DEPARTMENT OF ENGLISH LINGUISTICS
OF KOBE SHOIN GRADUATE SCHOOL OF LETTERS
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

Adviser: GUNJI Takao

By
HASHIMOTO Chikara
November, 2004

Dissertation Committee

Chairperson:	GUNJI Takao
Member:	Joseph Emonds
Member:	Taisuke Nishigauchi

Acknowledgments

I gratefully acknowledge many people who supported me with deepest appreciation. I first express my appreciation to my dissertation committee members, Takao Gunji, Joseph Emonds, and Taisuke Nishigauchi. Takao Gunji, my adviser, gave me many theoretical ideas and inspired me in diverse ways. Actually, his writings interested me in theoretical linguistics and made me turn to grammar development in which linguistics and NLP collaborate. I enjoyed arguing with Joseph Emonds about mathematical aspects of linguistic theory. I was really impressed with his wise comments. My gratitude also goes Taisuke Nishigauchi, whose guidance has been invaluable since I decided to major in linguistics.

I am also deeply grateful to Francis Bond for lots of advice ranging from engineering and technical aspects to spiritual side. I owe him almost all of the technical expertise that are central to my dissertation. He also gave me a chance to take part in the HINOKI project, through which I could meet many interesting people and have a good experience. I learned so many things about NLP from Takaaki Tanaka and Sanae Fujita through the HINOKI project. Their help constitutes an integral part of my dissertation.

I am indebted to Yuji Matsumoto, one of the groundbreakers of HPSG-based NLP in Japan, for insightful comments. As well, I received many helpful and stimulating comments from Akira Ohtani. Thanks to them, I could think about a computational Japanese grammar in depth.

I have benefited from the discussion with Dan Flickinger a lot. He patiently answered my forest of questions about the ERG grammar. I am also deeply appreciative of his many kindnesses during my stay in Stanford. Melanie Siegel and Emily Bender, the principal JACY developers, also gave me valuable comments, and solved many problems that I encountered in developing my implementation. Without their development of JACY, I could not have finished the dissertation. Timothy Baldwin pointed out several drawbacks of my implementation, which were really helpful. Also, he kindly spared a lot of time for the discussion with me. I am grateful to Stephan Oepen, the developer of `[incr tsdb()]`, too. I could conduct the evaluation smoothly thanks to his amazing creation. Kiyoko Uchiyama, Colin Bannard, and Koji Tsukamoto were also very kind enough to contribute to the significant engineering parts of my treatment of verbal compound.

I also have to acknowledge the great debts of gratitude that I owe to Ivan Sag, Peter Sells, Anthony Davis, Jong-Bok Kim, and John Beavers. Indeed, their theoretical advice and kindnesses were indispensable.

In a similar way, I have received lots of insightful advice from Kenjiro Matsuda, Michinao Matsui and Philipe Spaelti. I owe most of my knowledge of linguistics to them.

I cannot list the names of all people who I am indebted to. But I particularly would like to thank Yoshinobu Aoki, David Sheridan, and the graduate students of Kobe Shoin for their very warm encouragements and supports.

Finally, I dedicate my dissertation with my sincere gratitude to my parents and brother.

Contents

Acknowledgments **iv**

1 Introduction **1**

1.1 Overview 1

1.2 Background: the need for a linguistic treatment 2

1.2.1 Theoretical linguistics and NLP 2

1.2.2 Multiword Expressions: an obstacle to NLP 5

1.2.3 V_1-V_2 Compounds as Multiword Expressions 7

1.3 Summary 9

1.4 The purposes of the dissertation 11

2 Linguistic analyses of V_1-V_2 compounds **12**

2.1 Linguistic theory: from an engineering perspective 12

2.2 Kageyama (1993) 15

2.2.1 Syntactic V_1-V_2s vs Lexical V_1-V_2s 15

2.2.2 Syntactic V_1-V_2 compounds 20

2.2.3 Lexical V_1-V_2 compounds 23

2.2.4 Engineering problems 26

2.3 Matsumoto (1996) 26

2.3.1 Lexical V_1-V_2 compounds 27

2.3.2 Engineering problems 31

2.4 Summary 32

3 Engineering oriented analysis of V_1-V_2 compounds **35**

3.1 Grammar development: from an engineering perspective 35

3.2 JACY: a linguistically precise grammar of Japanese 37

3.3 Some basics of HPSG 39

3.3.1 Types, type hierarchy, and feature structures 39

3.3.2 Lexicon 40
3.3.3 Syntax 44
3.3.4 Semantics 49
3.3.5 Word order 53
3.3.6 Lexical rule 54
3.4 LKB: a grammar and lexicon development environment 55
3.5 Syntactic V_1-V_2 compounds 55
3.5.1 Hashimoto (2003b, 2003c) 55
3.5.2 Adapting Hashimoto (2003b, 2003c) to JACY: word order problem 69
3.6 Lexical V_1-V_2 compounds 79
3.6.1 Introducing ARG-ST to JACY 81
3.6.2 Right headed V_1-V_2s 83
3.6.3 Argument mixing V_1-V_2s 92
3.6.4 V_1-V_2s with semantically deverbalized V_1 97
3.6.5 V_1-V_2s with semantically deverbalized V_2 99
3.6.6 Non-compositional V_1-V_2s 101
3.7 Summary 103

4 Evaluation **106**
4.1 What does a good computational grammar look like? 106
4.2 The details of evaluation 107
4.2.1 [incr tsdb()]: competence and performance laboratory 107
4.2.2 Lexeed: a fundamental vocabulary database 107
4.2.3 Evaluation procedure 108
4.3 Result 110
4.3.1 Competence 110
4.3.2 Performance 111
4.4 Discussion 112
4.4.1 Competence 112
4.4.2 Performance 120
4.5 Summary 121

5 Concluding remarks **122**
5.1 Future work 122
5.1.1 Problematic cases of V_1-V_2 compounds 122

5.1.2 Automatic detection of non-compositional V_1-V_2 compounds 126
5.1.3 Machine Translation of V_1-V_2 compounds 128
5.2 Conclusion: the prospect of the two studies of language 131
5.2.1 Proper division of labor 131
5.2.2 Airplane or bird? 132

A Grammar source code **134**
A.1 Basics 135
A.2 Syntactic V_1-V_2 compound 139
A.2.1 VP embedding structure 139
A.2.2 V embedding structures 144
A.3 Lexical V_1-V_2 compound 147
A.3.1 General rule types 148
A.3.2 Right headed V_1-V_2 150
A.3.3 Argument mixing V_1-V_2 154
A.3.4 V_1-V_2 with semantically deverbalized V_1 155
A.3.5 V_1-V_2 with semantically deverbalized V_2 156
A.3.6 Verb hierarchy 157

B Additional illustration of the analysis **164**
B.1 Preliminaries 164
B.2 Syntactic V_1-V_2 compound 169
B.2.1 A type 169
B.2.2 B type 172
B.2.3 C type 175
B.3 Lexical V_1-V_2 compound 176
B.3.1 Right headed V_1-V_2 176
B.3.2 Argument mixing V_1-V_2 179
B.3.3 V_1-V_2 with semantically deverbalized V_1 184
B.3.4 V_1-V_2 with semantically deverbalized V_2 186

Bibliography **191**

List of Tables

1 Table representation of the type hierarchies . . . 41
2 The classification of syntactic V_1-V_2s . . . 57
3 The type hierarchy of ARG-ST . . . 83
4 The 29 rules for Right headed V_1-V_2s . . . 85
5 The two versions of the JACY grammar . . . 108
6 The two evaluation corpora . . . 109
7 The two grammars' competence with respect to ALL corpus . . . 110
8 The two grammars' competence with respect to V-V corpus . . . 111
9 The two grammars' performance with respect to ALL corpus . . . 111
10 The two grammars' performance with respect to V-V corpus . . . 112
11 The frequencies of V_1-V_2 compounds . . . 115
12 The frequencies of Right headed V_1-V_2s . . . 115
13 The three problems that prevent JACY-vv from getting more coverage . . . 118
14 The semantic composition of *Ken-ga hon-o yomu* . . . 165
15 The semantic composition of *Ken-ga Naomi-ga kasikoi-to omou* . . . 168
16 The semantic composition of *Ken-ga hon-o yomi-kakeru* . . . 171
17 The semantic composition of *Ken-ga hon-o yomi-sokoneru* . . . 174
18 The semantic composition of *Ken-ga hon-o yomi-naosu* . . . 177
19 The semantic composition of *huku-ga ki-kuzureru* . . . 179
20 The semantic composition of *Ken-ga sake-o nomi-aruku*: am-V_1 . . . 182
21 The semantic composition of *Ken-ga sake-o nomi-aruku*: am-V_2 . . . 183
22 The semantic composition of *Ken-ga zyugyoo-o kaki-midasu* . . . 185
23 The semantic composition of *oto-ga hibiki-wataru* . . . 187
24 The semantic composition of *Ken-ga hon-o yomi-konasu* . . . 189

List of Figures

1 Syntactically derived V_1-V_2 compound . . . 15
2 Lexically derived V_1-V_2 compound . . . 15
3 Depth-first grammar development . . . 35
4 Breadth-first grammar development . . . 36
5 Input through Emacs . . . 38
6 The example of JACY output . . . 38
7 Example of Indexed MRS . . . 50
8 LKB . . . 56
9 A part of JACY's verb hierarchy . . . 56
10 The JACY output of *Ken-ga hon-o yomi-kakeru* . . . 61
11 The JACY output of *Ken-ga hon-o yomi-sokoneru* . . . 61
12 The JACY output of *Ken-ga hon-o yomi-naosu* . . . 62
13 The JACY output of *hon-o Ken-ga yomi-sokoneru* . . . 73
14 The example of Right headed V_1-V_2 . . . 84
15 The JACY output of *huku-ga ki-kuzureru* . . . 85
16 A V_1-V_2 compound consisting of two motion verbs . . . 94
17 The two possibilities of Argument mixing V_1-V_2 . . . 95
18 The JACY output of *Ken-ga sake-o nomi-aruku*: syntax . . . 95
19 The JACY output of *Ken-ga sake-o nomi-aruku*: semantics . . . 96
20 The JACY output of *Ken-ga zyugyoo-o kaki-midasu* . . . 100
21 The JACY output of *oto-ga hibiki-wataru* . . . 102
22 The JACY output of the non-compositional V_1-V_2 *tori-simaru* . . . 103
23 [incr tsdb()] . . . 107
24 Entry for the word *hinoki* "Japanese Cedar" (with English glosses) . . . 108
25 The proportion of *monotrans-monotrans* to all kinds of V_1-V_2 compound . 116
26 The JACY output of *suuryo-o hakari-kazoeru* . . . 117
27 *Ken-ga hon-o yomu* . . . 129
28 *Ken reads a book* . . . 129

29 The semantic transfer with JACY and ERG 130
30 *Ken-ga hon-o yomu* 164
31 *Ken-ga Naomi-ga kasikoi-to omou* 167
32 *Ken-ga hon-o yomi-kakeru* 170
33 *Ken-ga hon-o yomi-sokoneru* 173
34 *Ken-ga hon-o yomi-naosu* 175
35 *huku-ga ki-kuzureru* 178
36 *Ken-ga sake-o nomi-aruku*: syntax 180
37 *Ken-ga sake-o nomi-aruku*: semantics 181
38 *Ken-ga zyugyoo-o kaki-midasu* 186
39 The JACY output of *oto-ga hibiki-wataru* 187
40 *Ken-ga hon-o yomi-konasu* 190

Chapter 1 Introduction

1.1 Overview

The purposes of this study are two-fold. One is to implement the linguistic analyses of Japanese verbal compounds in a computational grammar of Japanese, and the other is to discuss why and how Natural Language Processing (NLP) should benefit from theoretical linguistics.

Japanese verbal compounds consist of an infinitive verb (V_1) followed by another verb (V_2), and I will call them V_1-V_2 compounds or just V_1-V_2s hereafter.[1] Japanese, as an agglutinative language, has a huge number of V_1-V_2 compounds, examples of which are illustrated below.

(1) a. *moosi-deru* (say-come.out) 'propose'

b. *mi-watasu* (look-give.over) 'overlook'

c. *ii-haru* (say-stretch) 'insist'

They are frequently used in both colloquial speech and written documents, but their usages and meanings are not easy to articulate despite their surface simplicity. Thus, they have been extensively studied in Japanese linguistics and Japanese-language education. However, there have been few works on V_1-V_2 compounds in NLP that make full use of the insightful analyses developed in the linguistic studies. Actually, linguistic analyses other than those of V_1-V_2 compounds have also been paid almost no attention by NLP works. I will clarify in the course of the discussion presented in this dissertation that the reason for this concerns the differences between the two studies of language, theoretical linguistics and NLP, in terms of their purposes and approaches, and show a particular instance where the two studies reconcile with each other. In other words, I will show the implementation of V_1-V_2 compounds that harmonizes theoretical insights with NLP practice.

The remainder of this chapter discusses the background for the discrepancy between the two studies of language and the motivation that I take up V_1-V_2 compounds here. In the second chapter, I will look over theoretical analyses of V_1-V_2s and criticize them from

[1] In Japanese, there is another kind of verbal compound, in which *te*, an affix, intervenes between V_1 and V_2. In this dissertation, however, I do not discuss that.

the engineering point of view. The third chapter is devoted to my analysis of V_1-V_2s that makes use of theoretical analyses and yet is computationally efficient and practical. The fourth chapter contains the evaluation of my implementation that is conducted through corpus annotation. Finally, the fifth chapter concludes the dissertation.

1.2 Background: the need for a linguistic treatment

1.2.1 Theoretical linguistics and NLP

Theoretical linguistics, as science of language, tries to explain a variety of grammatical phenomena, to understand how a child acquires a language, and to find language universals. On the other hand, NLP, as engineering of language, aims to the ways to make a computer process or understand human languages as naturally as we do when we are reading a book or talking with someone. Accordingly, they differ in the following respects.

- Theoretical linguistics works on idealized data, which is natural as a scientific study. NLP, on the other hand, works on unidealized data, since the input to an NLP system is necessarily spontaneous utterances.
- Theoretical studies usually deal with one particular phenomenon, and analyze it in depth. In contrast, an NLP system has to process all sentences in the input, in which all kinds of phenomena, including even pragmatic ones, show up.
- There would be various kinds of ambiguities in a sentence. For example, considering the facts that a dative phrase in Japanese can be either an argument or an adjunct and that Japanese allows pro-drop, then the sentence in (2a) can be given two readings corresponding to (2b) and (2c).

  (2) a. Ken-ga X-ni au
  Ken-NOM X-DAT (argument or adjunct) meet

  b. 'Ken meet X.'

  c. 'Ken meet someone at X (time).'

  au (meet) in (2a) can take a dative argument that stands for a person who *Ken* met. In that case, (2a) means (2b). In the case of (2c), on the other hand, the dative phrase, *X-ni*, represents a time or the date when Ken met someone. Theoretical linguistics can ignore such an ambiguity if it is irrelevant to the discussion. However, NLP systems are expected to resolve any kind of ambiguity, or to pick the most plausible analysis among ambiguities.

- Linguistic theories usually do not have to worry about whether their analyses or computations are executed efficiently, whereas processing efficiency is one of the biggest issues in NLP.

Because of these differences, NLP has adopted shallow processing techniques, rather than theoretically-motivated deep linguistic analyses. Shallow techniques have brought NLP the following advantages.

- It became easy to build many rules that can describe unidealized data exhaustively.
- It became possible to select the best parse among ambiguities by means of statistical methods, which have been extensively developed for shallow processing techniques.
- Utterances can be parsed by efficient algorithms, which have also been explored mainly for shallow techniques.

However, shallow techniques lead to the defects below.

- They don't provide us with the meaning of utterances.
- In order to select the best parse, they depend heavily on the statistical information concerning syntactic structures that we can obtain only by analyzing huge amounts of text data.

Deep linguistic treatments for NLP, if possible, can get around these problems.

- Semantic representations are constructed in parallel with phrase structures.
- They don't need to rely as heavily on statistical information, as long as a grammar describes a language precisely.

Some NLP applications, such as Information Retrieval and Automatic Text Summarization, have been thought to be able to dispense with deep linguistic semantic information. Nevertheless, other applications like Machine Translation and Automatic E-mail Response certainly require deep semantic information. Naturally, once such information is available, we can expect applications like Automatic Text Summarization to be more sophisticated and improve dramatically, too. Furthermore, linguistic semantic representations can be seen as independent from languages, so it is reasonable to think that the semantic representation helps develop multilingual NLP applications. Statistical information to resolve ambiguities comes with the cost of annotating huge amount of text data, and is sometimes affected by characteristics of particular text data. We can assume that deep linguistic treatments are exempt from the problem if the grammar reflects linguistic facts properly. In addition, such a linguistic approach to NLP might be able to supply developers who adopt statistical methods with better language models whose importance is recognized by Charniak (2001).

For deep linguistic treatments to be usable, they have to meet the following three conditions.

- They must be executed in an efficient way.
- They must be able to find the best parse among ambiguities.
- They must have broad coverage.

As for the first condition, Callmeier (2000) and Maxwell and Kaplan (1993), among others, propose efficient ways to process linguistic grammars, especially those that are called constraint-based lexical grammars like HPSG and LFG. The second condition can be met by the statistical disambiguation technique developed by Toutanova et al. (2002), which is applicable to computational HPSG grammars. The third condition, achieving broad coverage, is one of the main concerns of the projects that aim at building a large-scale linguistic grammar for NLP. Some of these projects are listed below.

- ParGram (Butt et al., 2002) ··· LFG
- DELPH-IN (Oepen et al., 2002) ··· HPSG
  - The HINOKI Treebank (Bond et al., 2004a, 2004b, 2004c)
- XTAG (The XTAG Research Group, 1995) ··· TAG
- Edinburgh Wide-Coverage CCG Parsing Project (Clark et al., 2002) ··· Combinatory Categorial Grammar

In these projects, various kinds of constraint-based lexicalist frameworks are being used, and almost all of them have achieved a moderate success in building a large-scale linguistic grammar.

I take up, for example, the HINOKI treebank project, which is regarded as the subproject of DELPH-IN in that it follows the DELPH-IN framework. The HINOKI treebank project aims to make computer understand human languages. In order to achieve the ultimate goal, the project is trying to build a large-scale linguistically precise computational grammar of Japanese,[2] a very large corpus that is parsed with the grammar (treebank),[3] a statistical model for the grammar that is obtained from the treebank, and a knowledge base that is also obtained from the treebank. With the grammar they are developing, they are able to cover over 80% of 81,000 sentences, and still they manage to keep the average number of ambiguities per sentence reasonably low. In addition, 94% of parsed sentences are given a correct analysis.

Considering these recent developments in the field of building a computational grammar, it is reasonable to think that linguistic treatments for NLP are becoming available and NLP applications should take better advantage of theoretical analyses and observations.

1.2.2 Multiword Expressions: an obstacle to NLP

There are several general problems in NLP: ambiguity, efficiency, and unknown words, to name a few. Recently, another problem, which is somewhat related to the unknown words problem, has been recognized especially by Sag et al. (2002) and Baldwin and Bond (2002). That is what they call **Multiword Expressions** (**MWEs**).

Sag et al. (2002) and Baldwin and Bond (2002) define MWEs as "idiosyncratic interpretations that cross word boundaries (or spaces)," and classify MWEs into two kinds: **lexicalized phrases** and **institutionalized phrases**. Below is the explanation of MWEs cited from Sag et al. (2002).

> Lexicalized phrases have at least partially idiosyncratic syntax and semantics, or contain 'words' which do not occur in isolation; they can be further broken down into **fixed expressions**, **semi-fixed expressions** and **syntactically-flexible expressions**, in roughly decreasing order of lexical regidity. Institutionalized phrases are syntactically and semantically compositional, but occur with markedly high frequency (in a given context). ··· Fixed expressions are fully lexicalized and undergo neither morphosyntactic variation (cf. **in shorter*) nor internal modification (cf. **in very short*). ··· Semi-fixed expressions adhere to strict constraints on word order and composition, but undergo some degree of lexical variation, e.g. in the form of inflection, variation in the reflexive form, and determiner selection. ··· Whereas semi-fixed expressions retain the same basic word order throughout, syntactically-flexible expressions exhibit a much wider range of syntactic variability. ··· Institutionalized phrases are semantically and syntactically compositional, but statistically idiosyncratic.

[2] Actually, they do not develop a grammar from scratch. Rather they are extending the existing computational grammar of Japanese called JACY, which I will describe in §3.2

[3] They parse the Lexeed corpus, the description of which is given in §4.2.2

English and Japanese examples of MWEs, cited from Sag et al. (2002) and Baldwin and Bond (2002), are illustrated below.

Lexicalized phrases:

Fixed expressions:

- *by and large*
- *in short*
- *mae-muki* (forward-turn.on) ‘positive’

Semi-fixed expression:

- *kick the bucket*
- *trip the light fantastic*
- *o-me-ni kakaru* (HON-eye-DAT hang) ‘meet’

Syntactically-flexible expressions:

- *call up*
- *take a walk*
- *ude-o ageru* (arm-ACC raise) ‘improve one’s skill’

Institutionalized phrases:

- *traffic light*
- *telephone booth*
- *kikai-hon’yaku* (machine-translation) ‘machine translation’

Though fixed expressions allow no variation, other MWEs show varying degrees of flexibility and compositionality. Semi-fixed expressions, for instance, allow inflection, so *kicked the bucket* and *o-me-ni kaka-tta* (HON-eye-DAT hang-PAST), are grammatical. Syntactically-flexible expressions allow an element to appear between constituents that consist of the MWEs, e.g. *call him up*, *ude-o takaku ageru* (arm-ACC up.high raise). Institutionalized phrases are compositional both syntactically and semantically, but they are not fully productive; while *traffic light* and *kikai-hon’yaku* sound natural, neither *traffic director* nor *konpyûtâ hon’yaku* (computer translation) are natural if the latter two are intended to mean the same things as the corresponding former phrases. The reason for this is purely statistical. Put another way, *traffic light* and *kikai-hon’yaku* are so familiar that they sound more natural than *traffic director* and *konpyûtâ hon’yaku.*

Sag et al. (2002) and Baldwin and Bond (2002) claim that because of their complex natures, MWEs resist the following simple solutions.

- Regard all MWEs as totally compositional, and derive all of them by means of some sort of rule.
- Regard all MWEs as single words, and register all of them in the lexicon.

According to Sag et al. (2002), the first approach would suffer from an **overgeneration problem**, that is, we would generate or accept word sequences that are not attested, such as *traffic director* and *konpyûtâ hon'yaku*. The approach would also suffer from an **idiomaticity problem**. This problem involves how we know the idiomatic meanings of MWEs that cannot be predicted from their components. The second approach does not capture the characteristics of MWEs, either. Sag et al. (2002) regards the approach as insufficient since it would face a **flexibility problem**. For instance, if *call up* is registered in the lexicon as one word, then how can *call her up* be treated? Besides, this approach would also suffer from a **lexical proliferation problem**. It would have to list *take a walk*, *take a hike*, *take a trip*, and *take a flight* separately in the lexicon, resulting in loss of generality and lack of prediction.

So far, we have seen that MWEs pose serious problems for NLP, since their characteristics are hard to explain for simple treatments like those mentioned above; MWEs need sophisticated linguistic treatments. However, MWEs would show different characteristics from language to language, and the theory of MWEs is underdeveloped. Dealing with MWEs properly is a key aspect of deep linguistic treatments for NLP.

1.2.3 V_1-V_2 Compounds as Multiword Expressions

In Japanese, V_1-V_2 compounds abound in both spontaneous speech and written documents, and their surface compositions are quite simple: an infinitive verb followed by another verb. However, their usages and meanings are so complex that they have been one of the central issues of Japanese linguistics.

Some V_1-V_2s are productive and transparent in their meanings, while others show highly lexicalized characteristics. Below are examples of V_1-V_2s.

(3) Productive and compositional V_1-V_2s

a. *aruki-kakeru* (walk-be.about.to) 'be about to walk'

b. *huri-hazimaru* (fall-begin) 'start falling'

c. *kaki-tuzukeru* (write-continue) 'continue writing'

d. *ai-sobireru* (meet-fail) 'fail to meet'

e. *yomi-ayamaru* (read-mistake) ‘make a mistake in reading’

f. *tabe-akiru* (eat-get.bored.with) ‘get bored with eating’

(4) Less productive and less compositional V_1-V_2s

a. *naki-sakebu* (cry-scream) ‘cry and scream’

b. *odori-tukareru* (dance-get.tired) ‘get tired from dancing’

c. *tobi-okiru* (jump-get.up) ‘get up swiftly’

d. *tataki-waru* (hit-break.in.half) ‘break in half by hitting’

e. *hare-wataru* (clear.up-cross.over) ‘break into sunshine’

f. *yuzuri-ukeru* (yield-receive) ‘inherit’

(5) Idiosyncratic V_1-V_2s

a. *kuri-kaesu* (turn.over-give.back) ‘repeat’

b. *uti-kiru* (hit-cut) ‘abort’

c. *uti-tokeru* (hit-thaw) ‘come out of one’s shell’

d. *tori-midasu* (take-disturb) ‘come apart’

e. *tori-simaru* (take-fasten) ‘police’

f. *hiki-tatu* (pull-stand) ‘look well’

The V_1-V_2s listed in (3) are very productive, compositional, and transparent as to how their meanings are constructed from their component verbs. Semantically speaking, the V_2s in (3) take V_1’s meaning as a semantic argument, or embed V_1’s semantics. The V_1-V_2s illustrated in (4) are compositional in some way, but it seems difficult to find a regularity governing these V_1-V_2s. In (4b), we would find some kind of causation relation held between *odoru* (dance) and *tukareru* (get.tired), but in (4c), *tobu* (jump) seems to describe a manner in which someone gets up. Besides, these V_1-V_2s are restricted in variation; while we can say *hare-wataru* (clear.up-cross.over), we would never say something like **kumori-wataru* (cloud.up-cross.over), even though it makes sense semantically or pragmatically. (5) shows us V_1-V_2s that are non-compositional and highly lexicalized. In the V_1-V_2 in (5b), *uti-kiru* (hit-cut) 'abort', for instance, neither *utu* nor *kiru* contributes their meanings to the compound's meaning 'abort'. V_1-V_2s of this kind are much more restricted in variation than those in (3) and (4).

Baldwin and Bond (2002) regards the V_1-V_2s in (4) and (5) as MWEs, especially semi-fixed expressions, since those V_1-V_2s represent, more or less, "idiosyncratic interpretations that cross word boundaries," and they undergo inflections but do not allow syntactic modifications to intervene, for example. As for the V_1-V_2s in (3), on the other hand, Baldwin and Bond (2002) does not include them as MWEs because of their productive and compositional nature. However, as we will see later on, the V_1-V_2s in (3) should be distinguished into three types according to their syntactic and semantic characteristics, and hence should be analyzed in terms of a linguistic point of view. Therefore, in this dissertation, I would like to regard all kinds of V_1-V_2s, no matter whether they are (non-)compositional, as MWEs in light of the fact that they all need sophisticated linguistic treatments.

In spite of their pervasiveness, variety, and complexity, little attention was been paid to V_1-V_2 compounds in the previous studies of computational grammars of Japanese (Mitsuishi et al., 1998; Ohtani et al., 2000; Siegel & Bender, 2002; Masuichi & Ôkuma, 2003). Siegel and Bender (2002), for example, merely try to register all V_1-V_2s in the lexicon, identifying them as single words. However, it is certain from the discussion in the previous section that since V_1-V_2 compounds in Japanese are MWEs, and especially those in (3) are very productive, this approach would suffer from a lexical proliferation problem, among other things. Moreover, dealing with the V_1-V_2s in a fully compositional way with no distinction would face problems involving an overgeneration problem and an idiomaticity problem. As mentioned above, not all imaginable combinations of verb, like **kumori-wataru* (cloud.up-cross.over), are really possible as V_1-V_2 compounds. The fully compositional approach cannot rule out those impossible cases. Besides, the approach has no way of predicting differing compositions of meanings of V_1-V_2s. Indeed, the meanings of V_1-V_2s in (3), (4), and (5) seem to be formed by different rules or principles.

V_1-V_2 compounds in Japanese, as MWEs, also pose serious problems for Japanese NLP. Thus, it is apparent that for a deep linguistic approach to Japanese NLP to be feasible, we have to find an appropriate way to deal with V_1-V_2s.

1.3 Summary

In chapter one, I gave the background and motivation of this dissertation. I first discussed differences between theoretical linguistics and Natural Language Processing (NLP), which are summarized as follows.

- Theoretical linguistics works on idealized data, which is natural as a scientific study. NLP, on the other hand, works on unidealized data, since the input to an NLP system is necessarily spontaneous utterances.

- Theoretical studies usually deal with one particular phenomenon, and analyze it in depth. In contrast, an NLP system has to process all sentences in the input, in which all kinds of phenomena, including even pragmatic ones, show up.

- Theoretical linguistics can ignore ambiguity as long as it is irrelevant to the discussion. However, NLP systems are expected to resolve any kind of ambiguity, or to pick the most plausible analysis among them.

- Linguistic theories usually do not have to worry if their analyses or computations are executed efficiently, whereas processing efficiency is one of the biggest issues in NLP.

In spite of the differences, I motivated taking advantage of deep linguistic treatments for NLP on the following grounds.

- Semantic representations, which are hard to come by with shallow processing techniques, are constructed in parallel with phrase structures by means of a deep linguistic analysis.

- Deep linguistic treatments do not need to rely as heavily on statistical information as shallow processing techniques, as long as a grammar describes a language precisely.

And we saw that recent developments in deep linguistic treatments have made it possible to use them in practical applications since they are now able to be executed in an efficient way, can find the best parse among ambiguities, and can have broad coverage.

Next I discussed how so-called **Multiword Expressions** (**MWEs**) pose serious problems to NLP; they resist the following simple solutions.

- Regard all MWEs as totally compositional, and derive all of them by means of some sort of rule.

- Regard all MWEs as single words, and register all of them in the lexicon.

The first solution would suffer from an **overgeneration problem** and **idiomaticity problem**. Put differently, it would not be able to rule out unattested MWEs nor capture their idiomatic characteristics. The second solution would face a **flexibility problem** and **lexical proliferation problem**. That is, since MWEs are sometimes syntactically flexible and productive, treating them as single words cannot be realistic. These strongly suggest that we should incorporate a linguistic analysis into the treatment of MWEs.

I regarded V_1-V_2 compounds in Japanese as MWEs because of the fact that they all need sophisticated linguistic treatments. As such, Japanese V_1-V_2 compounds would pose serious problems to Japanese NLP in the same way as other MWEs, although previous computational grammars of Japanese have not addressed this problem.

1.4 The purposes of the dissertation

In the following chapters, I will show how we can analyze V_1-V_2 compounds in Japanese, and how the analyses can be implemented in a large-scale computational grammar of Japanese. My analysis is based on linguistic analyses and observations that have been adopted by previous linguistic studies on V_1-V_2s, particularly Kageyama (1993) and Matsumoto (1996), but is arranged to be more suitable for NLP. In doing so, I will make reference to criteria proposed by Hasida (1997), by which a linguistic theory is judged to be appropriate for NLP. By presenting my analysis, I will suggest what an NLP grammar should be and how NLP can take advantage of theoretical linguistics.

My analysis is implemented in the JACY grammar (Siegel & Bender, 2002), which is a grammar of Japanese given the formalism of Head-driven Phrase Structure Grammar (HPSG) (Pollard & Sag, 1987, 1994; Sag & Wasow, 1999). In implementing the analysis, I make use of the LKB system (Copestake, 2002), the grammar development environment.

Also, I will evaluate the coverage and precision of my implementation through corpus annotation. The [incr tsdb()] system (Oepen & Carroll, 2000) helps us conduct a grammar performance profiling of this kind.

Finally, I will discuss the desirable relationship between theoretical linguistics and NLP.

Chapter 2 Linguistic analyses of V_1-V_2 compounds

V_1-V_2 compounds in Japanese have been extensively studied by many linguists (Teramura, 1969; Yamamoto, 1983; Tagashira & Hoff, 1986; Kageyama, 1993; Matsumoto, 1996; Himeno, 1999; Fukushima, 2003). In this chapter, I will review some of these studies and critique them in terms of their potentials to contribute NLP.

2.1 Linguistic theory: from an engineering perspective

In the early stage of generative grammar, when the formalism of generative analyses was relatively simple and straightforward, there were fruitful interactions between theoretical linguistics and NLP. That is, generative grammarians analyzed linguistic phenomena and formalized them in a parsimonious way, and developers of NLP applications implemented the analysis on a computational system. This was possible because theoretical linguistics at that time aimed mainly at a formal or mathematically well-defined description of a wide variety of linguistic phenomena. Among such linguistic theories are context-free grammar (Hopcroft et al., 2001) and case grammar (Fillmore, 1968). Their formalisms were so simple and explicit that NLP researchers were willing to use them, resulting in efficient and practical parsing systems (Allen, 1994) or word sense disambiguation techniques (Yarowsky, 1992).

Around the 1970's, however, linguistic theories were getting more abstract and losing their mathematical explicitness in order to achieve explanatory adequacy (Chomsky, 1965). Probably this tendency itself might be on the right track for a science of language, but NLP, as engineering of language, still needed a simple, explicit, and efficient theory. Consequently, theoretical linguistics and NLP began diverging; theories became more and more abstract and concentrated on very specific problems that might not be problems for NLP, while NLP turned to shallow statistical processing techniques, which have been studied eagerly until today and achieved moderate success, but seemed to give up a systematic and parsimonious description of language. The divergence of the two studies of language resulted in the differences described in §1.2.1, but, as mentioned there, NLP should benefit from theoretical insights about language.

Due to the divergence, it is not an easy job to implement theoretical analyses on a computational system directly. Relevant to this issue are the criteria proposed by Hasida (1997), which are described below.

Importance of Phenomenon: The phenomena a theory tries to explain should be important not only for linguistics but also for NLP.

Simplicity of Design: A theory should make NLP systems simple.

Efficiency of Computation: It must be possible to execute the computation posited by a theory efficiently.

Availability of Input: The inputs that a theory makes reference to should be easily available to NLP systems.

Let me take examples to make these statements more comprehensible. Regarding the first criterion, **Importance of Phenomenon**, Hasida (1997) cites a case of parasitic gaps illustrated in (6).

(6) kore-wa watasi-ga yom-azu-ni ka-tta hon-desu
this-TOP I-NOM read-not-DAT buy-PAST book-COP
'This is the book that I bought without reading.'

In (6), the complement positions of *yom* (read) and *kau* (buy) are occupied by gaps, and both gaps seem to be interpreted as referring to the same thing, namely *hon* (book). Linguistic theory has to predict which gap refers to which element. Thus it has to deal with a sentence like (6). On the other hand, Hasida claims that NLP does not have to explain the behavior of parasitic gaps, since sentences containing parasitic gaps rarely appear in conversation or writings. Thus, NLP systems should not include a theory of parasitic gaps.

Simplicity of Design is intended to mean that a less *ad hoc* and computationally explicit theory contributes to NLP as well. To illustrate this, Hasida takes parasitic gaps as an example again. Look at the contrast below.

(7) a. She is a girl whom I talk to *t* whenever I see *e*.

b.*She is a girl who *t* talk to me whenever I see *e*.

For example, a tentative hypothesis that a parasitic gap (*e* above) cannot be parasitic on a subject gap (*t* in (7b)) is too *ad hoc* to satisfy the Simplicity of Design. Another illustration Hasida cites concerns relevance theory (Sperber & Wilson, 1986). Relevance theory tries to explain pragmatic phenomena by means of a general principle, and is therefore far from *ad hoc*. However, the theory is neither explicit nor mathematically well-defined, and hence does not meet the requirement of Simplicity of Design.

Efficiency of Computation means that it is important for NLP to be able to execute linguistic theory efficiently. In other words, any theory whose design prevents us from processing it efficiently is more or less useless for NLP. Hasida takes the case of Montague semantics (Dowty et al., 1981) to illustrate this condition. As is well known, Montague semantics presupposes logical omniscience. Introducing this notion to NLP would require us to prepare all facts that hold in this world and to compute a meaning of a given phrase in light of the huge amount of facts.

Availability of Input involves whether information that a theory makes reference to has to be easily available to computational systems. Needless to say, preparing all facts in the world, as Montague semantics does, definitely violates this condition. World knowledge is hard to encode in a way computational systems can easily access, since **i)** the amount of such knowledge is so huge that it is almost impossible to exhaust, and **ii)** such knowledge could change its content in accord with contexts in which it appears, thus we have to give it a representation that can capture its dynamic nature.

Similarly, Raskin and Nirenburg (1998) also provides the following properties that a linguistic analysis must have for it to be useful within a natural language processing system.

- Wide coverage (can handle a variety of input)
- Tractable (can be implemented in a working system)
- Robust (can handle unknown or ill-formed input)
- Better than a base-line [e.g. all NPs are singular, definite]
- Portable to new domains and language pairs

The first two of these criteria overlap with those of Hasida (1997), while the other three are more application-oriented.

In the rest of the dissertation, I concentrate on the criteria proposed by Hasida (1997). The four criteria enable us to evaluate linguistic theories that have been proposed until today. In the following section, I will take up some linguistic analyses of V_1-V_2 compounds and examine them to see if they meet the criteria.

2.2 Kageyama (1993)

2.2.1 Syntactic V_1-V_2s vs Lexical V_1-V_2s

Kageyama (1993) divides V_1-V_2 compounds into two types according to where they are formed; One is **Syntactic V_1-V_2 compounds**, which are formed in the syntax, and the other type is **Lexical V_1-V_2 compounds**, which are formed in the lexicon. These two are illustrated in Figures 1 and 2, respectively. Appearing below are examples of each type;

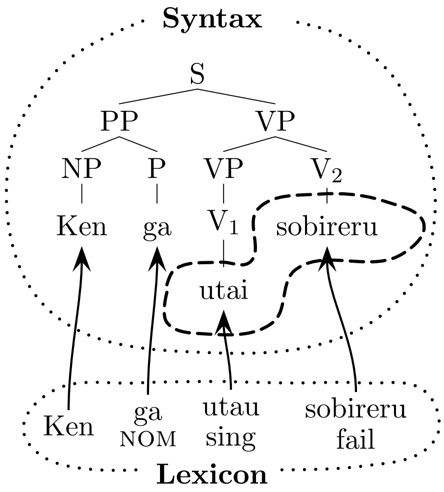


Figure 1: Syntactically derived V_1-V_2 compound

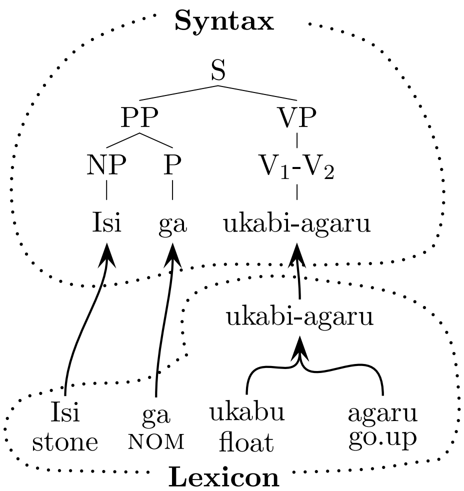


Figure 2: Lexically derived V_1-V_2 compound

V_1-V_2s in (8) correspond to syntactic V_1-V_2 compounds, while (9) contains lexical V_1-V_2 compounds.

(8) a. *kaki-hazimeru* (write-begin) 'begin to write'

b. *tabe-owaru* (eat-finish) 'finish eating'

c. *hanasi-tuzukeru* (speak-continue) 'continue speaking'

d. *ugoki-dasu* (move-take.out) 'begin to move'

e. *tabe-kakeru* (eat-be.about.to) 'be about to eat'

(9) a. *uti-korosu* (shoot-kill) 'shoot to death'

b. *nomi-aruku* (drink-walk) 'tour bars'

c. *si-nokosu* (do-leave) 'leave undone'

d. *kiki-kaesu* (ask-return) 'ask back'

e. *oi-dasu* (chase-take.out) 'send out'

Kageyama shows a variety of syntactic and semantic phenomena that support the distinction between syntactic V_1-V_2s and lexical V_1-V_2s. An example of a syntactic difference noted by Kageyama is that syntactic V_1-V_2s allow phrasal V_1, whereas lexical V_1-V_2s do not. For example, only syntactic V_1-V_2s allow verbal proform *soo suru* (so do) 'do so' in V_1, since *soo suru* is a phrase, and lexical V_1-V_2s exclude phrases in it. (10) illustrates syntactic V-Vs with the verbal proform in V_1, while (11) shows the impossibility of such an occurrence with lexical V-Vs.

(10) a. *kaki-hazimeru* (write-begin) 'begin to write'

a′. *soo si-hazimeru* (so do-begin) 'begin to do so'

b. *tabe-owaru* (eat-finish) 'finish eating'

b′. *soo si-owaru* (so do-finish) 'finish doing so'

(11) a. *uti-korosu* (shoot-kill) 'shoot to death'

a′. **soo si-korosu* (so do-kill) '?'

b. *nomi-aruku* (drink-walk) 'tour bars'

b′. **soo si-aruku* (so do-walk) '?'

Honorifics in Japanese also support his distinction. Honorific verbs in Japanese *o-V-ni naru* (HON-V-DAT become) should be thought of as a phrase. It follows that such verbs can appear only in the V_1 position of syntactic V_1-V_2. (12) and (13) show the behavior of syntactic V_1-V_2s and lexical V_1-V_2s, respectively.

(12) a. *kaki-hazimeru* (write-begin) 'begin to write'

a′. *o-kaki-ni nari-hazimeru* (HON-write-DAT become-begin) 'begin to write'

b. *tabe-owaru* (eat-finish) 'finish eating'

b′. *o-tabe-ni nari-owaru* (HON-eat-DAT become-finish) 'finish eating'

(13) a. *uti-korosu* (shoot-kill) 'shoot to death'

a′.**o-uti-ni nari-korosu* (HON-shoot-DAT become-kill) 'shoot to death'

b. *nomi-aruku* (drink-walk) 'tour bars'

b′.**o-nomi-ni nari-aruku* (HON-drink-DAT become-walk) 'tour bars'

Passivization of V_1 also seems to be a good indicator in deciding which category a given compound belongs to, though there have been continuous debates on whether (direct) passivization in Japanese is a syntactic process or not. Relevant data shown in (14) and (15) illustrate the impossibility of passivizing V_1 of lexical V_1-V_2s.

(14) a. *kaki-hazimeru* (write-begin) 'begin to write'

a′. *kak-are-hazimeru* (write-PASS-begin) 'begin to be written'

b. *tabe-owaru* (eat-finish) 'finish eating'

b′. *tabe-rare-owaru* (eat-PASS-finish) 'finish being eaten'

(15) a. *uti-korosu* (shoot-kill) 'shoot to death'

a′.**ut-are-korosu* (shoot-PASS-kill) '?'

b. *nomi-aruku* (drink-walk) 'tour bars'

b′.**nom-are-aruku* (drink-PASS-walk) '?'

Only syntactic V_1-V_2s allow a *VN-suru* (VN-do) in V_1 in accord with Kageyama's claim that *VN-suru* is formed by incorporation of *VN-wo* (VN-ACC) into *suru*. This tells us whether a given compound is lexical or not. (17a′,b′) indicate that we cannot substitute *VN-suru* into the V_1 of lexical V_1-V_2s.

(16) a. *kaki-hazimeru* (write-begin) 'begin to write'

a′. *hikki-si-hazimeru* (write-do-begin) 'begin to write'

b. *tabe-owaru* (eat-finish) 'finish eating'

b′. *syokuzi-si-owaru* (eat-do-finish) 'finish eating'

(17) a. *uti-korosu* (shoot-kill) 'shoot to death'

a′.**zyûgeki-si-korosu* (shoot-do-kill) 'shoot to death'

b. *nomi-aruku* (drink-walk) 'tour bars'

b′.**insyu-si-aruku* (drink-do-walk) 'tour bars'

In Kageyama (1993), it is argued that verbs in a reduplicated form, *nomi-ni nomu* (drink-DAT drink) 'drink so much', *hataraki-ni hataraku* (work-DAT work) 'work so hard', are syntactically derived words, since this allows a causativized verb and a passivized verb, which should be thought of as syntactically derived, to be used as its base verb.

(18) a. *yom-ase-ni yom-aseru* (read-CAUS-DAT read-CAUS) 'make someone read continuously'

b. *nak-are-ni nak-areru* (cry-PASS-DAT cry-PASS) 'adversely affected by someone's crying continuously'

Then we predict that a lexical V_1-V_2 prohibits its V_1 from being reduplicated, and this is indeed the case as in (20a′,b′).

(19) a. *kaki-hazimeru* (write-begin) 'begin to write'

a′. *kaki-ni kaki-hazimeru* (write-DAT write-begin) 'begin to write continuously'

b. *tabe-owaru* (eat-finish) 'finish eating'

b′. *tabe-ni tabe-owaru* (eat-DAT eat-finish) 'finish eating continuously'

(20) a. *uti-korosu* (shoot-kill) 'shoot to death'

a′.**uti-ni uti-korosu* (shoot-DAT shoot-kill) 'shoot to death continuously'

b. *nomi-aruku* (drink-walk) 'tour bars'

b′.**nomi-ni nomi-aruku* (drink-DAT drink-walk) 'tour bars continuously'

As for their semantics, syntactic V_1-V_2s consistently show that V_2 embeds V_1 as exemplified in glosses at the above-mentioned examples, while the semantics of lexical ones are sometimes idiomatic and hard to generalize about even when they seem compositional. Idiomatic cases are shown in (21), which repeats (5) on page 8.

(21) a. *kuri-kaesu* (turn.over-give.back) 'repeat'

b. *uti-kiru* (hit-cut) 'abort'

c. *uti-tokeru* (hit-thaw) 'come out of one's shell'

d. *tori-midasu* (take-disturb) 'come apart'

e. *tori-simaru* (take-fasten) 'police'

f. *hiki-tatu* (pull-stand) 'look well'

It is difficult to find any semantic relation between a compound as a whole and its component verbs. In (21e), for instance, neither *toru* (take) nor *simaru* (fasten) contributes to the meaning of the compound *tori-simaru* (police). (4) on page 8, repeated here as (22), includes compositional cases, but no generalization seems possible as to how their meanings are composed, as the translations in (22) indicate.

(22) a. *naki-sakebu* (cry-scream) 'cry and scream'

b. *odori-tukareru* (dance-get.tired) 'get tired from dancing'

c. *tobi-okiru* (jump-get.up) 'get up swiftly'

d. *tataki-waru* (hit-break.in.half) 'break in half by hitting'

e. *hare-wataru* (clear.up-cross.over) 'break into sunshine'

f. *yuzuri-ukeru* (yield-receive) 'inherit'

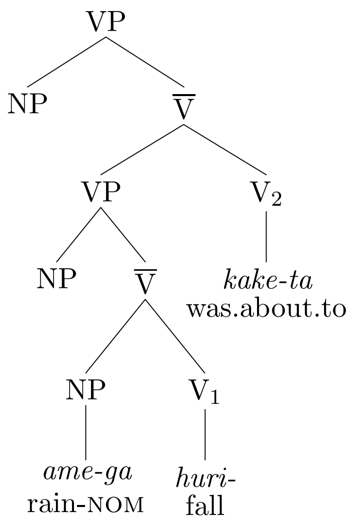
So far we have seen Kageyama's observations that strongly imply that we need to distinguish two types of V_1-V_2 compounds: syntactic V_1-V_2 compounds and lexical V_1-V_2 compounds.

2.2.2 Syntactic V_1-V_2 compounds

Kageyama classifies syntactic V_1-V_2s into three types, and analyzes them in the Government and Binding (GB) framework (Chomsky, 1981). The first one has a raising structure in which V_2 syntactically embeds a VP that is headed by V_1. The second one is somewhat similar to the first one in that V_2 of this type also embeds a VP headed by V_1, but this type of V_1-V_2 has a control structure. The other type involves $\overline{\mathrm{V}}$ complementation where V_2 embeds $\overline{\mathrm{V}}$ rather than VP. (23) illustrates these three types.

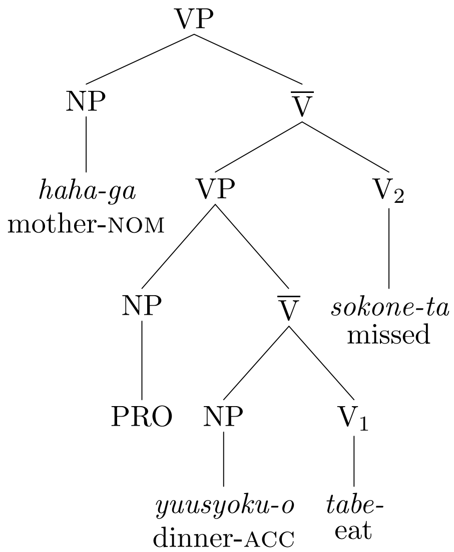
(23) a. Raising type

ame-ga huri-kake-ta
rain-NOM fall-be.about.to-PAST
'It was about to rain.'



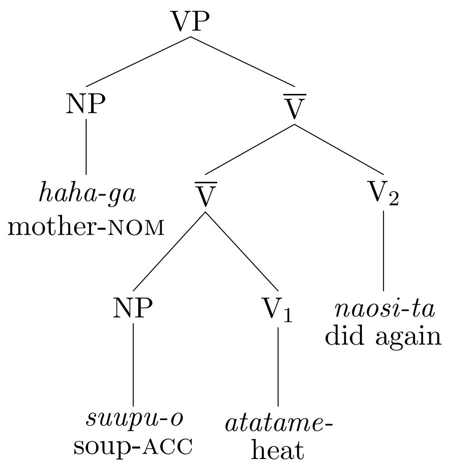
b. Control type

haha-ga yuusyoku-o tabe-sokone-ta
mother-NOM dinner-ACC eat-miss-PAST
'Mother missed eating dinner.'



c. $\overline{V}$ complementation type

haha-ga suupu-o atatame-naosi-ta
mother-NOM soup-ACC heat-do.again-PAST
'Mother reheated the soup.'



We can confirm this distinction by several tests Kageyama proposes. Raising and control types behave differently from the $\overline{V}$ complementation type in terms of direct passivization of a V_1-V_2 compound as a whole, among other thing; raising and control type do not allow passivization of V_1-V_2, while $\overline{V}$ complementation type allows it.

(24) a.*hon-ga Ken-ni [$_{V-V}$ yomi-{kake / sokone}]-rareru
book-NOM Ken-DAT [$_{V-V}$ read-{be.about.to / fail}]-PASS
'The book is {been about / failed} to read by Ken.'

b. hon-ga Ken-ni [$_{V-V}$ yomi-naos]-areru
book-NOM Ken-DAT [$_{V-V}$ read-do.again]-PASS
'The book is reread by Ken.'

Kageyama's analysis of direct passivization follows Sugioka (1984), where the direct passive morpheme *rareru* subcategorizes for $\overline{\mathrm{V}}$. According to Kageyama, *rareru*'s functions are twofold; **i)** it suppresses the passivized verb's external argument and renders the argument an adjunct marked by *ni* or *niyotte*, and **ii)** it absorbs the passivized verb's accusative case, so that the object argument of the verb would move to the subject position where the argument is given nominative case. It follows that the raising type cannot be passivized since the V_2 does not have an external argument. Note that *ame-ga* (rain-NOM) in (23a) is an internal argument. Kageyama attributes the impossibility of passivization of the control type to the Relativized Minimality of Rizzi (1990). That is, *yuusyoku-o* (dinner-ACC) in (23b), for instance, cannot move to a subject position by direct passivization because of the existence of intervening PRO. On the other hand, the $\overline{\mathrm{V}}$ complementation types does not violate the two passivization conditions.

Kageyama also observes semantic differences among the three types. They differ in whether V_2s thematically restrict their subjects and objects. The raising type puts no restriction on a subject, while control and $\overline{\mathrm{V}}$ complementation types do.

(25) ame-ga huku-o nurasi-{kakeru/*sokoneru/*naosu}
rain-NOM clothes-ACC humidify-{be.about.to/fail/do.again}
'Rain {is about/fail/work over} to humidify clothes.'

sokoneru (fail) and *naosu* (do.again) are supposed to put a thematic restriction, such as agentivity, on the subject, but *ame* (rain) violates it. The structures in (23) explain the difference; the control and $\overline{\mathrm{V}}$ complementation types select a subject, while the raising type does not. Similarly, only the $\overline{\mathrm{V}}$ complementation type puts restrictions on its object. Note that *atama-o hiyasu* 'cool off' is an idiom. Thus *atama-o* in this idiom cannot meet any thematic condition. This prediction is borne out.

(26) Ken-ga atama-o hiyasi-{kakeru/sokoneru/⁇naosu}
Ken-NOM heat-ACC cool-{be.about.to/fail/do.again}
'Ken {is about/fail/work over} to cool off.'

Kageyama's account is that an embedded object in $\overline{\mathrm{V}}$ of the $\overline{\mathrm{V}}$ complementation type is theta-marked not only by V_1 but also V_2 that embeds $\overline{\mathrm{V}}$. That is, *suupu-o* (soup-ACC) in

(23c) is theta-marked by both *atatameru* (heat) and *naosu* (do.again). This assumption is possible thanks to the idea proposed by Baker (1989) that the theta-marking of an internal argument is done within the projection of $\overline{\mathrm{V}}$. On the other hand, according to Kageyama, the V_2s that select VP complements take as an argument the whole embedded clause rather than the embedded object alone. In this way, the contrast in (26) is explained.

Theoretical framework aside, evidence for the three-way distinction made by Kageyama is overwhelming, and Matsumoto (1996) and Hashimoto (2003b, 2003c) posit similar analyses. Later I will present my analysis of syntactic V_1-V_2s that owes much to Kageyama (1993).

2.2.3 Lexical V_1-V_2 compounds

Kageyama proposes that a principle involving argument structure governs lexical V_1-V_2 compounding. Then he proposes the following principle.

(27) **The Transitivity Harmony Principle**

Given the three argument structures below, lexical compound verbs are built by combining two verbs with the same type of argument structure.

(a) transitive verbs: $(x \langle y \rangle)$

(b) unergative verbs: $(x \langle\rangle)$

(c) unaccusative verbs: $\langle y \rangle$

Below is the explanation of the principle cited from Kageyama (1999), citing his earlier 1993 paper.

> In the argument structures above, x represents external argument, and y internal argument. Since both transitive and unergative verbs have external argument, their argument structures are deemed of the same type, while unaccusative verbs, lacking external argument, are assumed to constitute a separate type. Kageyama argues that the formation of lexical compound verbs is contingent upon the argument-structure types, on the grounds that in addition to the combinations of transitive-transitive and unergative-unergative, mixed combinations of transitives and unergatives are attested while unaccusatives may be compounded only with unaccusatives.

The examples below that are also cited from Kageyama (1999) support his claim.

(28) a. transitive V_1 + unergative V_2

(teki-o) mati-kamaeru "(enemies) wait-be.prepared"

b. unergative V_1 + transitive V_2

(me-o) naki-harasu "(eyes) cry-cause.swell"

(29) a.*transitive V_1 + unaccusative V_2

**tuki-otiru* "push-fall" (cf. transitive + transitive: *tuki-otosu* "push-make.fall")

b.*unaccusative V_1 + transitive V_2

**ore-mageru* "snap-bend" (cf. transitive + transitive: *ori-mageru* "fold-bend")

c.*unergative V_1 + unaccusative V_2

**(me-ga) naki-hareru* "(eyes) cry-get.swollen" (cf. (28b))

d.*unaccusative V_1 + unergative V_2

**koroge-oriru* "tumble-step.down" (cf. unaccusative + unaccusative: *koroge-otiru* "tumble-fall")

However, as Matsumoto (1996) and Kageyama himself point out, there are grammatical cases that the Transitivity Harmony Principle incorrectly rules out, examples of which are shown below.

(30) a. *uti-agaru* (hit-go.up) (unergative-unaccusative)

b. *hari-tuku* (paste-be.attached) (transitive-unaccusative)

c. *yaki-tuku* (burn-be.attached) (transitive-unaccusative)

d. *musubi-tuku* (fasten-be.attached) (transitive-unaccusative)

For these counterexamples, Kageyama postulates that they are formed through back formation (intransitivization in this case) from their 'canonical' counterparts illustrated below.

(31) a. *uti-ageru* (hit-raise) (unergative-transitive)

b. *hari-tukeru* (paste-attach) (transitive-transitive)

c. *yaki-tukeru* (burn-attach) (transitive-transitive)

d. *musubi-tukeru* (fasten-attach) (transitive-transitive)

According to Kageyama, there are other cases that do not obey his Transitivity Harmony Principle. The remaining counterexamples have *-komu* (go.in), *-saru* (leave), and *-dasu* (take.out) as their V_2.

(32) a. *oi-komu* (chase-go.in) (transitive V_1)

b. *kake-komu* (run-go.in) (unergative V_1)

c. *oti-komu* (fall-go.in) (unaccusative V_1)

(33) a. *moti-saru* (have-leave) (transitive V_1)

b. *tobi-saru* (jump-leave) (unergative V_1)

c. *sugi-saru* (pass-leave) (unaccusative V_1)

(34) a. *oi-dasu* (chase-take.out) (transitive V_1)

b. *tobi-dasu* (jump-take.out) (unergative V_1)

c. *waki-dasu* (bord-take.out) (unaccusative V_1)

As these examples indicate, *-komu* (go.in), *-saru* (leave), and *-dasu* (take.out) can constitute V_1-V_2s with transitive, unergative, and unaccusative verbs, resulting in the violation of the Transitivity Harmony Principle. Kageyama observes several semantic changes that take place when these compounds are formed. *-komu*, for instance, adds a dative argument that stands for something like a goal to V_1's arguments.

(35) a. neko-ga nezumi-o ou
cat-NOM mouse-ACC chase
'A cat chases a mouse.'

b. neko-ga nezumi-o kabe-ni oi-komu
cat-NOM mouse-ACC wall-DAT chase-go.in
'A cat runs a mouse to a wall.'

Besides, attaching *-komu* changes V_1's aspect from durative to perfective. As a consequence, Kageyama posits principles that involve changes of Lexical Conceptual Structure (LCS) (Hale & Keyser, 1987; Rappaport & Levin, 1988).

2.2.4 Engineering problems

I claim that Kageyama's treatments of syntactic V_1-V_2s have several problems in light of Hasida (1997). The problems are due to the GB framework. Most constraint-based lexicalist grammars, such as HPSG, Lexical Functional Grammar (LFG) (Bresnan & Kaplan, 1982), and Categorial Grammar (CCG) (Wood, 1993), can be given mathematically well-defined formalizations. HPSG, for instance, is formalized within some particular version of typed feature structure logic such as Carpenter (1992), and its computational characteristics have been studied so extensively that several efficient processing techniques are now available: Callmeier (2000) and Maxwell and Kaplan (1993), to name two. In contrast, mathematical foundations for such grammatical frameworks as GB and the Minimalist Program (Chomsky, 1995) have not been established to date. As a result, there have been few proposals for efficient processing techniques for those frameworks, and thus I claim that adopting GB analyses for NLP would lead to a violation of **Efficiency of Computation**. Notably problematic theoretical mechanisms in the GB theory are various kinds of movement operations and empty categories. Nevertheless, to deal with the fact that arguments within a VP or $\overline{\mathrm{V}}$ complement can be scrambled, Kageyama would have to posit some kind of movement such as those assumed in Saito (1985) and Hoji (1985). He also relies on an NP movement to account for Japanese passives, as briefly mentioned on page 22. Besides, his account of passivizability of V_1-V_2 relies on the (non)existence of the empty category, PRO. In contrast, my analysis below dispenses with both movement and empty categories.

Kageyama's theory of lexical V_1-V_2s should also be criticized in terms of Hasida's criteria, and I point out two problems. One is the back formation analysis. Even if this analysis is theoretically correct, positing this for NLP would cost too much, since the analysis would require us to posit machinery besides those for the Transitivity Harmony Principle, and it is unclear which compounds the back formation is applied to. Consequently, I have to say the analysis would violate **Simplicity of Design**. Kageyama's LCS analysis is the other problem, since this analysis also need another mechanism, resulting in the violation of **Simplicity of Design**. As well, it might also violate **Importance of Phenomenon**, since the analysis seems to apply only to *V_1-komu*, *V_1-saru*, and *V_1-dasu*. I will show an analysis of lexical V_1-V_2 compounds that is simpler and yet has broader coverage.

2.3 Matsumoto (1996)

Matsumoto (1996) develops a theory of syntactic V_1-V_2s that is very similar to Kageyama (1993), though Matsumoto (1996) adopts an LFG framework. Because of this similarity, this section concentrates on Matsumoto's treatments of lexical V_1-V_2.

2.3.1 Lexical V_1-V_2 compounds

Matsumoto recognizes several kinds of semantic relations holding between two component verbs that constitute lexical V_1-V_2s. (36) describes the semantic relations.

(36) a. Pair compounds

b. Cause compounds

c. Manner compounds

d. Means compounds

e. Compounds exhibiting other relations

f. Compounds with semantically deverbalized V_2

g. Compounds with semantically deverbalized V_1

Matsumoto illustrates in detail how each kind of V_1-V_2 is formed in terms of argument structure with thematic roles. These illustrations give us valuable insights into what computational implementations of lexical V_1-V_2s should be, so let us look closely at examples taken up by Matsumoto.

A pair compound consists of two verbs whose argument structures are of the same type, and arguments of V_1 and V_2 that are thematically alike are co-indexed.[1]

(37) a. V_1 ⟨th⟩ + V_2 ⟨*th*⟩ = V_1-V_2 ⟨*th*⟩

b. *hikari-kagayaku* (shine-shine) 'shine brightly'

(38) a. V_1 ⟨ag, (loc/go/etc.)⟩ + V_2 ⟨*ag*, (*loc/go/etc.*)⟩ = V_1-V_2 ⟨*ag*, (*loc/go/etc.*)⟩

b. *tobi-haneru* (jump-leap) 'jump repeatedly'

A cause compound's V_2 is always unaccusative. The internal arguments of V_2 are usually co-indexed with the internal arguments of V_1.

(39) a. V_1 ⟨th⟩ + V_2 ⟨*th*⟩ = V_1-V_2 ⟨*th*⟩

[1]Thematic role abbreviations used in the examples are as follows.

(i) **ag**ent, **pat**ient, **th**eme, **loc**ation, **go**al, sou**rc**e

The line linking arguments indicates which of the arguments are referentially identical. The italicized arguments are reflected in the argument structure of the whole.

b. *yake-sinu* (burn-die) 'die from burning'

(40) a. $\mathrm{V}_1\ \langle \mathrm{ag},\ \mathrm{pt} \rangle + \mathrm{V}_2\ \langle \mathit{th} \rangle = \mathrm{V}_1\text{-}\mathrm{V}_2\ \langle \mathit{th} \rangle$

b. *humi-katamaru* (tramp-harden) 'be tramped hard'

Manner compounds are possible with all types of V_2: unaccusative, unergative, and transitive.

(41) a. $\mathrm{V}_1\ \langle \mathrm{th} \rangle + \mathrm{V}_2\ \langle \mathit{th},\ \mathit{go/loc} \rangle = \mathrm{V}_1\text{-}\mathrm{V}_2\ \langle \mathit{th},\ \mathit{go/loc} \rangle$

b. *nagare-otiru* (flow-fall) 'flow down'

(42) a. $\mathrm{V}_1\ \langle \mathrm{ag},\ \mathrm{pt/etc.} \rangle + \mathrm{V}_2\ \langle \mathit{ag\text{-}th},\ \mathit{loc/go/src} \rangle = \mathrm{V}_1\text{-}\mathrm{V}_2\ \langle \mathit{ag\text{-}th},\ \mathit{loc/go/src} \rangle$

b. *fumi-todomaru* (stamp-remain) 'refrain from going'

They are mostly right-headed, but when V_2 is an unergative verb representing spatial motion or a related meaning, like *aruku* (walk) and *mawaru* (go.around), the arguments of V_1 and V_2 are mixed in the argument structure of the compound.

(43) a. $\mathrm{V}_1\ \langle \mathrm{ag},\ \mathit{pt} \rangle + \mathrm{V}_2\ \langle \mathit{ag\text{-}th},\ \mathit{loc/go/src} \rangle = \mathrm{V}_1\text{-}\mathrm{V}_2\ \langle \mathit{ag\text{-}th},\ \mathit{pt},\ \mathit{loc/go/src} \rangle$[2]

b. *nomi-aruku* (drink-walk) 'drink around'

c. *tabe-aruku* (eat-walk) 'eat around'

d. *sagasi-mawaru* (search-go.around) 'go around, searching'

e. *atume-mawaru* (collect-go.around) 'go around, collecting'

[2] It is not the case that those V_1-V_2s in (43) can take two accusative objects, *pt* and *loc/go/src*, because of the Double-*o* Constraint (Harada, 1973). as illustrated in (i).

(i) *Ken-ga sake-o Tokyo-o nomi-aruku
Ken-NOM sake-ACC Tokyo-ACC drink-walk
'Ken drinks sake around Tokyo.'

Matsumoto (1996) seems to assume that one of the two object arguments, *pt* and *loc/go/src*, is realized with a case marker other than the accusative marker *o*.

(ii) Ken-ga sake-o Tokyo-de nomi-aruku
Ken-NOM sake-ACC Tokyo-at drink-walk
'Ken drinks sake around Tokyo.'

My analysis, however, assumes that those V_1-V_2s in (43) take only one object argument, either *pt* or *loc/go/src*.

f. *moti-saru* (have-leave) 'go away with'

g. *sute-saru* (throw.away-leave) 'leave after throwing away'

A means compound's V_2 is either unergative or transitive, and so is its V_1.

(44) a. V_1 ⟨ag, pt, ...⟩ + V_2 ⟨*ag*, *?*⟩ = V_1-V_2 ⟨*ag*, *?*⟩

b. *nage-katsu* (throw-win) 'defeat ... by throwing (or 'win a pitching game')'

(45) a. V_1 ⟨ag, pt⟩ + V_2 ⟨*ag*, *pt*⟩ = V_1-V_2 ⟨*ag*, *pt*⟩

b. *naguri-korosu* (strike-kill) 'kill by striking'

There are relations holding between V_1 and V_2 other than described above. (46) is an example of such relation, which is difficult to characterize.

(46) a. V_1 ⟨ag, pt/loc, (...)⟩ + V_2 ⟨*ag*, *pt*, *loc/go*⟩ = V_1-V_2 ⟨*ag*, *pt*, *loc/go*⟩

b. *nori-suteru* (ride-abandon) 'abandon (something) one rides'

(47) a. V_1 ⟨ag-src, th, go⟩ + V_2 ⟨*ag-go*, *th*, *src*⟩ = V_1-V_2 ⟨*ag-go*, *th*, *src*⟩

b. *yuzuri-ukeru* (yield-receive) 'inherit'

Compounds with semantically deverbalized V_2 can be regarded as left-headed. Their V_2s have lost their original meanings and ARG-STs, and have taken on adverbial meanings.

(48) a. *-tsukeru* (attach) 'hard, harshly'

sikari-tsukeru (scold-) 'scold harshly'
nirami-tsukeru (stare-) 'glare at'

b. *-tateru* (stand (transitive)) 'actively'

donari-tateru (yell-) 'yell violently'
seme-tateru (attack-) 'attack violently'

There are compounds in which V_1 has lost its meaning.

(49) a. *sasu-* (thrust) 'urgently, forcefully'

sasi-semaru (*sasu*-come.close) 'become near at hand, become urgent'
sasi-osaeru (*sasu*-hold) 'seize'
sasi-tomeru (*sasu*-stop) 'suspend'

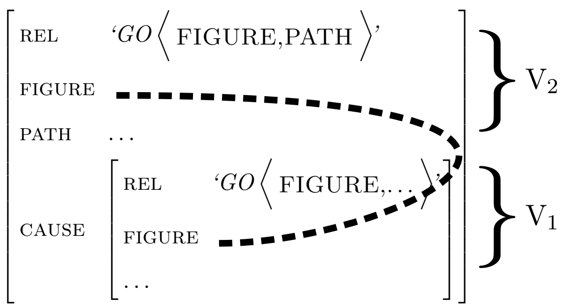
Based on these observations, Matsumoto first proposes a very general principle.

(50) **Shared Participant Condition**

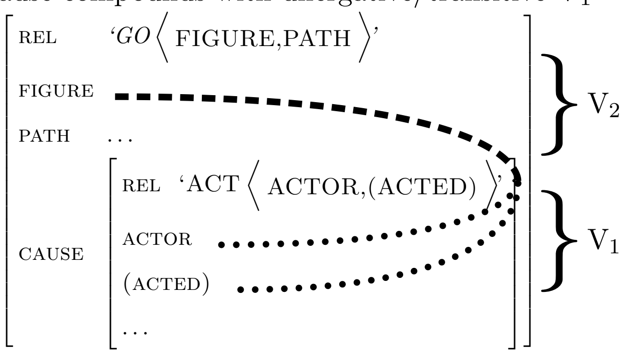
Each of the component verbs forming a compound must have at least one argument which is semantically linked to an argument of the other component verb.

In addition, he sets out semantic linking patterns for some of the semantic relations as follows.

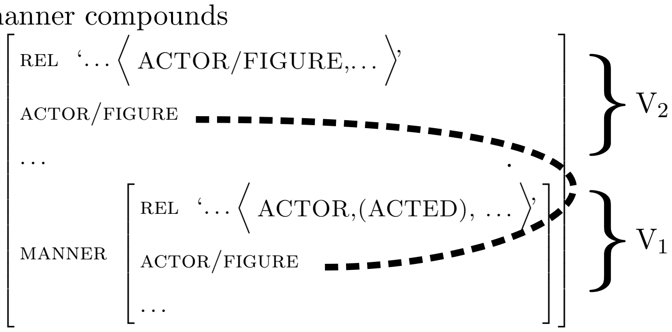
(51) a. cause compounds with unaccusative V_1



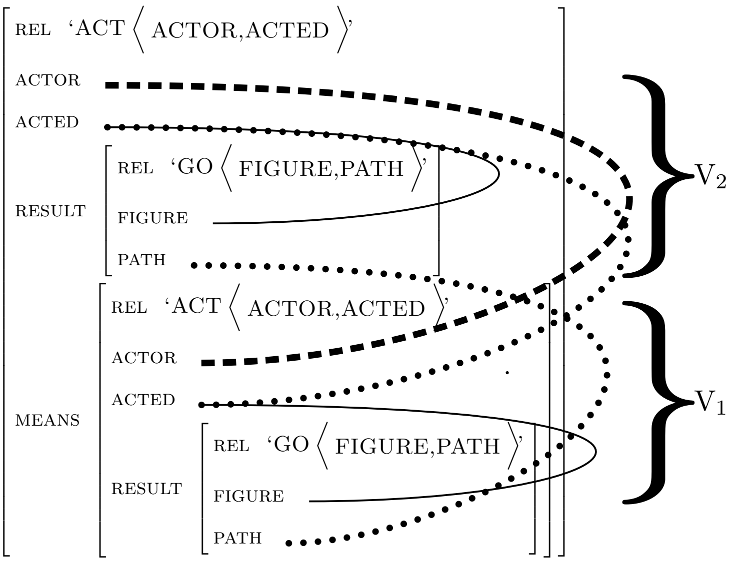
b. cause compounds with unergative/transitive V_1



c. manner compounds



d. means compounds



As indicated in (51), Matsumoto posits a lexical semantic analysis for lexical V_1-V_2s, while Kageyama's (1993) account is based on argument structure. Specific details aside, we should notice that Matsumoto's analysis seems to depend on what relation – *pair*, *cause*, *manner*, *means* – holds for each particular pair of verbs, and that it posits fine-grained LFG semantics and mapping theory.

2.3.2 Engineering problems

Matsumoto's approach has two defects. First, his assumption that what semantic relation holds between two component verbs plays a crucial role in understanding the structure of lexical V_1-V_2 compound would pose a problem for NLP, since recognizing the semantic relation depends heavily on world knowledge and we cannot expect a computational system to be equipped with such a recognition ability. Furthermore, the judgments about the semantic relations are often not so clear-cut, so it would be difficult to build a large-scale and consistent rule system to deal with them. Hence I claim Matsumoto's approach to lexical V_1-V_2s cannot meet the condition of **Availability of Input**.

Second, Matsumoto's analysis would require us to adopt very fine-grained semantic notions like ACTOR, FIGURE, and PATH and mapping theory to regulate correspondences between semantic structure and argument structure. It is easily conceivable that we will have difficulty building a large-scale grammar and lexicon with such complicating notions and apparatuses. Therefore, I would have to say that they would violate **Simplicity of Design**. As we will see later, my analysis of lexical V_1-V_2s only needs the distinction between internal arguments and external arguments, which is relatively stable and easy to implement on a large-scale grammar and lexicon.

Of course, fine-grained semantic information, if available and computationally tractable, would be a big help. Actually, there are a few computerized lexical semantic databases developed by NLP researchers. Dorr (2001) constructed a large-scale Lexical Conceptual Structure (LCS) lexicon for English and made use of it for several NLP applications such as machine translation. Takeuchi et al. (2003) built an LCS database for Japanese that was used for the computational analysis of the structure of compound nouns. These resources are precise and computationally explicit, and yet cover most verbs of the language. However, they seem to be used as a separated module. LCS is an independently motivated linguistic notion, and hence, as is commonly assumed in linguistics, it should be embedded in a larger grammatical resource such as the one I use in the dissertation.[3] Otherwise it cannot fully demonstrate its capacities and powers. It would be challenging to incorporate these lexical semantic resources into large-scale computational grammars, although I do not try that in the dissertation.

2.4 Summary

In chapter two, I looked over two linguistic studies of V_1-V_2 compounds, Kageyama (1993) and Matsumoto (1996), and criticized them from the engineering point of view proposed by Hasida (1997). Hasida (1997) gave the criteria by which a linguistic theory is judged to be suitable for NLP, which I repeat below.

Importance of Phenomenon: The phenomena a theory tries to explain should be important not only for linguistics but also for NLP.

Simplicity of Design: A theory should make NLP systems simple.

Efficiency of Computation: It must be possible to execute the computation posited by a theory efficiently.

Availability of Input: The inputs that a theory makes reference to should be easily available to NLP systems.

Then I provided Kageyama's (1993) argument that there are two kinds of V_1-V_2 compound in Japanese: those that are derived syntactically and those lexically derived. The evidence for the claim that Kageyama uses includes phenomena involving the verbal proform *soo suru* (so do) 'do so', honorification in Japanese *o-V-ni naru* (HON-V-DAT become), passivization of V_1, VN-suru (VN-do) in V_1, verbs in a reduplicated form, and a semantic compositionality. His distinction is crucial to properly handle V_1-V_2 compounds' productivity and compositionality, and indeed, my implementation owes much to Kageyama (1993).

[3] I use the broad coverage computational grammar of Japanese named JACY. I will describe it in §3.2.

Next I reviewed his analysis of syntactic V_1-V_2 compounds. He classifies syntactic V_1-V_2s into three types: the Raising type, the Control type, and the $\overline{V}$ complementation type. His argument for the classification concerns passivization of V_1-V_2 and the ability of each V_2 to assign a thematic role to a subject or an object. This classification is also important in dealing with the MWE nature of V_1-V_2 compounds.

Kageyama's analysis of lexical V_1-V_2 compounds, the Transitivity Harmony Principle, is based on argument structure and is applicable to many cases. Kageyama himself notices two kinds of exception, and he posits the back formation analysis and the LCS analysis for each.

Following my review of Kageyama (1993), I argued against several points of the study from an engineering perspective. First I claimed that adopting such GB notions as movement or empty categories, which are indispensable for his analysis of syntactic V_1-V_2 compounds, would violate **Efficiency of Computation**. Second, the back formation analysis and LCS analysis posited for exceptions to the Transitivity Harmony Principle would need additional machinery, and their domain of applicability seems unclear. Hence I concluded that Kageyama's analysis of lexical V_1-V_2s violates **Simplicity of Design**. As well, it might also violate **Importance of Phenomenon**, since the analysis seems to be applied only to *V_1-komu*, *V_1-saru*, and *V_1-dasu*.

I also discussed the analysis of lexical V_1-V_2 compounds proposed by Matsumoto (1996). He classifies lexical V_1-V_2s into seven types: Pair compounds, Cause compounds, Manner compounds, Means compounds, Compounds exhibiting other relations, Compounds with semantically deverbalized V_2, and Compounds with semantically deverbalized V_1. He proposed a fine-grained semantic account based on observations about their semantic properties and the ways in which arguments between V_1 and V_2 are co-indexed. My analysis of lexical V_1-V_2 compounds is in large part due to Matsumoto (1996).

However, I criticized Matsumoto's analysis in light of Hasida (1997), too. Matsumoto's assumption that what semantic relation holds between two component verbs plays a crucial role in understanding the structure of lexical V_1-V_2 compound would violate the condition of **Availability of Input**, since recognizing this semantic relation depends heavily on world knowledge and the judgments about the semantic relations are often not so clear-cut. Besides, his analysis would require us to adopt very fine-grained semantic notions and mapping theory, but then we would have trouble in building a large-scale grammar and lexicon with such complicating notions and apparatus. Therefore, I concluded that it would violate **Simplicity of Design**.

Despite their inadequacy from the engineering point of view, we have learned from Kageyama (1993) and Matsumoto (1996) that V_1-V_2 compounds in Japanese, as MWEs, have very rich internal structures that computational grammars should be capable of accounting for. Obviously, the simple approach in which all V_1-V_2 compounds are treated as single words cannot deal with them, and cannot get around the lexical proliferation problem, among others.

Chapter 3 Engineering oriented analysis of V_1-V_2 compounds

In this chapter, I will present an analysis of V_1-V_2s that is computationally more efficient and yet captures their linguistic characteristics reasonably.

3.1 Grammar development: from an engineering perspective

As discussed in §1.2.1 on page 2, a theoretical analysis usually takes up one particular phenomenon to find general principles which might gradually be extended to account for other phenomena, while a grammar used in NLP has to deal with all phenomena that appear in the input data. As a result, grammar development processes are also different between the two studies of language. I would call the grammar development process of theoretical linguistics the "depth-first" way, and that of NLP can be called the "breadth-first" way, as illustrated in Figure 3 and Figure 4.

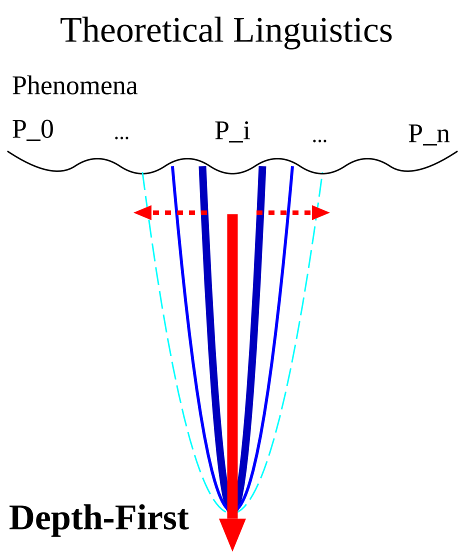


Figure 3: Depth-first grammar development

Considering the breadth-first nature of grammar development of NLP, there has to be a balance between analyses of phenomena in terms of how fine-grained they are. Otherwise, we

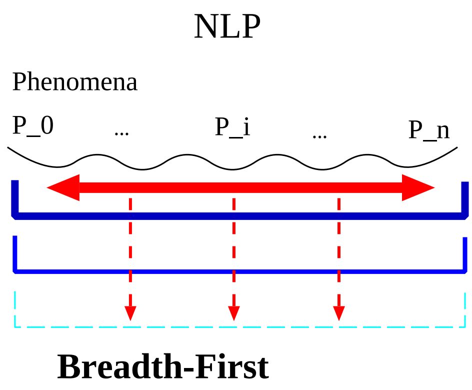


Figure 4: Breadth-first grammar development

would suffer from unexpected interactions between rules, and the grammars would quickly become unmaintainable, since the number of linguistic phenomena runs up to a considerable figure. This situation reminds us of software development where maintainability is one of the primary concerns.

At the outset, I propose the following general policies of NLP grammar development.

- NLP grammars should concentrate on phenomena that occur frequently, and should not be complicated to explain "exceptional" cases. Also, data that NLP grammars deal with should be restricted to those where speaker's grammaticality judgments are consistent.

  → **Importance of Phenomena**

- NLP grammars should be conservative or somewhat descriptive, and should not adopt theoretically advanced but controversial analyses.

  → **Simplicity of Design**

- Information that has to be stipulated in each lexical item and rule should be something that we can easily determine which category it belongs to, so that a large-scale NLP lexicon and rules could be easy to build and maintain.

  → **Availability of Input**

In addition, as stated previously, NLP grammars have to be executed in an efficient way. The PET system (Callmeier, 2000) processes a grammar very efficiently, as long as the grammar is described in (a subset of) Type Description Language ($\mathcal{TDL}$) (Krieger

& Schafer, 1994). $\mathcal{TDL}$ allows us to write a grammar and lexicon in a way which is a readable by a machine. I implement my analysis in the grammar by ? (?) (see the next section), which is written in $\mathcal{TDL}$. Thus, I assume that my analysis satisfies **Efficiency of Computation**.

All in all, there is a tension between robustness and linguistic preciseness in NLP, in a way similar to theoretical linguistics, where there has been a continuous tension between descriptive adequacy and explanatory adequacy.

3.2 JACY: a linguistically precise grammar of Japanese

I implement my analysis in the grammar of Japanese named **JACY** (Siegel, 1998, 1999, 2000b, 2000a; Siegel & Bender, 2002),[1] and it is based on a particular version of **Head-driven Phrase Structure Grammar** (**HPSG**, Pollard and Sag (1987, 1994), Sag and Wasow (1999)) with **Minimal Recursion Semantics** (**MRS**, Copestake et al. (1999, 2001), Flickinger and Bender (2003)). The first application of JACY was the Verbmobil system, a spoken language machine translation project (Wahlster, 2000), where Japanese HPSG was used in deep processing of appointment scheduling and travel reservation dialogues. The grammar was also used in an industrial application of automatic email response. Recently, the grammar contributes to the EU project DeepThought (Callmeier et al., 2004), where the main focus is on building applications for combined shallow and deep natural language processing.

Figure 5 shows the picture in which the sentence *Ken-ga neru* 'Ken sleeps.' is input through Emacs. Before parsing, the ChaSen morphological analyzer (Matsumoto et al., 2000) is invoked. Figure 6 shows the JACY output for the sentence. The left side of the figure shows the syntactic structure of the sentence, and the semantic structure is illustrated in the right side.[2] Below is an example of grammar description of JACY, which is written in the $\mathcal{TDL}$ language.[3]

```
head-subject-rule := head-final-type &
                     orth-princ &
                     scp-sbj &
                     que-princ &
                     adjacent_subjbind_check &
                     adjacent_nonhead_check &
                     affix-list &
 [J-NEEDS-AFFIX -,
```

[1] I thank Melanie Siegel and Emily Bender for their helpful comments and heartfelt support. You can download JACY from the website below.
http://www.dfki.uni-sb.de/~siegel/grammar-download/JACY-grammar.html

[2] I will describe the semantic formalism of JACY in §3.3.4.

[3] In appendix A, I will give a more comprehensive illustration of grammar description of JACY.

Figure 5: Input through Emacs

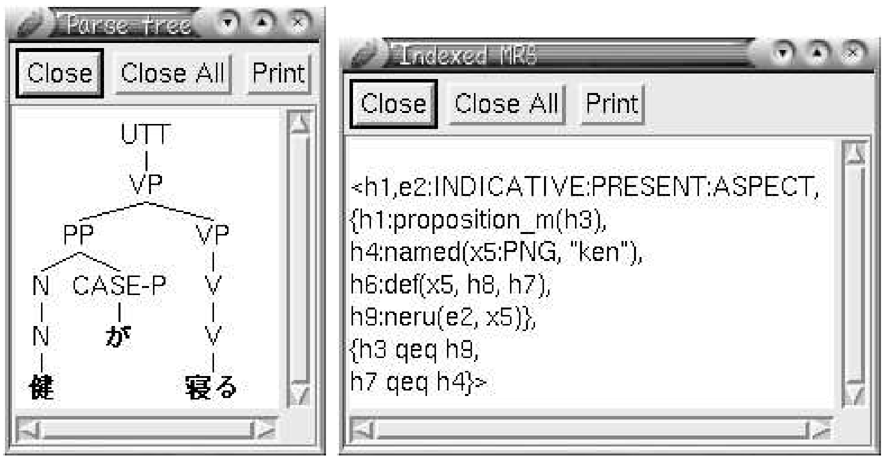


Figure 6: The example of JACY output

```
SYNSEM [LOCAL [CONT [MOD-IND #ind,
                     MOD-HAND #top],
               CAT.POSTHEAD +],
        LEX -],
C-CONT [RELS <! !>,
        HCONS <! !>],
HEAD-DTR [J-NEEDS-AFFIX -,
          SYNSEM [LOCAL [CONT [MOD-IND #ind,
                               MOD-HAND #top],
                         BAR +]]]].
```

3.3 Some basics of HPSG

Before moving on to the specific details of my implementation, let us look over relevant aspects of HPSG.[4]

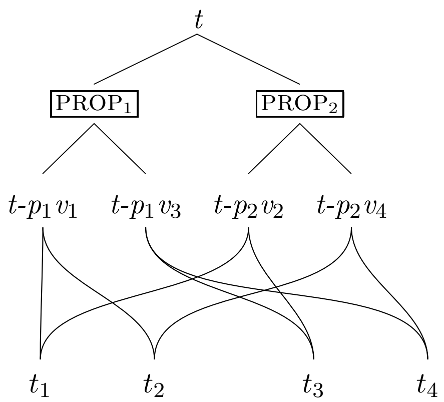
3.3.1 Types, type hierarchy, and feature structures

In HPSG, all linguistic notions, ranging from parts of speech to grammatical constructions, are represented as **type**s. Types are not totally independent of each other; some types might have the same property and/or the same value for a property in common, or they might form a group that a principle makes reference to. In order to express these kinds of generalizations, HPSG usually makes use of **type hierarchy**. Suppose that types, t_1, t_2, t_3, and t_4, have properties as follows.

(52) $$ {}^{t_1}\begin{bmatrix} \textsc{prop}_1 & val_1 \\ \textsc{prop}_2 & val_2 \end{bmatrix}, \ {}^{t_2}\begin{bmatrix} \textsc{prop}_1 & val_1 \\ \textsc{prop}_2 & val_4 \end{bmatrix}, \ {}^{t_3}\begin{bmatrix} \textsc{prop}_1 & val_3 \\ \textsc{prop}_2 & val_2 \end{bmatrix}, \ {}^{t_4}\begin{bmatrix} \textsc{prop}_1 & val_3 \\ \textsc{prop}_2 & val_4 \end{bmatrix} $$

In (52), $\textsc{prop}_n$s stand for properties and val_ns correspond to the values for the properties. Note that val_ns are also types. Then a type hierarchy expresses generalizations among the four types, from t_1 to t_4, as in (53).

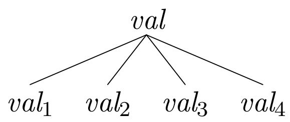
(53)



[4]The actual organization of JACY is far more complicated than the one presented in this section. But explicating JACY's architecture exhaustively is not necessarily helpful to understand my analysis.

In the hierarchy, abstract types like $t\text{-}p_1v_1$ generalize common properties among the types. For example, t_1 and t_2 have the same value val_1 for the property $\textsc{prop}_1$, and the abstract type $t\text{-}p_1v_1$ expresses the generalization. Likewise, $t\text{-}p_1v_3$ indicates that the values for the property $\textsc{prop}_1$ of t_3 and t_4 are the same. We should notice that $\boxed{\textsc{prop}_1}$ and $\boxed{\textsc{prop}_2}$ are not types, but merely imply the way in which the hierarchy is organized; in other words, they tell us that the types $t\text{-}p_1v_1$ and $t\text{-}p_1v_3$ determine $\textsc{prop}_1$, while $t\text{-}p_2v_2$ and $t\text{-}p_2v_4$ express a generalization about $\textsc{prop}_2$. Finally, t at the top of the hierarchy states that all t_ns have the properties $\textsc{prop}_1$ and $\textsc{prop}_2$. (54) shows the hierarchy of val_ns.

(54)



These generalizations that the hierarchies in (53) and (54) imply can be represented more clearly by the table 1. The table shows us how FEATURES/CONSTRAINTSs for each type are being specified. In the case of t_1, its FEATURES/CONSTRAINTS are determined by the **inheritance** of relevant information from t_1's supertypes, $t\text{-}p_1v_1$, $t\text{-}p_2v_2$, and t. As a result, t_1's properties are fully specified as in (52). Note that t's constraint [$\textsc{prop}_1$ *val*] does not contradict $t\text{-}p_1v_1$'s constraint [$\textsc{prop}_1$ val_1], since, being a subtype of *val*, val_1 is not assigned any information that is inconsistent with *val*.

Types in HPSG are represented by feature structures, or an **Attribute Value Matrix** (**AVM**) as illustrated in (52). Feature names are in ALL CAPS and type names are in *italics*. Values of features are types, so the values can be AVMs, as in (55).

(55)

$$
{}^{type}\begin{bmatrix} \textsc{feature}_1 & type_1 \\ \textsc{feature}_2 & {}^{type_2}\begin{bmatrix} \textsc{feature}_3 & type_3 \\ \textsc{feature}_4 & {}^{type_4}\begin{bmatrix} \textsc{feature}_5 & type_5 \\ \textsc{feature}_6 & type_6 \end{bmatrix} \end{bmatrix} \end{bmatrix}
$$

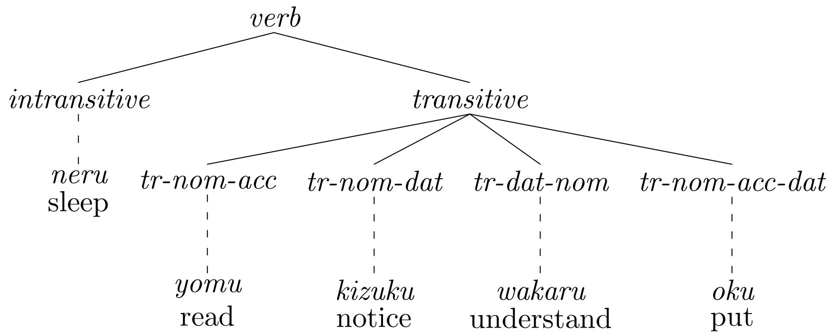
3.3.2 Lexicon

Each lexical item is also a type and thus expressed by an AVM. As such, a lexicon has a hierarchical structure that is similar to (53). Take the case of (a part of) JACY's verb hierarchy, the simplified version of which is shown in (56a).

Table 1: Table representation of the type hierarchies

TYPE	FEATURES/CONSTRAINTS	IMMEDIATE SUPER TYPE
t	$\begin{bmatrix} \text{PROP}_1 & val \\ \text{PROP}_2 & val \end{bmatrix}$	
$t\text{-}p_1 v_1$	$\begin{bmatrix} \text{PROP}_1 & val_1 \end{bmatrix}$	t
$t\text{-}p_1 v_3$	$\begin{bmatrix} \text{PROP}_1 & val_3 \end{bmatrix}$	t
$t\text{-}p_2 v_2$	$\begin{bmatrix} \text{PROP}_2 & val_2 \end{bmatrix}$	t
$t\text{-}p_2 v_4$	$\begin{bmatrix} \text{PROP}_2 & val_4 \end{bmatrix}$	t
t_1		$t\text{-}p_1 v_1$ & $t\text{-}p_2 v_2$
t_2		$t\text{-}p_1 v_1$ & $t\text{-}p_2 v_4$
t_3		$t\text{-}p_1 v_3$ & $t\text{-}p_2 v_2$
t_4		$t\text{-}p_1 v_3$ & $t\text{-}p_2 v_4$
val		
val_1		val
val_2		val
val_3		val
val_4		val

(56) a.



b. Ken-ga neru
Ken-NOM sleep
'Ken sleeps.'

c. Ken-ga hon-o yomu
Ken-NOM book-ACC read
'Ken reads a book.'

d. Ken-ga Naomi-ni kizuku
Ken-NOM Naomi-DAT notice
'Ken notices Naomi.'

e. Ken-ni eigo-ga wakaru
Ken-DAT English-NOM understand
'Ken understands English.'

f. Ken-ga hon-o tukue-ni oku
Ken-NOM book-ACC table-DAT put
'Ken puts a book on the table.'

As indicated above, JACY's verb hierarchy is organized according to transitivity and case-marking. The type *verb* is specified as being [HEAD *verb_head*]. The feature HEAD roughly corresponds to part-of-speech information. *intransitive* and *transitive* are given the constraints in (57a) and (57b), respectively.[5]

(57) a. $\textit{intransitive}\begin{bmatrix}\text{SUBJ} & \left\langle \textit{phrase}\,[\,\text{HEAD}|\text{CASE}\ \ \textit{nom}\,] \right\rangle \\ \text{COMPS} & \langle\ \rangle\end{bmatrix}$

b. $\textit{transitive}\begin{bmatrix}\text{SUBJ} & \langle\ \textit{phrase}\ \rangle \\ \text{COMPS} & \langle\ [\ ]\ \rangle\end{bmatrix}$

[5]There is no semantic feature in (57). But I will describe it in §3.3.4.

They state that both *intransitive* and *transitive* take a subject, the syntactic category of which is a phrase headed by a case particle (a phrase consisting of a noun and a case particle). The feature HEAD|CASE shows what case it bears. Only the subject of *intransitive* is specified as being in nominative case, since some *transitive* verbs like *wakaru* 'understand' can have a dative subject. On the other hand, while the value of *transitive* COMPS is [ ], the HPSG notation indicating that there should be something, i.e. not empty, *intransitive* gives an empty value for COMPS. The subtypes of *transitive* such as *tr-nom-acc* and *tr-nom-dat* are further specified their COMPS value as follows.

(58) a. $$\textit{tr-nom-acc}\begin{bmatrix} \text{SUBJ} & \left\langle {}^{\textit{pharse}}\left[\text{HEAD|CASE}\ \ \textit{nom}\right] \right\rangle \\ \text{COMPS} & \left\langle {}^{\textit{pharse}}\left[\text{HEAD|CASE}\ \ \textit{acc}\right] \right\rangle \end{bmatrix}$$

b. $$\textit{tr-nom-dat}\begin{bmatrix} \text{SUBJ} & \left\langle {}^{\textit{pharse}}\left[\text{HEAD|CASE}\ \ \textit{nom}\right] \right\rangle \\ \text{COMPS} & \left\langle {}^{\textit{pharse}}\left[\text{HEAD|CASE}\ \ \textit{dat}\right] \right\rangle \end{bmatrix}$$

c. $$\textit{tr-dat-nom}\begin{bmatrix} \text{SUBJ} & \left\langle {}^{\textit{pharse}}\left[\text{HEAD|CASE}\ \ \textit{dat}\right] \right\rangle \\ \text{COMPS} & \left\langle {}^{\textit{pharse}}\left[\text{HEAD|CASE}\ \ \textit{nom}\right] \right\rangle \end{bmatrix}$$

d. $$\textit{tr-nom-acc-dat}\begin{bmatrix} \text{SUBJ} & \left\langle {}^{\textit{phrase}}\left[\text{HEAD|CASE}\ \ \textit{nom}\right] \right\rangle \\ \text{COMPS} & \left\langle {}^{\textit{phrase}}\left[\text{HEAD|CASE}\ \ \textit{acc}\right], {}^{\textit{phrase}}\left[\text{HEAD|CASE}\ \ \textit{dat}\right] \right\rangle \end{bmatrix}$$

Here is an example of a *tr-nom-acc* verb, *yom* (read).

(59) $$\textit{tr-nom-acc}\begin{bmatrix} \text{PHON} & /\text{yom}/ \\ \text{HEAD} & \textit{verb\_head} \\ \text{SUBJ} & \left\langle {}^{\textit{phrase}}\left[\text{HEAD|CASE}\ \ \textit{nom}\right] \right\rangle \\ \text{COMPS} & \left\langle {}^{\textit{phrase}}\left[\text{HEAD|CASE}\ \ \textit{acc}\right] \right\rangle \end{bmatrix}$$

The constraint [HEAD *verb*] is inherited from the type *verb*. The feature PHON stands for phonological information of the verb.

There are also hierarchies for nouns, adjectives, adverbs, and so forth. Here I present simplified lexical entries for exposition.

(60) a. A simplified common noun standing for *hon* (book)

$$\textit{common-noun}\begin{bmatrix} \text{PHON} & /\text{hon}/ \\ \text{HEAD} & \textit{common-noun\_head} \\ \text{SUBJ} & \langle\ \rangle \\ \text{COMPS} & \langle\ \rangle \end{bmatrix}$$

b. A simplified proper name standing for *Ken*

$$
{}^{\textit{proper-noun}}\begin{bmatrix} \textsc{phon} & /\text{Ken}/ \\ \textsc{head} & \textit{proper-noun\_head} \\ \textsc{subj} & \langle\ \rangle \\ \textsc{comps} & \langle\ \rangle \end{bmatrix}
$$

(61) a. A simplified accusative case particle

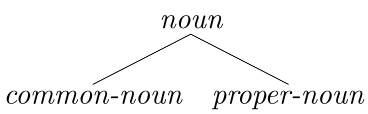
$$
{}^{\textit{case-p}}\begin{bmatrix} \textsc{phon} & /\text{o}/ \\ \textsc{head} & {}^{\textit{case-p\_head}}\begin{bmatrix} \textsc{case} & \textit{acc} \end{bmatrix} \\ \textsc{subj} & \langle\ \rangle \\ \textsc{comps} & \langle\ \textit{noun}\ \rangle \end{bmatrix}
$$

b. A simplified nominative case particle

$$
{}^{\textit{case-p}}\begin{bmatrix} \textsc{phon} & /\text{ga}/ \\ \textsc{head} & {}^{\textit{case-p\_head}}\begin{bmatrix} \textsc{case} & \textit{nom} \end{bmatrix} \\ \textsc{subj} & \langle\ \rangle \\ \textsc{comps} & \langle\ \textit{noun}\ \rangle \end{bmatrix}
$$

I am assuming that (some part of) the type hierarchy of nouns is organized as follows.

(62)



And notice that accusative and nominative case markers are specified as [HEAD [CASE *acc*]] and [HEAD [CASE *nom*]], respectively. We can also represent them as [HEAD|CASE *acc*] and [HEAD|CASE *nom*].

3.3.3 Syntax

Now let us see how those lexical items given above are assembled to form grammatical expressions. In particular, I present three rules and two principles that would constitute parts of "Universal Grammar" in HPSG.

A **head-complement rule** forms a phrase consisting of a head daughter and a non-head daughter that satisfies the restriction imposed by the head daughter's COMPS list.

(63) **Head-complement rule**

$$
{}^{\textit{phrase}}\begin{bmatrix} \textsc{comps} & \langle \boxed{A} \rangle \end{bmatrix} \quad \rightarrow \quad \boxed{1} \quad \mathbf{H}\begin{bmatrix} \textsc{comps} & \langle \boxed{1}, \boxed{A} \rangle \end{bmatrix}
$$

In (63), the left side of the arrow represents the phrase that is formed by the rule. $\boxed{1}$ is the non-head daughter subcategorized for by the head daughter (indicated by **H**) as one of its complements. The head daughter's complements other than $\boxed{1}$ are represented by $\boxed{A}$, which is sometimes used to indicate a list consisting of some elements, along with $\boxed{B}$, $\boxed{C}$, and so on. These complements represented by $\boxed{A}$ remain in the COMPS list of the phrase.[6] Below are examples of the phrases licensed by the rule.

(64) a. Ken-ga ··· *ga*, a *case-p*, is the head and *Ken*, a *proper-noun*, is the complement.
Ken-NOM

b. hon-o ··· *o*, a *case-p*, is the head and *hon*, a *common-noun*, is the complement.
book-ACC

c. hon-o yom ··· *yom*, a *tr-nom-acc*, is the head, and *hon-o*, an accusative *case-p*
book-ACC read
phrase is the complement.

ga, *o*, and *yom* subcategorize for only one complement. Thus, $\boxed{A}$ in (63) is empty.

A **head-subject rule** constructs a sentence from a subject phrase and a phrase headed by a predicate in a way that is very similar to the head-complement rule.[7]

(65) **Head-subject rule**

$$\textit{phrase}\begin{bmatrix}\text{SUBJ } \langle\ \rangle\end{bmatrix} \rightarrow \boxed{1} \quad \mathbf{H}\begin{bmatrix}\text{SUBJ } \langle \boxed{1} \rangle\end{bmatrix}$$

A sentence usually contains only one subject, namely $\boxed{1}$. Hence there is no element that corresponds to $\boxed{A}$ in (63) in the value for the SUBJ feature above. The head-subject rule licenses the following expressions, for instance.

[6]This is JACY's formulation of a head-complement rule, and there are a variety of ways in which the rule is formulated. In some HPSG literature, Sag and Wasow (1999) for instance, a head-complement rule looks like the following.

(i) Head-complement rule of Sag and Wasow (1999) for English:

$$\textit{phrase}\begin{bmatrix}\text{COMPS } \langle\ \rangle\end{bmatrix} \rightarrow \mathbf{H}\ \textit{word}\begin{bmatrix}\text{COMPS } \langle \boxed{A} \rangle\end{bmatrix} \quad \boxed{A}$$

That is, all complements that a head daughter subcategorizes for are licensed once, resulting in a flat structure (remember that $\boxed{A}$ is a list of elements).

[7]Again, this is JACY's formulation of a head-subject rule, and there are other formulations of it. Sag and Wasow (1999) proposes the corresponding rule as follows.

(i) Sag and Wasow's (1999) formulation of head-subject rule for English (their Head-Specifier Rule)

$$\textit{phrase}\begin{bmatrix}\text{SUBJ } \langle\ \rangle\end{bmatrix} \rightarrow \boxed{1} \quad \mathbf{H}\ \textit{phrase}\begin{bmatrix}\text{SUBJ} & \langle \boxed{1} \rangle \\ \text{COMPS} & \langle\ \rangle\end{bmatrix}$$

They require that a verb take a subject after the verb is combined with all complements. Consequently, a subject is always higher in the phrase structure tree than complements. But this is not the case for the formulation in (65).

(66) a. Ken-ga yom · · · *yom* is the head and *Ken-ga* is the subject.
Ken-NOM read

b. Ken-ga hon-o yom · · · *hon-o yom* is the head and *Ken-ga* is the subject.
Ken-NOM book-ACC read

A **head-specifier rule** determines phrases that are formed from a determiner and a noun, or a verb phrase and an auxiliary verb, among other things.

(67) a. kono hon · · · *hon* is the head and *kono* is the specifier.
this book

b. hon-o yon-de miru · · · *miru* is the head and *hon-o yon-de* is the specifier.
book-ACC read-TE see (try)
'try reading a book'

I am assuming that the auxiliary verb *miru* 'try' has the following (simplified) lexical entry.

(68)
$$ {}^{aux}\begin{bmatrix} \text{PHON} & /\text{mi}/ \\ \text{HEAD} & \textit{aux\_head} \\ \text{SUBJ} & \left\langle {}^{phrase}[\text{HEAD}|\text{CASE}\ \textit{nom}] \right\rangle \\ \text{COMPS} & \langle\ \rangle \\ \text{SPR} & \left\langle {}^{phrase}\begin{bmatrix} \text{HEAD} & \textit{verb\_head} \\ \text{SUBJ} & \langle [\ ] \rangle \\ \text{COMPS} & \langle\ \rangle \end{bmatrix} \right\rangle \end{bmatrix} $$

SPR stands for SPecifieR, and the specifier that *miru* subcategorizes for is a verb phrase that contains its complements (but not its subject). Thus, the phrase structure for (67b) is [[*hon-o yon-de*] *miru*].

Roughly speaking, within the JACY framework, phrases formed by the head-specifier rule contain a non-head daughter that cannot be scrambled away from a head daughter.

(69) a. Ken-ga kono hon-o yomu
Ken-NOM this book-ACC read
'Ken reads this book.'

b.*kono Ken-ga hon-o yomu
this Ken-NOM book-ACC read
'Ken reads this book.'

(70) a. Ken-ga hon-o yon-de miru
Ken-NOM book-ACC read-TE see (try)
'Ken tries reading a book.'

b.*hon-o yon-de Ken-ga miru
book-ACC read-TE Ken-NOM see (try)
'Ken tries reading a book.'

The formulation of the head-specifier rule is similar to the head-subject rule.

(71) **Head-specifier rule**

$$_{phrase}\left[\text{SPR}\ \langle\ \rangle\right] \rightarrow \boxed{1}\quad \mathbf{H}\left[\text{SPR}\ \langle\boxed{1}\rangle\right]$$

So far we have seen three syntactic rules. They are all regulated by general principles. Here I take up the **Head Feature Principle** (**HFP**) and the **Valence Principle**. The HFP requires that a head daughter's HEAD value be identical to that of the mother's. In this way, we can capture a universal regularity; a phrase whose head daughter is a verb must be a verb phrase, a phrase that a noun heads must be a noun phrase, and so on. This principle can be represented in the following way.

(72) **Head Feature Principle**

$$_{phrase}\left[\text{HEAD}\ \boxed{1}\right] \rightarrow \cdots\quad \mathbf{H}\left[\text{HEAD}\ \boxed{1}\right]$$

The Valence Principle is employed to express a tendency that, in general, the values for valence features (SUBJ, COMPS, and SPR) are the same between a mother and a head daughter, unless syntactic rules say otherwise. This principle can be given the formalization in (73).

(73) **Valence Principle**

$$_{phrase}\begin{bmatrix}\text{SUBJ} & \langle\ /\ \boxed{1}\rangle\\ \text{COMPS} & \langle\ /\ \boxed{A}\rangle\\ \text{SPR} & \langle\ /\ \boxed{2}\rangle\end{bmatrix} \rightarrow \cdots\quad \mathbf{H}\begin{bmatrix}\text{SUBJ} & \langle\ /\ \boxed{1}\rangle\\ \text{COMPS} & \langle\ /\ \boxed{A}\rangle\\ \text{SPR} & \langle\ /\ \boxed{2}\rangle\end{bmatrix}$$

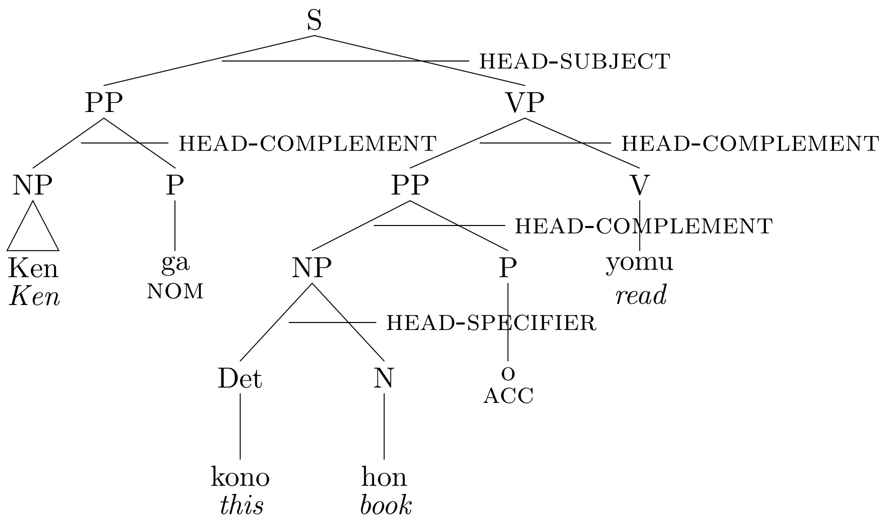
The '/' notation indicates that this is a **default constraint**. This means that the principle can be overridden by other rules or constraints. For example, if the head-subject rule is applied to a phrase and overrides the Valence Principle, the identity between the mother's SUBJ and the head daughter's SUBJ no longer holds.[8]

[8]In JACY, the tendency that the Valence Principle expresses is dealt with by other techniques that are a bit more redundant.

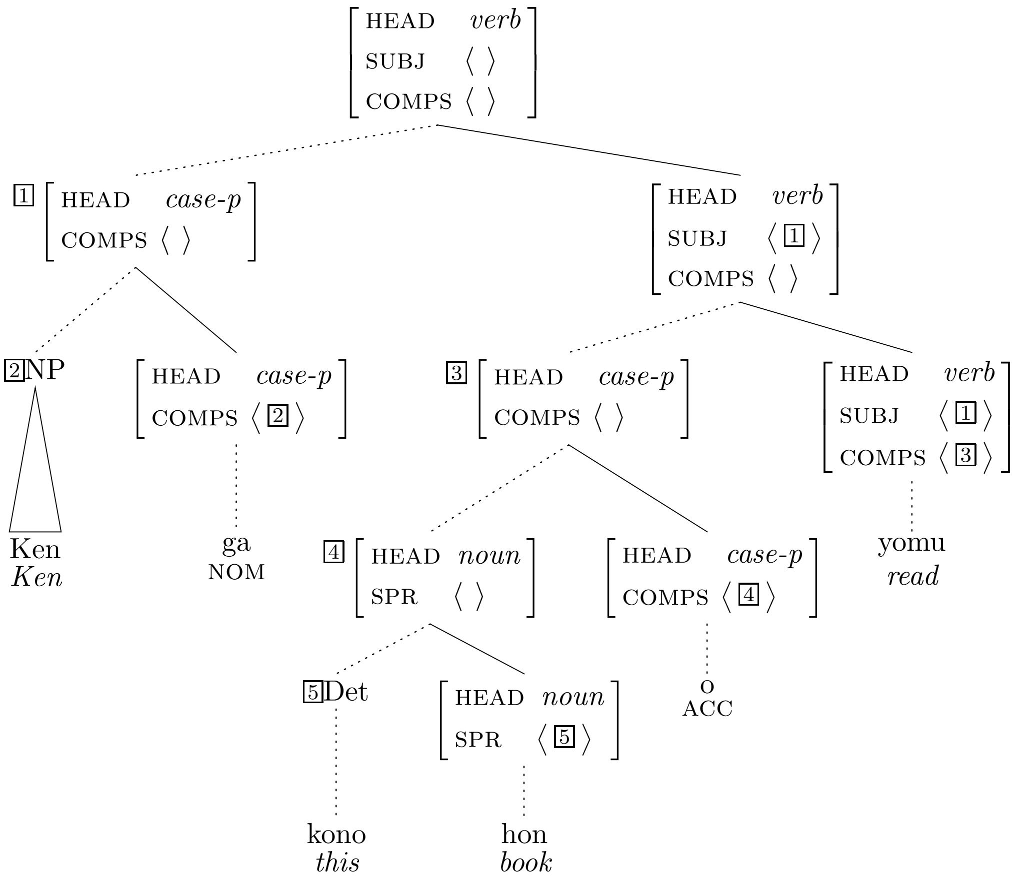
The rules and principles we have seen in this section are all built into the type hierarchy, as they are all types (i.e. linguistic notions) in the same way as lexical items.

(74) a. Ken-ga kono hon-o yomu
Ken-NOM this book-ACC read
'Ken reads this book.'

b.



c.



(74) shows an example of HPSG analysis; (74b) illustrates which rule applies to which node, and how the Head Feature Principle and the Valence Principle work is described in (74c). The two principles are working at the solid lines in (74c). Note that the Head Feature Principle assures the identity of HEAD feature value, *verb*, among the S, VP, and V nodes, for example. Also, the rule applications cancel the Valence Principle. For example, the application of HEAD-SUBJECT RULE cancels the default constraint that requires the identity of VALENCE feature values, SUBJ in this case, between the S and the VP. But the identity of COMPS feature values are not canceled by the HEAD-SUBJECT RULE, since the rule does not affect the COMPS feature.

3.3.4 Semantics

Linguistic expressions are constrained not only by syntactic rules or principles but also by principles involving phonology, morphology, semantics, and contexts. Among them, two semantic principles will be discussed here.

JACY adopts **Minimal Recursion Semantics** (**MRS**), which was designed to enable semantic composition using only unification of typed feature structures, producing for each phrase or sentence a description of the meaning representation sufficient to support logical inference. We can use various ways to show an MRS representation, and in this dissertation, I use one called the Indexed MRS. Figure 7 is the Indexed MRS of the sentence in (75).

(75) Ken-ga hon-o yomu
Ken-NOM book-ACC read
'Ken reads a book.'

`read(Ken, book)`

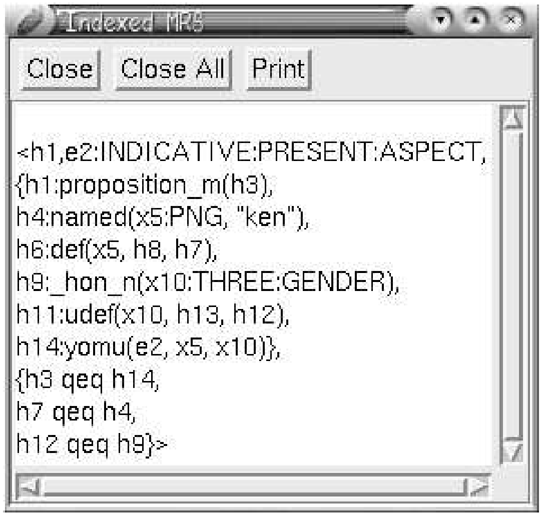


Figure 7: Example of Indexed MRS

We do not have to know everything about MRS to understand the dissertation, but we should notice that the reference index of the sentence is `e2` on the first line, which is shared by the seventh line, `h14:yomu(e2, x5, x10)`. That is, the line is the core semantic representation of the sentence. We also see that `x5` and `x10` refer to *Ken* and *hon* (book), respectively. As a result, we find that the meaning of the sentence roughly corresponds to `read(Ken, book)`.

In this section, however, I take up a simpler version of the theory presented in Sag and Wasow (1999), for ease of exposition, instead of a much more elaborated version such as Copestake et al. (1999). First, let us see how the meaning of each lexical item is represented. I previously gave several lexical items where the semantic features had been omitted. Below are examples with semantic features included.[9]

[9] In (76), features other than PHON are divided into two groups. One contains SYNtactic features: HEAD, SUBJ, and COMPS. The other contains SEMantic features that have just been introduced here: MODE, INDEX, and RESTR.

(76) a.

$$\textit{tr-nom-acc}\left[\begin{array}{ll}\text{PHON} & /\text{yom}/\\ \text{SYN} & \left[\begin{array}{ll}\text{HEAD} & \textit{verb\_head}\\ \text{SUBJ} & \left\langle \textit{phrase}\left[\begin{array}{ll}\text{SYN|HEAD|CASE} & \textit{nom}\\ \text{SEM|INDEX} & \boxed{2}\end{array}\right]\right\rangle\\ \text{COMPS} & \left\langle \textit{phrase}\left[\begin{array}{ll}\text{SYN|HEAD|CASE} & \textit{acc}\\ \text{SEM|INDEX} & \boxed{3}\end{array}\right]\right\rangle\end{array}\right]\\ \text{SEM} & \left[\begin{array}{ll}\text{MODE} & \textit{prop}\\ \text{INDEX} & \boxed{1}\ i\\ \text{RESTR} & \left\langle \textit{pred}\left[\begin{array}{ll}\text{RELATION} & \textit{read}\\ \text{SITUATION} & \boxed{1}\\ \text{READER} & \boxed{2}\ n\\ \text{READ} & \boxed{3}\ o\end{array}\right]\right\rangle\end{array}\right]\end{array}\right]$$

b.

$$\textit{common-noun}\left[\begin{array}{ll}\text{PHON} & /\text{hon}/\\ \text{SYN} & \left[\begin{array}{ll}\text{HEAD} & \textit{common-noun\_head}\\ \text{SUBJ} & \langle\ \rangle\\ \text{COMPS} & \langle\ \rangle\end{array}\right]\\ \text{SEM} & \left[\begin{array}{ll}\text{MODE} & \textit{ref}\\ \text{INDEX} & \boxed{4}\ j\\ \text{RESTR} & \left\langle \textit{pred}\left[\begin{array}{ll}\text{RELATION} & \textit{book}\\ \text{INSTANCE} & \boxed{4}\end{array}\right]\right\rangle\end{array}\right]\end{array}\right]$$

c.

$$\textit{proper-noun}\left[\begin{array}{ll}\text{PHON} & /\text{Ken}/\\ \text{SYN} & \left[\begin{array}{ll}\text{HEAD} & \textit{proper-noun\_head}\\ \text{SUBJ} & \langle\ \rangle\\ \text{COMPS} & \langle\ \rangle\end{array}\right]\\ \text{SEM} & \left[\begin{array}{ll}\text{MODE} & \textit{ref}\\ \text{INDEX} & \boxed{5}\ k\\ \text{RESTR} & \left\langle \textit{pred}\left[\begin{array}{ll}\text{RELATION} & \textit{name}\\ \text{NAME} & \textit{Ken}\\ \text{NAMED} & \boxed{5}\end{array}\right]\right\rangle\end{array}\right]\end{array}\right]$$

d.

$$\textit{case-p}\left[\begin{array}{ll}\text{PHON} & /\text{o}/\\ \text{SYN} & \left[\begin{array}{ll}\text{HEAD} & \textit{case-p\_head}\left[\text{CASE}\ \ \textit{acc}\right]\\ \text{SUBJ} & \langle\ \rangle\\ \text{COMPS} & \left\langle \textit{noun}\left[\text{SEM|INDEX}\ \ \boxed{6}\right]\right\rangle\end{array}\right]\\ \text{SEM} & \left[\begin{array}{ll}\text{MODE} & \textit{ref}\\ \text{INDEX} & \boxed{6}\ l\\ \text{RESTR} & \langle\ \rangle\end{array}\right]\end{array}\right]$$

e. $$^{case\text{-}p}\begin{bmatrix} \text{PHON} & /ga/ \\ \text{SYN} & \begin{bmatrix} \text{HEAD} & ^{case\text{-}p\_head}\begin{bmatrix}\text{CASE} & nom\end{bmatrix} \\ \text{SUBJ} & \langle\ \rangle \\ \text{COMPS} & \left\langle {}^{noun}\begin{bmatrix}\text{SEM}|\text{INDEX} & \boxed{7}\end{bmatrix} \right\rangle \end{bmatrix} \\ \text{SEM} & \begin{bmatrix} \text{MODE} & ref \\ \text{INDEX} & \boxed{7}\ m \\ \text{RESTR} & \langle\ \rangle \end{bmatrix} \end{bmatrix}$$

Semantic information is represented by the features MODE, INDEX, and RESTR. MODE indicates what kind of meaning a given expression takes on. For instance, a verb usually stands for a *prop*osition, while a noun *ref*ers to something. The value of INDEX is a variable corresponding to what a given expression means. RESTR tells us the condition that the meaning of an expression must satisfy.

According to (76a), the meaning of *yom* 'read' is regarded as a proposition i in which the reader n who is expressed by the subject phrase (as indicated by $\boxed{2}$) reads o that is realized as the complement phrase (as indicated by $\boxed{3}$). As illustrated in (76b) and (76c), *hon* (book) means something that is referred to by j in this case, and the meaning of *Ken*, a proper name, is someone whose name is Ken. Case particles such as *ga* and *o* do not mean anything by themselves, but they can refer to the same thing as their complement nouns. Note that those case particles' indices, l and m, are shared by their complement nouns by virtue of $\boxed{6}$ and $\boxed{7}$.

Now we are ready to discuss the semantic principles. As is well known, the meaning of a linguistic expression shows compositionality; components that constitute a given expression contribute the whole of their meanings to what the expression means. The **Semantic Compositionality Principle** (**SCP**) states this general characteristic. More specifically, this principle assembles the daughters' values of RESTR and puts them into the mother's RESTR. Take *hon-o yom* for example.

(77) The RESTR value of *hon-o yom*

$$\begin{bmatrix} \text{RESTR} & \left\langle {}^{pred}\begin{bmatrix} \text{RELATION} & book \\ \text{INSTANCE} & \boxed{1} \end{bmatrix}, {}^{pred}\begin{bmatrix} \text{RELATION} & read \\ \text{SITUATION} & i \\ \text{READER} & n \\ \text{READ} & \boxed{1}\ o \end{bmatrix} \right\rangle \end{bmatrix}$$

Notice that the indices for *hon* and what is read are the same as indicated by $\boxed{1}$. In this way, it is assured that in the reading situation, what is read is the thing which is expressed by the expression *hon*.

Next, the **Semantic Inheritance Principle** (**SIP**) requires that mother's MODE and INDEX values be identical to those of a head daughter. This amounts to saying that a phrase that is headed by a verb should represent a proposition, while a noun phrase should refer to the same thing as the head noun. In other words, this principle of semantics functions in a way similar to the HFP, one of the syntactic principles. Take *hon-o yom* as an example, again.

(78) The SEM value of *hon-o yom*

$$\begin{bmatrix} \text{MODE} & \textit{prop} \\ \text{INDEX} & \boxed{2} \\ \text{RESTR} & \left\langle {}^{\textit{pred}}\begin{bmatrix} \text{RELATION} & \textit{book} \\ \text{INSTANCE} & \boxed{1} \end{bmatrix}, {}^{\textit{pred}}\begin{bmatrix} \text{RELATION} & \textit{read} \\ \text{SITUATION} & \boxed{2}\ i \\ \text{READER} & n \\ \text{READ} & \boxed{1}\ o \end{bmatrix} \right\rangle \end{bmatrix}$$

In (78), [MODE *prop*] is inherited from the head daughter *yom* by virtue of the SIP, and so is the INDEX value (see (76a)). Consequently, *hon-o yom* stands for the proposition in which the reading situation described by the verb *yom* holds.

The semantic principles in this section are also built onto the type hierarchy in some way or another.

3.3.5 Word order

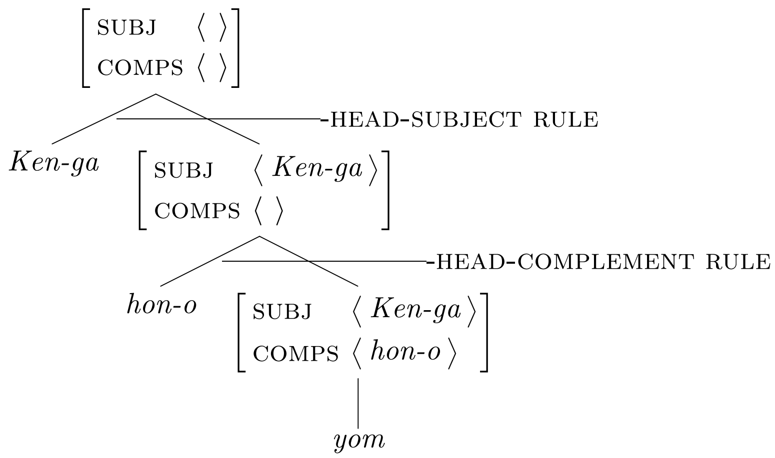
Japanese allows scrambling, so the word order of arguments is not fixed, unlike English.

(79) a. Ken-ga hon-o yomu
Ken-NOM book-ACC read
'Ken reads a book.'

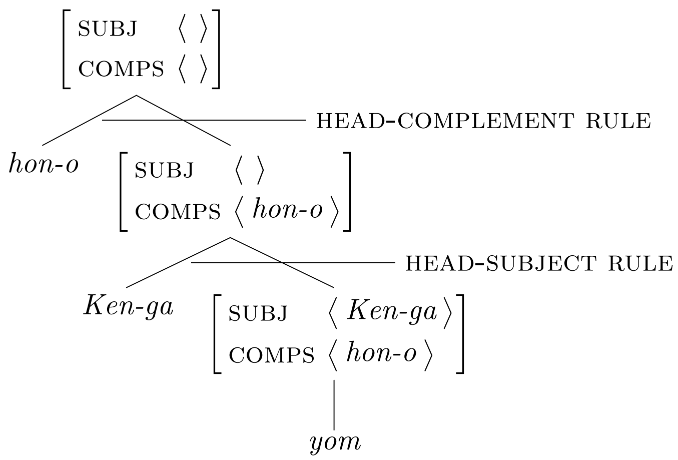
b. hon-o Ken-ga yomu
book-ACC Ken-NOM read
'Ken reads a book.'

There have been various proposals to account for scrambling including the movement approach (Saito, 1985; Hoji, 1985), the SUBCAT approach (Gunji, 1987), the SLASH approach (Gunji, 1987), and the linearization approach (Yatabe, 1996; Gunji, 1999). JACY deals with the free word order of Japanese by an approach similar to the SUBCAT approach. That is, in JACY, the subject and complements are allowed to be realized in any order. (80) shows JACY's phrase structures assigned to (79a) and (79b), respectively.

(80) a.



b.

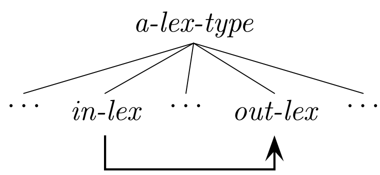


This is quite different from Sag and Wasow (1999), where a subject must be realized in a phrase structure position higher than complements.

3.3.6 Lexical rule

Roughly speaking, lexical rules map a lexical type (input) to another type (output). In addition, lexical rules are also formalized as types and built onto the type hierarchy in HPSG. Let us see an example. Assume that we have a lexical type hierarchy, *a-lex-type*, and a lexical type, *out-lex*, is always derived from another lexical type, *in-lex*. In that case, we posit a lexical rule like (81b).

(81) a.



b. $$\textit{out-lex}\begin{bmatrix}\textsc{feat1} & \boxed{1} \\ \textsc{feat2} & \textit{val2} \\ \textsc{input} & \textit{in-lex}\begin{bmatrix}\textsc{feat1} & \boxed{1}\end{bmatrix}\end{bmatrix}$$

Note that the lexical rule itself is assigned the type, *out-lex*, and that the *in-lex* type is the value for INPUT feature. Note also that the *out-lex* inherits the value for FEAT1 feature from the *in-lex* but the value for FEAT2 feature is changed.

Many linguists have proposed various formulations of a lexical rule. Among them, the proposals that are directly relevant to HPSG are Flickinger (1987), Pollard and Sag (1987, Chapter 8), Copestake (1992), Briscoe and Copestake (1999), and Meurers (2001), among others.

3.4 LKB: a grammar and lexicon development environment

In developing a large-scale computational grammar, it would be very difficult to investigate the grammar's consistency if the developer's introspection were the only tool available. Thus, in the grammar development projects that I mentioned on page 4, the developers use various tools that facilitate the development of grammar. The ParGram project (Butt et al., 2002), for instance, utilizes the Xerox Linguistic Environment (XLE, Kaplan and Newman (1997)) as a platform for grammar development.

In implementing my treatment of V_1-V_2 compound, I use the LKB system (Copestake, 2002), which is now extensively used in the DELPH-IN project (Oepen et al., 2002). The LKB system loads a grammar source code that is written in (a subset of) the $\mathcal{TDL}$ language, and parses or generates grammatical sentences according to the grammar. Furthermore, it provides a nice graphical interface that helps developers look into the internal structure of grammar. Figure 8 is the screenshot of LKB top after the JACY grammar is loaded. In Figure 9, the graphical interface that presents a portion of JACY's verb hierarchy is given.

3.5 Syntactic V_1-V_2 compounds

My analysis of syntactic V_1-V_2s follows Hashimoto (2003b, 2003c), which are in turn based on Kageyama (1993) and Matsumoto (1996). Therefore we first give an overview of the account proposed by Hashimoto (2003b, 2003c).

3.5.1 Hashimoto (2003b, 2003c)

I develop an HPSG account of syntactic V_1-V_2s that is more or less theory oriented, and I implement the analysis on my quite small scale computational grammar of Japanese (Hashimoto, 2003a).

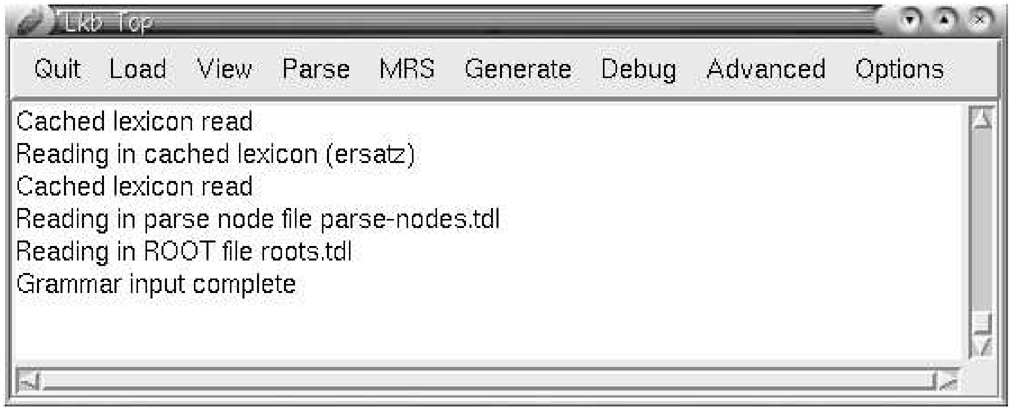


Figure 8: LKB

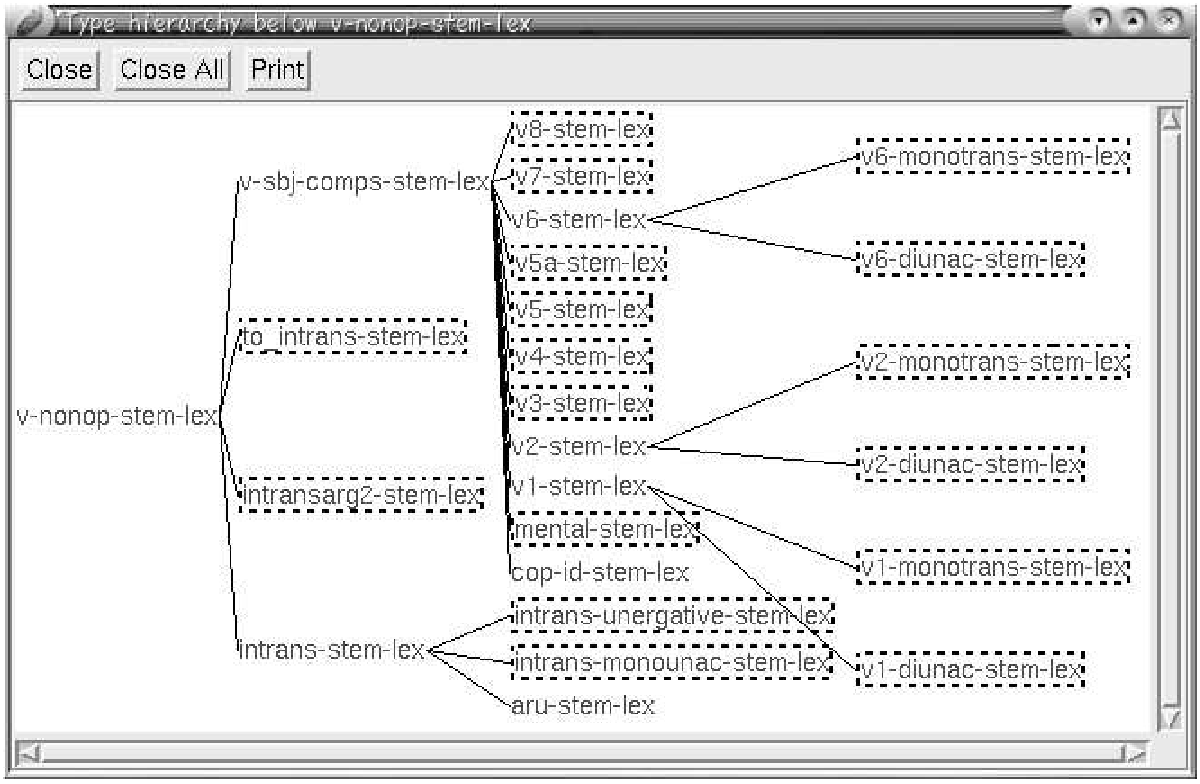


Figure 9: A part of JACY's verb hierarchy

I classify syntactic V_1-V_2s into three types, A type, B type, and C type, that correspond to Kageyama's (1993) three-way distinction as described in Table 2. With regard

Table 2: The classification of syntactic V_1-V_2s

Kageyama (1993)	Raising	Control	$\overline{\text{V}}$ type
Hashimoto (2003b, 2003c)	A	B	C

to Hashimoto's A and B types, they are analyzed in almost the same way as Kageyama, but are given the HPSG formalization of raising and control. Therefore let me begin with looking over the HPSG analysis of raising and control for the following discussion.

In Sag and Wasow (1999), a raising verb like *continue* has a lexical specification illustrated in (82a).

(82) a.
$$\textit{raising-verb}\begin{bmatrix} \text{PHON} & \text{/continue/} \\ \text{ARG-ST} & \left\langle \boxed{1}\ ,\ \text{CP}\begin{bmatrix} \text{SUBJ} & \langle \boxed{1}\ \text{NP} \rangle \\ \text{SEM} & [\text{INDEX}\ \boxed{2}] \end{bmatrix} \right\rangle \\ \text{SEM} & \begin{bmatrix} \text{RESTR} & \left\langle \textit{pred}\begin{bmatrix} \text{RELN} & \textit{continue} \\ \text{ARG} & \boxed{2} \end{bmatrix} \right\rangle \end{bmatrix} \end{bmatrix}$$

b. It$\boxed{1}$ continues $_{\text{CP}}$[($\boxed{1}$ NP) to rain].

ARG-ST stands for ARGument STructure where arguments are aligned in the order of obliqueness. Thus the first argument, $\boxed{1}$, is realized as the subject of a sentence that is headed by *continue* , and the CP that follows the $\boxed{1}$ is realized as the complement of *continue*. We should notice that the first argument is shared by the subject of the following CP as the two $\boxed{1}$s indicate. This amounts to "raising" of NP from the subject position of an embedded CP to the matrix subject position. And we should also note that *continue* assigns its thematic role only to (the INDEX of) the CP complement (as indicated by $\boxed{2}$) but not to the subject. In this way, the semantic behavior of raising is accounted for; in (82b), for instance, the expletive *it* can be the subject of the sentence because no semantic restriction is imposed by *continue*. Regarding a control verb such as *try*, Sag and Wasow (1999) posit the following features and constraints.

(83) a. *control-verb*

$$\begin{bmatrix} \text{PHON} & /\text{try}/ \\ \text{ARG-ST} & \left\langle \text{NP}_i \text{ , CP}\begin{bmatrix} \text{SUBJ} & \langle \text{NP}_i \rangle \\ \text{SEM} & [\text{INDEX } \boxed{2}] \end{bmatrix} \right\rangle \\ \text{SEM} & \begin{bmatrix} \text{RESTR} & \left\langle {}^{pred}\begin{bmatrix} \text{RELN} & try \\ \text{TRIER} & i \\ \text{ARG} & \boxed{2} \end{bmatrix} \right\rangle \end{bmatrix} \end{bmatrix}$$

b. Ken_i tries $_{\text{CP}}[$ (NP_i) to sleep].

c.*It_i tries $_{\text{CP}}[$ (NP_i) to rain].

NP_i is the abbreviation of the feature structure below.

(84)

$$\begin{bmatrix} \text{SYN} & \begin{bmatrix} \text{HEAD} & noun\_head \\ \text{SUBJ} & \langle\ \rangle \\ \text{COMPS} & \langle\ \rangle \end{bmatrix} \\ \text{SEM} & \begin{bmatrix} \text{MODE} & ref \\ \text{INDEX} & i \end{bmatrix} \end{bmatrix}$$

That is, NP_i is a noun phrase that makes reference to something represented by an index *i*. In contrast to a raising verb, a control verb puts semantic restrictions on not only the CP complement but also its subject. In the case of *try*, the verb restricts its subject to those that can be interpreted as TRIER. Hence, the example in (83c), where the subject is expletive, is ruled out.

Note that the constraint-based lexical analysis of raising and control proposed by Sag and Wasow (1999) does not involve any kind of movement.

Now we turn to the analysis by Hashimoto (2003b, 2003c). I assign each type of syntactic V_1-V_2 the following structures.[10]

(85) a. A type

Ken-ga hon-o yomi-kakeru
Ken-NOM book-ACC read-be.about.to
'Ken is about to read a book.'

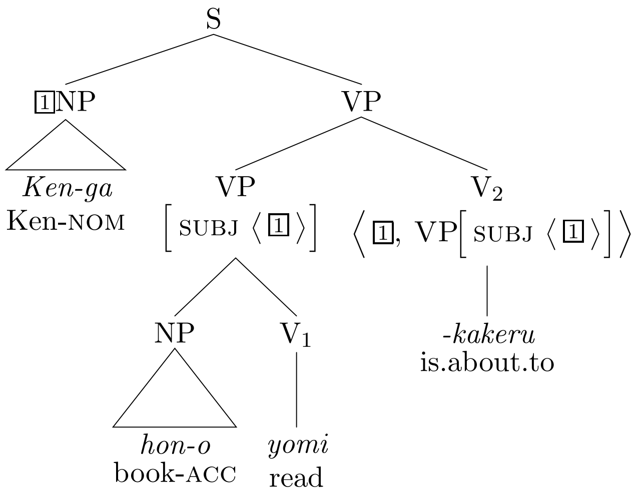
[10]Node labels posited in Hashimoto (2003b, 2003c) are those that are adopted by Gunji (1987).

S: A verbal projection that is saturated with both a subject and all complements.

VP: A verbal projection that is saturated with all complements, but not with a subject.

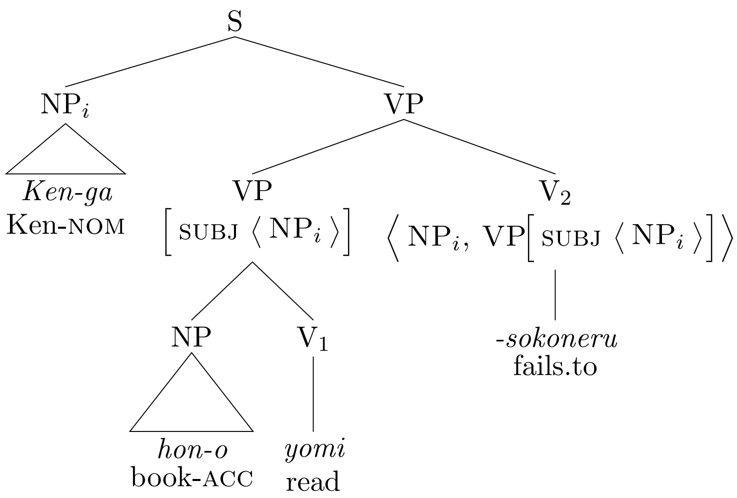
TVP: A verbal projection that is neither saturated with a subject nor complements.

And the lists under each V_2 represent the V_2's ARG-ST.



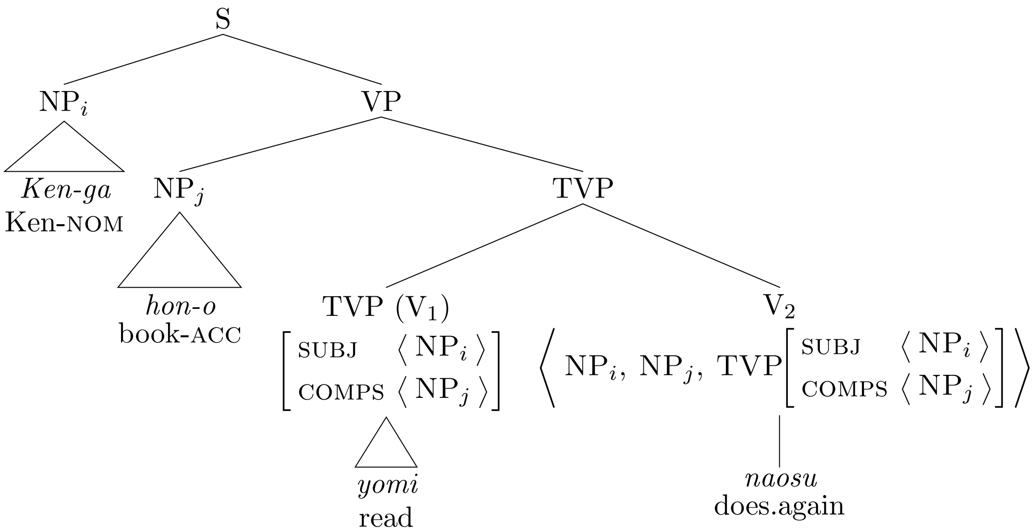
b. B type

Ken-ga hon-o yomi-sokoneru
Ken-NOM book-ACC read-fail
'Ken fails to read a book.'



c. C type

Ken-ga hon-o yomi-naosu
Ken-NOM book-ACC read-do.again
'Ken reads a book again.'



(85a) shows that the A type has a raising structure in accord with Kageyama (1993), in which the subject of the embedded VP is "raised" to the matrix subject position. Note that as a raising construction, the V_2, *kake* 'be.about.to', does not assign any thematic role. In (85b), the B type is analyzed as having a control structure, again following Kageyama (1993). That is, the matrix subject, *Ken-ga*, controls the subject of the embedded VP. As such, the V_2 of B type, *sokone* 'fail', semantically restricts its subject to ones that are agentive, or at least sentient. According to (85c), in the C type, there are two control relations holding between the matrix subject, *Ken-ga*, and the embedded TVP's subject, on the one hand, and between the matrix object, *hon-o*, and the embedded TVP's complement, on the other. Note that since the "object-to-object" control relation holds in (85c) in addition to the "subject-to-subject" control, not only the subject, *Ken-ga*, but also the object, *hon-o*, is assigned thematic restrictions by V_2.

I implemented this analysis in JACY. Figure 10, 11, and 12 show the JACY outputs of the sentences in (85a), (85b), and (85c).[11] Notice that my analysis can obtain correct semantic representations. The right side of Figure 10 (Indexed MRS) illustrates that the raising verb, *kakeru*, assigns a thematic role only to the proposition indicated by `h17` (=`h18`=`h14`), which corresponds to *Ken-ga hon-o yomu* 'Ken read a book'. The Indexed MRS representation of Figure 11 shows that the control verb, *sokoneru*, semantically restricts not only the proposition `h17` but also the subject, *Ken-ga*, indicated by `x5`. Similarly, from the Indexed MRS representation of Figure 12, we find that the C type V_2, *naosu*, puts semantic restrictions on the proposition `h17`, the subject, *Ken-ga*, indicated by `x5`, and the

[11] Node labels in these figures are different from those in Hashimoto (2003b, 2003c) because of the little differences of framework between JACY and Hashimoto (2003b, 2003c).

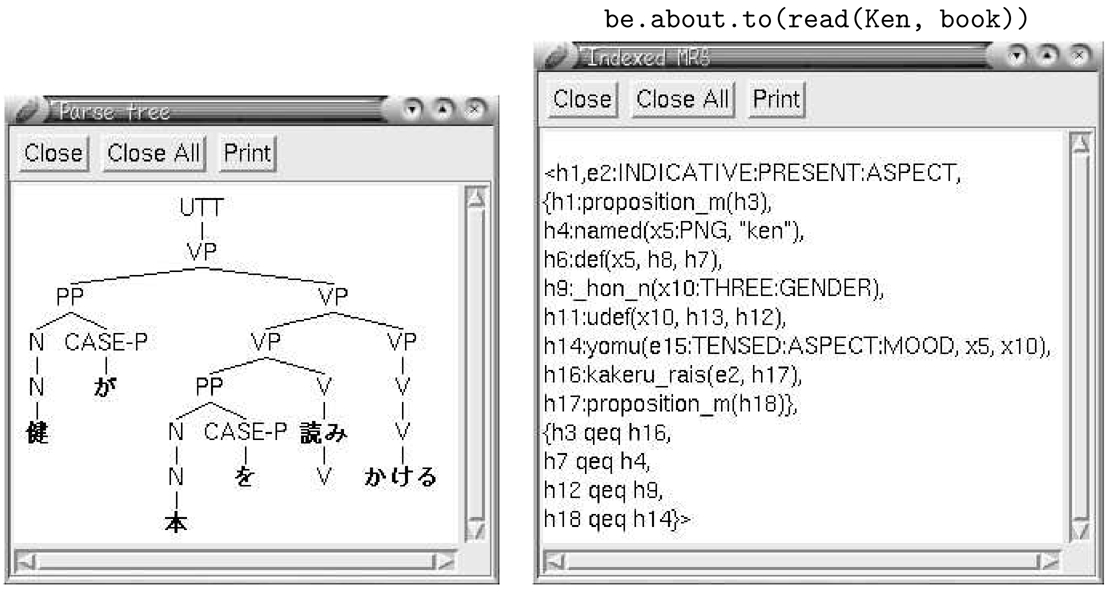


Figure 10: The JACY output of *Ken-ga hon-o yomi-kakeru*

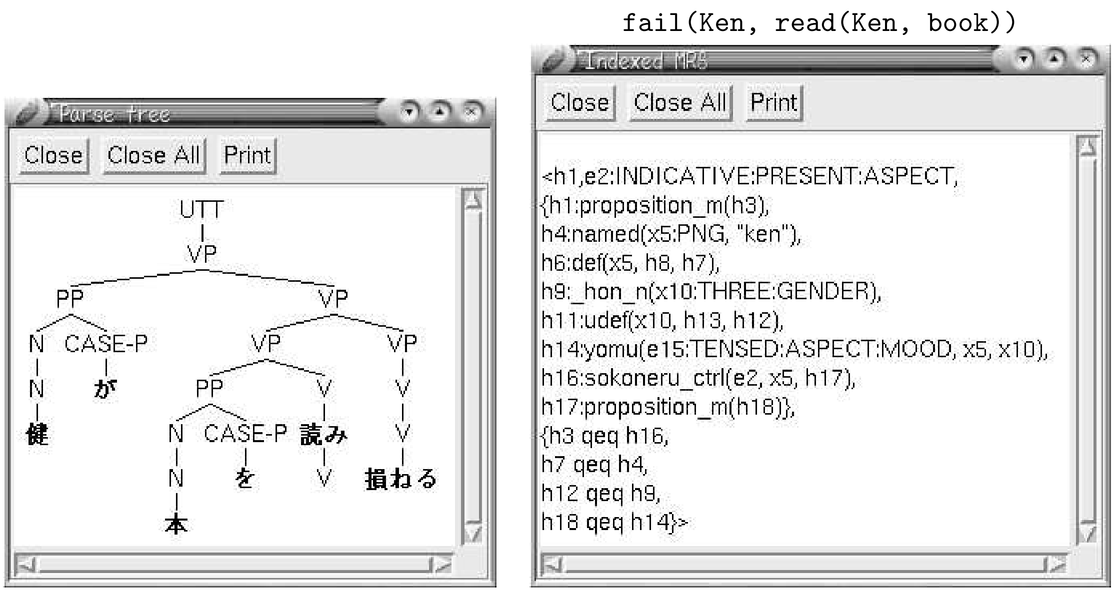


Figure 11: The JACY output of *Ken-ga hon-o yomi-sokoneru*

do.again(Ken, book, read(Ken, book))

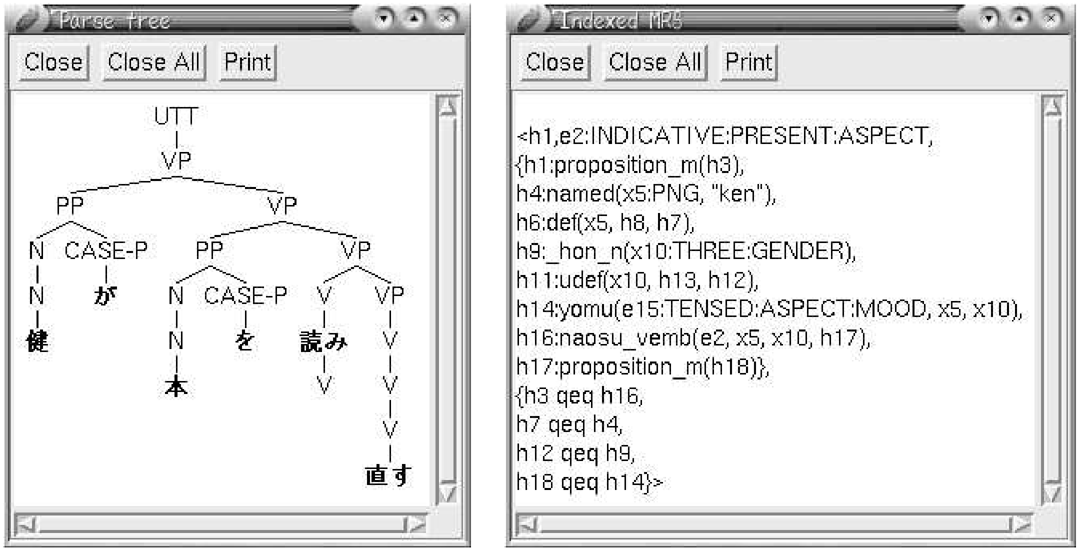


Figure 12: The JACY output of *Ken-ga hon-o yomi-naosu*

object, *hon-o*, that is marked up by `x10`.

These analyses explain the semantic differences among these three types observed in (25) and (26) on page 22, which are repeated in (86) and (87) respectively.

(86) ame-ga huku-o nurasi-{kakeru/*sokoneru/*naosu}
rain-NOM clothes-acc humidify-{be.about.to/fail/do.again}
'Rain {is about/fail/work over} to humidify clothes.'

(87) Ken-ga atama-o hiyasi-{kakeru/sokoneru/⁇naosu}
Ken-NOM heat-ACC cool-{be.about.to/fail/do.again}
'Ken {is about/fail/work over} to cool off.'

In (86), *kake*, as a raising verb, puts no semantic restriction on its subject, while, in (87), *naosu* restricts its object to something that can be done again, since the verb involves "object-to-object" control.

Now we can also account for the passivizability of V_1-V_2 that is described in (24) in §2.2.2. The examples are repeated as (88) below.

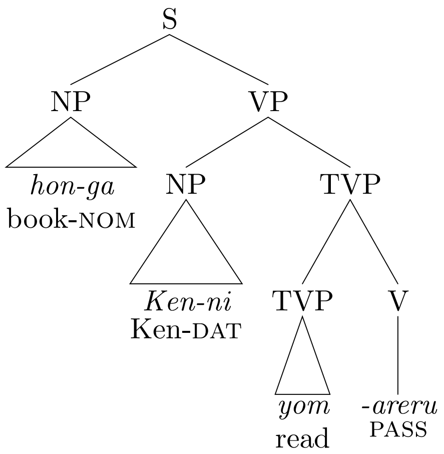
(88) a.*hon-ga Ken-ni [$_{V-V}$ yomi-{kake / sokone}]-rareru
book-NOM Ken-DAT [$_{V-V}$ read-{be.about.to / fail}]-PASS
'The book is {been about / failed} to read by Ken.'

b. hon-ga Ken-ni [$_{V-V}$ yomi-naos]-areru
book-NOM Ken-DAT [$_{V-V}$ read-do.again]-PASS
'The book is reread by Ken.'

The account of direct passivization in Hashimoto (2003b, 2003c) follows that of Gunji (1987), where the Japanese passive verb, *-rareru*, is analyzed as subcategorizing for a TVP that corresponds to a passivized stem verb (i.e. *yom* in (89)).

(89) Direct passivization

hon-ga Ken-ni yom-areru
book-NOM Ken-DAT read-PASS
'A book is read by Ken.'

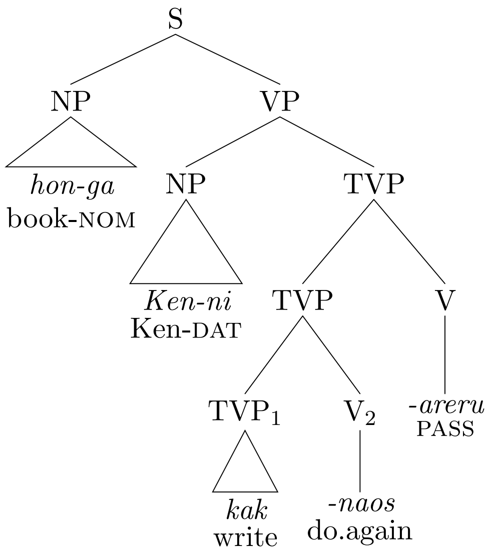


As we saw in (85), only the C type has a TVP node that contains both V_1 and V_2. It follows that only the V_1-V_2s of C type can be passivized.[12]

(90) The passivized C type V_1-V_2

hon-ga Ken-ni yomi-naos-areru
book-NOM Ken-DAT read-do.again-PASS
'A book is read by Ken again.'

[12] The node that is labeled "TVP(V_1)" in (85) is indicated by "TVP_1" in (90).



I give further evidence to support the analysis. First I take up the analysis of the verbal proform, *soo suru* 'do so', and next show how I account for phenomena involving the honorification expression, *o*-V-*ninaru*.

On page 16, it is argued that *soo suru* is a phrase that can appear in the V_1 position of syntactic V_1-V_2s, but lexical V_1-V_2s do not allow *soo suru* in their V_1 position. However, Kageyama (1993) and Matsumoto (1996) notice that the C type, although it is one kind of syntactic V_1-V_2s, prohibits *soo suru* from showing up as its V_1. Relevant examples are illustrated below.

(91) A type

a. Ken-ga hon-o yomi-kakeru
Ken-NOM book-ACC read-be.about.to
'Ken is about to read a book,'

b. sosite Naomi-mo soo si-kakeru
and Naomi-too so do-be.about.to
'and Naomi is about to do so, too.'

(92) B type

a. Ken-ga hon-o yomi-sokoneru
Ken-NOM book-ACC read-fail.to
'Ken fails to read a book.'

b. sosite Naomi-mo soo si-sokoneru
   and Naomi-too so do-fail.to
   'and Naomi fails to do so, too.'

(93) C type

a. Ken-ga hon-o yomi-naosu
   Ken-NOM book-ACC read-do.again
   'Ken reads a book, again.'

b.*sosite Naomi-mo soo si-naosu
   and Naomi-too so do-do.again
   'and Naomi does so again, too.

I assume that *soo suru* constitutes a VP, although there has been debate on the syntactic status of *soo suru* in the field of Japanese linguistics (Hinds, 1973; Inoue, 1976; Hasegawa, 1980; Gunji, 1987). Accordingly, the impossibility of the *soo suru* substitution only for the C type is explained, since, unlike the A and B types, the C type does not contain a VP node that includes only V_1, as (85) describes.

(94) A type

a. Ken-ga [$_{VP}$ hon-o yomi]-kakeru
   Ken-NOM [$_{VP}$ book-ACC read]-be.about.to
   'Ken is about to read a book,'

b. sosite Naomi-mo [$_{VP}$ soo si]-kakeru
   and Naomi-too [$_{VP}$ so do]-be.about.to
   'and Naomi is about to do so, too.'

(95) B type

a. Ken-ga [$_{VP}$ hon-o yomi]-sokoneru
   Ken-NOM [$_{VP}$ book-ACC read]-fail.to
   'Ken fails to read a book,'

b. sosite Naomi-mo [$_{VP}$ soo si]-sokoneru
   and Naomi-too [$_{VP}$ so do]-fail.to
   'and Naomi fails to do so, too.'

(96) C type

a. Ken-ga [VP hon-o [TVP [TVP yomi]-naosu]]
Ken-NOM [VP book-ACC [TVP [TVP read]-do.again]]
'Ken reads a book, again.'

b.*sosite Naomi-mo [VP soo si]-naosu
and Naomi-too [VP so do]-do.again
'and Naomi does so again, too.

(96a) shows that in the C type, the VP node consists of the object and V_1-V_2. That is, *hon-o yomi-naosu* constitutes a VP. Hence the *soo suru* substitution like (97) is possible.

(97) C type

a. Ken-ga [VP hon-o [TVP [TVP yomi]-naosu]]
Ken-NOM [VP book-ACC [TVP [TVP read]-do.again]]
'Ken reads a book, again.'

b. sosite Naomi-mo [VP soo suru]
and Naomi-too [VP so do]
'and Naomi does so, too.

Like *soo suru*, the honorific verb phrase, *o*-V-*ni naru*, is also prohibited from appearing in the V_1 position of C type, though we saw on page 17 Kageyama's (1993) observation that syntactic V_1-V_2s allow the honorific verbal expression to appear as V_1. (98) illustrates this.

(98) a. A type
sensei-ga hon-o o-yomi-ni nari-kakeru
teacher-NOM book-ACC HON-read-DAT become-be.about.to
'A teacher is about to read a book.'

b. B type
sensei-ga hon-o o-yomi-ni nari-sokoneru
teacher-NOM book-ACC HON-read-DAT become-fail.to
'A teacher fails to read a book.'

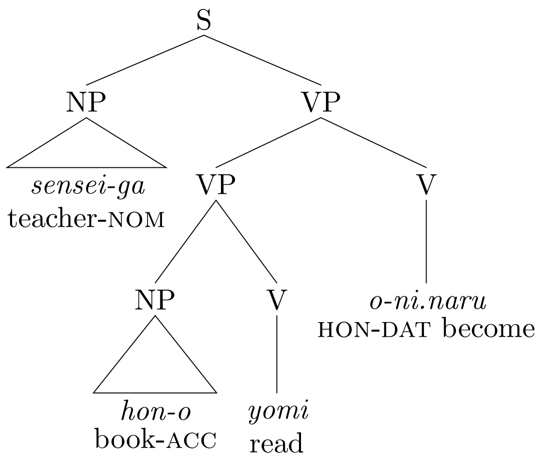
c. C type
⁇sensei-ga hon-o o-yomi-ni nari-naosu
teacher-NOM book-ACC HON-read-DAT become-do.again
'A teacher reads a book, again.'

My analysis of honorification follows Gunji (1999), where the honorific verb phrase, *o-ni.naru* (HON-DAT become), constitutes one verb that subcategorizes for VP to form VP. On the other hand, phonologically speaking, the *o-* in the honorific verb phrase is prefixed to its adjacent verb (i.e. *yomi* in (99)).

(99) Honorification

a. sensei-ga hon-o o-yomi-ni naru
teacher-NOM book-ACC HON-read-DAT become
'A teacher reads a book.'

b.

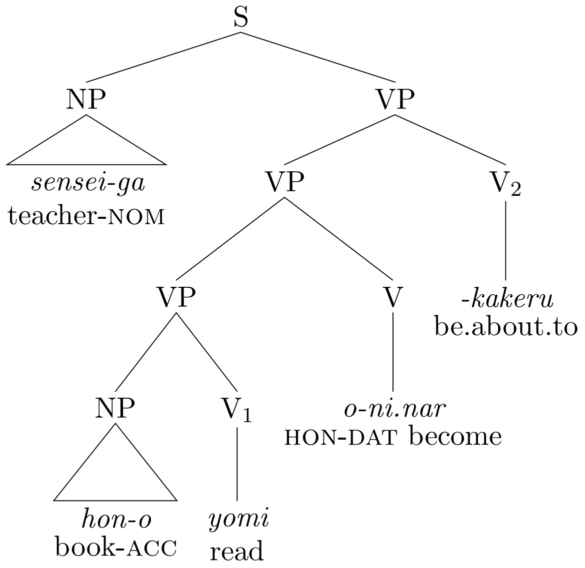


Given this analysis, the sentences in (98) should have the following syntactic structures.

(100) A type

a. sensei-ga hon-o o-yomi-ni nari-kakeru
teacher-NOM book-ACC HON-read-DAT become-be.about.to
'A teacher is about to read a book.'

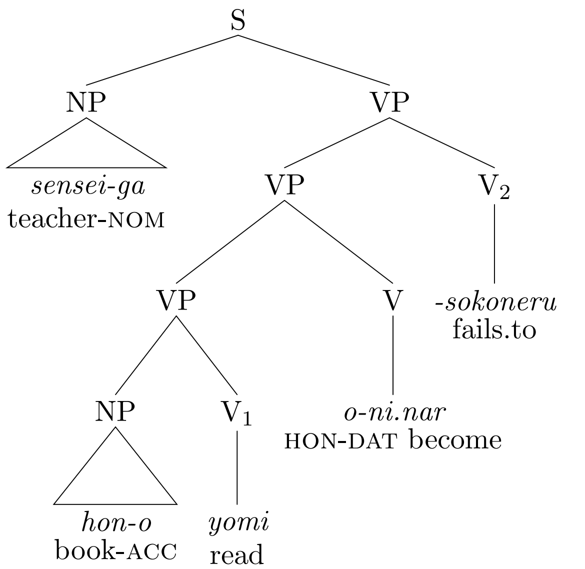
b.



(101) B type

a. sensei-ga hon-o o-yomi-ni nari-sokoneru
teacher-NOM book-ACC HON-read-DAT become-fail
'A teacher fails to read a book.'

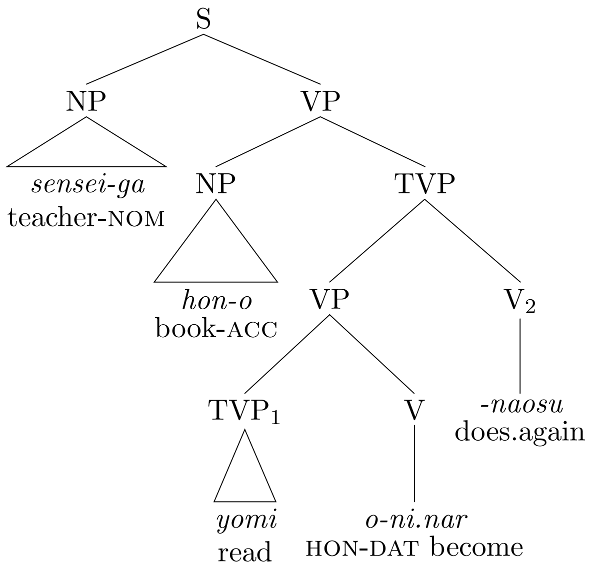
b.



(102) C type

a. sensei-ga hon-o o-yomi-ni nari-naosu
teacher-NOM book-ACC HON-read-DAT become-do.again
'A teacher reads a book again.'

b.*



We should notice that in (102b) the honorific verb, *o-ni.nar*, takes TVP, resulting in violation of its subcategorization requirement. As well, the C type V_2, *naosu*, takes the VP that is headed by the honorific verb, although C type V_2s must take TVP as we saw previously.

As is clear from the discussion so far, Hashimoto (2003b, 2003c) posits neither empty category nor movement, but is successful in accounting for the relevant data. Furthermore, my analysis is simple and explicitly formalized within the $\mathcal{TDL}$ format. Thus, I conclude that the analysis of syntactic V_1-V_2s posited by Hashimoto (2003b, 2003c) is theoretically reliable, and yet meets the conditions of **Simplicity of Design** and **Efficiency of Computation**.

3.5.2 Adapting Hashimoto (2003b, 2003c) to JACY: word order problem

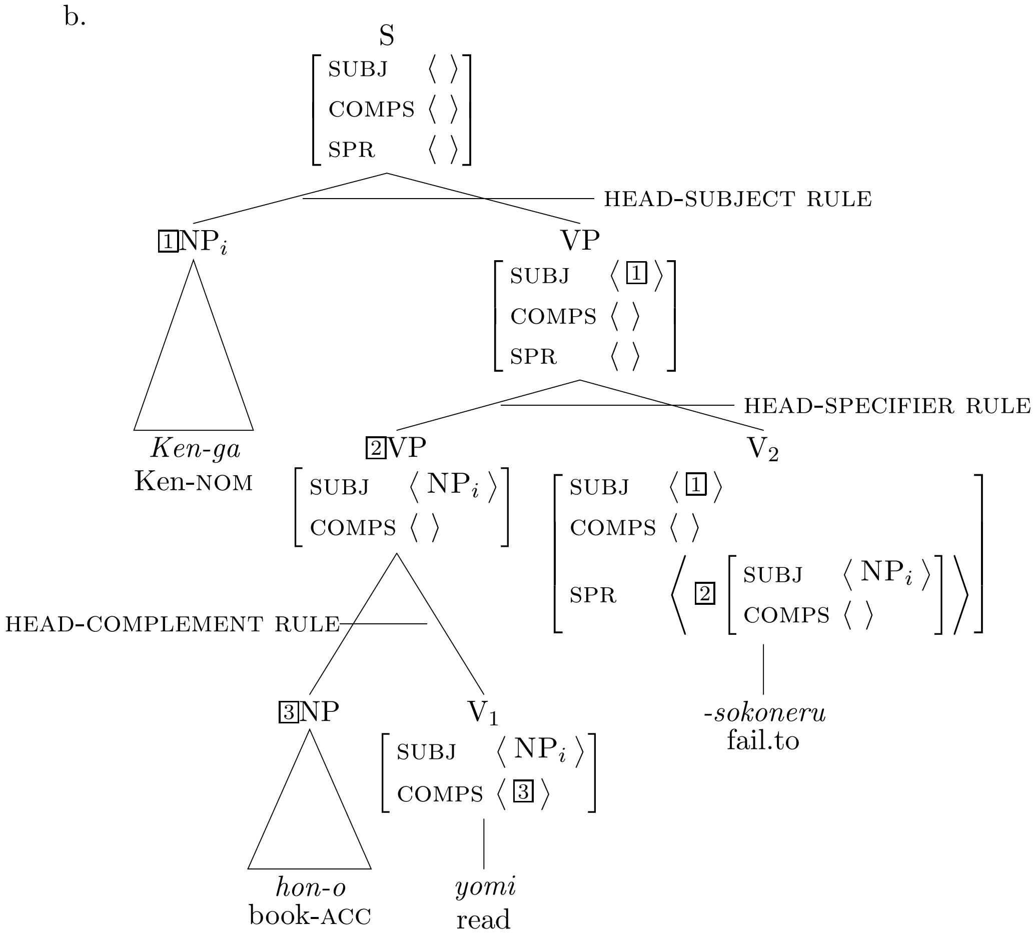
With some modifications,[13] I adapt the analysis proposed by Hashimoto (2003b) to JACY, the formalism of which my implementation is based on.

However, JACY's treatment of word order that we saw in §3.3.5 on page 53 presents a problem concerning scrambling of arguments from an embedded VP. Look at (103) that exemplifies the problem.

(103) a. Ken-ga [$_{VP}$ hon-o yomi]-sokoneru
Ken-NOM [$_{VP}$ book-ACC read]-fail.to
'Ken fails to read a book.'

[13] Among those modifications are the formulations of raising and control. Following Sag and Wasow (1999), Hashimoto (2003b, 2003c) utilizes ARG-ST as a locus where the constraints about raising and control are stated. I will introduce ARG-ST to JACY, but, in my current implementation, the valence features, SUBJ and SPR, are used to state the constraints of raising and control. This arrangement can make the implementation simpler.

b.



Note first that *-sokoneru* heads a control construction where a matrix subject controls the subject of an embedded VP, and the VP is subcategorized for by V_2 as its SPR value, since the embedded VP cannot be scrambled away from V_2.[14] The problem is that we incorrectly predict from the structure of (103b) that the object, *hon-o*, cannot be scrambled, since the V_2, *sokoneru*, is supposed to subcategorize for a verb phrase that is saturated with a complement, namely, VP. As a result, the object, *hon-o*, must stay inside the VP. But, in fact, the scrambled counterpart of (103a) shown in (104) is completely grammatical.

(104) hon-o Ken-ga [$_{VP}$ ϕ yomi]-sokoneru
book-ACC Ken-NOM [$_{VP}$ ϕ read]-fail.to
'Ken fails to read a book.'

Somehow we need to be able to allow an object within an embedded VP to be scrambled, but at the same time we have to stay within JACY's framework. My solution to the

[14] Remember the analysis of the auxiliary verb, *miru* 'try', discussed on page 46, which also subcategorizes for a verb as its SPR value. This formulation is different from Hashimoto (2003b, 2003c) where an embedded VP is realized as V_2's COMPS value rather than as SPR.

problem is to posit an **Argument Attraction** approach (Hinrichs & Nakazawa, 1994; Gunji, 1999). Roughly speaking, VP embedding verbs that undergo Argument Attraction treat a complement inside the embedded VP as their own complement. Below is the illustration of Argument Attraction.

(105) a. *-sokoneru*

$$\begin{bmatrix} \textsc{subj} & \langle \text{NP}_i \rangle \\ \textsc{comps} & \langle\ \rangle \\ \textsc{spr} & \left\langle \text{VP}\begin{bmatrix} \textsc{subj} & \langle \text{NP}_i \rangle \\ \textsc{comps} & \langle\ \rangle \end{bmatrix} \right\rangle \end{bmatrix}$$

b. Argument Attracted *-sokoneru*

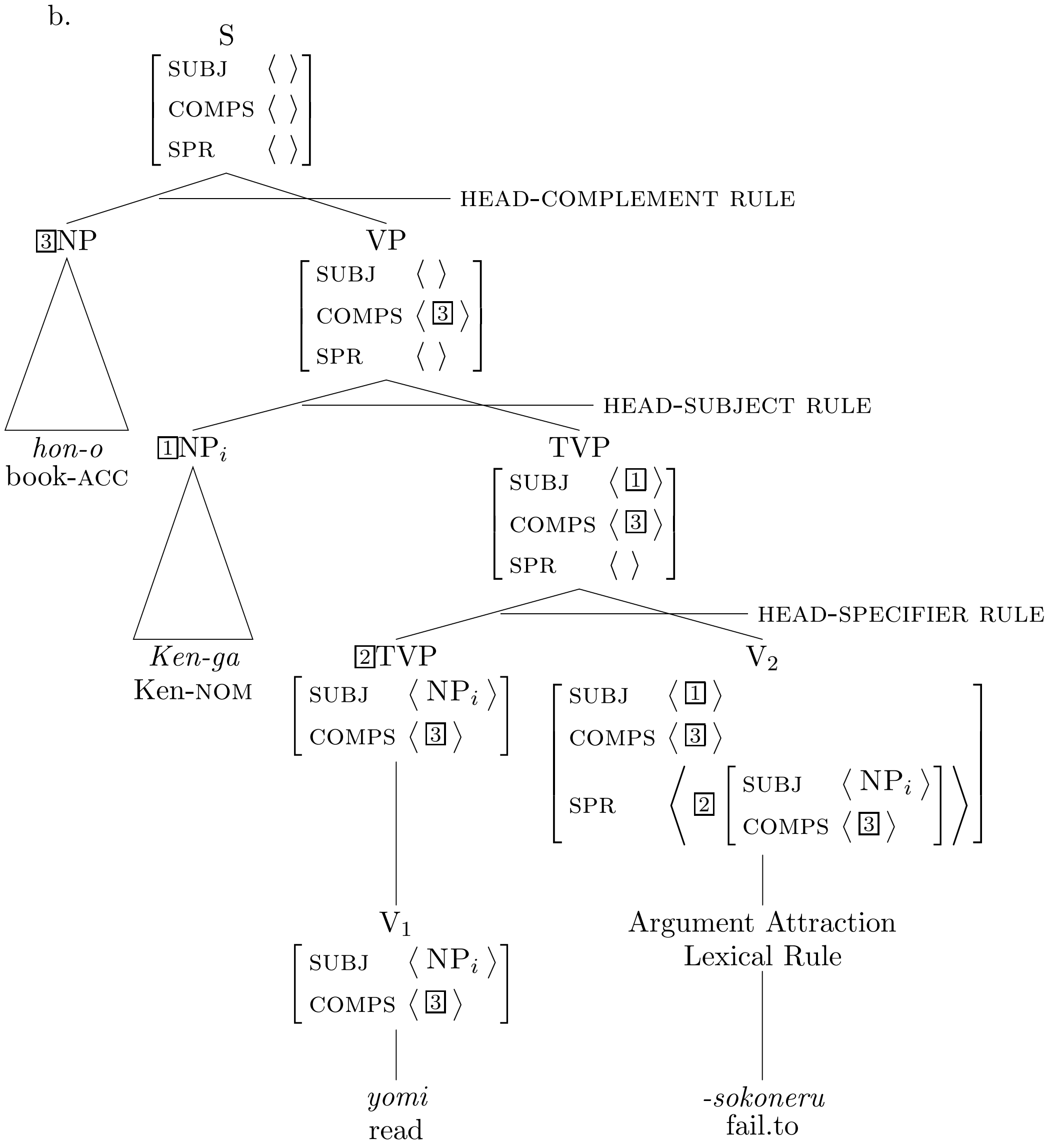
$$\begin{bmatrix} \textsc{subj} & \langle \text{NP}_i \rangle \\ \textsc{comps} & \langle \boxed{1}\text{NP} \rangle \\ \textsc{spr} & \left\langle \text{TVP}\begin{bmatrix} \textsc{subj} & \langle \text{NP}_i \rangle \\ \textsc{comps} & \langle \boxed{1}\text{NP} \rangle \end{bmatrix} \right\rangle \end{bmatrix}$$

By means of Argument Attraction, the complement of the embedded VP becomes shared with *-sokoneru* as indicated by $\boxed{1}$, and consequently *-sokoneru* is changed to subcategorize for TVP rather than VP. A VP embedding verb and its Argument Attracted counterpart are related to each other by the **Argument Attraction Lexical Rule**.

Now the grammar can deal with scrambling of a complement inside VP. Let us see a relevant example in (106).

(106) a. hon-o Ken-ga yomi-sokoneru
book-ACC Ken-NOM read-fail.to
'Ken fails to read a book.'

b.



By virtue of Argument Attraction, the object, *hon-o*, is scrambled to the sentence-initial position, and still is construed as the complement of the V_1, *yomi*, as 3 indicates in the example. Figure 13 shows the JACY output of (106). Note that the Argument Attraction does not affect the meaning of a sentence. Compare the Indexed MRS of Figure 13 with that of Figure 11.[15]

However, Argument Attraction as formulated above would bring us problems involving spurious ambiguities. Without additional stipulations, Argument Attraction would also apply to a verb that heads a non-scrambled sentence, and we would get two parses for the sentence.

[15]Notice that, in Figure 11, *Ken* and *hon* (book) are represented by `x5` and `x10`, respectively, which is the reverse of Figure 13. This is because of the different word order.

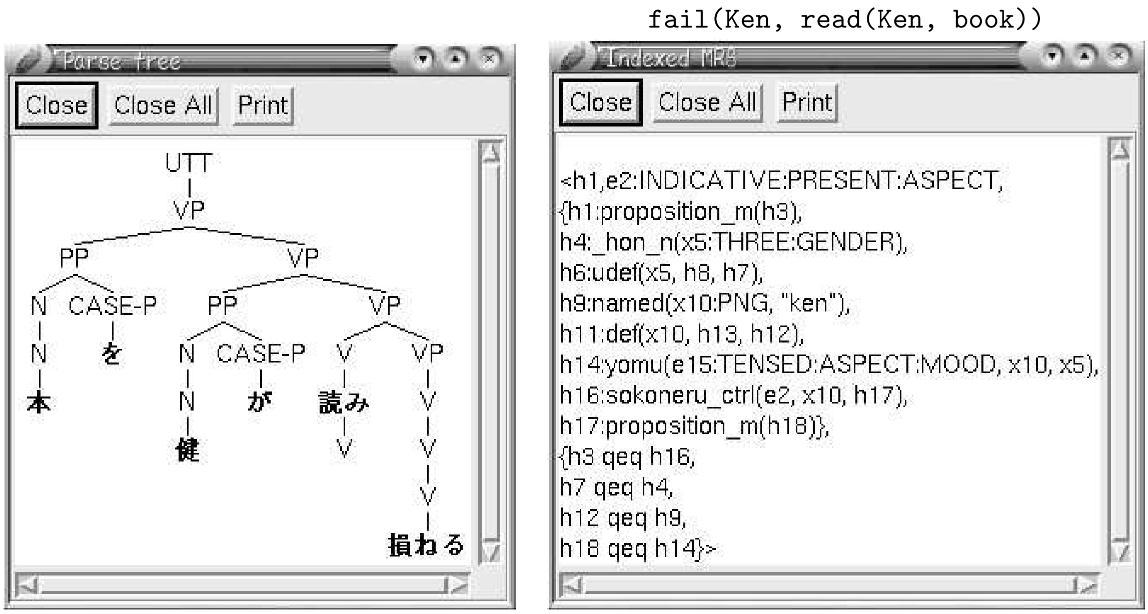


Figure 13: The JACY output of *hon-o Ken-ga yomi-sokoneru*

(107) a. without Argument Attraction

Ken-ga [$_{\text{VP}}$ hon-o yomi]-sokoneru
Ken-NOM [$_{\text{VP}}$ book-ACC read]-fail.to
'Ken fails to read a book.'

b. with Argument Attraction

Ken-ga hon-o [$_{\text{TVP}}$ yomi]-sokoneru
Ken-NOM book-ACC [$_{\text{TVP}}$ read]-fail.to
'Ken fails to read a book.'

We only need (107a), and the application of Argument Attraction should be restricted to a verb that heads a scrambled sentence like (106a). Thus I stipulate that the complement of Argument Attracted verb must be combined with the head daughter after the verb is saturated with the subject. In this way, unnecessary parses like (107b) are ruled out.

We would also face a problem in dealing with sentences that contain a phonologically empty pronoun, *pro*.

(108) a. without Argument Attraction

Ken-ga [$_{\mathrm{VP}}$ ϕ yomi]-sokoneru
Ken-NOM [$_{\mathrm{VP}}$ *pro* read]-fail.to
'Ken fails to read (a book).'

b. with Argument Attraction

ϕ Ken-ga [$_{\mathrm{TVP}}$ yomi]-sokoneru
pro Ken-NOM [$_{\mathrm{TVP}}$ read]-fail.to
'Ken fails to read (a book).'

(109) a. without Argument Attraction

ϕ [$_{\mathrm{VP}}$ hon-o yomi]-sokoneru
pro [$_{\mathrm{VP}}$ book-ACC read]-fail.to
'(Ken) fails to read a book.'

b. with Argument Attraction

hon-o ϕ [$_{\mathrm{TVP}}$ yomi]-sokoneru
book-ACC *pro* [$_{\mathrm{TVP}}$ read]-fail.to
'(Ken) fails to read a book.'

Note that the two sentences in (108) are the same in their phonological form, and so are the two sentences in (109). But, in (108) and (109), two parses would be given the phonologically same sentences, and we do not want (108b) nor (109b) as before. To get around this problem, I also stipulate that arguments of Argument Attracted verbs must be obligatory. Clearly, this stipulation enables us to rule out (108b) and (109b).[16]

Surprisingly, this stipulation also prevents another annoying consequence of Argument Attraction, which involves the passivizabilities of syntactic V_1-V_2s. As we saw in §3.5.1 on page 55, V_1-V_2s that are headed by a VP embedding V_2, namely the A type and the B

[16] However, as Timothy Baldwin pointed out to me, this stipulation brings us a problem concerning topicalization.

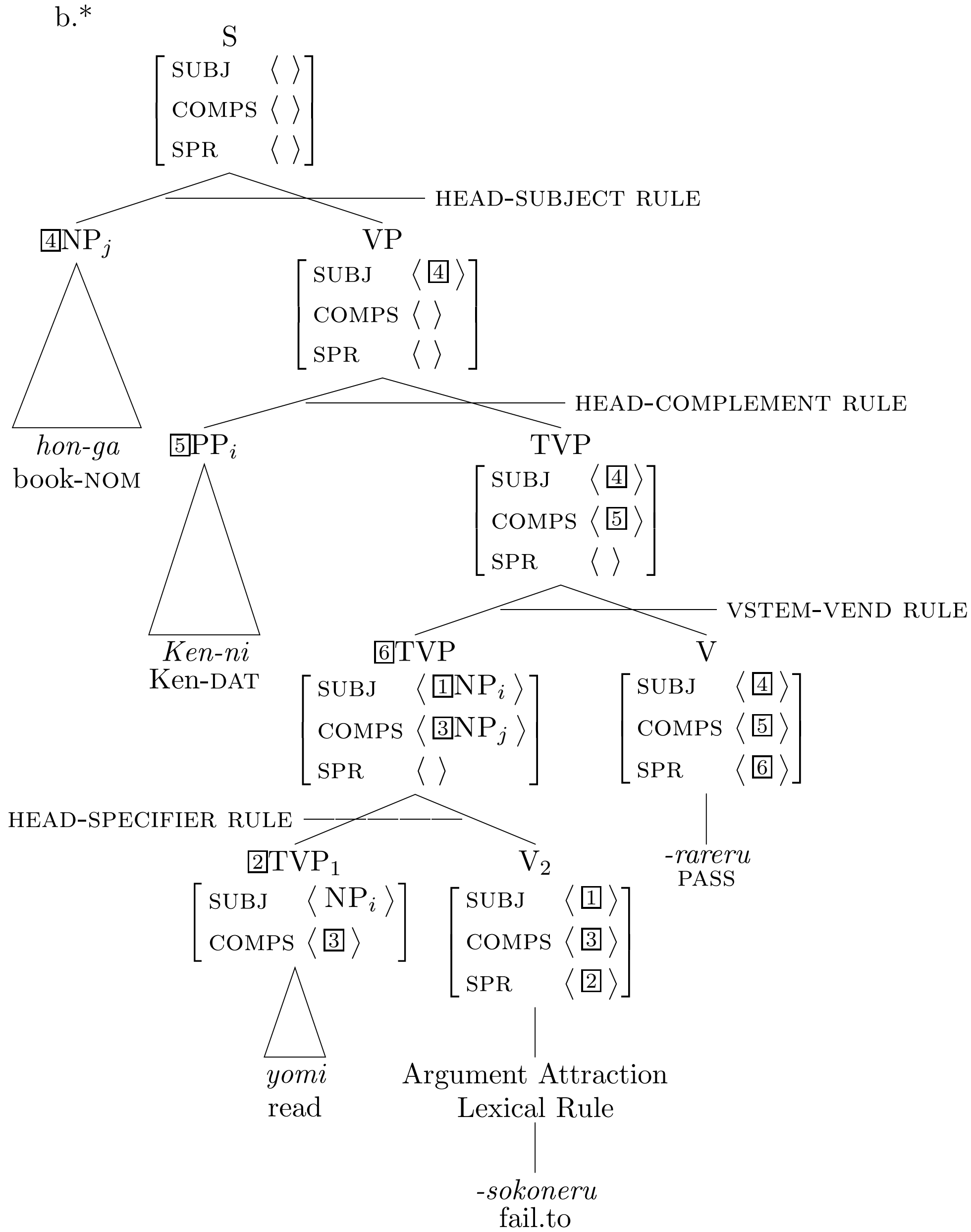
(i) ?hon-o Ken-wa [$_{\mathrm{TVP}}$ yomi]-sokoneru
book-ACC Ken-TOP [$_{\mathrm{TVP}}$ read]-fail.to
'Ken fails to read a book.'

In this example, the topic phrase, *Ken-wa*, must be dependent on *-sokoneru* or *yomi-sokoneru* as a whole. Therefore, the object, *hon-o*, must be Argument Attracted as the arrow indicates. In spite of the similarity between (i) and (106), (i) is predicted to be ungrammatical, since the obligatory subject argument is absent because of the topic phrase. Even though this example is not very natural, it might be wrong to rule this out.

type, are not passivizable unlike the C type. Nevertheless, Argument Attraction gets those unpassivizable V_1-V_2s to have the structure that looks like that of C type's. In other words, Argument Attracted V_1-V_2s of A type or B type have a TVP node that includes both V_1 and V_2, and hence you might think that it is incorrectly predicted that Argument Attracted V_1-V_2s of these types can be passivized. However, the prediction is not made, since the constraint that arguments of an Argument Attracted verb are obligatory is violated if an Argument Attracted verb is passivized. (110b) shows what would happen if the (impossible) passivization occurred. It is important to note that the Argument Attracted verb *sokoneru* is not saturated with its obligatory arguments, $\boxed{1}$NP$_i$ and $\boxed{3}$NP$_j$. Hence this is ruled out.

(110) a.*hon-ga Ken-ni yomi-sokone-rare
book-NOM Ken-DAT read-fail.to-PASS
'A book is failed to read by Ken.'

b.*



I have shown here that Argument Attraction helps us deal with scrambling from an embedded VP. Siegel and Bender (2002) does not adopt Argument Attraction or some other machinery to deal with the problematic scrambling, and they suffer from the VP embedding structure as I will mention in §4.3.1. Accordingly, they try to avoid the structure as much as possible.[17] Indeed, they do not posit a VP embedding even for Japanese causatives which most linguists have analyzed with VP embedding structures. Ohtani et al. (2000) also avoid VP embeddings so as not to be troubled with the word order problem. However, in order to

[17]Their analysis of aspectual constructions involves VP embedding structures.

capture the characteristics of syntactic V_1-V_2s, VP embeddings are indispensable not only to theoretical grammars but also to computational grammars.

At first glance, Argument Attraction might seem *ad hoc*. However, many linguists have posited Argument Attraction for many languages; Hinrichs and Nakazawa (1994) analyzed auxiliaries in German with Argument Attraction, and Gunji (1999) posited it to account for Japanese causatives. More specifically, Gunji (1999) observes that if the argument within an embedded VP is scrambled to the sentence initial position, the ARG-ST of the verb that embeds the VP seems to be affected, although he claims that scrambling does not affect the sentence's syntactic-semantic structure. In addition, Gunji (p.c., 2003) points out that not only scrambling but also such dislocation as relativization and topicalization of the argument within an embedded VP do affect the ARG-ST of the VP embedding verb. Following are examples of the scrambling (111b), the relativization (111c), and the topicalization (111d).

(111) a. Ken_i-ga Naomi-ni [$_{\text{VP}}$ kare_i-o mi]-saseru
Ken_i-NOM Naomi-DAT [$_{\text{VP}}$ he_i-ACC see]-CAUS
'Ken makes Naomi see himself.'

b.*kare_i-o Ken_i-ga Naomi-ni [$_{\text{VP}}$ ϕ_i mi]-saseru
he_i-ACC Ken_i-NOM Naomi-DAT [$_{\text{VP}}$ (ACC) see]-CAUS
'Ken makes Naomi see himself.'

c.*[Ken_i-ga Naomi-ni [$_{\text{VP}}$ ϕ_i mi]-saseru] kare_i
[Ken_i-NOM Naomi-DAT [$_{\text{VP}}$ (ACC) see]-CAUS] he_i
'Ken, who makes Naomi see himself.'

d.*kare_i-wa Ken_i-ga Naomi-ni [$_{\text{VP}}$ ϕ_i mi]-saseru
he_i-TOP Ken_i-NOM Naomi-DAT [$_{\text{VP}}$ (ACC) see]-CAUS
'Himself, Ken makes Naomi see.'

In each case, the affected argument is *kare* (he) within the embedded VP, *kare-o mi* (he-ACC see). The changes in the interpretations of *kare* above indicate that the ARG-ST of the VP embedding verb *saseru* (CAUS) changes in the following way.

(112) a. the ARG-ST of *saseru* in (111a)

$$\left[\text{ARG-ST}\ \left\langle\ \textit{Ken, Naomi,}\ \text{VP}\left[\text{ARG-ST}\ \langle\ \textit{Naomi, Kare}\ \rangle\right]\right\rangle\right]$$

b. the ARG-ST of *saseru* in (111b)–(111d)

$$\left[\text{ARG-ST}\ \left\langle\ \textit{Ken, Naomi, kare,}\ \text{VP}\left[\text{ARG-ST}\ \langle\ \textit{Naomi, Kare}\ \rangle\right]\right\rangle\right]$$

To deal with this, Gunji (1999) proposed Argument Attraction, which attracts the argument within the VP complement to the ARG-ST of the VP embedding verb. Likewise, we notice the same contrast in the A and B types.

(113) a. Ken$_i$-ga [$_{VP}$ kare$_i$-o mi]-kakeru/sokoneru
Ken$_i$-NOM [$_{VP}$ he$_i$-ACC see]-be.about.to/fail
'Ken {is about to / fails to} see himself.'

b.*kare$_i$-o Ken$_i$-ga [$_{VP}$ ϕ_i mi]-kakeru/sokoneru
he$_i$-ACC Ken$_i$-NOM [$_{VP}$ (ACC) see]-be.about.to/fail
'Ken {is about to / fails to} see himself.'

c.*[Ken$_i$-ga [$_{VP}$ ϕ_i mi]-kakeru/sokoneru] kare$_i$
[Ken$_i$-NOM [$_{VP}$ (ACC) see]-be.about.to/fail] he$_i$
'Ken, who {is about to / fails to} see himself.'

d.*kare$_i$-wa Ken$_i$-ga [$_{VP}$ ϕ_i mi]-kakeru/sokoneru
he$_i$-TOP Ken$_i$-NOM [$_{VP}$ (ACC) see]-be.about.to/fail
'Himself, Ken {is about to / fails to} see.'

Evidently, dealing with this also requires us to posit machinery similar to (112). Thus, I would say that Argument Attraction is far from *ad hoc* in the field of theoretical linguistics.[18]

There are various alternatives to the Argument Attraction approach: the movement approach (Saito, 1985; Hoji, 1985), the SLASH approach (Gunji, 1987) and the Linearization approach (Yatabe, 1996; Gunji, 1999), among others. Although these alternatives have well-motivated theoretical bases, they lack efficient processing techniques for them.[19] As discussed in §2.2.4, Kageyama's (1993) framework suffers from being insufficient in terms of mathematical preciseness and hence the unavailability of efficient processing systems. The approaches of Gunji (1987), Yatabe (1996), and Gunji (1999) are mathematically precise and explicit enough to be executed on computers. Nevertheless, it is well-known that these approaches to word order are too powerful to be executed in an efficient and fast way. In contrast, my approach is not only mathematically precise and explicit but also restricted

[18] My version of Argument Attraction is different from that of Gunji (1999) in that, in my version, the constraint on Argument Attraction is stated in the VALENCE feature rather than ARG-ST. However, my version is compatible with Gunji's, and it would not be difficult to rearrange the Argument Attraction in my implementation so that it can account for the binding facts in (111) and (113).

[19] Daniels and Meurers (2004) discusses the processing technique of Linearization theory. It remains to be seen how efficient it really is.

enough to be processed very efficiently, as it can be described in the $\mathcal{TDL}$ language. Therefore, I conclude that the Argument Attraction approach discussed here is more suitable for NLP than the other approaches in terms of **Efficiency of Computation**.

3.6 Lexical V_1-V_2 compounds

My analysis of lexical V_1-V_2s roughly follows Matsumoto (1996). However, as an engineering oriented analysis, it is a bit coarser, but it is computationally simpler and more explicit than Matsumoto's analysis. Besides, it has reasonably broad coverage comparable to his.

Below is my classification of lexical V_1-V_2s.

(114) a. **Right headed V_1-V_2s**

b. **Argument mixing V_1-V_2s**

c. **V_1-V_2s with semantically deverbalized V_1**

d. **V_1-V_2s with semantically deverbalized V_2**

e. **Non-compositional V_1-V_2s**

Right headed and Argument mixing V_1-V_2s cover most of Matsumoto's (1996) *pair*, *cause*, *manner*, and *means* compounds. However, unlike Matsumoto, I underspecify the semantic relation holding between the two component verbs of the Right headed V_1-V_2, since recognizing such relations depends heavily on world knowledge and current computational systems are not able to be equipped with such a recognition ability, as discussed in §2.3.2.[20] Non-compositional V_1-V_2s are those that should be treated as single words.

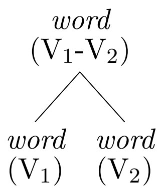
In spite of the classification parallel to Matsumoto's, the machinery I have posited to account for lexical V_1-V_2s is similar to that of Kageyama (1993). To be more precise, my analysis is given a simple ARG-ST based formulation, rather than a fine-grained semantics based one. This way, we can expect my analysis to meet the condition of **Simplicity of Design**. Furthermore, since the analysis does not require us to know in advance which semantic relation holds between V_1 and V_2, **Availability of Input** is also observed. Besides, the underspecification of the semantic relation can also be justified in light of **Importance of Phenomena**, since our judgments of such a semantic relation are occasionally not stable.

Lexical compounds should be created by lexical machinery, but the LKB system, by which JACY has been developed, does not support any lexical rules for compounding. Therefore, I made extensive use of the machinery for phrase structure rules to develop

[20] I restrict the semantic relation between V_1 and V_2 of the Argument mixing V_1-V_2 to the *manner* relation. See §3.6.3.

Pseudo Lexical Rules.[21] The Pseudo Lexical Rules are compounding rules that form a single word from two component words, as described in (115).

(115) **Pseudo Lexical Rule**



This helps us explain the difference between syntactic V_1-V_2s and lexical V_1-V_2s. As discussed in §2.2.1, lexical, as opposed to syntactic, V_1-V_2s do not allow phrases to appear in the V_1 position. Below are the data illustrating this point.

(116) a. *kaki-hazimeru* (write-begin) 'begin to write'

a′. *soo si-hazimeru* (so do-begin) 'begin to do so'

b. *tabe-owaru* (eat-finish) 'finish eating'

b′. *soo si-owaru* (so do-finish) 'finish doing so'

(117) a. *uti-korosu* (shoot-kill) 'shoot to death'

a′.**soo si-korosu* (so do-kill) '?'

b. *nomi-aruku* (drink-walk) 'tour bars'

b′.**soo si-aruku* (so do-walk) '?'

(118) a. *kaki-hazimeru* (write-begin) 'begin to write'

a′. *o-kaki-ni nari-hazimeru* (HON-write-DAT become-begin) 'begin to write'

b. *tabe-owaru* (eat-finish) 'finish eating'

b′. *o-tabe-ni nari-owaru* (HON-eat-DAT become-finish) 'finish eating'

(119) a. *uti-korosu* (shoot-kill) 'shoot to death'

a′.**o-uti-ni nari-korosu* (HON-shoot-DAT become-kill) 'shoot to death'

b. *nomi-aruku* (drink-walk) 'tour bars'

[21] I will present the grammar source code of the Pseudo Lexical Rules in Appendix A.

b′.**o-nomi-ni nari-aruku* (HON-drink-DAT become-walk) 'tour bars'

(120) and (122) are the examples of syntactic V_1-V_2s, while (121) and (123) describe lexical ones. As I described previously, *soo suru* (so do) and *o*-V-*ni naru* (HON-V-DAT become) are phrases, and as shown above, they cannot appear in the V_1 position of lexical V_1-V_2s. Within my analysis, lexical V_1-V_2s are formed by means of Pseudo Lexical Rules, which correctly permit only a word to be a V_1 as in (115).

3.6.1 Introducing ARG-ST to JACY

I analyze lexical V_1-V_2s by referring to ARG-STs of their component verbs. Other computational grammars, such as Ohtani et al. (2000) and Masuichi and Ôkuma (2003), have not adopted ARG-ST (or a-structure) to avoid complexity. On the other hand, I think a relatively simple formulation of it has the advantage of being able to account for many linguistic phenomena and yet keeping a grammar reasonably concise.[22]

I formulate ARG-ST following Imaizumi and Gunji (2000).[23]

(120) $${}^{\textit{arg-st}}\begin{bmatrix} \textsc{ext} & \textit{index} \\ \textsc{int1} & \textit{index} \\ \textsc{int2} & \textit{index} \end{bmatrix}$$

ARG-ST consists of one EXTernal argument and two INTernal arguments. Roughly speaking, an external argument is mapped onto a subject, while internal arguments correspond to complements.[24] The type hierarchy of *arg-st* is organized according to which argument(s) it has (see Table 3 and (121)).

[22]JACY already contains the apparatus named ARG-S, which is similar to ARG-ST that I present here, but ARG-S differs from ARG-ST in that the former does not distinguish an external argument from internal ones. As we will see soon, the distinction plays a crucial role in my analysis. Thus I decided that I newly introduce ARG-ST to JACY, but leave ARG-S as it is in the grammar for safety.

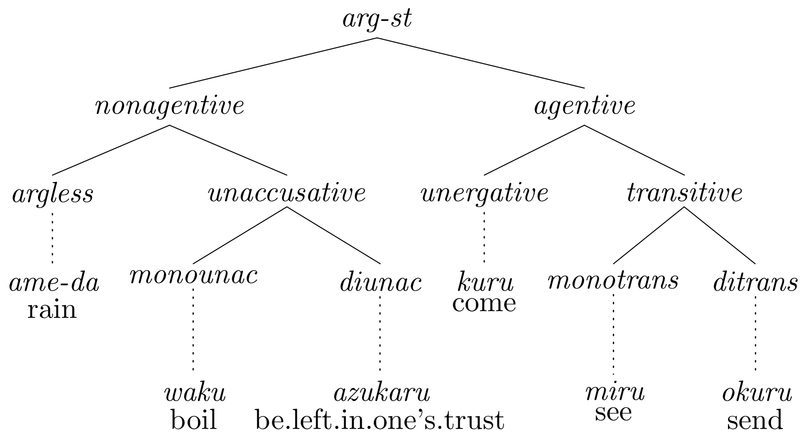
[23] The values of EXT, INT1, and INT2 are the type *index*, rather than *synsem*. Otherwise we would face problems concerning case-marking. That is, when combining a nominative-accusative transitive and a nominative-dative transitive to form a compound in which the accusative argument and the dative argument are to be co-indexed, the two internal arguments cannot structure-share due to the case-marking difference (accusative vs dative) if we adopt *synsem* as the values.

[24]Imaizumi and Gunji (2000) propose the mapping constraint called the **Argument Realization Principle** (**ARP**) as follows.

(i) a. $$\begin{bmatrix} \textsc{subj} & \langle \boxed{1} \rangle \\ \textsc{comps} & \langle \boxed{2} \rangle \\ \textsc{arg-st} & \begin{bmatrix} \textsc{ext} & \langle\ \rangle \\ \textsc{int1} & \langle \boxed{1} \rangle \\ \textsc{int2} & \langle \boxed{2} \rangle \end{bmatrix} \end{bmatrix}$$ b. $$\begin{bmatrix} \textsc{subj} & \langle \boxed{1} \rangle \\ \textsc{comps} & \langle \boxed{2}, \boxed{3} \rangle \\ \textsc{arg-st} & \begin{bmatrix} \textsc{ext} & \langle \boxed{1} \rangle \\ \textsc{int1} & \langle \boxed{2} \rangle \\ \textsc{int2} & \langle \boxed{3} \rangle \end{bmatrix} \end{bmatrix}$$

Currently, verbs that have a non-empty SPR value like auxiliaries are not equipped with ARG-ST.

(121) a.



b. ame-da
rain-COP
'It is raining.'

c. yu-ga waku
hot.water-NOM boil
'Water boils.'

d. Ken-ga nimotu-o azukaru
Ken-NOM baggage-ACC be.left.in.one's.trust
'Ken is left a piece of baggage in his trust.'

e. Ken-ga kuru
Ken-NOM come
'Ken comes.'

f. Ken-ga Naomi-o miru
Ken-NOM Naomi-ACC see
'Ken sees Naomi.'

g. Ken-ga Naomi-o eki-ni okuru
Ken-NOM Naomi-ACC station-DAT send
'Ken sends Naomi to the station.'

monotrans, for example, has one external argument and one internal argument (INT1).

Imaizumi and Gunji (2000) make use of ARG-ST to account for the particular V_1-V_2 compounds that include *deru* (let.out) or *dasu* (take.out) as V_2. The V_1-V_2 compounds are used in various ways. In fact, they can be used either as syntactically derived or lexically

Table 3: The type hierarchy of ARG-ST

			EXT	INT1	INT2
nonagentive		*argless*	×	×	×
	unaccusative	*monounac*	×	○	×
		diunac	×	○	○
agentive		*unergative*	○	×	×
	transitive	*monotrans*	○	○	×
		ditrans	○	○	○

derived. On top of this, lexical usages of them are further divided into several types. Imaizumi and Gunji (2000) correctly predict their usages thanks to ARG-ST (and LCS).

Incorporating ARG-ST into JACY's verb hierarchy enables us to distinguish external arguments from internal arguments. In connection with this, the hierarchy of ARG-ST also allows us to distinguish the two kinds of intransitive verb, *monounac* and *unergative*, and the two kinds of transitive verb, *diunac* and *monotrans*. Most linguists recognize the needs for these distinctions in order to account for various linguistic phenomena. For example, Burzio (1986) proposes the possibly universal constraint that a verb which lacks an external argument fails to assign accusative case. Also, Grimshaw (1990) analyzes nominalization and passivization making reference to the (non)existence of an external argument. The JACY without my implementation, which is not equipped with ARG-ST, has no way to account for these phenomena.

As we will see in the next section, ARG-ST, in the dissertation, is used to restrict possible combinations of V_1 and V_2 and to restrict the co-indexing of their two arguments. For example, an *unergative* verb and a *monotrans* verb can be combined only by co-indexing their EXTernal arguments, while an *unergative* verb and a *monounac* verb cannot form a lexical V_1-V_2, since they do not share same arguments in terms of the EXT/INT distinction.

3.6.2 Right headed V_1-V_2s

In §2.3.1, I presented data indicating the semantic or argument structure properties of lexical V_1-V_2s observed by Matsumoto (1996). Although his data described show various patterns in which V_1 and V_2 are combined to form a lexical V_1-V_2, if we underspecify semantic relations between the two component verbs, most of them can be identified as **Right headed V_1-V_2**s, to which the constraints in (122) are applied.

(122) The constraints of **Right headed V_1-V_2**

a. ARG-ST, transitivity and case-marking are determined by V_2.

b. Arguments that agree in the EXT/INT distinction are co-indexed.

As for (122b), EXTernal argument corresponds to Matsumoto's **ag**ent argument, while INTernal arguments include **pa**t**i**ent, **th**eme, **loc**ation, **go**al, and **s**ou**rc**e arguments.

Figure 14 is an example showing how Right headed V_1-V_2, *ki-kuzureru* (wear-get.out.of.shape) 'lose its original shape due to wearing it', in which the V_1 is *monotrans* and the V_2 is *monounac*, is formed. In Figure 14, the ARG-ST, transitivity, and case-marking of the V_1-V_2 is identical to those of V_2, in accord with (122a). In addition, (122b) enforces the co-indexing between the two INTernal arguments. (123) includes the relevant sentences.

(123) a. Ken-ga huku-o kiru
Ken-NOM clothes-ACC wear
'Ken wears his clothes.'

b. huku-ga kuzureru
clothes-NOM get.out.of.shape
'The clothes get out of the shape.'

c. huku-ga ki-kuzureru
clothes-NOM wear-get.out.of.shape
'The clothes get out of the shape due to (someone's) wearing it.'

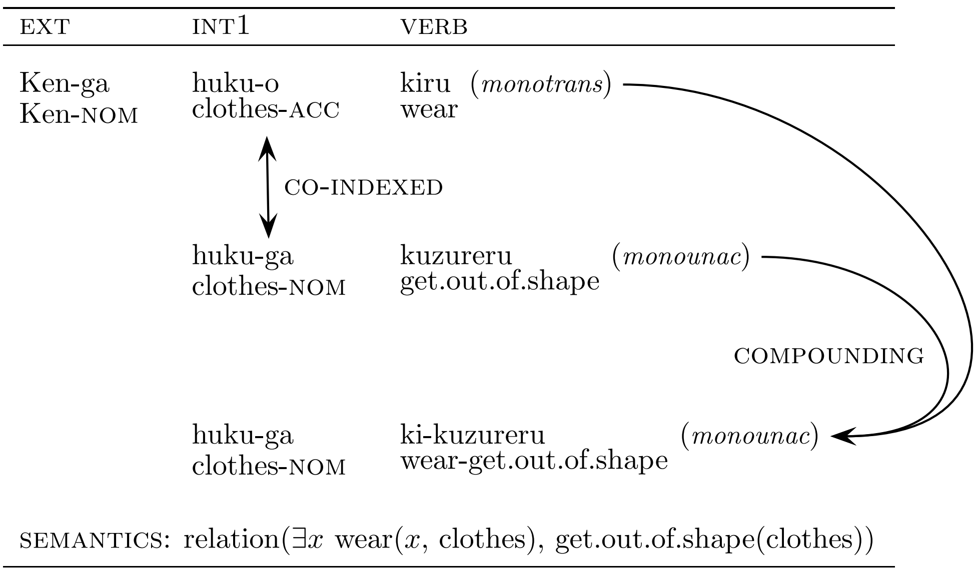


Figure 14: The example of Right headed V_1-V_2

Figure 15 is the JACY output of *huku-ga ki-kuzureru*. We should note that the correct

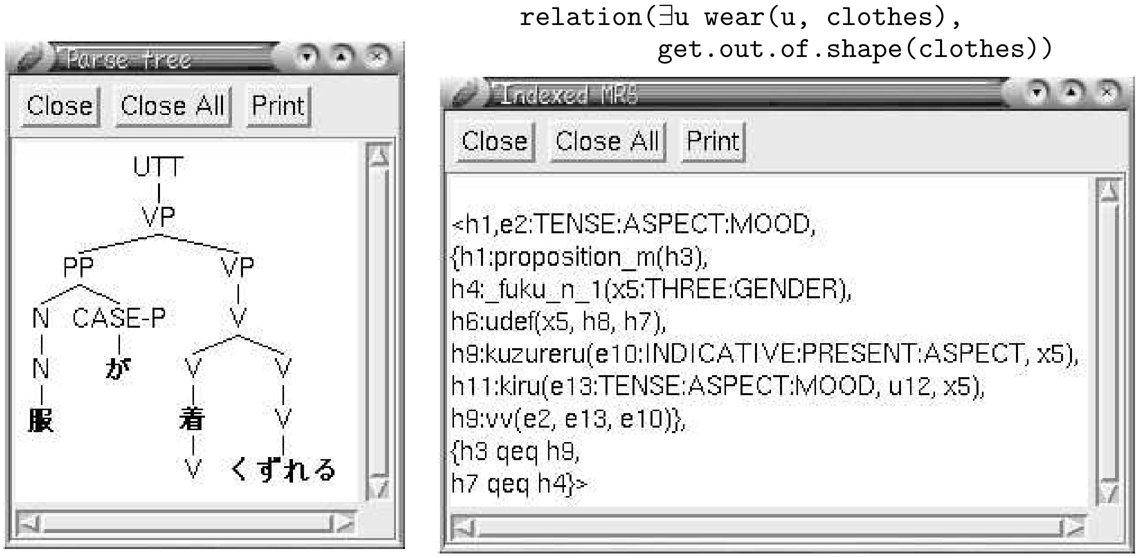


Figure 15: The JACY output of *huku-ga ki-kuzureru*

MRS representation for the sentence (the right side of the figure) is obtained. That is, the MRS says that the subject, *huku*, indicated by `x5` is the thing that gets out of shape, and at the same time, it is the thing that is worn rather than someone who wears something. The actor of *kiru* (wear) that does not surface in the sentence is expressed by `u12`, implying that the reference of the actor would be determined by contextual or pragmatic factors.

Based on the analysis in (122), I have implemented 29 rules that exhaust possible ways of co-indexing arguments between the V_1 and V_2 of Right headed V_1-V_2. Each grid cor-

Table 4: The 29 rules for Right headed V_1-V_2s

V_2 / V_1	*monounac*	*diunac*	*unergative*	*monotrans*	*ditrans*
monounac	**I1-I1**	**I1-I1** **I1-I2**	×	**I1-I1**	**I1-I1** **I1-I2**
diunac	**I1-I1** **I1-I2**	**I1-I1, I2-I2**	×	**I1-I1** **I2-I1**	**I1-I1, I2-I2**
unergative	×	×	**E-E**	**E-E**	**E-E**
monotrans	**I1-I1**	**I1-I1** **I1-I2**	**E-E**	**E-E, I1-I1**	**E-E, I1-I1** **E-E, I1-I2**
ditrans	**I1-I1** **I2-I1**	**I1-I1, I2-I2**	**E-E**	**E-E, I1-I1** **E-E, I2-I1**	**E-E,I1-I1,I2-I2**

responds to one rule, except for the grids with two rows, which have two rules. **E**, **I1** and **I2** are abbreviations of EXT, INT1 and INT2, respectively. ×s indicate that I assume the corresponding V_1-V_2 compounds do not exist.

Here are examples. According to Table 4, a *ditrans* V_1 and a *diunac* V_2 form a V_1-V_2 that shows the co-indexing pattern as follows (**I1-I1, I2-I2**).

(124) a. V_1 : [EXT *index*, INT1 *index*, INT2 *index*]

V_2 : [EXT <>, INT1 *index*, INT2 *index*]

On the other hand, if a *monotrans* V_1 merges with a *ditrans* V_2, the compound results in either (125a) (**E-E, I1-I1**) or (125b) (**E-E, I1-I2**).

(125) a. V_1 : [EXT *index*, INT1 *index*, INT2 <>]

V_2 : [EXT *index*, INT1 *index*, INT2 *index*]

b. V_1 : [EXT *index*, INT1 *index*, INT2 <>]

V_2 : [EXT *index*, INT1 *index*, INT2 *index*]

The example in Figure 14 is a *monotrans-monounac* V_1-V_2 that is assigned a co-indexing pattern of **I1-I1**.

We should notice that Table 4 embodies the **Shared Participant Condition** of Matsumoto (1996). That is to say, the 29 rules of Right headed V_1-V_2 do not allow V_1 and V_2 that share no argument to form a V_1-V_2.

Let us examine how well my approximative analysis can explain the behavior of Matsumoto's *pair*, *cause*, *manner*, and *means* compounds. (126) is the *pair* V_1-V_2 that I took up in §2.3.1.

(126) a. V_1 ⟨th⟩ + V_2 ⟨*th*⟩ = V_1-V_2 ⟨*th*⟩

b. *hikari-kagayaku* (shine-shine) 'shine brightly'

c. isi-ga hikaru
stone-NOM shine
'A stone shines.'

d. isi-ga kagayaku
stone-NOM shine
'A stone shines.'

e. isi-ga hikari-kagayaku
stone-NOM shine-shine
'A stone shines brightly.'

According to (126a), both V_1 and V_2 are *monounac*. Thus the resulting V_1-V_2 is also *monounac* with the two internal arguments co-indexed (**I1-I1**). From (126c-e), we find that the V_1-V_2's transitivity and case-marking are the same as those of both V_1 and V_2. These are all consistent with the constraints on Right headed V_1-V_2.

(127), the *cause* V_1-V_2 cited from §2.3.1, also follows the constraints.

(127) a. V_1 ⟨ag, pt⟩ + V_2 ⟨*th*⟩ = V_1-V_2 ⟨*th*⟩

b. *humi-katamaru* (tramp-harden) 'be tramped hard'

c. Ken-ga zimen-o humu
Ken-NOM ground-ACC tramp
'Ken tramps on the ground.'

d. zimen-ga katamaru
ground-NOM harden
'The ground hardens.'

e. zimen-ga humi-katamaru
ground-NOM tramp-harden
'The ground is tramped hard.'

In this case, the ARG-STs of V_1 and V_2 are different, and that of V_2 is inherited by V_1-V_2. The co-indexing pattern of the two internal arguments, pt and *th*, is **I1-I1**. With regard to transitivity and case-marking, V_1-V_2 and V_2 share the same properties, as (127c-d) indicate.

Below is the *manner* compound described in §2.3.1, whose component verbs are *monounac* and *diunac*.

(128) a. V_1 ⟨th⟩ + V_2 ⟨*th*, *go/loc*⟩ = V_1-V_2 ⟨*th*, *go/loc*⟩

b. *nagare-otiru* (flow-fall) 'flow down'

c. mizu-ga nagareru
water-NOM flow
'Water flows.'

d. mizu-ga sita-ni otiru
water-NOM bottom-DAT fall
'Water falls downward.'

e. mizu-ga sita-ni nagare-otiru
water-NOM bottom-DAT flow-fall
'Water flows down.'

The V_1-V_2's ARG-ST is identical to that of V_2 as expected. Since this V_1-V_2 consists of *monounac* and *diunac*, there are two possible ways of co-indexing for it, namely **I1-I1** or **I1-I2**, although only the former happens to be the correct way for *nagare-otiru*. The transitivity and case-marking of V_1-V_2 are inherited from V_2, following Right headed V_1-V_2's constraints in (122).

Finally, it is obvious from (129) that the *means* compound, *naguri-korosu*, follows the constraints, too.

(129) a. $V_1 \langle \text{ag, pt} \rangle + V_2 \langle ag, pt \rangle = V_1\text{-}V_2 \langle ag, pt \rangle$

b. *naguri-korosu* (strike-kill) 'kill by striking'

c. Ken-ga hito-o naguru
Ken-NOM person-ACC strike
'Ken strikes a person.'

d. Ken-ga hito-o korosu
Ken-NOM person-ACC kill
'Ken kills a person.'

e. Ken-ga hito-o naguri-korosu
Ken-NOM person-ACC strike-kill
'Ken kills a person by striking.'

As the V_1-V_2 is formed from two *monotrans* verbs, arguments of the two verbs are co-indexed in accord with both **E-E** and **I1-I1**.

We have seen some examples of V_1-V_2 presented in Matsumoto (1996) that are consistent with the constraints on Right headed V_1-V_2. As I mentioned previously, Right headed

V_1-V_2s cover most lexical V_1-V_2s, as long as the semantic relations holding between two component verbs are underspecified.

Needless to say, counterexamples to the constraints of Right headed V_1-V_2 do exist. Let me take up some examples described in Matsumoto (1996).

(130) a. $V_1 \langle \text{ag} \rangle + V_2 \langle th \rangle = V_1\text{-}V_2 \langle th \rangle$

b. *ne-midareru* (sleep-get.disorderly) 'get disorderly because of sleeping'

c. *naki-nureru* (weep-get.wet) 'get wet because of weeping'

Both of them consist of an unergative verb and a monounac verb, and are incorrectly ruled out by my analysis. Matsumoto (1996) speculates that they might involve the notion of frame (Fillmore, 1982), though he does not give any detailed analysis of it. Fukushima (2003) posits proto-roles (Dowty, 1991) to account for them, but it would be difficult to implement a proto-role account on the JACY framework. These V_1-V_2 compounds seem to pose a serious problem for my treatment. However, I claim that my treatment is not damaged by them in light of **Importance of Phenomena**. In other words, it is sufficient to give them some *ad hoc* treatment, or to enter them in the lexicon as single words, since counterexamples like those in (130) seem to be rare and the grammar would certainly terribly overgenerate if we allow for *unergative-monounac*. The approach, in which we enter them in the lexicon as a whole, seems appropriate considering the sentences below.

(131) a. kami-ga ne-midareru
hair-NOM sleep-get.disorderly
'Hairs get disorderly because of sleeping.'

b. hoho-ga naki-nureru
cheek-NOM weep-get.wet
'Cheeks get wet because of weeping.'

Both V_1-V_2s in (130) sound the most natural if they are used with the subjects, *kami* (hair) and *hoho* (cheek), respectively. Otherwise, their acceptability would degrade.

(132) a.⁇huku-ga ne-midareru
clothes-NOM sleep-get.disorderly
'Clothes get disorderly because of sleeping.'

b.⁇kao-ga naki-nureru
face-NOM weep-get.wet
'One's face gets wet because of weeping.'

This fact indicates that these two V_1-V_2 compounds are highly lexicalized even though they seem to show a semantic compositionality.

(133), one of the other counterexamples, is peculiar in that even though both of the two component verbs have an internal argument, the two internal arguments are not co-indexed.

(133) a. V_1 ⟨ag, pt/etc., (...)⟩ + V_2 ⟨*ag*, *pt*⟩ = V_1-V_2 ⟨*ag*, *pt*⟩

b. *kui-tubusu* (eat-waste) ‘use up ... by eating’

c. *nomi-tubusu* (drink-waste) ‘use up ... by drinking’

However, I cannot come up with other examples of this type. In addition, these two compounds seems highly lexicalized, too. Look at the examples in (134).

(134) a. Ken-ga zaisan-o kui-tubusu
Ken-NOM property-ACC eat-waste
‘Ken uses up his (or his parents) properties by eating wastefully.’

b. Ken-ga zaisan-o nomi-tubusu
Ken-NOM property-ACC drink-waste
‘Ken uses up his (or his parents) properties by drinking wastefully.’

I, a native speaker of Japanese, fell really odd if *kui-tubusu* or *nomi-tubusu* is used with an object other than *zaisan* (property) (or some other closely related words).

(135)⁇Ken-ga kyuryoo-o {kui-tubusu / nomi-tubusu}
Ken-NOM salary-ACC {eat-waste / drink-waste}
‘Ken uses up his salary by {eating / drinking} wastefully.’

Thus, on the ground of **Importance of Phenomena**, I suppose that a computational grammar should regard them as single words along with the compounds in (130), rather than try to deal with them by means of rules.

Below are other counterexamples, which are also cited from Matsumoto (1996).

(136) a. V_1 ⟨ag-src, th, go⟩ + V_2 ⟨*ag-go*, *th*, *src*⟩ = V_1-V_2 ⟨*ag-go*, *th*, *src*⟩

b. *yuzuri-ukeru* (yield-receive) ‘inherit’

c. *moosi-ukeru* (say-receive) ‘accept the statement of’

d. V_1 ⟨ag-go, th, src⟩ + V_2 ⟨*ag-src*, *th*, *go*⟩ = V_1-V_2 ⟨*ag-src*, *th*, *go*⟩

e. *uke-watasu* (receive-give) 'give away'

These cases, *mirror image* compounds (Fukushima, 2003), seem to obey thematic distinctions rather than the EXT/INT distinction, violating the co-indexing constraint in (122b), though their ARG-ST, transitivity, and case-marking observe (122a).

(137) a. V_1 ⟨ag, (pt)⟩ + V_2 ⟨*th*⟩ = V_1-V_2 ⟨*th*⟩

b. *aruki-kutabireru* (walk-get.tired) 'get tired from walking'

c. *hasiri-tukareru* (run-get.tired) 'get tired from running'

d. *tatakai-yabureru* (battle-be.ruined) 'fall because of fighting'

e. *nomi-tubureru* (drink-get.out.of.shape) 'pass out from drinking'

(137) contains V_1-V_2s that violate the co-indexing constraint; external arguments (ag) and internal arguments (*th*) are co-indexed.

(138) a. V_1 ⟨ag, *pt*⟩ + V_2 ⟨*ag-th*, *loc/go/src*⟩ = V_1-V_2 ⟨*ag-th*, *pt*, *loc/go/src*⟩

b. *nomi-aruku* (drink-walk) 'tour bars'

c. *tabe-aruku* (eat-walk) 'eat around'

d. *sagasi-mawaru* (search-go.around) 'go around, searching'

e. *atume-mawaru* (collect-go.around) 'go around, collecting'

The V_1-V_2s in (138) show a peculiarity similar to (133). That is, the internal arguments of V_1 and V_2 are not co-indexed. Besides, they diverge from Right headed V_1-V_2s in that they can take an object argument from either V_2 or V_1, as I will discuss in the next section.

While admitting that we should be able to deal with all the problematic cases in (136)–(138), I have decided to concentrate on the cases in (138), which I call **Argument mixing $\mathbf{V}_1$-$\mathbf{V}_2$**s in this dissertation, since they are especially productive. Besides, devising theoretical machinery to handle all problematic cases tends to bring us unmanageable complexity. This is the situation that developers of computational grammars must avoid. As for (136) and (137), I will discuss them in §5.1.1.

The next section tackles the Argument mixing V_1-V_2s.

3.6.3 Argument mixing V_1-V_2s

We first look closely at the characteristic behavior of the V_1-V_2s with the example in (139).

(139) *nomi-aruku* (drink-walk) 'drink something while walking'

a. ┌── × ──┐
Ken-ga sake-o nomi-aruku
Ken-NOM sake-ACC drink-walk
└─ ○ ─┘
'Ken drinks sake around somewhere.' (*sake* is V_1's argument.)

b. ┌── ○ ──┐
Ken-ga Tokyo-o nomi-aruku
Ken-NOM Tokyo-ACC drink-walk
└─ × ─┘
'Ken drinks around Tokyo.' (*Tokyo* is V_2's argument.)

As is clear from (139), Argument mixing V_1-V_2s are often ambiguous in that they can take an object argument from either V_2 or V_1. In the case of (139a), the object, *sake*, is what Ken drinks, while the object in (139b), *Tokyo*, is the place where Ken walks while drinking something.

Some Argument mixing V_1-V_2s are used only in the manner of (139a). Indeed, the V_1-V_2s in (140), *moti-saru* for example, cannot be used as taking V_2's object as its matrix direct object, despite the perfect grammaticality of (140c).

(140) *moti-saru* (have-leave) 'go away with'

a. ┌── × ──┐
Ken-ga hon-o moti-saru
Ken-NOM hon-ACC have-leave
└─ ○ ─┘
'Ken goes away with a book from somewhere.' (*hon* is V_1's argument.)

b.* ┌── ○ ──┐
Ken-ga Tokyo-o moti-saru
Ken-NOM Tokyo-ACC have-leave
└─ × ─┘
'Ken goes away with something from Tokyo.' (*Tokyo* is V_2's argument.)

c. Ken-ga Tokyo-o saru
Ken-NOM Tokyo-ACC leave
'Ken leaves Tokyo.'

Nevertheless, many Argument mixing V_1-V_2s also have the usage of (139b). Here is the case of (138d), *sagasi-mawaru*.

(141) *sagasi-mawaru* (search-go.around) 'look for something around'

a. ┌── × ──┐
Ken-ga hon-o sagasi-mawaru
Ken-NOM hon-ACC search-go.around
└── ○ ──┘
'Ken looks for a book around somewhere.' (*hon* is V_1's argument.)

b. ┌── ○ ──┐
Ken-ga Tokyo-o sagasi-mawaru
Ken-NOM Tokyo-ACC search-go.around
└── × ──┘
'Ken looks for something around Tokyo.' (*Tokyo* is V_2's argument.)

Hence, in this dissertation, I regard Argument mixing V_1-V_2s as basically having the two usages of (139a) and (139b), and tentatively treat the V_1-V_2s like (140) as exceptional cases that happen to be used only in the manner of (139a).

Matsumoto (1996) remarks that V_2s that can take part in Argument mixing V_1-V_2 are agentive verbs that express some kind of spatial motion, and V_1 and V_2 share their agent arguments, implying that V_1 must be agentive, too. On top of these, the V_1 of the compound is not a motion verb, as opposed to V_2; when V_1 and V_2 are both motion verbs, the V_1-V_2 formed from the two verbs does not show the property of Argument mixing V_1-V_2, and can be accounted for by one of the rules for Right headed V_1-V_2.[25] Figure 16 exemplifies this. Furthermore, as the name "Argument mixing" implies, V_1 is supposed to be able to contribute arguments other than its agent argument. Hence, I assume that V_1 must be *transitive*, i.e. *monotrans* or *ditrans*. As for a semantic relation, only the *manner* relation can apply to the compound.

With these six assumptions in mind, let's see how the constraints of Argument mixing V_1-V_2s are formulated. As in (142), V_1 is restricted to [ARG-ST *transitive*] and [MOTION −], whereas [ARG-ST *monotrans*] and [MOTION +] are specified for V_2. Note that [3] indicates that an EXTernal argument is shared by V_1 and V_2. This amounts to the co-indexing of EXTs between the two verbs. It is also important to note that the V_1-V_2 is ambiguous in that the V_1-V_2 inherits its ARG-ST and VAL from V_1 or V_2. The left side of the disjunction

[25] Note that in JACY, a locative argument of motion verb is treated as a complement, so spatial motion verbs are usually *monotrans* verbs in my framework. Also, I deal with the V_1-V_2 with two motion verbs by means of a special rules that is not in Table 4. I will mention the rule in Appendix A.

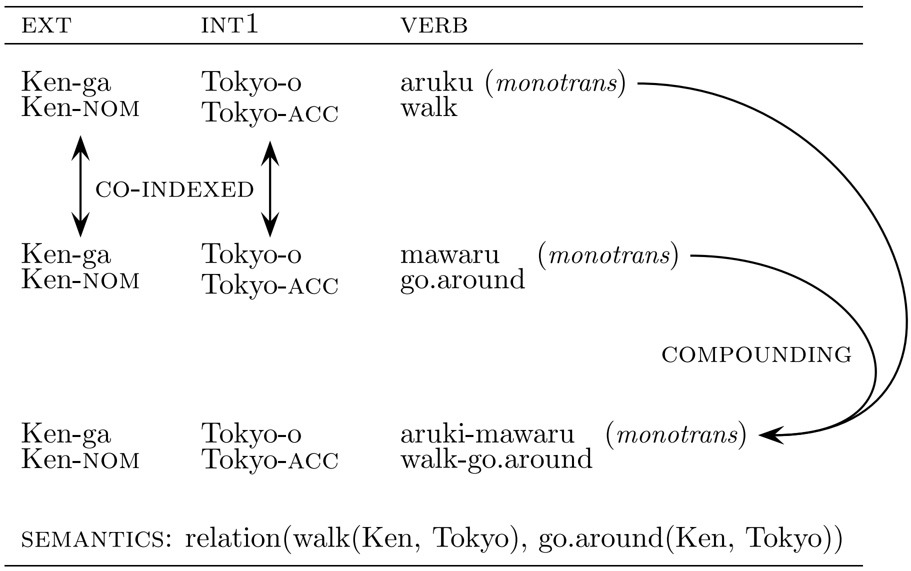
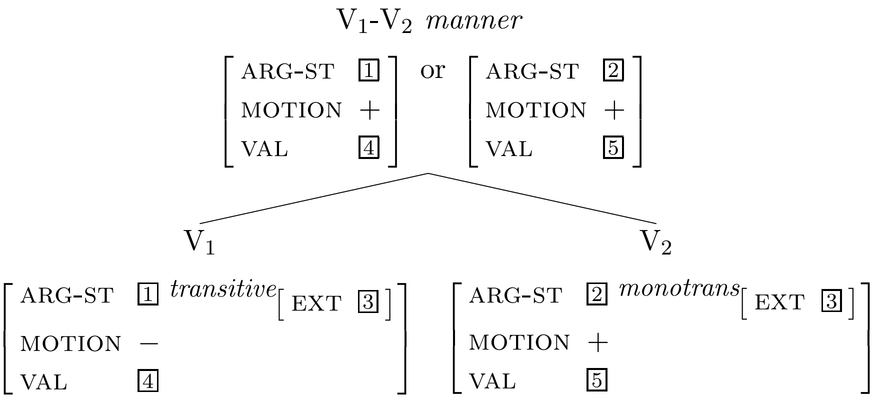


Figure 16: A V_1-V_2 compound consisting of two motion verbs

in (142) corresponds to (139a), the case where an matrix object is coming from V_1, while the right side would result in (139b), in which the V_1 contributes no argument to the V_1-V_2.

(142) The constraints of **Argument mixing V_1-V_2**



These two possibilities are illustrated in Figure 17.

Figure 18 and 19 show the JACY output of (139a). There are two MRS representations in Figure 19. Note that the formulation of Argument mixing V_1-V_2 presented in (142) necessarily generates ambiguity corresponding to COMPOUNDING1, where the object is related to the theme of V_1, and COMPOUNDING2, where the object represents the location of V_2, in Figure 17. The two MRSs in Figure 19 correspond to the ambiguity, with the upper one generated by COMPOUNDING1 and the lower one generated by COMPOUNDING2.

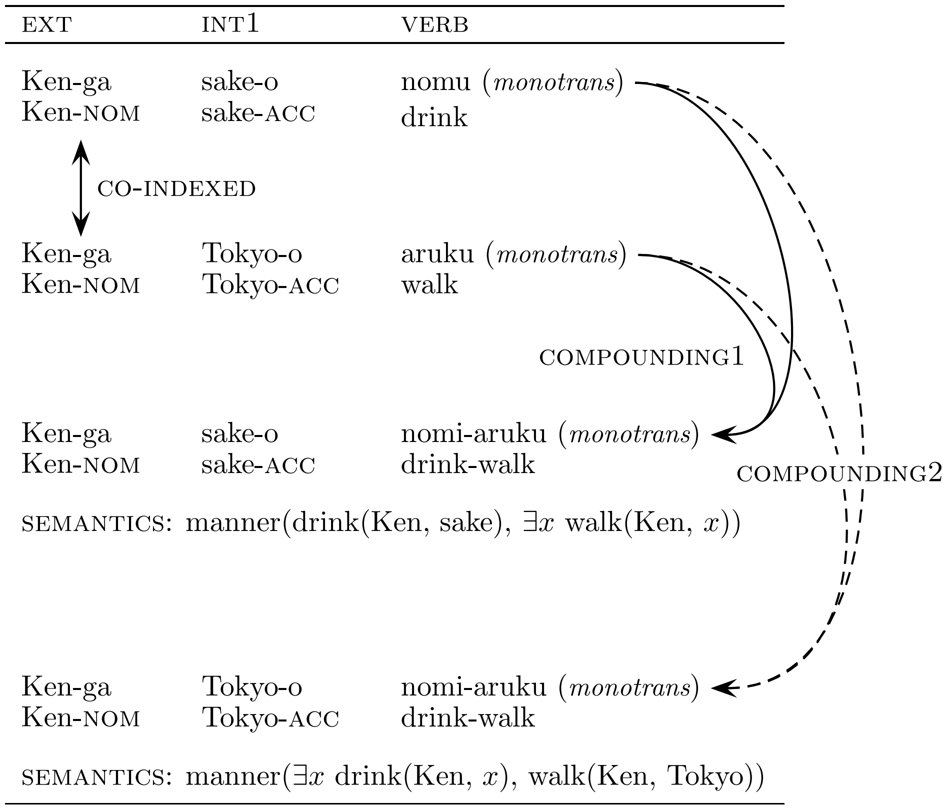


Figure 17: The two possibilities of Argument mixing V_1-V_2

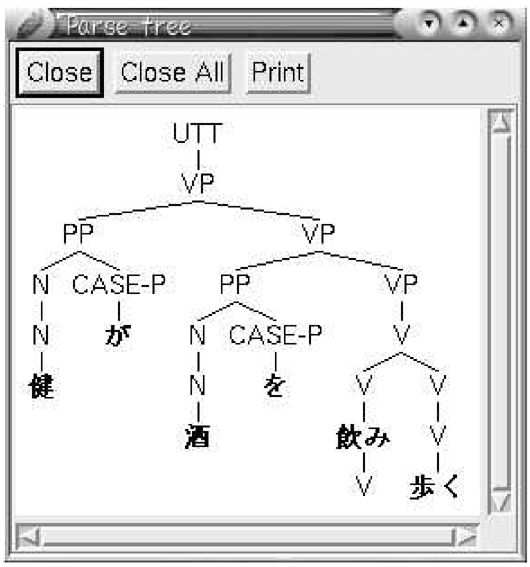
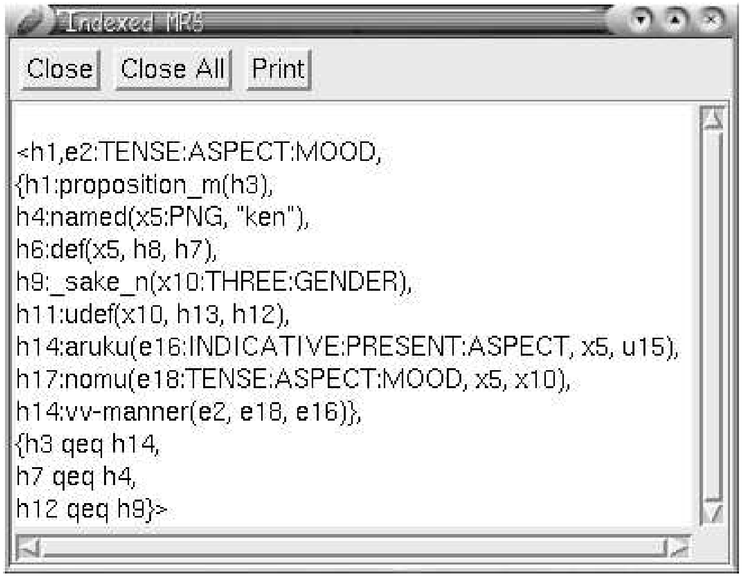


Figure 18: The JACY output of *Ken-ga sake-o nomi-aruku*: syntax

`relation(drink(Ken, sake), ∃u walk(Ken, u))`



`relation(∃u drink(Ken, u), walk(Ken, sake))`

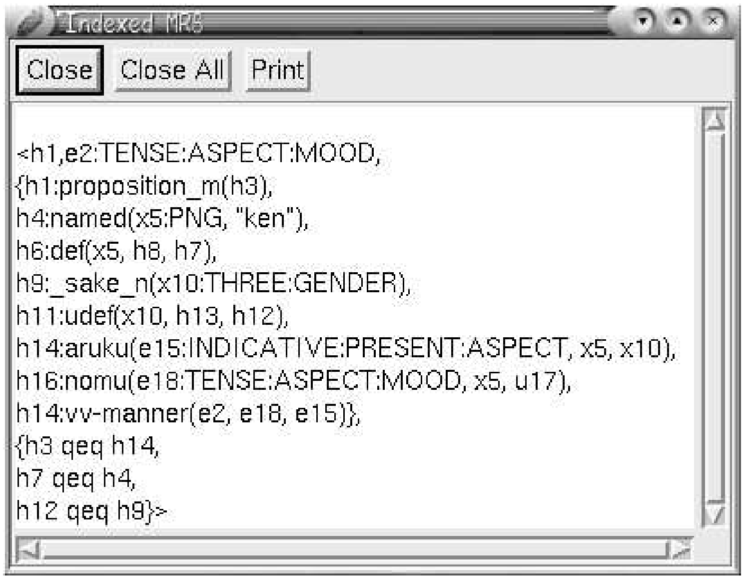


Figure 19: The JACY output of *Ken-ga sake-o nomi-aruku*: semantics

Evidently, the upper one is the correct MRS, in which *sake*, `x10`, represents the theme of *nomu* (drink). At this point, you might think that it is not necessary to get two MRSs for Argument mixing V_1-V_2s such as *nomi-aruku*, since we human beings are able to see, with no trouble, that *sake* is something to drink rather than somewhere to walk. However, it is well-known that building the large-scale database of world knowledge of this kind consistently is unexpectedly difficult, and thus, it is unlikely at least for now that computational systems understand sentences as flexibly as human beings. On top of this, contextual information might allow the interpretation that *sake* is the name of place. We can easily imagine the fiction in which there are countries named *whiskey*, *beer*, and *sake*.

So far I have presented the rules and constraints for the two kinds of lexical V_1-V_2: Right headed V_1-V_2 and Argument mixing V_1-V_2, and discussed how they jointly give rise to a wide range of phenomenon relating to lexical V_1-V_2s. One might argue that the analysis presented here is too coarse for sophisticated linguistic explanations, but it is still justified from the engineering point of view; thanks to the simple but reliably broad coverage nature of the rules and constraints, I dispense with, for example, Back Formation and the LCS analysis posited by Kageyama (1993), so that my analysis can meet the conditions of **Simplicity of Design** and **Importance of Phenomena**.

3.6.4 V_1-V_2s with semantically deverbalized V_1

Both Right headed V_1-V_2s and Argument mixing V_1-V_2s show compositionality in their meanings, but there are lexical V_1-V_2s that are not fully compositional in Japanese. They are divided into the following three types.

(143) a. V_1-V_2s whose V_1 does not contribute its meaning to the whole compound

b. V_1-V_2s whose V_2 exhibits adverbial meaning rather than verbal meaning

c. V_1-V_2s that are totally idiomatic in the sense that neither V_1 nor V_2 are reflected in V_1-V_2's meaning

In this section, the treatment of (143a), **V_1-V_2s with semantically deverbalized V_1**, is discussed.

V_1-V_2s with semantically deverbalized V_1 include those illustrated in (144).

(144) a. Ken-ga zyugyoo-o kaki-midasu
Ken-NOM class-ACC scratch-disturb
'Ken disturbs the class.'

b. ginkoo-ga zaisan-o sasi-osaeru
bank-NOM property-ACC thrust-seize
'A bank seizes the property.'

c. musi-ga huku-ni tori-tuku
bug-NOM clothes-DAT take-stick
'A bug sticks on clothes.'

Kageyama (1993) claims that the V_1 of this kind only emphasizes V_2's meaning. Matsumoto (1996) makes a similar comment about them. Besides, it is argued in Kageyama (1993) that the V_1-V_2s seem to have no restriction on possible combinations of V_1 and V_2 in terms of ARG-ST, unlike Right headed and Argument mixing V_1-V_2s. Here are data supporting the argument cited from Kageyama (1993).

(145) V_1 + *monounac*

a. *kaki-kumoru* (scratch-cloud.up)
kaki-kureru (scratch-be.stumped)

b. *sasi-hibiku* (thrust-ring.out)
sasi-semaru (thrust-close)

c. *tori-magireru* (take-be.distracted)
tori-kakaru (take-hang)

(146) V_1 + *monotrans*

a. *kaki-midasu* (scratch-disturb)
kaki-atumeru (scratch-group.together)

b. *sasi-hasamu* (thrust-shut.in)
sasi-tateru (thrust-sway.up)

c. *tori-kimeru* (take-decide)
tori-kowasu (take-break)

(145) and (146) show that V_1s of this type of V_1-V_2 can attach to both *nonagentive* and *agentive* V_2s, and hence provide evidence that V_1-V_2s with semantically deverbalized V_1 are not restricted by ARG-ST.

With these in mind, I treat this kind of V_1-V_2 as formed by prefixation with V_1 losing its original meaning in the compound, and V_1 can be prefixed to any kind of V_2 in principle.[26] Given this analysis, the (simplified) semantics of the sentences in (144) would look as follows.

(147) a. disturb(Ken, class)

b. seize(bank, property)

c. stick(bug, clothes)

Notice that the V_1s make no semantic contribution to the V_1-V_2s. The JACY output of (144a) is illustrated in Figure 20. It is shown that what *Ken*, `x5`, does to *zyugyoo* (class), `x10`, is only to *midasu* (disturb), with V_1, *kaki*, only emphasizing the proposition, `h18` (=`h20`), which corresponds to *Ken-ga zyugyoo-o midasu* 'Ken disturbs the class,' as indicated by `h19:vv-prefix-v1(e2, h20)`. This is the correct semantics, considering the observation of Kageyama (1993) and Matsumoto (1996).

3.6.5 V_1-V_2s with semantically deverbalized V_2

In this section, I take up (143b), **V_1-V_2s with semantically deverbalized V_2**.

The semantics of this type of V_1-V_2 has been analyzed as an embedding structure where V_2 embeds V_1's semantics (Kageyama, 1993), or as V_2 taking on adverbial meaning that modifies V_1 (Matsumoto, 1996). Either way, the V_2s seem to have lost their original verbal meanings. The examples in (148) and (149) are cited from Kageyama (1993) and Matsumoto (1996), respectively.

(148) a. *hibiki-wataru* (ring.out-cross) 'ring through'

b. *kumi-kawasu* (ladle-exchange) 'hobnob with'

c. *tukai-konasu* (use-deal.with) 'master'

(149) a. *sikari-tukeru* (scold-attach) 'scold harshly'

[26] In fact, this is not really true, as there are several cases where a prefix V_1 cannot attach to V_2.

(i) a. **kaki-warau* (scratch-laugh)
**kaki-waku* (scratch-boil)

b. **sasi-tobu* (thrust-jump)
**sasi-oreru* (thrust-fracture)

c. **tori-hasiru* (take-run)
**tori-uku* (take-float)

These would be what Fukushima (2003) calls *accidental gaps*, the treatment of which might be beyond the current state of the art of linguistics, and I suppose my solution would be as good as any among those available for now.

disturb(Ken, class)

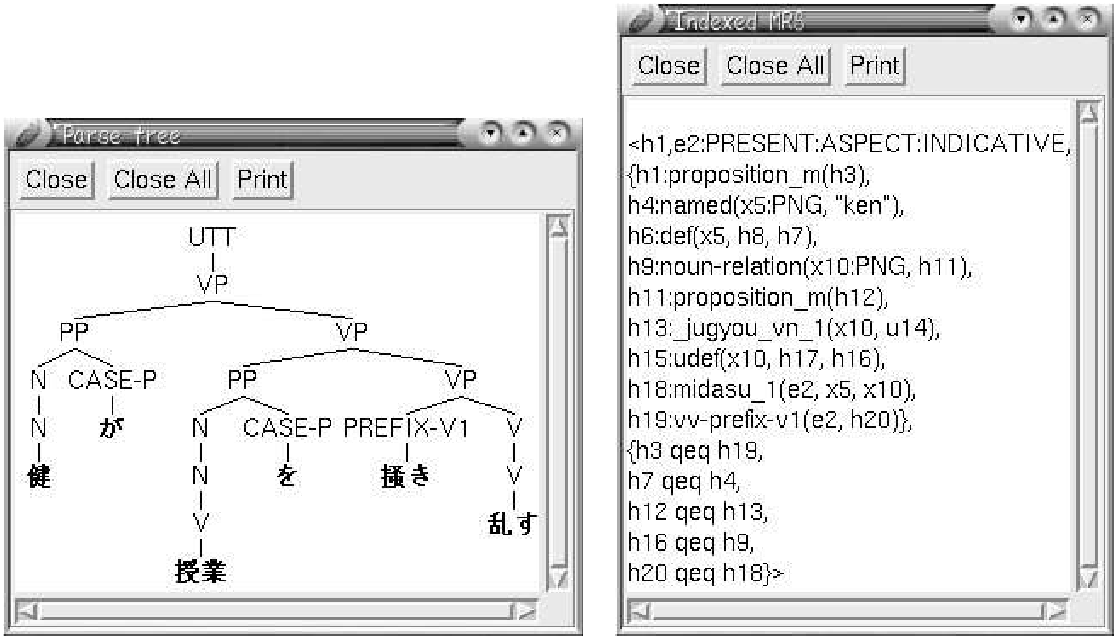


Figure 20: The JACY output of *Ken-ga zyugyoo-o kaki-midasu*

b. *omoi-tuku* (think-be.attached) 'think of, hit upon'

c. *hiki-kaesu* (draw.back-return) 'retreat'

Kageyama (1993) claims that this type of V_1-V_2 observes his **Transitivity Harmony Principle**. For example, The V_1-V_2 in (148a) consists of *nonagentive* V_1 and *nonagentive* V_2, while, in (148b), the V_1 and V_2 are both *agentive*, and they sound completely natural. In constrast, those V_1-V_2s in (150) that violate the principle show ungrammaticality.

(150) a.**sakebi-wataru* (shout-cross) '?'

b.**ukabi-kawasu* (float-exchange) '?'

The V_1-V_2 in (150a) consists of *agentive* V_1 and *nonagentive* V_2, and the V_1-V_2 in (150b) are formed from *nonagentive* V_1 and *agentive* V_2.

Based on these observations, I propose constraints for the V_1-V_2s with semantically deverbalized V_2 in the following way.

(151) The constraints of **V_1-V_2s with semantically deverbalized V_2**

a. V_1 and V_2 must agree in agentivity; This type of V_1-V_2 is required to consist of either one of the two below.

- *agentive* V_1 and *agentive* V_2
- *nonagentive* V_1 and *nonagentive* V_2

b. The V_2 semantically embeds V_1's semantics.

Let us see some examples.

(152) a. Ken-ga hon-o yomi-konasu
Ken-NOM book-ACC read-deal.with
'Ken reads a book competently.'

b. oto-ga (ie-zyu-ni) hibiki-wataru
sound-NOM (house-around-DAT) ring.out-cross
'The sound echoes (throughout the house).'

(152a) is the case of *agentive* V_1 and *agentive* V_2, while the V_1-V_2 in (152b) is formed from *nonagentive* V_1 and *nonagentive* V_2. Their (simplified) semantics would be those in (153).

(153) a. deal.with(Ken, read(Ken, book))

b. cross(ring.out(sound))

Figure 21 shows the JACY output of (153b). As in the right side of Figure 21, the V_2, *wataru*, semantically embeds the proposition that corresponds to *oto-ga hibiku*, 'the sound echoes', indicated by `h10` (=`h11`=`h12`).

3.6.6 Non-compositional V_1-V_2s

Non-compositional V_1-V_2 compounds are distinguished from (partially) compositional V_1-V_2s by their characteristic that neither V_1 nor V_2 contributes to the meaning of the V_1-V_2. Below are the examples of non-compositional V_1-V_2, which repeat those in (5) on page 8.

(154) a. *kuri-kaesu* (turn.over-give.back) 'repeat'

b. *uti-kiru* (hit-cut) 'abort'

c. *uti-tokeru* (hit-thaw) 'come out of one's shell'

d. *tori-midasu* (take-disturb) 'come apart' `cross(ring.out(sound))`

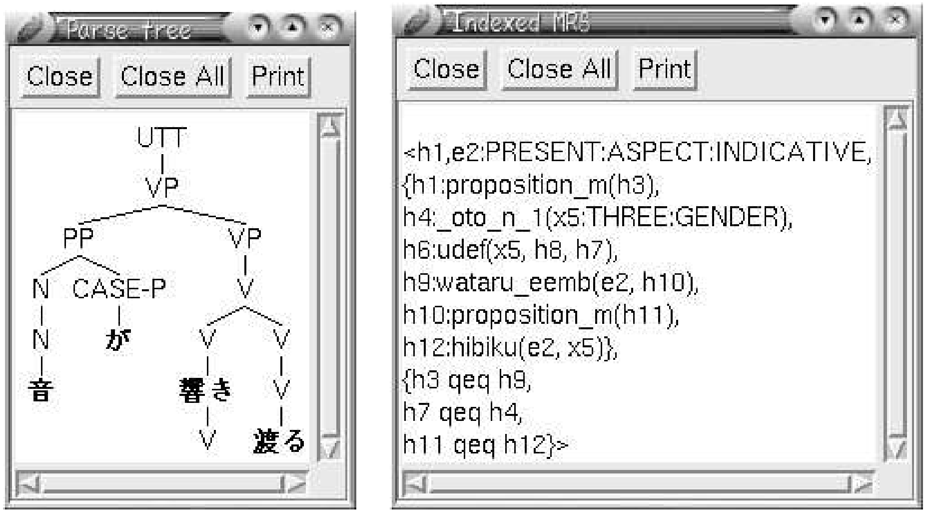


Figure 21: The JACY output of *oto-ga hibiki-wataru*

e. *tori-simaru* (take-fasten) 'police'

f. *hiki-tatu* (pull-stand) 'look well'

As I will discuss later in §5.1.2, there have been several researches in the field of NLP that try to automatically detect non-compositional V_1-V_2 compounds from corpora. On the other hand, it seems that few linguists have shown interest in non-compositional V_1-V_2s. Probably the reason for this would be that non-compositional V_1-V_2s, which are idiosyncratic and have almost no regularity, cannot be the subject of study of linguistics, which investigates regularity of language.

Considering their characteristic, we should treat them as single words and at present I enter V_1-V_2 compounds that seem non-compositional into the lexicon by hand. The parse example of the sentence, *keisatu-ga hanzai-o tori-simaru* 'Police controls crimes', which includes the non-compositional V_1-V_2, *tori-simaru* (take-fasten) 'police' in Figure 22. It is shown that *tori-simaru* is treated as a single word.

However, this simple strategy would suffer from two problems. One is that it should be automated to enter non-compositional V_1-V_2s into the lexicon, since the task would be time-consuming. I will discuss the issue in §5.1.2. The other, and probably more vexing problem, concerns how we can define the non-compositionality formally. The notion seems completely semantic, and therefore, we would have to rely on nothing but semantic intuition, which would not be very stable. Nevertheless, we should find the formal characterization of police(police, crime)

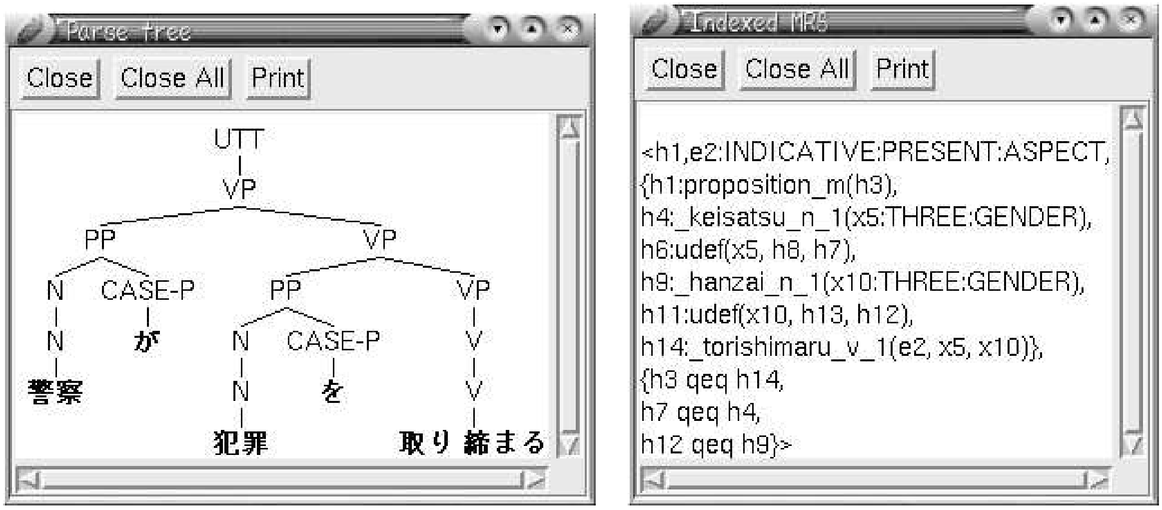


Figure 22: The JACY output of the non-compositional V_1-V_2 *tori-simaru*

non-compositionality so that the resulting grammar and lexicon can be consistent. Bannard et al. (2003) conducted the experiment where they asked 28 participants whether or not a given verb-particle construction in English was compositional, and built the resource by which we can judge verb-particle constructions' compositionality relatively stably. Maybe this kind of resource, if available, can be helpful, although that is obviously not a formal characterization. Anyway, I leave this problem open in the dissertation.

3.7 Summary

In chapter 3, I presented my implementation of V_1-V_2 compounds. To begin with, I discussed general policies by which I develop the computational analysis of V_1-V_2 compounds.

- NLP grammars should concentrate on phenomena that occur frequently, and should not be complicated to explain "exceptional" cases. Also, data that NLP grammars deal with should be those which people judge the grammaticality of consistently.

  → **Importance of Phenomena**

- NLP grammars should be conservative or somewhat descriptive, and should not adopt theoretically advanced but controversial analyses.

  → **Simplicity of Design**

- Information that has to be stipulated in each lexical item and rule should be things that we can easily determine the category it belongs to, so that a large-scale NLP lexicon and rules could be easy to build and maintain.

→ **Availability of Input**

In addition, in order to assure **Efficiency of Computation**, a description language of a grammar should be (a subset of) the $\mathcal{TDL}$ language (Krieger & Schafer, 1994).

After I described the JACY grammar, some basics of HPSG, and the LKB system, my analysis of syntactic V_1-V_2 compounds and lexical V_1-V_2 compounds was presented. Following Kageyama (1993), I classified syntactic V_1-V_2 compounds into three types as in (155).

(155) Syntactic V_1-V_2 compound

a. **A type**

b. **B type**

c. **C type**

I showed that my analysis correctly predicted several phenomena involving syntactic V_1-V_2s that Kageyama accounts for. These phenomena include the difference in the possibility of theta-marking, the passivizability of V_1-V_2, the verbal proform *soo suru* 'do so', and honorification. My phrase structure analysis posited neither movements nor empty categories, unlike Kageyama (1993). Hence, I concluded that my analysis of syntactic V_1-V_2s would meet the condition of **Efficiency of Computation**, and yet would be theoretically adequate.

As a computational implementation embedded in JACY, there was a caveat in my analysis of syntactic V_1-V_2 compounds. That involved how we could allow for scrambling from embedded VPs such as those in the A type and the B type. JACY does not allow for scrambling from any embedded structure. Accordingly, I introduced Argument Attraction to JACY to handle the scrambling. But, at the same time, several kinds of unnecessary ambiguity were brought about. I got around these problems successfully by restricting the grammar with some stipulations. Although these stipulations might not be theoretically adequate, my Argument Attraction approach has computational advantages over other approaches such as the movement approach (Saito, 1985; Hoji, 1985), the SLASH approach (Gunji, 1987) and the Linearization approach (Yatabe, 1996), among others.

My analysis of lexical V_1-V_2 compound was indebted to Matsumoto's (1996) observation, and was based on the ARG-ST of Imaizumi and Gunji (2000). I began with the classification of lexical V_1-V_2s into five categories.

(156) Lexical V_1-V_2 compound

a. **Right headed V_1-V_2s**

b. **Argument mixing V_1-V_2s**

c. **V_1-V_2s with semantically deverbalized V_1**

d. **V_1-V_2s with semantically deverbalized V_2**

e. **Non-compositional V_1-V_2s**

My classification was similar to Matsumoto (1996), but I intentionally underspecified the semantic relation between the V_1 and the V_2 of the Right headed V_1-V_2, so that **Availability of Input** and **Importance of Phenomena** could be satisfied. Furthermore, my analysis was so concise and explicit that it could meet **Simplicity of Design**. Introducing the Pseudo Lexical Rules to the grammar enabled me to distinguish lexical V_1-V_2s from syntactic V_1-V_2s.

A VP embedding structure and ARG-ST were the most notable features of my implementation. Almost all of the previous computational grammars of Japanese, to avoid complexity, have not adopted both of them, but they certainly have a wide applicability and are helpful in restricting a grammar properly. Indeed, these two mechanisms allowed me to deal with the MWE nature of V_1-V_2 compounds.

Of course, I could not solve all of the linguistic problems involving V_1-V_2 compounds in Japanese. My analysis is admittedly coarser than existing linguistic analyses on V_1-V_2 compound such as Kageyama (1993) and Matsumoto (1996). However, I have claimed that the coarse nature of my analysis is justifiable on the ground of the criterion of Hasida (1997); **Importance of Phenomena**, **Simplicity of Design**, **Availability of Input**, and **Efficiency of Computation**. Moreover, my treatment of V_1-V_2 compounds is much more sophisticated than those of the previous NLP grammars as well as getting more coverage as I will show in chapter 4. Besides, my treatment would be compatible with theoretically more advanced analyses, and it is very likely that my treatment will develop, even in the "breadth-first" way, so as to be able to make extensive use of, say, the analysis of Kageyama (1993) with the LCS database of Takeuchi et al. (2003), which I briefly mentioned in §2.3.2. My approach thus makes use of *Successive Approximation*. The term means that scientific understanding is not always correct but is getting closer to correct. In a similar way, my engineering oriented analysis of V_1-V_2 compounds would also get closer to correct.

Chapter 4 Evaluation

Theoretical linguists evaluate a grammar or theory against idealized data, while NLP engineers use a huge amount of text data to investigate the coverage and/or precision of their grammars. Throughout the previous chapter, it has been shown that my analysis deals moderately with the theoretical aspects of V_1-V_2 compounds. In this chapter, then, we will evaluate how well the implementation deals with a large corpus from an engineering point of view.

4.1 What does a good computational grammar look like?

In §2.1, on page 13, I described Hasida's (1997) criteria by which a linguistic theory is judged to be suitable for NLP, which are repeated below.

Importance of Phenomenon: The phenomena a theory tries to explain should be important not only for linguistics but also for NLP.

Simplicity of Design: A theory should make NLP systems simple.

Efficiency of Computation: It must be possible to execute the computation posited by a theory efficiently.

Availability of Input: The inputs that a theory makes reference to should be easily available to NLP systems.

The theory or analysis behind a good computational grammar should meet the criteria, and we argued that my analysis presented in the previous chapter indeed met them.

Furthermore, there are other relevant issues that are somewhat related to Hasida's criteria: the conditions that deep linguistic treatments of NLP in general should satisfy, which are described in 1.2.1, page 4 and repeated below.

1. They must be executed in an efficient way.
2. They must be able to find the best parse among ambiguities.
3. They must have broad coverage.

In a grammar development context, the second item should be replaced with the claim that a good computational grammar should get as few ambiguities as possible.

Therefore, in this chapter, I evaluate my implementation in the following respects.

1. Coverage
2. The amount of ambiguity
3. Processing efficiency

That is, my implementation is judged to be better than the original version of JACY if the former has broader coverage, less ambiguity, and works more efficiently than the latter.

4.2 The details of evaluation

4.2.1 [incr tsdb()]: competence and performance laboratory

In the evaluation, I use [incr tsdb()] (Oepen & Carroll, 2000), the grammar profiling environment (Figure 23). [incr tsdb()] helps us look into a grammar's *competence* and *performance*.

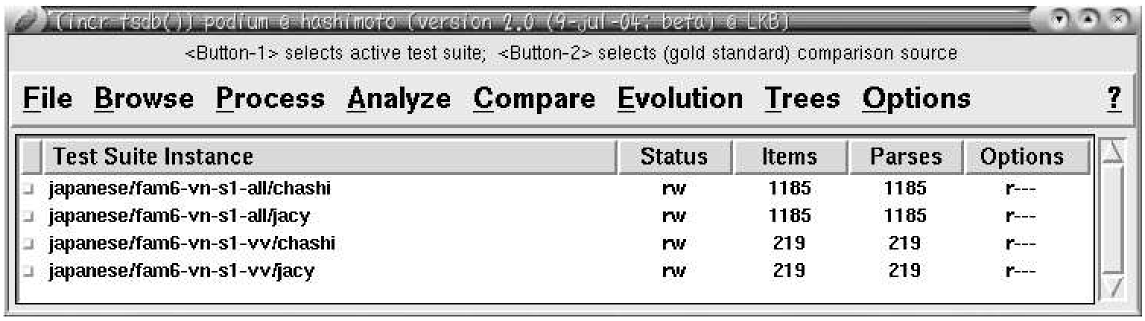


Figure 23: [incr tsdb()]

Note that in the grammar profiling context of [incr tsdb()], *competence* means, among other things, a grammar's coverage and the amount of ambiguity the grammar produces, and *performance* means how efficiently the grammar works: processing time, memory consumption, and the number of operations while parsing.

[incr tsdb()] is independent of what grammar, parser, and corpus are used. This feature facilitates a comparison of different versions of grammar.

4.2.2 Lexeed: a fundamental vocabulary database

The Lexeed Semantic Database of Japanese aims to cover the most common words in Japanese. It was built based on a series of psycholinguistic experiments where words from two existing machine-readable dictionaries were presented to subjects, and they were asked to rank them on a familiarity scale from one to seven, with seven being the most familiar (Amano & Kondo, 1999; Kanasugi et al., 2002; Kasahara et al., 2004). Lexeed consists of all words with a familiarity greater than or equal to five. There are 28,000 words in all. Many words have multiple senses, and there were 46,347 different senses. Definition sentences for these sentences were rewritten by four different analysts to use only the 28,000 familiar words and the best definition chosen by a second set of analysts. In the final configuration, 18,700 different words (66% of all possible words) were actually used in the definition sentences. An example entry for the word 檜 *hinoki* "Japanese Cedar" is given in Figure 24, with

DEFINITION WORD 檜
Japanese Cedar
VARIANTS 桧
White Cedar
FAMILIARITY 5.469 [1-7]
DEFINITION *sense 1*
s1 一年を通して葉が緑色の高い木。
tall tree whose leaves remain green at all times of the year.
s2 高さが *30* メートル、直径 *1* メートルに達する。
height reaches 30 meters, diameter 1 meter.
...
s3 材は優良な建築材。
lumber is an excellent building material.

Figure 24: Entry for the word *hinoki* "Japanese Cedar" (with English glosses)

English glosses added. In all there are 81,000 definition sentences.

In the evaluation, I use the definition sentences of Japanese verbal nouns in Lexeed. The total number of the definition sentences I use is 1,185.

4.2.3 Evaluation procedure

I prepared two versions of JACY: JACY-plain and JACY-vv (Table 5). JACY-plain is

Table 5: The two versions of the JACY grammar

	V_1-V_2 implementation	**V_1-V_2 entries**
JACY-plain	*No*	*Yes*
JACY-vv	*Yes*	*No*

not given the V_1-V_2 implementation, but contains 1,325 lexical entries of V_1-V_2 compounds, which were gotten together from several corpora including the corpus of Verbmobil

(Wahlster, 2000). (157) illustrates some of the 1,325 lexical entries.

(157) a. *mori-ageru* (pile.up-raise) 'heap up'

b. *mi-mamoru* (look-protect) 'watch attentively'

c. *sinobi-yoru* (undergo-get.up) 'creep on'

d. *kai-toru* (buy-take) 'buy out'

e. *humi-kiru* (tramp-cut) 'embark on'

f. *tati-naoru* (stand.up-recover) 'regain one's footing'

g. *mi-tumoru* (look-accumulate) 'estimate'

h. *ni-tumaru* (boil.up-clog) 'simmer down'

i. *kati-nokoru* (win-stay.in) 'remain in competition'

Most of them are more or less lexicalized or conventionalized expressions, and frequently occur in writings and utterances.

On the other hand, JACY-vv is equipped with the rules and lexical types[1] that I have discussed so far, but there is no V_1-V_2 entry in its lexicon, except for the two non-compositional V_1-V_2 compounds in (158).

(158) a. *nari-tatu* (become-stand) 'consist'

b. *mousi-komu* (say-go.in) 'apply'

I regard JACY-plain as the baseline in evaluating my implementation's (JACY-vv's) competence and performance.

I also prepared two corpora for the evaluation: ALL and V-V (Table 6). The ALL

Table 6: The two evaluation corpora

	the number of sentences
ALL corpus	1,185
V-V corpus	219

corpus is the definition sentences of Japanese verbal nouns I mentioned above, in which

[1]We added 58 lexical entries that are instances of the newly introduced lexical types.

there are 1,185 sentences. The V-V corpus is the subset of the ALL corpus, in which each sentence contains at least one V_1-V_2.[2] The V-V corpus consists of 219 sentences. While the V-V corpus serves as a measure of how well the implementation covers V_1-V_2 compounds occurring in one corpus, the ALL corpus tells us what interaction is brought about between the implementation for V_1-V_2 and the other parts of the grammar.

Using [incr tsdb()], I evaluate the competence and performance of the two versions of the JACY grammar against the two corpora from Lexeed.

4.3 Result

In this section, I will show the results of evaluation I conducted to reveal my implementation's competence and performance. **(g)old** and **new** in the tables below represent JACY-plain and JACY-vv, respectively.

4.3.1 Competence

Let us first look at the two grammars' coverage (**in**) and the average number of (syntactic) ambiguities (**parser**) with respect to ALL corpus (Table 7) and V-V corpus (Table 8).[3]

Table 7: The two grammars' competence with respect to ALL corpus

Phenomenon	(g)old				new			
	lexical Ø	parser Ø	in Ø	out Ø	lexical Ø	parser Ø	in Ø	out Ø
Total	**−0.01**	**49.00**	**70.9**	**100.0**	**−0.01**	**36.59**	**71.8**	**100.0**

(generated by [incr tsdb()] at 22−sep−04 (13:18))

The result in Table 8 is surprising because it shows that JACY-plain's strategy, in which all V_1-V_2 compounds are treated as single words and entered in the lexicon, only covers 52.1% of 219 V_1-V_2 compounds in the corpus even though there are as many as 1,325 entries in the grammar's lexicon. This certainly means that an exhaustive listing approach

[2] I automatically collected these sentences by a perl script utilizing ChaSen (Matsumoto et al., 2000), the morphological analyzer of Japanese.

[3] All of these numbers represent percentages as below.

lexical: the average number of lexical ambiguity per word

parser: the average number of syntactic ambiguity per sentence

in: the overall coverage percentage

out: the overgeneration percentage

The **lexical** measure does not work in this experiment, and the **out** measure is not relevant to the discussion.

Table 8: The two grammars' competence with respect to V-V corpus

Phenomenon	(g)old				new			
	lexical Ø	parser Ø	in Ø	out Ø	lexical Ø	parser Ø	in Ø	out Ø
Total	**−0.01**	**53.41**	**52.1**	**100.0**	**−0.01**	**50.78**	**63.5**	**100.0**

(generated by [incr tsdb()] at 22−sep−04 (13:18))

to V_1-V_2 compounds is simply far from adequate. On the other hand, JACY-vv covers V_1-V_2 compounds in the corpus more broadly than JACY-plain. Remember that no entry for V_1-V_2 compounds (except for the two in (158)) is entered in the lexicon of this version. Table 7 shows a moderate increase in coverage in the ALL corpus.

As for the average number of ambiguities illustrated in Tables 7 and 8, we should notice that JACY-vv exhibits less ambiguity than JACY-plain. This is again surprising because it is usually the case that the amount of ambiguity would increase as we introduce new rules to a grammar. As I will discuss the decrease later, this involves the difference of treatment of VP embedding constructions between the two grammars, which is relevant to the implementation of syntactic V_1-V_2 compounds. In any case, the point is that even if we leave the difference concerning VP embedding constructions out of consideration, my addition of the rules and lexical types for V_1-V_2 compounds to the grammar would not increase the amount of ambiguity drastically.

4.3.2 Performance

Next let us move on to the grammars' performance, which Table 9 and 10 summarize.

Table 9: The two grammars' performance with respect to ALL corpus

Phenomenon	(g)old			new			reduction		
	tasks Ø	time Ø	space Ø	tasks Ø	time Ø	space Ø	tasks %	time %	space %
Total	**61028**	**3.95**	**573284**	**93205**	**4.56**	**680759**	**−52.7**	**−15.3**	**−18.7**

(generated by [incr tsdb()] at 15−oct−04 (16:09))

In these tables, **tasks**, **time**, and **space** stand for the average number of operations while

Table 10: The two grammars' performance with respect to V-V corpus

Phenomenon	(g)old			new			reduction		
	tasks Ø	time Ø	space Ø	tasks Ø	time Ø	space Ø	tasks %	time %	space %
Total	**79783**	**4.85**	**816779**	**137851**	**6.43**	**995681**	**−72.8**	**−32.5**	**−21.9**

(generated by [incr tsdb()] at 15−oct−04 (16:14))

parsing, processing time, and memory consumption, respectively. A smaller number means better performance, and the three measures are usually related to each other to some extent.

Unfortunately, we find from the tables that JACY-vv works less efficiently than JACY-plain in each of the three respects. This is evidently the effect of adding the rules and lexical types for V_1-V_2 compounds: in most cases, more rules lead to less efficiency as well as more ambiguity.

4.4 Discussion

We have seen that my implementation, namely JACY-vv, shows better competence but worse performance. In this section, I look closely into these results and justify my treatment of V_1-V_2 compounds.

4.4.1 Competence

COVERAGE

Clearly, the implementation for V_1-V_2 compounds, which JACY-vv is equipped with, caused the broader coverage. (159) shows some instances that JACY-vv could parse while JACY-plain could not.

(159) a. *narabe-kaeru* (get.lined.up-change) 'rearrange the order'

Right-headed (*monotrans-monotrans*)

b. *kakawari-au* (be.involved-conform) 'be involved with each other'

Syntactic C type

c. *seme-utu* (attack-shoot) 'invade'

Right-headed (*monotrans-monotrans*)

d. *sasi-sadameru* (point-determine) 'ordain'

Right-headed (*monotrans-monotrans*)

e. *tôri-sugiru* (run.through-run.over) 'go by'

Right-headed (*monotrans-monounac*)

f. *nari-sugiru* (become-run.over) 'get something excessively'

Syntactic A type

g. *maze-au* (mix-conform) 'mix up'

Syntactic C type

As I will describe below, there are many more lexical V_1-V_2 compounds than syntactic ones in the evaluation corpus. On top of this, JACY-plain is given 1,325 V_1-V_2 entries, most of which are lexical ones, as we have seen above. Considering these conditions, JACY-plain might have shown better coverage than JACY-vv, but, in fact, it is just the contrary. This surprising result can be attributed to the remarkably high productivity of some lexical V_1-V_2 compounds. Look at the examples of them below.

(160) V_1-*wakeru* (V_1-distinguish)

a. *mi-wakeru* (see-distinguish) 'distinguish by looking' (Tagashira & Hoff, 1986)

b. *kagi-wakeru* (smell-distinguish) 'distinguish by smelling' (Tagashira & Hoff, 1986)

c. *sawari-wakeru* (touch-distinguish) 'distinguish by touching' (Tagashira & Hoff, 1986)

d. *name-wakeru* (lick-distinguish) 'distinguish by licking'

e. *momi-wakeru* (massage-distinguish) 'distinguish by massaging'

f. *nage-wakeru* (throw-distinguish) 'distinguish by throwing'

g. *kiki-wakeru* (listen-distinguish) 'distinguish by listening'

(161) V_1-*korosu* (V_1-kill)

a. *sasi-korosu* (stab-kill) 'kill by stabbing' (Tagashira & Hoff, 1986)

b. *inori-korosu* (pray-kill) 'kill by praying' (Tagashira & Hoff, 1986)

c. *noroi-korosu* (curse-kill) 'kill by cursing'

d. *musi-korosu* (fume-kill) 'kill by fuming'

e. *nirami-korosu* (stare.hard-kill) 'kill by staring hard'

f. *momi-korosu* (massage-kill) 'kill by massaging'

g. *nage-korosu* (throw-kill) 'kill by throwing'

You might think that all lexical V_1-V_2 compounds are restricted and unproductive. However, some of them, such as those in (160) and (161), can be used very productively as long as they are semantically and pragmatically plausible. Needless to say, the approach in which all V_1-V_2 compounds are regarded as single words and are entered in a lexicon would never be able to deal with this productive nature of some V_1-V_2s. Furthermore, once you try to deal with productive and compositional V_1-V_2 compounds by means of some kind of rule, you must be aware of their linguistic properties, as we have seen so far. The importance of linguistic treatment would be more prominent if the grammar was applied to a domain in which a lot of syntactic V_1-V_2 compounds occur.

Now let us look at Table 11, which shows what type of V_1-V_2 compound appears in the evaluation corpus how often.[4] From the table, we see that Right headed V_1-V_2, one type of lexical V_1-V_2 compound, has the highest frequency.[5] Table 12 summarizes the frequencies (and proportions) of each type of Right headed V_1-V_2. This table tells us that those lexical V_1-V_2s that consist of a *monotrans* verb, especially those of the *monotrans-monotrans* type, occur enormously frequently.[6] The proportion of the *monotrans-monotrans* type to all kinds of V_1-V_2 compound (including both syntactic V_1-V_2s and lexical V_1-V_2s), 57.89% ($=\frac{77}{133}$), is shown in Figure 25. This implies that *monotrans-monotrans* type is the most productive, and we indeed observe several creative usages of them in the evaluation corpus.

[4]I used 133 V_1-V_2 compounds out of 219 from the V-V corpus for this. To be more precise, 139 sentences in the V-V corpus were given one or more analysis (remember that the coverage is 63.5%), but it turned out that four sentences out of them were not given a correct analysis, and five sentences did not contain any V_1-V_2 compound because of the ChaSen's failure. As a result, I actually had 130 sentences, which contain one or more V-V compound and are given a correct parse. Three sentences out of 130 contain two V_1-V_2 compounds for each, and that is way, 133 V_1-V_2 compounds out of 219 were really available.

[5]This is unexpected since syntactic V_1-V_2 compounds are usually much more productive and abundant. The characteristics of the evaluation corpus, dictionary definition sentences, might be relevant to this: they have no context, no reference to past nor future.

[6]Kageyama (1993, p.120) discusses this tendency making reference to LCSs of transitive and intransitive verbs.

Table 11: The frequencies of V_1-V_2 compounds

Type	**Frequency**	**%**
Syntactic A Type	3	2.26
Syntactic B Type	1	0.75
Syntactic C Type	8	6.02
Right headed	108	81.20
Argument mixing	0	0.00
Deverbalized V_1	9	6.77
Deverbalized V_2	1	0.75
Non-compositional	3	2.26
Total	133	100

Table 12: The frequencies of Right headed V_1-V_2s

V_2 / V_1	*monounac*	*diunac*	*unergative*	*monotrans*	*ditrans*
monounac	**0**	**0** **0**	×	**2 (1.85%)**	**0** **0**
diunac	**0** **0**	**0**	×	**1 (0.93%)** **0**	**0**
unergative	×	×	**1 (0.93%)**	**6 (5.56%)**	**0**
monotrans	**3 (2.78%)**	**1 (0.93%)** **0**	**9 (8.33%)**	**77 (71.3%)**	**6 (5.56%)** **1 (0.93%)**
ditrans	**0** **0**	**0**	**1 (0.93%)**	**0** **0**	**0**

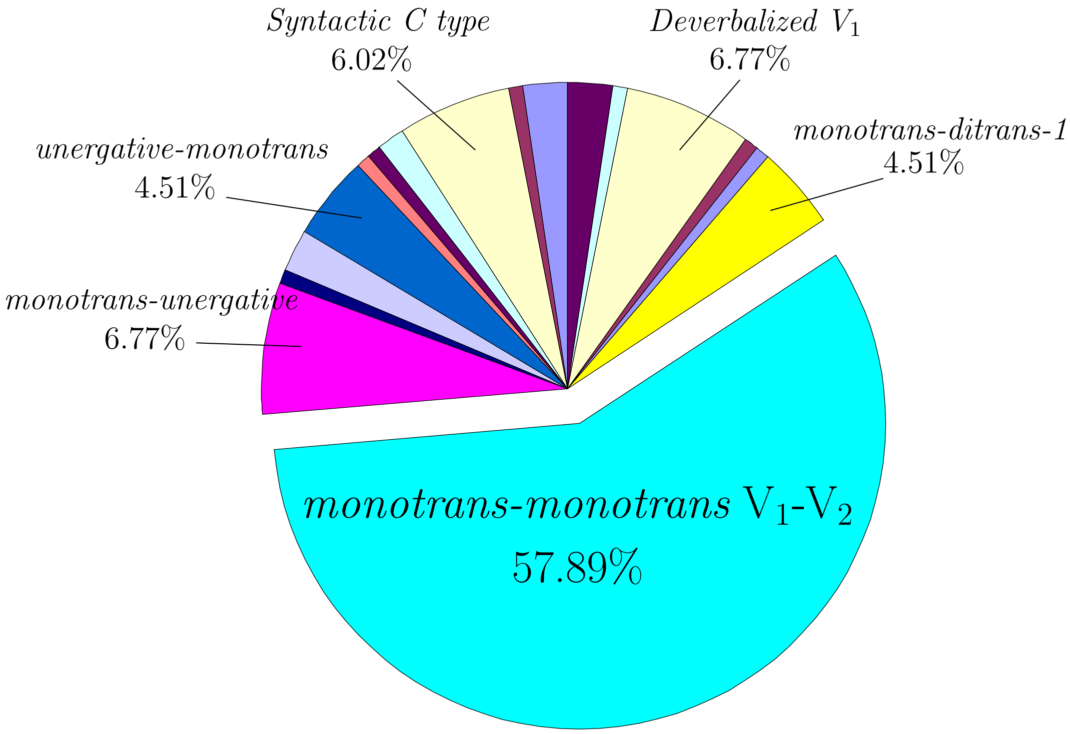


Figure 25: The proportion of *monotrans-monotrans* V_1-V_2 to all kinds of V_1-V_2 compound

(162) a. *hakari-kazoeru* (measure-count) 'measure and count'

b. *osie-mitibiku* (teach-lead) 'lead by teaching'

c. *tuge-siraseru* (report-inform) 'report and inform'

d. *tamesi-siraberu* (test-examine) 'examine by testing'

Some of the native speakers of Japanese might feel unfamiliar with these V_1-V_2s, but I am sure that they do not have any difficulty understanding them and accept them as Japanese words without hesitation. This is exactly what JACY-plain, in which all V_1-V_2 compounds are simply regarded as single words, suffers from: the lexical proliferation problem, which I mentioned in §1.2.2.[7] On the contrary, JACY-vv can deal with them by means of the suitable rule, namely the *monotrans-monotrans* Right headed V_1-V_2 rule. Let us take a look at the JACY output of (163), which includes (162a), for illustration (Figure 26).[8]

[7] I also mentioned the flexibility problem in conjunction with this problem. However, the flexibility problem might not be relevant to V_1-V_2 compounds in Japanese, since they always constitute single morphological words, unlike English phrasal verbs, for example.

[8] The actual sentence in the evaluation corpus that contains *hakari-kazoeru* is the one below.

(163) suuryou-o hakari-kazoeru
amount-ACC measure-count
'(Someone) measures and counts amount.'

```
∃u relation(measure(u, amount),
            count(u, amount))
```

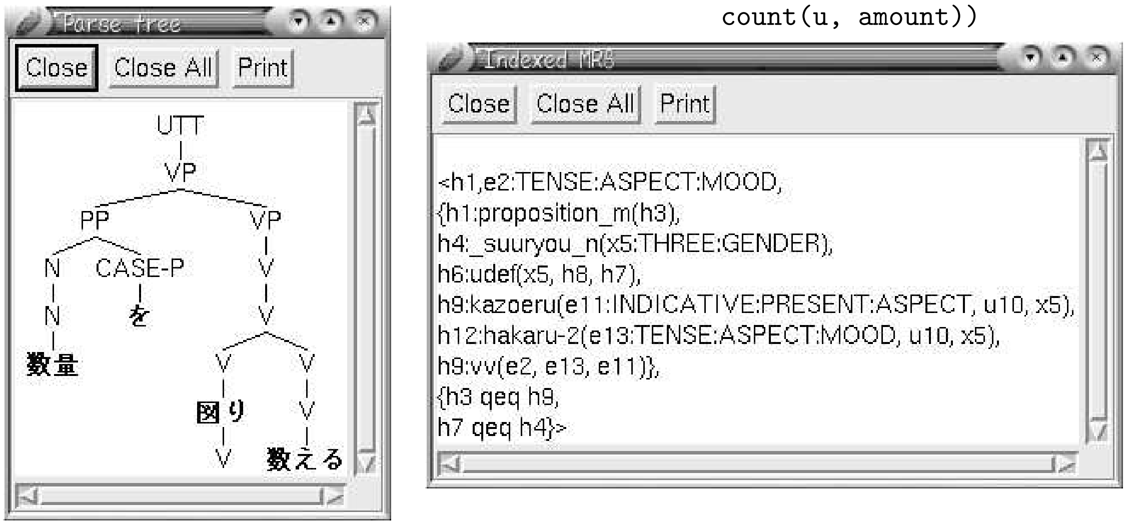


Figure 26: The JACY output of *suuryo-o hakari-kazoeru*

We find from the fifth and sixth lines of the figure that the sentence means something in which someone, who is unspecified in the sentence as indicated by `u10`, measures (*hakaru*) and counts (*kazoeru*) the amount of something (*suuryo*), `x5`. Obviously, this is the correct analysis.

I also discussed the other two problems concerning MWEs: the overgeneration problem and the idiomaticity problem. The strategies that make use of rules to handle V_1-V_2's compositionality, such as the one presented here, have possibilities of suffering from these problems. In fact, my treatment can reasonably get around them. First, in chapter 3, I classified V_1-V_2 compounds into eight sub-types according to their linguistic properties. As a result, it became clear how productive each type of V_1-V_2 compound is and under what condition it is allowed. Thus, I claim that my treatment is mostly immune to the overgeneration problem.[9] Second, I distinguished (partially) compositional V_1-V_2 compounds

(i) suuryou-o hakari-kazoeru koto
amount-ACC measure-count thing
'To measure and count amount.'

But I use the simplified sentence in (163) for ease of exposition.

[9]Nevertheless, one vexing problem, which is involving so-called *accidental gaps*, remains to be solved as I will discuss in §5.1.1.

from non-compositional ones. Thanks to the distinction, I could correctly assign idiomatic meaning to the latter one. However, as for some of compositional cases, some linguists have noticed their lexicalized characteristics. For example, Kageyama (1993, p.78) observes that *nomi-aruku* (drink-walk), which I took up in §3.6.3 as a compositional lexical V_1-V_2 compound, restricts the thing to drink to alcohol by convention as in (164a). It seems that the V_1-V_2 compound poses the idiomaticity problem since my treatment of the compound is totally compositional and cannot predict the idiomaticity. However, we can easily cancel the convention if the object is something other than alcohol. Look at the example in (164b).

(164) a. Ken-ga nomi-aruku
Ken-NOM drink-walk
'Ken tours bars.' / 'Ken walks while drinking (something).'

b. Ken-ga mizu-o nomi-aruku
Ken-NOM water-ACC drink-walk
'Ken walks while drinking water.'

Probably, it is *nomu* (drink) in intransitive use, rather than *nomi-aruku* as a whole, that induces the conventional meaning as indicated in (165).

(165) Ken-wa mainiti nomu
Ken-TOP everyday drink
'Ken drinks alcohol everyday.'

Sentences that JACY-vv could not parse are due to one of the three causes in Table 13. Notice that adding new rules for V_1-V_2 compounds to the grammar leads to more memory

Table 13: The three problems that prevent JACY-vv from getting more coverage

Cause	%
A lack of lexical item	16.25
A lack of memory	38.75
Grammar internal problems	45.00

consumption than JACY-plain. However, this means that we would get more coverage if we had more memory available, which is not so difficult. Adding lexical items to the lexicon is also an easy task. Thus, the first two problems, a lack of lexical items and memory, which comprise 55% of all the problems, can be resolved relatively easily.

AMBIGUITY

As in Tables 7 and 8, JACY-vv shows less ambiguity than JACY-plain though it is usually the case that more rules cause more ambiguity. Indeed, structures that involve V_1-V_2 compounds are given more ambiguity. However, those involving VP embedding structures, such as aspect constructions, are assigned far less ambiguity, resulting in reduction in the amount of ambiguity in total.

(166) is an example of a VP embedding structure.

(166) Ken-ga [$_{\text{VP}}$ hon-o yon-de] iru
Ken-NOM [$_{\text{VP}}$ book-ACC read-TE] BE
'Ken is reading a book.'

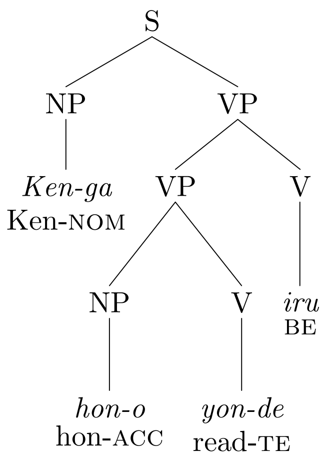
This structure allows an object inside VP to be scrambled out of the VP.

(167) hon-o Ken-ga ϕ [$_{\text{TVP}}$ yon-de] iru
book-ACC Ken-NOM ϕ [$_{\text{TVP}}$ read-TE] BE
'Ken is reading a book.'

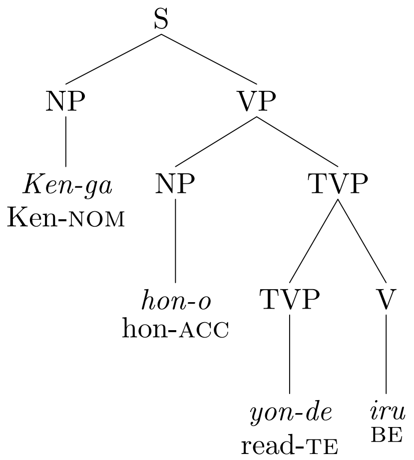
JACY-plain tries to handle this by underspecifying a category; an aspect verb like *iru* (be) subcategorizes for to either VP, a verbal projection saturated with objects, or TVP, a verbal projection that remains to be saturated with an object. As a result, however, JACY-plain necessarily gets two parses (168a,b) for a sentence like (166).

(168) a. Ken-ga hon- yon-de iru
Ken-NOM book-ACC read-TE be
'Ken is reading a book.'

b.



c.



Cleary, the parse in (168c) is unnecessary and should be ruled out. JACY-vv, on the other hand, adopts the Argument Attraction approach for VP embedding structures like this, which I described in §3.5.2 on page 69. Accordingly, JACY-vv is suitably restricted with respect to VP embedding structures and is immune to a spurious ambiguity like (168).

4.4.2 Performance

Performance got worse by introducing the implementation for V_1-V_2s to the grammar, but this is inevitable. More rules lead to more space for a parser to search, resulting in less efficiency. We should worry about efficiency only after a grammar attains a satisfactory coverage and precision, and, thanks to the LKB system and the $\mathcal{TDL}$ language, it still works reasonably efficiently.

However, as I mentioned in §1.2.1, NLP grammars should try to be as efficient as possible especially when they are used in practical NLP applications. Although most studies for better efficiency have been done mainly in the field of algorithms for parsing or generation, there have been some researches of getting better efficiency by revising a grammar: Verlinden (1999) and Flickinger (2000), to name two. Flickinger (2000) conducted three experiments in varying the choice of grammatical representation within his English Resource Grammar (ERG), a linguistically precise broad-coverage computational grammar of English, to see how much different grammatical representations affect processing efficiency. He succeeded in achieving better efficiency by changing grammatical representations without impairing descriptive accuracy and elegance. For example, he stopped using disjunctive feature specifications and made extensive use of the type hierarchy instead.

It is possible that changing grammatical representations in my implementation would lead to better performance, but I leave this problem open in this dissertation.

4.5 Summary

In chapter 4, I evaluated my implementation in terms of *competence* and *performance*, using the Lexeed corpus and [incr tsdb()]. First I reviewed the conditions that a good computational grammar should meet, and then, I briefly described [incr tsdb()] and the Lexeed corpus. I also described how I evaluated my implementation; I prepared two corpora and two versions of the grammar, JACY-vv and JACY-plain.

Consequently, my implementation, namely JACY-vv, outperformed JACY-plain in *competence*, that is, it gained more coverage and less ambiguity. JACY-vv got more coverage because of the remarkably high productivity of one of the lexical V_1-V_2 compounds, *monotrans-monotrans* one, which JACY-plain could not deal with through the approach in which all V_1-V_2 compounds were regarded as single words. In other words, JACY-plain suffered from the lexical proliferation problem, one of the problems concerning MWEs. On the other hand, I claimed that JACY-vv, or my treatment of V_1-V_2 compound, was sound from the MWE perspective. More than half (55%) of sentences that JACY-vv could not parse were due to the lack of lexical item or the lack of memory, which could be resolved relatively easily. The lesser ambiguity was brought about by the difference of the treatment of scrambling from an embedded VP. To be more precise, the restrictive nature of my Argument Attraction approach made us get less ambiguity.

However, as for performance, JACY-vv turned out to be working less efficiently than JACY-plain. Although I mentioned that this was unavoidable considering that JACY-vv were equipped with more rules, there might be a possibility that changing grammatical representations in my implementation would lead to better performance in a way similar to Flickinger (2000).

Chapter 5 Concluding remarks

5.1 Future work

I have shown throughout the dissertation that my deep linguistic treatment of V_1-V_2 compounds in Japanese is theoretically sound and yet practical from an engineering point of view. However, there are several issues that remain to be discussed, and I take up them in turn in this section.

5.1.1 Problematic cases of V_1-V_2 compounds

UNDERGENERATION

In §3.6.2, I illustrated several counterexamples, and some of them, (136) and (137), were really problematic. As for (136), Matsumoto (1996) and Fukushima (2003) notice that they are formed from V_1 and V_2 in the "mirror image" manner. I repeat (136) as (169).

(169) a. V_1 ⟨ag-src, th, go⟩ + V_2 ⟨*ag-go*, *th*, *src*⟩ = V_1-V_2 ⟨*ag-go*, *th*, *src*⟩

b. *yuzuri-ukeru* (yield-receive) 'inherit'

c. *moosi-ukeru* (say-receive) 'accept the statement of'

d. V_1 ⟨ag-go, th, src⟩ + V_2 ⟨*ag-src*, *th*, *go*⟩ = V_1-V_2 ⟨*ag-src*, *th*, *go*⟩

e. *uke-watasu* (receive-give) 'give away'

As illustrated in (169), arguments between the V_1 and V_2 are co-indexed in reverse order. Unfortunately, none of my implementation for lexical V_1-V_2 compounds can deal with this compounding. In addition, it might be difficult to maintain that these cases are included in non-compositional V_1-V_2s.

Fukushima (2003) accounts for these cases based on thematic proto-roles (Dowty, 1991). But his account would be difficult to implement on JACY since the account seems to rely on the ordering of rule or constraint application, which is incompatible with a constraint-based lexicalist grammar. Besides, I cannot come up with any way to implement a proto-role account on the JACY framework.

In such a case, we have two options. One is that we create new rules and lexical types to handle "mirror image" compounds, regarding them as totally compositional in the same way as other V_1-V_2 compounds such as the Right headed V_1-V_2. The other option is that we consider them as somewhat idiosyncratic in spite of their apparent compositionality, and add to the lexicon new V_1-V_2 lexical items that are given information relevant to their "mirror image" characteristics. Which position to take depends on how productive they are. If their productivity is limited, the latter position would be justifiable in light of **Importance of Phenomena**. Corpus study will tell us which path to take.

Regarding (137), which is repeated as (170), the situation might be different.

(170) a. V_1 ⟨ag, (pt)⟩ + V_2 ⟨*th*⟩ = V_1-V_2 ⟨*th*⟩

b. *aruki-kutabireru* (walk-get.tired) 'get tired from walking'

c. *hasiri-tukareru* (run-get.tired) 'get tired from running'

d. *tatakai-yabureru* (battle-be.ruined) 'fall because of fighting'

e. *nomi-tubureru* (drink-get.out.of.shape) 'pass out from drinking'

This type seems more productive than the "mirror image" compounds, and I can come up with many other V_1-V_2s of the same type.

(171) a. *syaberi-kutabireru* (chat-get.tired) 'get tired from chatting'

b. *warai-kutabireru* (laugh-get.tired) 'get tired from laughing'

c. *asobi-tukareru* (play-get.tired) 'get tired from playing'

d. *hataraki-tukareru* (work-get.tired) 'get tired from working'

Probably I might have to develop the new rule for them. According to Matsumoto (1996), all of the V_2s of this type are unaccusative and subcategorize for a human or animate subject. On top of this, there seems to be a semantic class that includes all of the V_2s in the examples, and I tentatively call it the "fatigue" class. These characteristics of the V_1-V_2s would help develop the new rule, but, in the dissertation, I will not be involved with this issue anymore.

Overgeneration

Not all combinations of V_1 and V_2 exist, even if a theory predicts their existence and they are pragmatically plausible. This is what Fukushima (2003) calls *accidental gaps*. Dealing with them is crucial because of the overgeneration problem of MWEs. For example, my analysis predicts the well-formedness of any V_1-V_2 in which V_1 and V_2 are both *monotrans*, but actually there are several unattested cases in *monotrans-monotrans* V_1-V_2s.

(172) Unattested *monotrans-monotrans* V_1-V_2 compounds

a.**naosi-tukau* (repair-use)

b.**yaburi-moyasu* (tear.apart-burn)

c.**nage-butukeru* (throw-strike)

d.**hakobi-tasukeru* (carry-help)

e.**nagasi-migaku* (flush-polish)

Regarding (172), Kageyama (1993, p.107) observes that a compound verb represents one single event, implying that if it is difficult to regard two situations described by V_1 and V_2 as representing a single event, the V_1-V_2 is deemed as unnatural. Indeed Kageyama's claim makes sense and can rule out the V_1-V_2s in (172), but how can we formalize the idea within a constraint-based lexicalist framework? Probably we can best couch this idea in a pragmatic component of an NLP system, which may well be outside the grammar component.

Below is another kind of accidental gap, which is also incorrectly accepted by my analysis.

(173) Unattested "reverse order" V_1-V_2 compounds

a.**tukare-odoru* (get.tired-dance)

cf. *odori-tukareru* (dance-get.tired) 'get tired from dancing'

b.**wari-tataku* (break.in.half-hit)

cf. *tataki-waru* (hit-break.in.half) 'break in half by hitting'

c.**katame-humu* (harden-tramp)

cf. *humi-katameru* (tramp-harden) 'be tramped hard'

d.**sini-yaku* (die-burn)

cf. *yake-sinu* (burn-die) 'die from burning'

e.**korosi-naguru* (kill-strike)

cf. *naguri-korosu* (strike-kill) 'kill by striking'

These impossible V_1-V_2s have grammatical counterparts, in which the two component verbs are aligned in reverse order. According to Kageyama (1993, p.114), this is due to an implicit, possibly universal, constraint on the ordering of V_1 and V_2 that require V_1 (representing 'cause') come first followed by V_2 (representing 'result'). This constraint also makes sense since it seems to reflect the characteristic of human cognition. Nonetheless, I would say this is also hard to implement on JACY. It seems to depend heavily on world knowledge to decide which verb represents 'cause' or 'result' in what situation. Again, I claim that this constraint should be captured by a pragmatic component of NLP system, rather than within JACY.

Another kind of accidental gap is more difficult to characterize. Examples below are cited from Himeno (1999).

(174) Unattested "synonymous" V_1-V_2 compounds

a.**kangae-egaku* (think-draw)

cf. *omoi-egaku* (think-draw) 'imagine'

b.**nizimi-tooru* (blot-pass)

cf. *simi-tooru* (blot-pass) 'sink in'

c.**kakae-yoseru* (fold-get.together)

cf. *daki-yoseru* (fold-get.together) 'take in one's arm'

d.**turusi-sageru* (hang-lower)

cf. *turi-sageru* (hang-lower) 'dangle'

e.**katari-kaesu* (say-return)

cf. *ii-kaesu* (say-return) 'talk back'

These are mysterious in that the unattested V_1-V_2s above literally mean the same things as their grammatical counterparts, and my analysis, as well as Kageyama (1993) and Matsumoto (1996), incorrectly predict their existence. Himeno (1999) speculates that there might be a subtle semantic difference between each grammatical and ungrammatical pair in (174), but the explanation looks *ad hoc* and difficult to formalize within the JACY framework. It seems to me that this problem is beyond the current scope of linguistics, and thus it poses a real problem not only to linguistics but also to NLP.

The accidental gaps we have seen can be regarded as a kind of *blocking* (Aronoff, 1976). Briscoe et al. (1995) propose the computationally tractable treatment of blocking that is compatible with the framework I adopt in this study.

5.1.2 Automatic detection of non-compositional V_1-V_2 compounds

I classified lexical V_1-V_2 compound into five categories, and one of them was called non-compositional V_1-V_2 and treated as single words. In §3.6.6, I did not mention how we can acquire them automatically, and I just entered them into the lexicon manually in the evaluation experiment in chapter 4. However, developing a way to automatically detect non-compositional V_1-V_2s is indispensable for the purpose of NLP; otherwise we would have to collect them manually, which would be very time-consuming and painstaking. In this section, I describe some studies for detecting non-compositional English phrasal verbs, and discuss its applicability to Japanese V_1-V_2 compounds.

English phrasal verbs, also known as verb particle constructions, consist of a head verb and one or more obligatory particles, in the form of intransitive prepositions, adjectives, or verbs. Examples of phrasal verbs that consist of a head verb and a particle are illustrated in (175). Below I concentrate on this type.

(175) a. *break down*
 b. *find out*
 c. *turn out*
 d. *look up*
 e. *make over*

According to Villavicencio and Copestake (2002), English phrasal verbs have the following characteristics. First, while some of them show compositionality, meanings of other English phrasal verbs cannot be derived from their components. For example, *make out* can mean *to see, hear or understand (something or someone) with difficulty*, as in *What I couldn't make out was your motive* (from the Cambridge International Dictionary of Phrasal Verbs, page 213). Clearly, neither *make* nor *out* has such a meaning. Second, they sometimes show polysemy; *make out* can also mean *to write all the necessary information on (an official form, document, etc.)*, as in *Did you make out a receipt?* (also from the Cambridge International Dictionary of Phrasal Verbs, page 214). Another characteristic of English phrasal verbs involves word order variation.

(176) a. Ken ate up all the dinner.

 b. Ken ate it up.

c. Ken came up with the idea.

d.*Ken came up it with.

As illustrated in (176), while *eat up* allows the object to occur between the verb and particle, *come up with* does not.

Among these characteristics, developing the technique of automatic detection of non-compositional phrasal verbs like *make out* has been extensively explored recently in connection with the study of MWEs (Lin, 1999; Bannard et al., 2003; McCarthy et al., 2003; Baldwin et al., 2003). Roughly speaking, they all make use of statistical techniques that measure the similarity between a phrasal verb and its components. The intuition behind these techniques can be summarized as follows.

1. If a phrasal verb is similar to both the head verb and the particle, it is fully compositional.

2. If a phrasal verb is similar to either the head verb or the particle, it is partially compositional.

3. If a phrasal verb is similar to neither the head verb nor the particle, it is non-compositional.

Similarity is measured according to their co-occurrence patterns. In other words, their meanings are approximated in terms of what subjects, objects, and modifiers these verbs take.

Naturally, we expect that this kind of technique should help us detect non-compositional V_1-V_2 compounds in Japanese, too. Take *tori-simaru* for example. (177) shows the possible co-occurrence words of the V_1-V_2, the V_1, and the V_2.

(177) *tori-simaru* (take-fasten) 'police'

a. **tori-simaru :** *police, crime, drugs, law, . . .*

b. **toru :** *license, balance, nutrition, fatigue, . . .*

c. **simaru :** *belt, tie, screw, tummy, . . .*

This is certainly a non-compositional V_1-V_2 compound as the possible co-occurrence words above indicate.

Previous studies on this problem seem to count any (content) words in measuring similarity, but I speculate that it would be better to count only a word that is an (semantic) argument of the verb in the measurement since arguments would better reflect a verb's meaning. In doing this, we definitely need the distinction between argument and adjunct, which a deep linguistic analysis can provide, while typical NLP parsers that only tell us surface dependency information cannot.

5.1.3 Machine Translation of V_1-V_2 compounds

I have shown that there are several classes in V_1-V_2 compounds in Japanese, and V_1-V_2s in each class behave differently both syntactically and semantically. Hence, a sophisticated linguistic treatment of V_1-V_2 compounds is crucial for an NLP application such as machine translation. So it would be an interesting challenge to develop a machine translation system that is capable of translating V_1-V_2 compounds, say, from Japanese to English, so that we can illustrate the advantage of my approach to V_1-V_2 compounds.

In the field of machine translation, several techniques have been developed, which can roughly be classified into example-based translation, statistical translation, and rule-based translation. The most compatible technique would be the rule-based one, especially the so-called semantic-transfer approach elaborated by Copestake et al. (1995). In this technique, a translation system works on two MRSs, which correspond to the meaning of utterances of source language and that of target language. Usually, MRSs of utterances that represent almost the same meaning are remarkably similar to each other. Look at the MRSs of the Japanese sentence, *Ken-ga hon-o yomu* 'Ken reads a book', and its English counterpart illustrated in the right side of Figures 27 and 28. These are produced by JACY and ERG (Flickinger, 2000), respectively. With the semantic-transfer module, ERG, and JACY, the translation system would be organized as in Figure 29. Because of MRS's independence from a particular language, the semantic transfer module can be concise.

As we have seen, V_1-V_2 compounds should be classified according to their syntactic and semantic properties. This is also indispensable regarding machine translation. That is, different types have different semantic structures. Accordingly, machine translation systems should treat them differently. My treatment presented here certainly plays an important role in the application.

However, in order to translate V_1-V_2 compounds in Japanese precisely, we might need additional machinery that resolves information underspecified by the grammar. Relevant to this is my analysis of lexical V_1-V_2 compounds, in which I underspecify the semantic relations holding between the V_1 and the V_2 of the Right headed V_1-V_2 such as *pair*, *cause*, *manner*, and *means* of Matsumoto (1996), as discussed on page 79. As the result of the read(Ken, book)

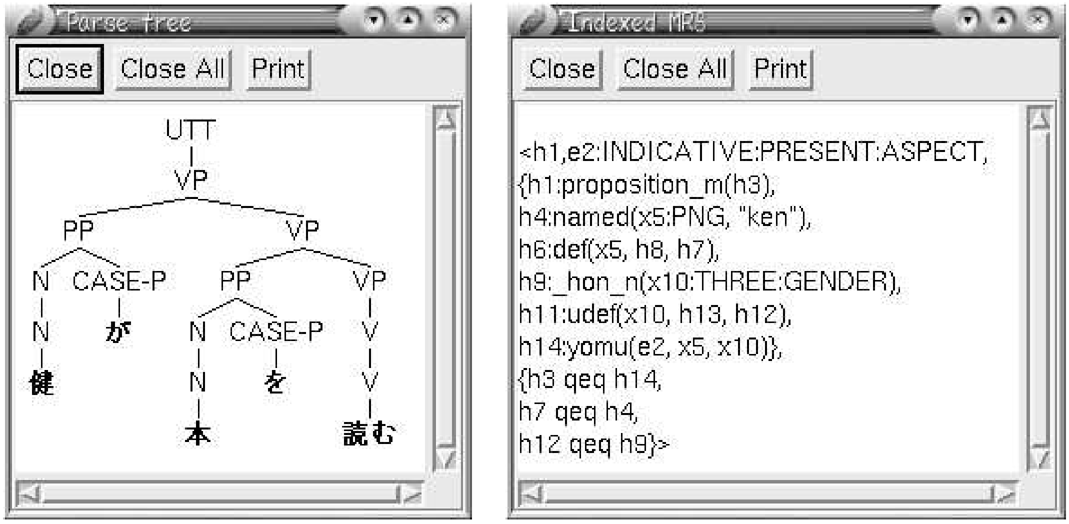


Figure 27: *Ken-ga hon-o yomu*

read(Ken, book)

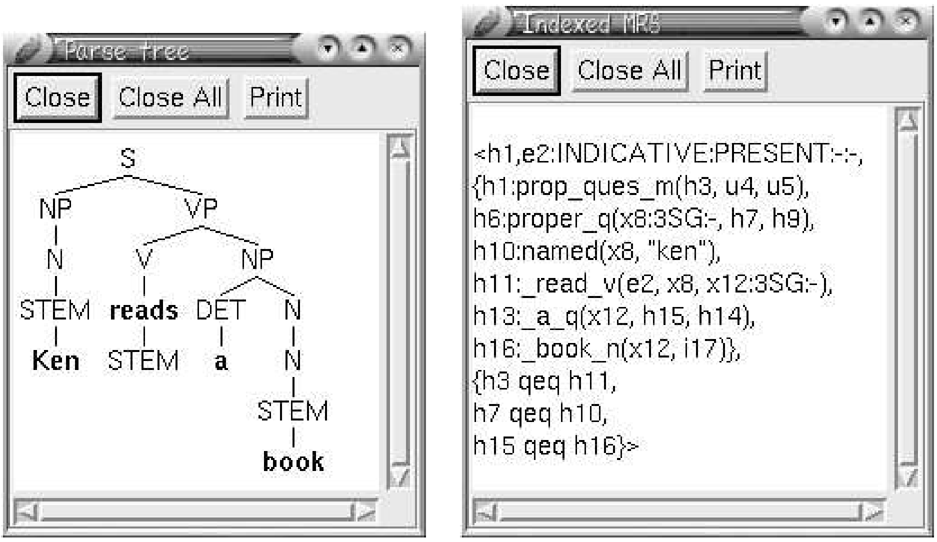


Figure 28: *Ken reads a book*

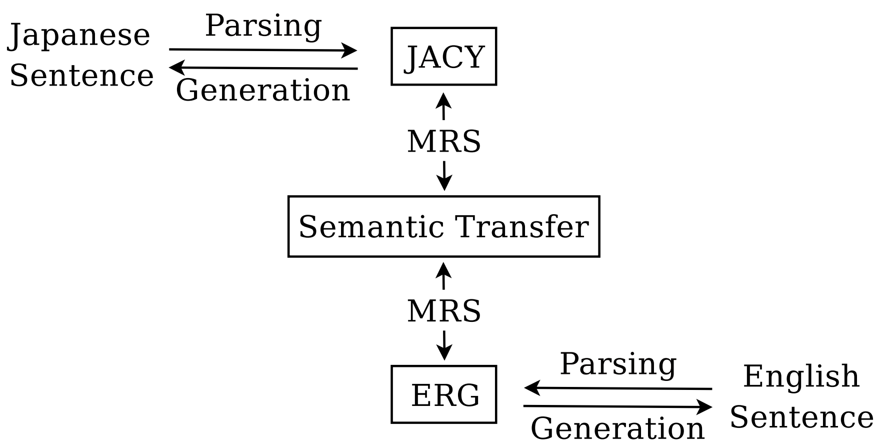


Figure 29: The semantic transfer with JACY and ERG

underspecification, we cannot precisely translate lexical V_1-V_2 compounds without further analysis.

Another relevant problem is exemplified by the lexical V_1-V_2 compounds in (178) and (179), which are discussed by Uchiyama and Baldwin (2003).

(178) a. *nage-ageru* 'throw up'

b. *moti-ageru* 'lift up'

c. *osi-ageru* 'push up'

(179) a. *yude-ageru* 'finish boiling'

b. *musi-ageru* 'finish steaming'

c. *yaki-ageru* 'finish baking'

They all contain *ageru* 'raise' as V_2. My implementation analyzes them as either the Right headed V_1-V_2 (*monotrans-monotrans*), in which case *ageru* would mean direction (*up*), or V_1-V_2 with semantically deverbalized V_2, in which case it would take on aspectual meaning (*finish*). Clearly, the grammar should assign the former analysis to (178), while (179) should be analyzed as the latter case. Uchiyama and Baldwin (2003) tackle this problem by means of NLP techniques.

In any case, a linguistic grammar cannot solve all of the NLP problems and needs to cooperate with NLP techniques. Nevertheless, it should be remembered that, without a deep linguistic treatment, we cannot obtain fine-grained semantic information, which is crucial for a semantic-transfer approach, in the first place.

5.2 Conclusion: the prospect of the two studies of language

I have argued throughout the dissertation that theoretical linguistics or grammatical theory should play an important role in NLP. Indeed, a deep linguistic treatment is indispensable for some NLP problems. Dealing with V_1-V_2 compounds in Japanese is one of them, and my treatment of V_1-V_2 compounds suggests in detail ways in which we can make use of linguistics for NLP.

Finally, I discuss the prospect of the relationship between theoretical linguistics and NLP to conclude the dissertation.

5.2.1 Proper division of labor

You might think that I am totally denying using shallow processing techniques for NLP, but, in fact, I claim that there should be a proper division of labor between linguistic theory and NLP techniques, and we should look for which business is best cared for by which.

Take a treatment of V_1-V_2 compound for example. To handle complicating characteristics of V_1-V_2 compound, I made use of linguistic analyses and observations. Different kind of V_1-V_2 compound has a different syntactic and semantic structure that we must rely on linguistics to account for. It is apparent from the discussion so far that shallow NLP techniques would not give us precise analyses for them. However, we should notice that we cannot replace all existing NLP techniques with linguistic theory. First, as I mentioned previously, it seems to depend on world knowledge to decide what semantic relation holds between V_1 and V_2 of Right headed V_1-V_2. Since linguistics has paid almost no attention to such a knowledge,[1] and, on the contrary, NLP has sought the best way how we characterize the knowledge on computer, deciding the semantic relation should be cared for by NLP techniques. Second, my treatment of V_1-V_2 compound assigns ambiguity to some cases like those in (180) and (181).

(180) *oki-wasureru* (put-forget)

a. 'forget to put (something somewhere)' ... Syntactic C type

b. 'put (something somewhere) and forget to bring it back' ... Right headed V_1-V_2

(181) *osi-dasu* (push-take.out)

a. 'begin to push' ... Syntactic A type

[1] Fillmore (1982) would be a notable exception.

b. ‘push out’ ... Right headed V_1-V_2

These ambiguities are not spurious; which interpretation is preferable should be decided by a contextual or pragmatic factor. Thus, it would be safer to delegate the decision to a NLP component that employs some heuristics or statistical techniques. Third, it goes without saying that linguistics is not concerned with Machine Translation, Automatic Text Summarization, Information Retrieval, and so forth. Hence, in order to build a translation system that is capable of translating V_1-V_2 compounds in Japanese into English, we must rely on something other than linguistics. Look at the examples below.

(182) a. *hataraki-sugiru* (work-run.over) ‘overwork’

b. *moti-ageru* (have-raise) ‘lift up’

c. *yomi-hazimeru* (read-begin) ‘begin to read’

As indicated above, V_1-V_2 compounds can be translated into single words (182a), verb particle constructions (182b), or infinitival constructions (182c). It should be the business of NLP techniques to translate which V_1-V_2 compound into which English construction. For example, Miyamoto et al. (2000) develop rules to translate Japanese compound verbs into English based on the possible combinations of V_1 and V_2. Finally, developing the automatic way of acquiring lexical information from corpora is crucial for NLP. It is not practical to build a large-scale lexicon only by hand. Although one of the most important problems of linguistics is how infants acquire language, such psycholinguistic theories would not be useful since inputs that are available to computer are usually so restricted compared to human babies that we need different assumptions. On the other hand, NLP researchers have explored the techniques eagerly (see Matsumoto and Utsuro (2000) for example). Relevant to this issue is the automatic inference of ARG-ST of a verb, among other things, since my treatment of lexical V_1-V_2 compound exploits ARG-ST. I speculate that various linguistic tests to distinguish argument structure types such as those provided in Kageyama (1993) would be useful for the task.

5.2.2 Airplane or bird?

In theoretical linguistics, there are several issues that have been extensively studied and, at the same time, controversial since Chomsky (1957), which include the universality of language and independence of knowledge of language from other cognitive domains. Sag and Wasow (1999, chapter 9) claims that the arguments have been unconvincing because of the lack of precise formulation of the issues, and that their explicit and precise linguistic theory, namely HPSG, can help clarify and solve the biggest problems in linguistics.

Along the lines of this claim, a deep linguistic treatment of NLP can contribute to the resolution of the issues. First of all, a large-scale, consistent, and explicit grammar, which is essential to the deep linguistic NLP, serves for considering what our knowledge of language looks like. In the grammar development projects that I listed on page 4, grammars are getting more and more comprehensive and precise. We can expect their grammars will be concrete models of human knowledge of language in the near future. Second, in those projects, researchers are developing grammars of several languages in parallel. The DELPH-IN project (Oepen et al., 2002), for example, have implemented English, German, Italian, Norwegian, Japanese, and Greek grammars so far. In addition, some of those projects are trying to extract commonalities among grammars (see Bender et al. (2002) for example). It is reasonable to regard the attempt as the computational approach to "Universal Grammar." Third, the ultimate goal of NLP or computational linguistics should be to establish the comprehensive computational model of natural language understanding. The model would include huge quantities of world knowledge, statistical information, several heuristics that are cognitively sound, and a large-scale linguistic grammar. Naturally, the grammar that is embedded in the model must be "realistic" in the sense of Sag and Wasow (1999). According to Sag and Wasow, a theory of grammar is realistic if it plays a role in explaining many features of linguistic performance, which undoubtedly includes natural language understanding. Probably it is not until we become familiar with the entire organization of natural language understanding that we are ready to attack the biggest issues of theoretical linguistics.

There is a famous metaphor in the field of artificial intelligence in which human beings are likened to birds while intelligent computational systems are compared to airplanes. Some computer scientists say that we do not have to develop "bird" because "airplane" will do, implying that in order to make computers behave as intelligently as human beings, we need not worry about how our brain works. To be sure, this is the right track as engineering, and it will take long time to achieve a success in NLP in the "bird" manner, but it must be fruitful to work on developing "bird" for both NLP and linguistics since human brain would be goldmine of a new information processing paradigm and give us a clue as to the biggest linguistic issues. Perhaps it might turn out to be impossible, but for the meanwhile, I insist that we should seek for "bird" or at least "sophisticated machine bird."

Appendix A Grammar source code

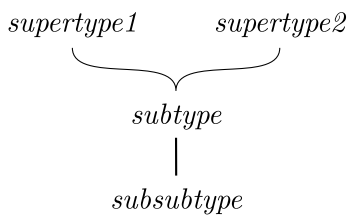
In this appendix, I present the grammar source codes of JACY that are relevant to the discussion of this dissertation. First of all, I take up a simple example that facilitates the understanding of grammar source code.[1]

EXAMPLE

```
subtype := supertype1 & supertype2 &
 [SYNSEM.LOCAL.CONT #tag & [INDEX index],
  HEAD-DTR sign & [SYNSEM.LOCAL [CONT #tag]]].

subsubtype := subtype &
 [SYNSEM.LOCAL.CONT.INDEX individual].
```

As indicated by (183), `subtype` inherits constraints from `supertype1` and `supertype2`. Likewise, `subsubtype` inherits constraints from `subtype`. These inheritance relations are represented by `:=`. `subtype` and `subsubtype` are further specified as having the constraints illustrated in (184) and (185).

(183)


(184) $\textit{subtype}\begin{bmatrix} \text{SYNSEM}|\text{LOCAL}|\text{CONT } \boxed{tag}\begin{bmatrix}\text{INDEX } \textit{index}\end{bmatrix} \\ \text{HEAD-DTR } \textit{sign}\begin{bmatrix}\text{SYNSEM}|\text{LOCAL }\begin{bmatrix}\text{CONT } \boxed{tag}\end{bmatrix}\end{bmatrix} \end{bmatrix}$

(185) $\textit{subsubtype}\begin{bmatrix}\text{SYNSEM}|\text{LOCAL}|\text{CONT}|\text{INDEX } \textit{individual}\end{bmatrix}$

Notice that `#tag` is an identification tag and that `subsubtype` further specifies `subtype`'s `INDEX` value as `individual`.

[1] However, in order to fully understand the grammar source codes in this appendix, you might have to be familiar with the $\mathcal{TDL}$ language and the entire organization of JACY. Krieger and Schafer (1994) and Copestake (2002) would help understand the $\mathcal{TDL}$ language. For the full source code of my version of JACY, namely JACY-vv, visit the following URL.
`http://sils.shoin.ac.jp/~chashi/jacy-vv.tgz`

A.1 Basics

I first present the codes of HPSG basics that I showed in §3.3: the head-complement rule, the head-subject rule, the head-specifier rule, the head feature principle, the valence principle, and the semantic compositionality principle. Almost all of them are developed by Melanie Siegel and Emily Bender.

There are several head-complement rules in JACY, although I take up only one, `head--complement-hf-rule`. The other three, `basic-head-complement-type`, `head-complement--type`, `head-complement-hf-type`, are the supertypes that generalize the commonalities among the head-complement rules.

HEAD-COMPLEMENT RULE

```
basic-head-complement-type := orth-princ &
                              que-princ &
                              adjacent_nonhead_check &
                              affix-list &
 [SYNSEM [MODIFIED.PERIPH #p,
          LEX -,
          LOCAL [CONT [MOD-IND #ind,
                       MOD-HAND #top],
                 NUCL #nucl]],
  C-CONT [RELS <! !>,
          HCONS <! !>],
  NON-HEAD-DTR j-sign & [J-NEEDS-AFFIX -,
                         SYNSEM.LOCAL.CAT.HEAD.EMPTY #empt],
  HEAD-DTR j-sign & [J-NEEDS-AFFIX -,
                     SYNSEM [MODIFIED.PERIPH #p,
                             LOCAL [CAT.HEAD.EMPTY #empt,
                                    CONT [MOD-IND #ind,
                                          MOD-HAND #top],
                                    NUCL #nucl]]]]].


head-complement-type := basic-head-complement-type &
                        scp-obj &
                        adjacent_objbind_check.


head-complement-hf-type := head-complement-type & head-final-type.


head-complement-hf-rule := head-complement-hf-type &
 [SYNSEM [LOCAL.BAR +,
          NON-LOCAL.AFFIX #afflist],
  HEAD-DTR.SYNSEM [LOCAL [CAT.HEAD final_head,
                          BAR +],
                   NON-LOCAL.AFFIX <! !>],
  NON-HEAD-DTR.SYNSEM j-non-argument-attracted-synsem &
         [NON-LOCAL.AFFIX #afflist,
          LOCAL.BAR +]].
```

There is only one head-subject rule in JACY, namely, `head_subj_rule`, whose supertype is `head-subject-rule`.

HEAD-SUBJECT RULE

```
head-subject-rule := head-final-type &
                     orth-princ &
                     scp-sbj &
                     que-princ &
                     adjacent_subjbind_check &
                     adjacent_nonhead_check &
                     affix-list &
 [J-NEEDS-AFFIX -,
  SYNSEM [LOCAL [CONT [MOD-IND #ind,
                       MOD-HAND #top],
                 CAT.POSTHEAD +],
          LEX -],
  C-CONT [RELS <! !>,
          HCONS <! !>],
  HEAD-DTR [J-NEEDS-AFFIX -,
            SYNSEM [LOCAL [CONT [MOD-IND #ind,
                                 MOD-HAND #top],
                           BAR +]]]].

head_subj_rule := head-subject-rule & [SYNSEM.LOCAL.BAR +].
```

The head-specifier rule is formulated in a similar way to the head-subject rule.

HEAD-SPECIFIER RULE

```
head-specifier-rule-type :=  head-final-type &
                             orth-princ &
                             scp-spec &
                             que-princ &
                             affix-list &
 [C-CONT [RELS <! !>,
          HCONS <! !>],
  SYNSEM [LOCAL.CONT.HOOK #hook],
  RMORPH-BIND-TYPE #rmorph,
  HEAD-DTR [SYNSEM [LOCAL [CAT.HEAD #head,
                           CONT #cont & [HOOK #hook]],
                    MODIFIED.PERIPH #per],
            RMORPH-BIND-TYPE #rmorph],
  NON-HEAD-DTR.SYNSEM.LOCAL.CAT.HEAD.J-SPEC < [LOCAL [CAT.HEAD #head,
                                                      CONT #cont],
                                               MODIFIED.PERIPH #per] > ].

head-specifier-rule := head-specifier-rule-type &
```

```
                     adjacent-spr-check &
 [SYNSEM [LOCAL [BAR #bar,
                CAT.VAL [UNSAT -,
                         SPR null],
                CONT [HOOK.XARG #xarg,
                      MOD-IND #mind,
                      MOD-HAND #mhand]],
         LEX -,
         MODIFIED.PERIPH #per],
  ARGS <[SYNSEM j-synsem & [MODIFIED.PERIPH #per],
        J-NEEDS-AFFIX -],
       [SYNSEM j-synsem & [LOCAL [BAR #bar,
                                  CONT [HOOK.XARG #xarg,
                                        MOD-IND #mind,
                                        MOD-HAND #mhand]]]]>].
```

The head feature principle is implemented as the type named **headed-phrase**. Any phrase that the head feature principle applies to must (directly or indirectly) inherit this type.

HEAD FEATURE PRINCIPLE

```
headed-phrase := phrase &
 [ ROOT -,
   SYNSEM.LOCAL [ CAT.HEAD head & #head,
                  AGR #agr ],
   HEAD-DTR.SYNSEM.LOCAL local &
        [ CAT.HEAD #head,
          AGR #agr ],
   NON-HEAD-DTR sign ].
```

JACY deals with the valence principle in a way different from Sag and Wasow (1999). JACY posits the three types, `scp-sbj`, `scp-obj`, and `scp-spec`, to emulate the principle.

VALENCE PRINCIPLE

```
scp-sbj := argument-binding-princ  &
 [SYNSEM.LOCAL [CAT.VAL [SUBJ null,
                         COMPS #comps,
                         SPR #spr,
                         UNSAT - ]],
  HEAD-DTR.SYNSEM.LOCAL.CAT [HEAD.FORMAL.SHON #shon,
                             VAL [SUBJ 1-list &
                                       [FIRST #1 &
                                       [LOCAL.CAT.HEAD.FORMAL.SHON #shon]],
                                  COMPS #comps,
                                  SPR #spr]],
  NON-HEAD-DTR.SYNSEM #1].
```

```
scp-obj :=  argument-binding-princ &
 [SYNSEM.LOCAL.CAT.VAL [SUBJ #sbjval,
                        COMPS 0-1-list & #obj2val,
                        SPR #sprval,
                        UNSAT - ],
  HEAD-DTR.SYNSEM.LOCAL.CAT.VAL [COMPS cons & [FIRST #1 & [OPT bool],
                                               REST #obj2val ],
                                 SUBJ #sbjval,
                                 SPR #sprval],
  NON-HEAD-DTR.SYNSEM #1].


scp-spec := argument-binding-princ &
 [SYNSEM.LOCAL.CAT.VAL [SUBJ #sbjval,
                        SPR 0-1-list,
                        COMPS #comps],
  HEAD-DTR.SYNSEM.LOCAL.CAT.VAL [SPR 1-list & [FIRST #1],
                                 SUBJ #sbjval,
                                 COMPS #comps],
  NON-HEAD-DTR.SYNSEM #1].
```

The semantic compositionality principle is implemented in the types named `basic--unary-phrase` and `basic-binary-phrase`. Notice that a phrase's (or construction's) meaning itself is also included in the semantics of the phrase as indicated by the `C-CONT` (construction's content) feature.

SEMANTIC COMPOSITIONALITY PRINCIPLE

```
basic-unary-phrase := phrase &
 [ STEM #stem,
   SYNSEM.LOCAL.CONT [ RELS [ LIST #first,
                              LAST #last ],
                       HCONS [ LIST #scfirst,
                               LAST #sclast ] ],
   C-CONT [ RELS [ LIST #middle,
                   LAST #last ],
            HCONS [ LIST #scmiddle,
                    LAST #sclast ] ],
   ARGS < sign & [ STEM #stem,
                   SYNSEM.LOCAL local &
                         [ CONT [ RELS [ LIST #first,
                                         LAST #middle ],
                                  HCONS [ LIST #scfirst,
                                          LAST #scmiddle ] ] ],
                   ROOT - ] > ].


basic-binary-phrase := phrase &
 [ SYNSEM.LOCAL.CONT [ RELS [ LIST #first,
                              LAST #last ],
                       HCONS [ LIST #scfirst,
```

```
                              LAST #sclast ] ],
   C-CONT [ RELS [ LIST #middle2,
                   LAST #last ],
            HCONS [ LIST #scmiddle2,
                    LAST #sclast ] ],
   ARGS < sign & [ SYNSEM.LOCAL local &
                               [ CONT [ RELS [ LIST #first,
                                               LAST #middle1 ],
                                        HCONS [ LIST #scfirst,
                                                LAST #scmiddle1 ] ] ],
                   ROOT - ],
           sign & [ SYNSEM.LOCAL local &
                                [ CONT [ RELS [ LIST #middle1,
                                                LAST #middle2 ],
                                         HCONS [ LIST #scmiddle1,
                                                 LAST #scmiddle2 ] ] ],
                    ROOT - ] > ].
```

As for the semantic inheritance principle, namely the identify of INDEX value between a head daughter and a mother, JACY treats the principle in a bit redundant way. In the head-complement rule, the head-subject rule, and the head-specifier rule, the identify of the two `MOD-IND` values between the head daughter and the mother represents the principle.

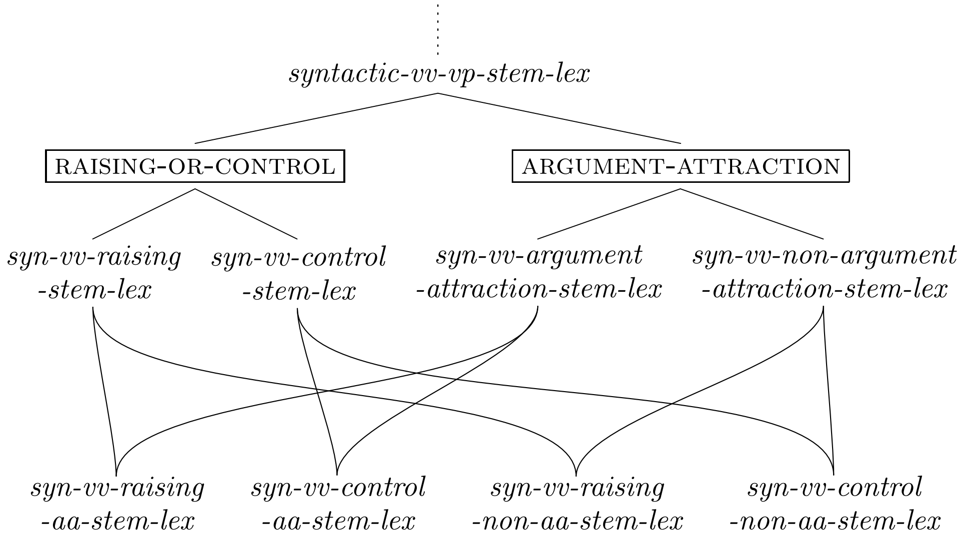
A.2 Syntactic V_1-V_2 compound

In this section, I present the source codes that embody my analysis of syntactic V_1-V_2 compounds. That is, it is shown how the V_2s of syntactic V_1-V_2 compounds are implemented.

A.2.1 VP embedding structure

There are two kinds of VP embedding structure in my analysis: the A type (the control type) and the B type (the raising type). In addition, they are classified into either Argument Attracted or non Argument Attracted. (186) is the (simplified) hierarchy of V_2s of VP embedding structures.

(186)



syntactic-vv-vp-stem-lex is the supertype for the V_2s of both *syn-vv-raising-stem-lex* (the A type) and *syn-vv-control-stem-lex* (the B type). *syn-vv-argument-attraction-stem-lex* and *syn-vv-non-argument-attraction-stem-lex* represent Argument Attracted phrases and non Argument Attracted phrases, respectively. There are four subtypes, *syn-vv-raising-aa-stem-lex*, *syn-vv-control-aa-stem-lex*, *syn-vv-raising-non-aa-stem-lex*, and *syn-vv-control-non-aa-stem-lex* in the hierarchy, according to the two dimensions, RAISING-OR-CONTROL and ARGUMENT-ATTRACTION. In addition, Argument Attracted V_2s are derived by means of a lexical rule. With these in mind, let us look at the implementations of these types in turn.

In the following implementation of *syntactic-vv-vp-stem-lex*, notice that an embedded VP (`SPR`) is intended to mean a proposition (`#ohand`=`#soa`).

SYNTACTIC-VV-VP-STEM-LEX

```
syntactic-vv-vp-stem-lex := verb-stem-lex &
                            lexical_sign-word &
 [SYNSEM [LOCAL [CAT [HEAD syn-vv-vp-stem_head & [H-TENSE #tense],
                      VAL [SUBJ.FIRST #sbj &
                                      [LOCAL [CAT.HEAD case-p_head &
                                                       [CASE ni-or-ga]]],
                           SPR.FIRST.LOCAL.CONT.HOOK.LTOP #ohand]],
                 CONT [HOOK.INDEX #event & [E [TENSE #tense]],
                       HCONS <! qeq & [LARG #ohand, HARG #soa] !>,
                       RELS diff-list & <! #key, message &
                                             [PRED proposition_m_rel,
                                              MARG #soa] !>],
                 ARG-S < #sbj >],
```

```
          LKEYS.KEYREL #key & [ARG0 #event],
          NON-LOCAL.QUE <! !>,
          MODIFIED.PERIPH +]].
```

As discussed in §3.5.1, the A type (raising) and the B type (control) are different in that the V_2 of the latter, but not the former, assigns a thematic role to the subject, as indicated below.

SYN-VV-RAISING-STEM-LEX

```
syn-vv-raising-stem-lex := syntactic-vv-vp-stem-lex &
 [SYNSEM [ LOCAL [ CONT.RELS <! #key, [ LBL #chand ] !> ],
           LKEYS.KEYREL #key & arg1-relation & [ ARG1 #chand ]]].
```

SYN-VV-CONTROL-STEM-LEX

```
syn-vv-control-stem-lex := syntactic-vv-vp-stem-lex &
 [SYNSEM [ LOCAL [ CAT.VAL.SUBJ < [ LOCAL.CONT.HOOK.INDEX #sbjind ] >,
                   CONT.RELS <! #key, [ LBL #chand ] !> ],
           LKEYS.KEYREL #key & arg12-relation &
                 [ ARG1 #sbjind,
                   ARG2 #chand ]]].
```

An Argument Attracted V_2, *syn-vv-argument-attraction-stem-lex*, and a non Argument Attracted V_2, *syn-vv-non-argument-attraction-stem-lex*, show a difference involving a VALENCE type. In Argument Attraction, the object in an embedded VP is "attracted" to the matrix object position. The identity tag, `#obj`, in `aspect-argument-attraction--obj_transitive` and the matrix object's `[OPT -]`, which means that the matrix object is obligatory, represent this.

SYN-VV-ARGUMENT-ATTRACTION-STEM-LEX

```
syn-vv-argument-attraction-stem-lex := syntactic-vv-vp-stem-lex.


syn-vv-aa-obj-stem-lex := syn-vv-argument-attraction-stem-lex &
 [SYNSEM.LOCAL.CAT.VAL aspect-argument-attraction-obj_transitive].


aspect-argument-attraction_transitive := j-valence &
 [SPR obl-1-arg &
      < [LOCAL [CAT [HEAD  verb_head & [MAIN-PRD -, COP -,
                                        VN -, AUX -],
```

```
                     VAL  [SUBJ <[LOCAL #sbj]>]],
                NUCL nucl_plus]] >,
  SUBJ <[LOCAL #sbj, OPT -]>].


aspect-argument-attraction-obj_transitive
 := aspect-argument-attraction_transitive &
 [SPR <[LOCAL.CAT.VAL.COMPS <[LOCAL #obj]>]>,
  COMPS < j-argument-attracted-synsem & [LOCAL #obj, OPT -]>].
```

On the other hand, in non Argument Attraction environments, the object inside an embedded VP must stay inside the VP. `#obj` and `<>` of the matrix `COMPS` and that of the embedded VP indicate this. That is, there must not be a matrix object and an embedded VP must be saturated with an object.

SYN-VV-NON-ARGUMENT-ATTRACTION-STEM-LEX

```
syn-vv-non-argument-attraction-stem-lex := syntactic-vv-vp-stem-lex &
 [SYNSEM.LOCAL.CAT.VAL aspect_transitive].


aspect_transitive := j-valence &
 [SPR obl-1-arg & < [LOCAL [CAT [HEAD  verb_head &
                                       [MAIN-PRD -, COP -,
                                        VN -, AUX -],
                                 VAL  [SUBJ  #sbj,
                                       COMPS #obj]],
                            NUCL nucl_plus]] >,
  SUBJ #sbj,
  COMPS #obj & <>].
```

Following are the four subtypes: *syn-vv-raising-aa-obj-stem-lex*, *syn-vv-raising-non-aa-obj-stem-lex*, *syn-vv-control-aa-obj-stem-lex*, and *syn-vv-control-non-aa-obj-stem-lex*.

SYN-VV-RAISING-AA-STEM-LEX

```
syn-vv-raising-aa-obj-stem-lex := syn-vv-raising-stem-lex &
                                  syn-vv-aa-obj-stem-lex.
```

SYN-VV-RAISING-NON-AA-STEM-LEX

```
syn-vv-raising-non-aa-stem-lex := syn-vv-raising-stem-lex &
                                  syn-vv-non-argument-attraction-stem-lex.
```

SYN-VV-CONTROL-AA-STEM-LEX

```
syn-vv-control-aa-obj-stem-lex := syn-vv-control-stem-lex &
                                  syn-vv-aa-obj-stem-lex.
```

SYN-VV-CONTROL-NON-AA-STEM-LEX

```
syn-vv-control-non-aa-stem-lex := syn-vv-control-stem-lex &
                                  syn-vv-non-argument-attraction-stem-lex.
```

Two lexical rules that derive Argument Attracted V_2s from non Argument Attracted counterparts are described below.

RAISING-ARGUMENT-ATTRACTION-OBJ-V-LRULE

```
raising-argument-attraction-obj-v-lrule
 := syn-vv-raising-aa-obj-v-stem-lex &
[ ORTH #orth,
  SYNSEM.LKEYS [KEYREL [PRED #pred]],
  ARGS < syn-vv-raising-non-aa-v-stem-lex
         & [ ORTH #orth,
             SYNSEM.LKEYS [KEYREL [PRED #pred]]]  >].
```

CONTROL-ARGUMENT-ATTRACTION-OBJ-V-LRULE

```
control-argument-attraction-obj-v-lrule
 := syn-vv-control-aa-obj-v-stem-lex &
[ ORTH #orth,
  SYNSEM.LKEYS [KEYREL [PRED #pred]],
  ARGS < syn-vv-control-non-aa-v-stem-lex
         & [ ORTH #orth,
             SYNSEM.LKEYS [KEYREL [PRED #pred]] ]  >].
```

Below are the V_2 of the A type, *-kakeru*, and the B type V_2, *-sokoneru.*

-KAKERU

```
kakeru-hiragana-syn-vv-raising-non-aa-stem
 := syn-vv-raising-non-aa-v-stem-lex &
[SYNSEM.LKEYS [KEYREL [PRED 'kakeru_rais]],
 ORTH <! "かける" !> ].
```

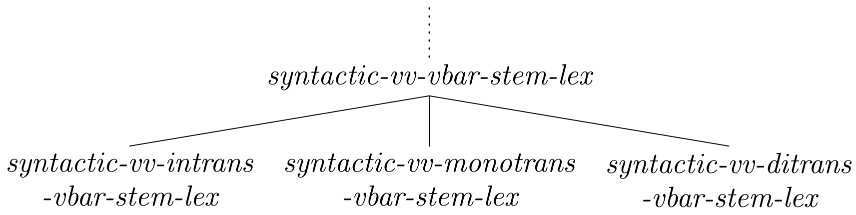
-SOKONERU

```
sokoneru-syn-vv-control-non-aa-stem
 := syn-vv-control-non-aa-v-stem-lex &
[SYNSEM.LKEYS [KEYREL [PRED 'sokoneru_ctrl]],
 ORTH <! "損ねる" !> ].
```

A.2.2 V embedding structures

The C type (Kageyama's (1993) $\overline{\mathrm{V}}$ complementation type) has a V embedding structure. (187) shows the hierarchy of V_2 of the C type, *syntactic-vv-vbar-stem-lex*.

(187)



As indicated in (187), *syntactic-vv-vbar-stem-lex* is classified into three subtypes according to transitivity. *syntactic-vv-monotrans-vbar-stem-lex* and *syntactic-vv-ditrans-vbar-stem-lex* are derived from *syntactic-vv-intrans-vbar-stem-lex* by means of monotransitivization or ditransitivization lexical rules.

All *syntactic-vv-vbar-stem-lex*s assign a thematic role to the subject. `[ARG1 #sbjind]` in `syntactic-vv-vbar-stem-lex` indicates this. `syn-vv-vbar-stem_transitive` is the VALENCE type for *syntactic-vv-vbar-stem-lex*. The VALENCE type requires that all complements of an embedded V should be structure-shared by matrix complement(s), which is indicated by `#comps`.

SYNTACTIC-VV-VBAR-STEM-LEX

```
syntactic-vv-vbar-stem-lex := verb-stem-lex &
                              lexical_sign-word &
 [SYNSEM [LOCAL [CAT [HEAD syn-vv-vbar-stem_head & [H-TENSE #tense],
                      VAL syn-vv-vbar-stem_transitive &
                           [SUBJ.FIRST #sbj &
                                       [LOCAL [CAT.HEAD case-p_head &
                                                        [CASE ni-or-ga],
                                               CONT.HOOK.INDEX #sbjind]],
                            SPR.FIRST.LOCAL.CONT.HOOK.LTOP #ohand]],
                 CONT [HOOK [INDEX #event & [E [TENSE #tense]],
                             XARG #sbjind],
```

```
                    HCONS <! qeq & [LARG #ohand,
                                    HARG #soa] !>,
                    RELS diff-list & <! #key, message &
                                       [PRED proposition_m_rel,
                                        MARG #soa] !>],
              ARG-S < #sbj >],
         LKEYS.KEYREL #key & arg1-relation &
               [ARG1 #sbjind,
                ARG0 #event],
         NON-LOCAL.QUE <! !>,
         MODIFIED.PERIPH +]].
```

```
syn-vv-vbar-stem_transitive := j-valence &
 [SPR  obl-1-arg &
       < [LOCAL [CAT [HEAD verb-stem_head & [MAIN-PRD -],
                      VAL [SUBJ #sbj,
                           COMPS #comps]],
                BAR +],
         LEX +] >,
  SUBJ #sbj,
  COMPS #comps].
```

syntactic-vv-intrans-vbar-stem-lex, *syntactic-vv-monotrans-vbar-stem-lex*, and *syntactic-vv-ditrans-vbar-stem-lex* are shown below. Naturally, the *syntactic-vv-intrans-vbar-stem-lex* has no object, as indicated by `[COMPS null]`, while the *syntactic-vv-monotrans-vbar-stem-lex* has one object (`[COMPS 1-list]`), and the *syntactic-vv-ditrans-vbar-stem-lex* has two objects (`[COMPS 2-comps-list]`).

SYNTACTIC-VV-INTRANS-VBAR-STEM-LEX

```
syntactic-vv-intrans-vbar-stem-lex := syntactic-vv-vbar-stem-lex &
 [SYNSEM [LOCAL [CONT.RELS <! [], [LBL #chand] !>,
                 CAT.VAL.SPR <[LOCAL.CAT.VAL [COMPS null]]>],
          LKEYS.KEYREL arg12-relation &
                [ARG2 #chand]]].
```

SYNTACTIC-VV-MONOTRANS-VBAR-STEM-LEX

```
syntactic-vv-monotrans-vbar-stem-lex := syntactic-vv-vbar-stem-lex &
 [SYNSEM [LOCAL [CAT [VAL [COMPS <[LOCAL.CONT.HOOK.INDEX #objind]>,
                           SPR.FIRST.LOCAL.CAT.VAL.COMPS 1-list]],
                 CONT.RELS <! [], [LBL #chand] !>],
          LKEYS.KEYREL arg123-relation &
                [ARG2 #objind,
                 ARG3 #chand]]].
```

SYNTACTIC-VV-DITRANS-VBAR-STEM-LEX

```
syntactic-vv-ditrans-vbar-stem-lex := syntactic-vv-vbar-stem-lex &
 [SYNSEM [LOCAL [CAT [VAL [COMPS <[LOCAL.CONT.HOOK.INDEX #objind],
                                  [LOCAL.CONT.HOOK.INDEX #obj2ind]>,
                           SPR.FIRST.LOCAL.CAT.VAL.COMPS 2-comps-list]],
                 CONT.RELS <! [], [LBL #chand] !>],
          LKEYS.KEYREL arg1234-relation &
                [ARG2 #objind,
                 ARG3 #obj2ind,
                 ARG4 #chand]]].
```

Following are lexical rules that derive *syn-vv-monotrans-vbar-v-stem-lex* and *syn-vv-ditrans-vbar-v-stem-lex* from *syn-vv-intrans-vbar-v-stem-lex*.

VBAR-MONOTRANSITIVIZATION-V-LEXICAL-RULE

```
vbar-monotransitivization-v-lrule := syn-vv-monotrans-vbar-v-stem-lex &
 [ORTH #orth,
  SYNSEM.LKEYS [KEYREL [PRED #pred]],
  ARGS < syn-vv-intrans-vbar-v-stem-lex &
         [ ORTH #orth,
           SYNSEM.LKEYS [KEYREL [PRED #pred]]] >].
```

VBAR-DITRANSITIVIZATION-V-LEXICAL-RULE

```
vbar-ditransitivization-v-lrule := syn-vv-ditrans-vbar-v-stem-lex &
 [ORTH #orth,
  SYNSEM.LKEYS [KEYREL [PRED #pred]],
  ARGS < syn-vv-intrans-vbar-v-stem-lex &
         [ ORTH #orth,
           SYNSEM.LKEYS [KEYREL [PRED #pred]]] >].
```

The V_2 of the C type, *-naosu*, is illustrated below.

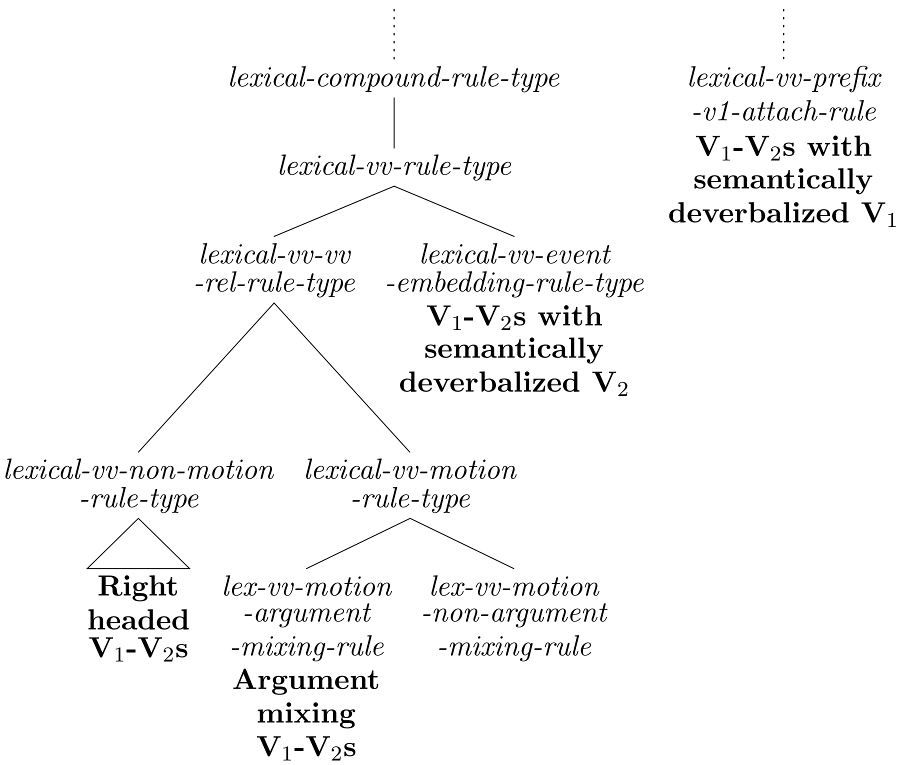
-NAOSU

```
naosu-syn-vv-intrans-vbar-stem := syn-vv-intrans-vbar-c-stem-lex &
[SYNSEM.LKEYS [KEYREL [PRED 'naosu_vemb]],
 ORTH <! "直す" !> ].
```

A.3 Lexical V_1-V_2 compound

In this section, the treatment of lexical V_1-V_2 compound is presented. As discussed in §3.6, lexical V_1-V_2 compounds are classified into five kinds, four of which are (partially) compositional: **Right headed V_1-V_2s**, **Argument mixing V_1-V_2s**, **V_1-V_2s with semantically deverbalized V_1**, and **V_1-V_2s with semantically deverbalized V_2**. They are organized into the type hierarchy as (188).

(188)



I developed *lexical-compound-rule-type* that is intended to be the supertype of all kinds of the Pseudo Lexical Rules (see §3.6, page 80). *lexical-vv-rule-type*, the subtype of *lexical-compound-rule-type*, is the supertype of all lexical V_1-V_2 compounds other than V_1-V_2s with semantically deverbalized V_1 and non-compositional V_1-V_2s. The *lexical-vv-rule-type* is inherited by *lexical-vv-vv-rel-rule-type* and *lexical-vv-event-embedding-rule-type*. The former represents V_1-V_2s in which one of the four semantic relations, *pair*, *cause*, *manner*, and *means*, can hold between the V_1 and the V_2, that is, either the Right headed V_1-V_2s or the Argument mixing V_1-V_2s. The latter is the implementation of the V_1-V_2s with semantically deverbalized V_2. *lexical-vv-non-motion-rule-type* and *lexical-vv-motion-rule-type* are the subtypes of *lexical-vv-vv-rel-rule-type*. The former is inherited by the 29 rules of Right headed V_1-V_2s (see Table 4 on page 85). The latter type consists of those in which a V_2 is a motion verb. *lexical-vv-motion-argument-mixing-rule* represents the Argument mixing V_1-V_2s, in which the V_1 is not a motion verb. On the other hand, the V_1 of *lexical-vv-motion-non-argument-mixing-rule* is a motion verb, and is treated as one of the Right headed V_1-V_2s (see page 93 and Figure 16). V_1-V_2s with semantically deverbalized V_1 is implemented as *lexical-vv-prefix-v1-attach-rule*, which is not a subtype of the *lexical-compound-rule-type*. This is because JACY already contains several prefix rules and I designed the *lexical-vv-prefix-v1-attach-rule* in the similar way.

A.3.1 General rule types

I first illustrate general rule types: *lexical-compound-rule-type*, *lexical-vv-rule-type*, *lexical-vv-vv-rel-rule-type*, and *lexical-vv-motion-rule-type*.

The *lexical-compound-rule-type* is intended to be the compounding lexical rule version of the semantic compositionality principle, and requires that two components of the compounding should be a lexical category rather than phrasal.[2]

LEXICAL-COMPOUND-RULE-TYPE

```
lexical-compound-rule-type := orth-princ &
                              affix-list &
                              que-princ &
 [SYNSEM [LOCAL [BAR -,
                CTXT.BACKGROUND diff-list & [LIST #1, LAST #2],
                CONT [HOOK [LTOP #top,
                            XARG #xarg],
                      MOD-IND #mod-ind,
                      MOD-HAND #mod-hand,
                      MSG #ms,
                      RELS [LIST #firstliszt,
                            LAST #lastliszt],
                      HCONS [LIST #scfirst,
                             LAST #sclast]]],
         NON-LOCAL [REL #rel],
         LEX +],
  C-CONT [RELS diff-list & [LIST #middle2liszt,
                           LAST #lastliszt],
         HCONS diff-list & [LIST #scmiddle2,
                            LAST #sclast]],
  ARGS <[SYNSEM [LOCAL [BAR -,
                       CTXT.BACKGROUND [LIST #3, LAST #2],
                       CONT [RELS [LIST #middle1liszt,
                                   LAST #middle2liszt],
                             HCONS [LIST #scmiddle1,
                                    LAST #scmiddle2]]],
                LEX +]],
        [SYNSEM [LOCAL [BAR -,
                       CTXT.BACKGROUND [LIST #1, LAST #3],
```

[2]The ARGS list represents constituents. In the case of lexical V_1-V_2 compound, two elements in the ARGS list correspond to a V_1 and a V_2.

```
                    CONT [HOOK [LTOP #top,
                                XARG #xarg],
                          MOD-IND #mod-ind,
                          MOD-HAND #mod-hand,
                          MSG #ms,
                          RELS [LIST  #firstliszt,
                                LAST  #middle1liszt],
                          HCONS [LIST #scfirst,
                                 LAST #scmiddle1]]],
                 NON-LOCAL [REL #rel],
                 LEX +]]>].
```

The *lexical-vv-rule-type* constrains the two components of compounding to verbs, although that is not explicitly stated below.

LEXICAL-VV-RULE-TYPE

```
lexical-vv-rule-type := lexical-compound-rule-type &
 [INFLECTED #infl,
  J-NEEDS-AFFIX #aff,
  RMORPH-BIND-TYPE #rmorphbind,
  ARGS <[SYNSEM [ LOCAL [CAT [HEAD [EMPTY -],
                              VAL [UNSAT +,
                                   SUBJ <[LOCAL.CAT.HEAD.CASE ga]>]]]]],
         RMORPH-BIND-TYPE #im & i-morph],
        [SYNSEM [LOCAL [CAT [HEAD [EMPTY -],
                             VAL.UNSAT +]]],
         INFLECTED #infl,
         J-NEEDS-AFFIX #aff,
         LMORPH-BIND-TYPE #im,
         RMORPH-BIND-TYPE #rmorphbind]>].
```

The *lexical-vv-vv-rel-rule-type*, which deals with Right headed V_1-V_2s and Argument mixing V_1-V_2s, introduces the semantic relation between two component verbs, `vv-relation`. Some pragmatic component might be further specify the relation as one of the four semantic relations: *pair*, *cause*, *manner*, and *means*.

LEXICAL-VV-VV-REL-RULE-TYPE

```
lexical-vv-vv-rel-rule-type := lexical-vv-rule-type &
 [SYNSEM.LOCAL.CONT.HOOK [INDEX #ind,
                          LTOP #lbl],
  C-CONT [RELS <! vv-relation &
                [LBL #lbl,
                 ARG0 #ind,
                 ARG1 #ind1,
                 ARG2 #ind2]!> ],
  ARGS <[SYNSEM [LOCAL [CONT.HOOK.INDEX #ind1]]],
        [SYNSEM [LOCAL [CONT.HOOK.INDEX #ind2]]]>].
```

As I described on page 148, the lexical-vv-motion-rule-type includes V_1-V_2s whose V_2 is a motion verb, as indicated by `[MOTION +]`.

LEXICAL-VV-MOTION-RULE-TYPE

```
lexical-vv-motion-rule-type := lexical-vv-vv-rel-rule-type &
 [SYNSEM.LOCAL [CAT.HEAD #head],
  C-CONT.RELS <! vv-relation & [PRED vv-manner_rel] !>,
  ARGS <[SYNSEM [LOCAL [CAT.HEAD v-stem_head,
                        ARG-ST transitive]]],
        [SYNSEM [LOCAL [CAT.HEAD v-stem_head & #head & [MOTION +],
                        ARG-ST monotrans]]]>].
```

A.3.2 Right headed V_1-V_2

In §3.6.2, it is shown that the Right headed V_1-V_2 is implemented by 29 rules that exhaust possible combinations of a V_1 and a V_2. But the *lexical-vv-non-motion-rule-type* generalizes commonalities among them; the compound and the V_2 must share the value of `ARG-ST` and `VAL` and the V_2 must not be `[MOTION +]`.

LEXICAL-VV-NON-MOTION-RULE-TYPE

```
lexical-vv-non-motion-rule-type := lexical-vv-vv-rel-rule-type &
 [SYNSEM [LOCAL [CAT [HEAD #head,
                      VAL #val],
                 ARG-ST #arg-st]],
  ARGS <[SYNSEM.LOCAL [CAT.HEAD v-stem_head]],
        [SYNSEM [LOCAL [CAT [HEAD v-stem_head & #head & [MOTION -],
                             VAL #val],
                        ARG-ST #arg-st]]]>].
```

Following are the 29 rules, which inherit *lexical-vv-non-motion-rule-type*. Notice how two arguments of a V_1 and a V_2 are co-indexed.

THE 29 RULES OF RIGHT HEADED VV

```
monounac-monounac-lex-vv-rule := lexical-vv-non-motion-rule-type &
 [ARGS.FIRST.SYNSEM.LOCAL.ARG-ST monounac & [INT1 #int1],
  ARGS.REST.FIRST.SYNSEM.LOCAL.ARG-ST monounac & [INT1 #int1]].


diunac-monounac-lex-vv-rule-1 := diunac-monounac-lex-vv-rule &
 [ARGS.FIRST.SYNSEM.LOCAL.ARG-ST [INT1 #int],
  ARGS.REST.FIRST.SYNSEM.LOCAL.ARG-ST [INT1 #int]].
```

```
diunac-monounac-lex-vv-rule-2 := diunac-monounac-lex-vv-rule &
 [ARGS.FIRST.SYNSEM.LOCAL.ARG-ST [INT2 #int],
  ARGS.REST.FIRST.SYNSEM.LOCAL.ARG-ST [INT1 #int]].


monotrans-monounac-lex-vv-rule := lexical-vv-non-motion-rule-type &
 [ARGS.FIRST.SYNSEM.LOCAL.ARG-ST monotrans & [INT1 #int],
  ARGS.REST.FIRST.SYNSEM.LOCAL.ARG-ST monounac & [INT1 #int]].


ditrans-monounac-lex-vv-rule-1 := ditrans-monounac-lex-vv-rule &
 [ARGS.FIRST.SYNSEM.LOCAL.ARG-ST [INT1 #int],
  ARGS.REST.FIRST.SYNSEM.LOCAL.ARG-ST [INT1 #int]].


ditrans-monounac-lex-vv-rule-2 := ditrans-monounac-lex-vv-rule &
 [ARGS.FIRST.SYNSEM.LOCAL.ARG-ST [INT2 #int],
  ARGS.REST.FIRST.SYNSEM.LOCAL.ARG-ST [INT1 #int]].


monounac-diunac-lex-vv-rule-1 := monounac-diunac-lex-vv-rule &
 [ARGS.FIRST.SYNSEM.LOCAL.ARG-ST [INT1 #int],
  ARGS.REST.FIRST.SYNSEM.LOCAL.ARG-ST [INT1 #int]].


monounac-diunac-lex-vv-rule-2 := monounac-diunac-lex-vv-rule &
 [ARGS.FIRST.SYNSEM.LOCAL.ARG-ST [INT1 #int],
  ARGS.REST.FIRST.SYNSEM.LOCAL.ARG-ST [INT2 #int]].


diunac-diunac-lex-vv-rule := lexical-vv-non-motion-rule-type &
 [ARGS.FIRST.SYNSEM.LOCAL.ARG-ST diunac & [INT1 #int1,
                                           INT2 #int2],
  ARGS.REST.FIRST.SYNSEM.LOCAL.ARG-ST diunac & [INT1 #int1,
                                                INT2 #int2]].


monotrans-diunac-lex-vv-rule-1 := monotrans-diunac-lex-vv-rule &
 [ARGS.FIRST.SYNSEM.LOCAL.ARG-ST [INT1 #int],
  ARGS.REST.FIRST.SYNSEM.LOCAL.ARG-ST [INT1 #int]].


monotrans-diunac-lex-vv-rule-2 := monotrans-diunac-lex-vv-rule &
 [ARGS.FIRST.SYNSEM.LOCAL.ARG-ST [INT1 #int],
  ARGS.REST.FIRST.SYNSEM.LOCAL.ARG-ST [INT2 #int]].


ditrans-diunac-lex-vv-rule := lexical-vv-non-motion-rule-type &
 [ARGS.FIRST.SYNSEM.LOCAL.ARG-ST ditrans & [INT1 #int1,
                                            INT2 #int2],
  ARGS.REST.FIRST.SYNSEM.LOCAL.ARG-ST diunac & [INT1 #int1,
                                                INT2 #int2]].


unergative-unergative-lex-vv-rule := lexical-vv-non-motion-rule-type &
```

```
 [ARGS.FIRST.SYNSEM.LOCAL.ARG-ST unergative & [EXT #ext],
  ARGS.REST.FIRST.SYNSEM.LOCAL.ARG-ST unergative & [EXT #ext]].

monotrans-unergative-lex-vv-rule := lexical-vv-non-motion-rule-type &
 [ARGS.FIRST.SYNSEM.LOCAL.ARG-ST monotrans & [EXT #ext],
  ARGS.REST.FIRST.SYNSEM.LOCAL.ARG-ST unergative & [EXT #ext]].

ditrans-unergative-lex-vv-rule := lexical-vv-non-motion-rule-type &
 [ARGS.FIRST.SYNSEM.LOCAL.ARG-ST ditrans & [EXT #ext],
  ARGS.REST.FIRST.SYNSEM.LOCAL.ARG-ST unergative & [EXT #ext]].

monounac-monotrans-lex-vv-rule := lexical-vv-non-motion-rule-type &
 [ARGS.FIRST.SYNSEM.LOCAL.ARG-ST monounac & [INT1 #int1],
  ARGS.REST.FIRST.SYNSEM.LOCAL.ARG-ST monotrans & [INT1 #int1]].

diunac-monotrans-lex-vv-rule-1 := diunac-monotrans-lex-vv-rule &
 [ARGS.FIRST.SYNSEM.LOCAL.ARG-ST diunac & [INT1 #int1],
  ARGS.REST.FIRST.SYNSEM.LOCAL.ARG-ST monotrans & [INT1 #int1]].

diunac-monotrans-lex-vv-rule-2 := diunac-monotrans-lex-vv-rule &
 [ARGS.FIRST.SYNSEM.LOCAL.ARG-ST diunac & [INT2 #int1],
  ARGS.REST.FIRST.SYNSEM.LOCAL.ARG-ST monotrans & [INT1 #int1]].

unergative-monotrans-lex-vv-rule := lexical-vv-non-motion-rule-type &
 [ARGS.FIRST.SYNSEM.LOCAL.ARG-ST unergative & [EXT #ext],
  ARGS.REST.FIRST.SYNSEM.LOCAL.ARG-ST monotrans & [EXT #ext]].

monotrans-monotrans-lex-vv-rule := lexical-vv-non-motion-rule-type &
 [ARGS.FIRST.SYNSEM.LOCAL.ARG-ST monotrans & [EXT #ext,
                                              INT1 #int],
  ARGS.REST.FIRST.SYNSEM.LOCAL.ARG-ST monotrans & [EXT #ext,
                                                   INT1 #int]].

ditrans-monotrans-lex-vv-rule-1 := ditrans-monotrans-lex-vv-rule &
 [ARGS.FIRST.SYNSEM.LOCAL.ARG-ST ditrans & [EXT #ext,
                                            INT1 #int],
  ARGS.REST.FIRST.SYNSEM.LOCAL.ARG-ST monotrans & [EXT #ext,
                                                   INT1 #int]].

ditrans-monotrans-lex-vv-rule-2 := ditrans-monotrans-lex-vv-rule &
 [ARGS.FIRST.SYNSEM.LOCAL.ARG-ST ditrans & [EXT #ext,
                                            INT2 #int],
  ARGS.REST.FIRST.SYNSEM.LOCAL.ARG-ST monotrans & [EXT #ext,
                                                   INT1 #int]].

monounac-ditrans-lex-vv-rule-1 := monounac-ditrans-lex-vv-rule &
 [ARGS.FIRST.SYNSEM.LOCAL.ARG-ST monounac & [INT1 #int],
```

```
  ARGS.REST.FIRST.SYNSEM.LOCAL.ARG-ST ditrans & [INT1 #int]].

monounac-ditrans-lex-vv-rule-2 := monounac-ditrans-lex-vv-rule &
 [ARGS.FIRST.SYNSEM.LOCAL.ARG-ST monounac & [INT1 #int],
  ARGS.REST.FIRST.SYNSEM.LOCAL.ARG-ST ditrans & [INT2 #int]].

diunac-ditrans-lex-vv-rule := lexical-vv-non-motion-rule-type &
 [ARGS.FIRST.SYNSEM.LOCAL.ARG-ST diunac & [INT1 #int1,
                                           INT2 #int2],
  ARGS.REST.FIRST.SYNSEM.LOCAL.ARG-ST ditrans & [INT1 #int1,
                                                 INT2 #int2]].

unergative-ditrans-lex-vv-rule := lexical-vv-non-motion-rule-type &
 [ARGS.FIRST.SYNSEM.LOCAL.ARG-ST unergative & [EXT #ext],
  ARGS.REST.FIRST.SYNSEM.LOCAL.ARG-ST ditrans & [EXT #ext]].

monotrans-ditrans-lex-vv-rule-1 := monotrans-ditrans-lex-vv-rule &
 [ARGS.FIRST.SYNSEM.LOCAL.ARG-ST monotrans & [EXT #ext,
                                              INT1 #int],
  ARGS.REST.FIRST.SYNSEM.LOCAL.ARG-ST ditrans & [EXT #ext,
                                                 INT1 #int]].

monotrans-ditrans-lex-vv-rule-2 := monotrans-ditrans-lex-vv-rule &
 [ARGS.FIRST.SYNSEM.LOCAL.ARG-ST monotrans & [EXT #ext,
                                              INT1 #int],
  ARGS.REST.FIRST.SYNSEM.LOCAL.ARG-ST ditrans & [EXT #ext,
                                                 INT2 #int]].

ditrans-ditrans-lex-vv-rule := lexical-vv-non-motion-rule-type &
 [ARGS.FIRST.SYNSEM.LOCAL.ARG-ST ditrans & [EXT #ext,
                                            INT1 #int1,
                                            INT2 #int2],
  ARGS.REST.FIRST.SYNSEM.LOCAL.ARG-ST ditrans & [EXT #ext,
                                                 INT1 #int1,
                                                 INT2 #int2]].
```

Some of the 29 rules have `-1` or `-2` as suffix. They are immediate subtypes of one of the following 8 general rule types, and the 8 types in turn directly inherit *lexical-vv-non-motion-rule-type*.

8 GENERAL RULE TYPES

```
diunac-monounac-lex-vv-rule := lexical-vv-non-motion-rule-type &
 [ARGS.FIRST.SYNSEM.LOCAL.ARG-ST diunac,
  ARGS.REST.FIRST.SYNSEM.LOCAL.ARG-ST monounac].
```

```
ditrans-monounac-lex-vv-rule := lexical-vv-non-motion-rule-type &
 [ARGS.FIRST.SYNSEM.LOCAL.ARG-ST ditrans,
  ARGS.REST.FIRST.SYNSEM.LOCAL.ARG-ST monounac].

monounac-diunac-lex-vv-rule := lexical-vv-non-motion-rule-type &
 [ARGS.FIRST.SYNSEM.LOCAL.ARG-ST monounac,
  ARGS.REST.FIRST.SYNSEM.LOCAL.ARG-ST diunac].

monotrans-diunac-lex-vv-rule := lexical-vv-non-motion-rule-type &
 [ARGS.FIRST.SYNSEM.LOCAL.ARG-ST monotrans,
  ARGS.REST.FIRST.SYNSEM.LOCAL.ARG-ST diunac].

diunac-monotrans-lex-vv-rule := lexical-vv-non-motion-rule-type &
 [ARGS.FIRST.SYNSEM.LOCAL.ARG-ST diunac,
  ARGS.REST.FIRST.SYNSEM.LOCAL.ARG-ST monotrans].

ditrans-monotrans-lex-vv-rule := lexical-vv-non-motion-rule-type &
 [ARGS.FIRST.SYNSEM.LOCAL.ARG-ST ditrans,
  ARGS.REST.FIRST.SYNSEM.LOCAL.ARG-ST monotrans].

monounac-ditrans-lex-vv-rule := lexical-vv-non-motion-rule-type &
 [ARGS.FIRST.SYNSEM.LOCAL.ARG-ST monounac,
  ARGS.REST.FIRST.SYNSEM.LOCAL.ARG-ST ditrans].

monotrans-ditrans-lex-vv-rule := lexical-vv-non-motion-rule-type &
 [ARGS.FIRST.SYNSEM.LOCAL.ARG-ST monotrans,
  ARGS.REST.FIRST.SYNSEM.LOCAL.ARG-ST ditrans].
```

As discussed in §3.6.2, if two component verbs are both `[MOTION +]`, the two verbs constitute the Right headed V_1-V_2. Following is the rule for this kind.

RIGHT HEADED VV CONSISTING OF TWO MOTION VERBS

```
lex-vv-motion-non-argument-mix-rule := lexical-vv-motion-rule-type &
 [SYNSEM [LOCAL [CAT [VAL #val],
                 ARG-ST #arg-st]],
  ARGS <[SYNSEM.LOCAL[CAT.HEAD.MOTION +,
                      ARG-ST #arg-st]],
        [SYNSEM [LOCAL [CAT [VAL #val],
                        ARG-ST #arg-st]]]>].
```

A.3.3 Argument mixing V_1-V_2

I discussed in §3.6.3 that the Argument mixing V_1-V_2 consists of a `[MOTION -]` V_1 and a `[MOTION +]` V_2, and the two `EXT`ernal arguments of the V_1 and the V_2 are co-indexed. The *lex-vv-motion-argument-mix-rule* accounts for these properties. Furthermore, the Argument mixing V_1-V_2 has an ambiguity that the `ARG-ST` and the `VAL` can be either from the V_1 or the V_2. The two rules, `lex-vv-motion-argument-mix-rule-1` and `lex-vv-motion--argument-mix-rule-2`, deal with this ambiguity.

ARGUMENT MIXING VV

```
lex-vv-motion-argument-mix-rule := lexical-vv-motion-rule-type &
 [ARGS <[SYNSEM.LOCAL [CAT.HEAD.MOTION -,
                       ARG-ST.EXT #ext]],
        [SYNSEM.LOCAL.ARG-ST.EXT #ext] >].


lex-vv-motion-argument-mix-rule-1 := lex-vv-motion-argument-mix-rule &
 [SYNSEM [LOCAL [CAT [VAL #val],
                 ARG-ST #arg-st]],
  ARGS <[SYNSEM [LOCAL [CAT [VAL #val],
                        ARG-ST #arg-st]]],
        [ ]>].


lex-vv-motion-argument-mix-rule-2 := lex-vv-motion-argument-mix-rule &
 [SYNSEM [LOCAL [CAT [VAL #val],
                 ARG-ST #arg-st]],
  ARGS <[ ],
        [SYNSEM [LOCAL [CAT [VAL #val],
                        ARG-ST #arg-st]]]>].
```

A.3.4 V_1-V_2 with semantically deverbalized V_1

The V_1-V_2 with semantically deverbalized V_1 does not obey the Transitivity Harmony Principle, as discussed in §3.6.4. The *vv with semantically deverbalized v1* represents the compound, and I designed this rule so as to be similar to other prefix rules in JACY.

VV WITH SEMANTICALLY DEVERBALIZED V1

```
lexical-vv-prefix-v1-attach-rule := head-final-type-mod &
                                    head-adjunct-p &
 [SYNSEM.LOCAL [CONT [HOOK [INDEX #ind,
                            LTOP #lbl,
                            XARG #xarg],
                      MOD-IND #mi,
                      MOD-HAND #mh],
                CTXT.BACKGROUND diff-list & [LIST #1, LAST #2],
                CAT.VAL #val],
  C-CONT [RELS <! vv-prefix-v1-relation &
                [LBL #lbl,
                  ARG0 #ind,
                 ARG1 #hind1]!>,
          HCONS <! qeq & [HARG #hind1, LARG #lind1] !>],
```

```
  J-NEEDS-AFFIX -,
  ARGS < [SYNSEM.LOCAL [BAR -,
                        CAT.HEAD lexical-vv-prefix-v1-stem_head,
                        CTXT.BACKGROUND [list #3,last #2]],
          J-NEEDS-AFFIX -],
         [SYNSEM.LOCAL [BAR bool,
                        CONT [HOOK [LTOP #lind1,
                                    INDEX #ind,
                                    XARG #xarg],
                              MOD-IND #mi,
                              MOD-HAND #mh],
                        CTXT.BACKGROUND [list #1,last #3],
                        CAT.VAL #val],
          J-NEEDS-AFFIX -] >].
```

A.3.5 V_1-V_2 with semantically deverbalized V_2

In §3.6.5, it is shown that the V_1-V_2 with semantically deverbalized V_2 has the semantic structure where the V_2 embeds V_1's semantics, and that the V_1-V_2 obeys Kageyama's (1993) Transitivity Harmony Principle. Hence we need two rules; the *lexical-vv-event-embedding-rule-agentive* deals with the case in which two component verbs are both agentive, and the *lexical-vv-event-embedding-rule-nonagentive* is for the case in which two component verbs are not agentive. Both rules embed V_1's semantics (`#chand`) within V_2's semantics.

VV WITH SEMANTICALLY DEVERBALIZED V2

```
lexical-vv-event-embedding-rule-type := lexical-vv-rule-type &
 [SYNSEM [LOCAL [CAT [HEAD #head,
                      VAL #val],
                 CONT.HOOK.INDEX #ind]],
  ARGS <[SYNSEM.LOCAL [CAT [HEAD v-stem_head,
                            VAL #val &
                                 [SUBJ <[LOCAL.CONT.HOOK.INDEX #sbjind]>]],
                       CONT [HOOK [LTOP #ohand,
                                   INDEX #ind]]]],
        [SYNSEM [LOCAL [CAT [HEAD lexical-vv-event-embedding-stem_head &
                                                                    #head],
                        CONT [RELS <! [ ], [MARG #soa] !>,
                              HCONS <! qeq & [LARG #ohand,
                                              HARG #soa]!>,
                              HOOK [XARG #sbjind,
                                    INDEX #ind]]]]]>].


lexical-vv-event-embedding-rule-agentive
 := lexical-vv-event-embedding-rule-type &
 [SYNSEM [LOCAL [CAT.VAL.SUBJ <[LOCAL.CONT.HOOK.INDEX #sbjind]>]],
  ARGS < [SYNSEM.LOCAL.ARG-ST agentive],
         [SYNSEM [LOCAL [ARG-ST agentive,
                         CONT.RELS <! [ ], [LBL #chand] !>],
                 LKEYS.KEYREL arg12-relation &
```

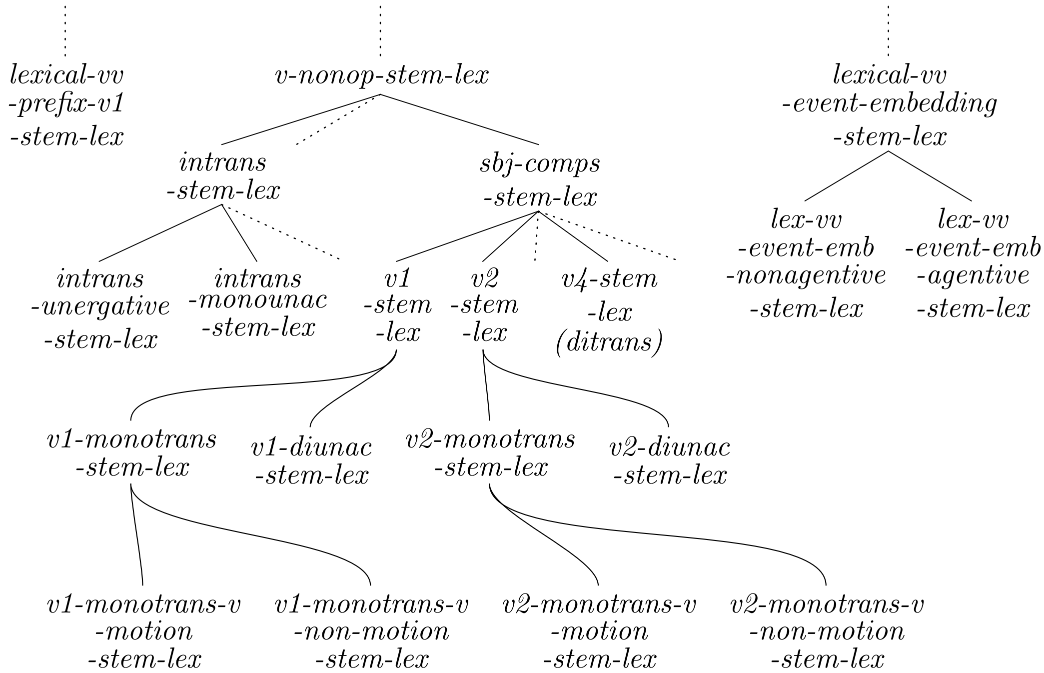
```
                       [ARG1 #sbjind,
                        ARG2 #chand]]] >].


lexical-vv-event-embedding-rule-nonagentive
 := lexical-vv-event-embedding-rule-type &
 [ARGS < [SYNSEM.LOCAL.ARG-ST nonagentive],
         [SYNSEM [LOCAL [ARG-ST nonagentive,
                        CONT.RELS <! [ ], [LBL #chand] !>],
                  LKEYS.KEYREL arg1-relation &
                       [ARG1 #chand]]] >].
```

A.3.6 Verb hierarchy

In this section, I present lexical types for each kind of V_1-V_2 compound. (189) shows (a part of) the relevant hierarchies.

(189)



Under the *v-nonop-stem-lex* are those that represent simple verbs like *neru* (sleep), *taberu* (eat), and *oku* (put). They constitute component verbs of the Right headed V_1-V_2 and the Argument mixing V_1-V_2. As well, they can be the V_2 of the V_1-V_2 with semantically deverbalized V_1 or the V_1 of the V_1-V_2 with semantically deverbalized V_2. The subtypes of the *v-nonop-stem-lex* include the *intrans-stem-lex*, representing intransitive verbs, and the *sbj-comps-stem-lex*, representing transitive verbs. Naturally, the former includes *unergative* verbs and *monounac* verbs, while the latter includes *monotrans* verbs, *diunac* verbs, and *ditrans* verbs. I also divide the two *monotrans* types into *motion* types and *non-motion* types.

The *lexical-vv-prefix-v1-stem-lex*, the lexical type for the V_1 of V_1-V_2 with semantically deverbalized V_1, and the *lexical-vv-event-embedding-stem-lex*, the lexical type for the V_2 of V_1-V_2 with semantically deverbalized V_2, are built onto outside the *v-nonop-stem-lex* hierarchy, since these two have several different characteristics from simple verbs. The *lexical-vv-event-embedding-stem-lex* is further divided into the *lex-vv-event-emb-nonagentive-stem-lex* and the *lex-vv-event-emb-agentive-stem-lex* since it is sensitive to agentivity, as discussed in §3.6.5.

Following are illustrations of those types in (189). First of all, I show the general lexical types. The *v-nonop-stem-lex*, the *intrans-stem-lex*, the *sbj-comps-stem-lex*, the *v1-stem-lex*, and the *v2-stem-lex* are developed by Melanie Siegel and Emily Bender.

V-NONOP-STEM-LEX

```
v-nonop-stem-lex := v-stem-lex &
 [SYNSEM [LOCAL.CONT verb-sem-type &
                     [HOOK.INDEX event,
                      RELS <! #key !>,
                      HCONS <! !>],
         LKEYS.KEYREL #key]].
```

The *v-nonop-stem-lex* requires that the reference of expression headed by a verb should be `event`.

INTRANS-STEM-LEX

```
intrans-stem-lex := v-nonop-stem-lex &
 [SYNSEM [LKEYS.KEYREL [ARG1 #sbjind],
         LOCAL [CAT [HEAD [COP -,
                           MOTION -],
                      VAL intransitive &
                          [SUBJ #sbj &
                                [FIRST.LOCAL.CONT [HOOK.INDEX #sbjind]]]],
                CONT  [HOOK.XARG #sbjind],
                ARG-S #sbj ]]].
```

SBJ-COMPS-STEM-LEX

```
v-sbj-comps-stem-lex := v-nonop-stem-lex &
 [SYNSEM.LOCAL [CAT.VAL [SUBJ.FIRST #sbj,
                         COMPS #comps],
               ARG-S < #sbj . #comps >]].
```

The *intrans-stem-lex* and the *sbj-comps-stem-lex* contain ARG-S feature, but the feature is not used any more. It stays in the codes only for safety. Notice that I restrict the *intrans-stem-lex* to [MOTION -] because, in my analysis, verbs that are [MOTION +] can take a locative object, and thus they must be *transitive*.

V1-STEM-LEX

```
v1-stem-lex := v-sbj-comps-stem-lex &
 [SYNSEM [LKEYS.KEYREL [ARG1 #sbjind,
                        ARG2 #objind],
          LOCAL [CAT [HEAD [COP -],
                      VAL ga-wo_transitive &
                           [SUBJ.FIRST.LOCAL.CONT.HOOK.INDEX #sbjind,
                            COMPS.FIRST.LOCAL.CONT.HOOK.INDEX #objind]],
                 CONT [HOOK.XARG #sbjind]]]].
```

V2-STEM-LEX

```
v2-stem-lex := v-sbj-comps-stem-lex &
 [SYNSEM [LOCAL [CAT [HEAD.COP -,
                      VAL ga-ni_transitive &
                              [COMPS.FIRST.LOCAL.CONT [HOOK.INDEX #objind],
                               SUBJ.FIRST.LOCAL.CONT [HOOK.INDEX #sbjind]]],
                 CONT [HOOK.XARG #sbjind]],
                        LKEYS.KEYREL [ARG1 #sbjind,
                                      ARG2 #objind],
          NON-LOCAL.QUE <! !>]].
```

The *v1-stem-lex* and the *v2-stem-lex* have different case-marking properties; the former has an accusative case-marker for an object, while the latter marks an object with a dative case-marker.

Next I show lexical types that can constitute the V_1 and V_2 of the Right headed and Argument mixing V_1-V_2s, the V_2 of the V_1-V_2 with semantically deverbalized V_1, and the V_1 of the V_1-V_2 with semantically deverbalized V_2; the *unergative* verbs, the *monounac* verbs, the *monotrans* verbs, the *diunac* verbs, and the *ditrans* verbs. I present them with lexical items of each type, for illustration.

INTRANS-UNERGATIVE-STEM-LEX

```
intrans-unergative-stem-lex := intrans-stem-lex &
 [SYNSEM.LOCAL [CAT [VAL.SUBJ < [LOCAL.CONT.HOOK.INDEX #sbjind] >],
```

```
                ARG-ST unergative & [EXT < #sbjind >]]].


hataraku-stem := intrans-unergative-c-stem-lex &
 [SYNSEM.LKEYS [KEYREL [PRED 'hataraku]],
  ORTH <! "働く" !> ].
```

INTRANS-MONOUNAC-STEM-LEX

```
intrans-monounac-stem-lex := intrans-stem-lex &
 [SYNSEM.LOCAL [CAT [VAL.SUBJ <  [LOCAL.CONT.HOOK.INDEX #sbjind] >],
                ARG-ST monounac & [INT1 < #sbjind >]]].


waku_1_4 := intrans-monounac-c-stem-lex &
 [SYNSEM.LKEYS.KEYREL.PRED 'waku_1_4,
  ORTH <! "沸く" !> ].
```

V1-MONOTRANS-STEM-LEX

```
v1-monotrans-stem-lex := v1-stem-lex &
 [SYNSEM.LOCAL [CAT [VAL [SUBJ < [LOCAL.CONT.HOOK.INDEX #sbjind] >,
                          COMPS < [LOCAL.CONT.HOOK.INDEX #objind] > ]],
                ARG-ST monotrans & [EXT < #sbjind >,
                                    INT1 < #objind > ]]].


v1-monotrans-c-non-motion-stem-lex := v1-monotrans-c-stem-lex &
 [SYNSEM.LOCAL.CAT.HEAD.MOTION -].


v1-monotrans-c-motion-stem-lex := v1-monotrans-c-stem-lex &
 [SYNSEM.LOCAL.CAT.HEAD.MOTION +].


waru-stem := v1-monotrans-c-non-motion-stem-lex &
 [SYNSEM.LKEYS [KEYREL [PRED 'waru]],
  ORTH <! "割る" !> ].


aruku-stem := v1-monotrans-c-motion-stem-lex &
 [SYNSEM.LKEYS [KEYREL [PRED 'aruku]],
  ORTH <! "歩く" !> ].
```

V1-DIUNAC-STEM-LEX

```
v1-diunac-stem-lex := v1-stem-lex &
 [SYNSEM.LOCAL [CAT [HEAD.MOTION -,
                     VAL [SUBJ < [LOCAL.CONT.HOOK.INDEX #sbjind] >,
                          COMPS < [LOCAL.CONT.HOOK.INDEX #objind] > ]],
                ARG-ST diunac & [INT1 < #sbjind >,
                                 INT2 < #objind > ]]].
```

```
azukaru-stem := v1-diunac-c-stem-lex &
 [SYNSEM.LKEYS [KEYREL [PRED 'azukaru]],
  ORTH <! "預かる" !> ].
```

V2-MONOTRANS-STEM-LEX

```
v2-monotrans-stem-lex := v2-stem-lex &
 [SYNSEM.LOCAL [CAT [VAL [SUBJ < [LOCAL.CONT.HOOK.INDEX #sbjind] >,
                          COMPS < [LOCAL.CONT.HOOK.INDEX #objind] > ]],
                ARG-ST monotrans & [EXT < #sbjind >,
                                    INT1 < #objind > ]]].
```

```
v2-monotrans-v-non-motion-stem-lex := v2-monotrans-v-stem-lex &
 [SYNSEM.LOCAL.CAT.HEAD.MOTION -].
```

```
v2-monotrans-v-motion-stem-lex := v2-monotrans-v-stem-lex &
 [SYNSEM.LOCAL.CAT.HEAD.MOTION +].
```

```
idomu-stem := v2-monotrans-c-non-motion-stem-lex &
 [SYNSEM.LKEYS [KEYREL [PRED 'idomu]],
  ORTH <! "挑む" !> ].
```

V2-DIUNAC-STEM-LEX

```
v2-diunac-stem-lex := v2-stem-lex &
 [SYNSEM.LOCAL [CAT [HEAD.MOTION -,
                     VAL [SUBJ < [LOCAL.CONT.HOOK.INDEX #sbjind] >,
                          COMPS < [LOCAL.CONT.HOOK.INDEX #objind] > ]],
                ARG-ST diunac & [INT1 < #sbjind >,
                                 INT2 < #objind > ]]].
```

```
somaru_1 := v2-diunac-c-stem-lex &
 [SYNSEM.LKEYS.KEYREL.PRED 'somaru_1,
  ORTH <! "染まる" !> ].
```

V4-STEM-LEX

```
v4-stem-lex := v-sbj-comps-stem-lex &
 [SYNSEM [LOCAL [CAT [HEAD [COP -],
                      VAL ditransitive &
                           [COMPS < [LOCAL.CONT [HOOK.INDEX #objind]],
                                    [LOCAL.CONT [HOOK.INDEX #obj2ind]] >,
                            SUBJ < [LOCAL.CONT [HOOK.INDEX #sbjind]] >]],
                 CONT [HOOK.XARG #sbjind],
                 ARG-ST ditrans & [EXT < #sbjind >,
                                   INT1 < #objind >,
                                   INT2 < #obj2ind > ]],
          LKEYS.KEYREL [ARG1 #sbjind,
                        ARG2 #objind,
                        ARG3 #obj2ind],
          NON-LOCAL.QUE <! !>]].


oku-stem := v4-c-non-motion-stem-lex &
 [SYNSEM.LKEYS [KEYREL [PRED 'oku]],
  ORTH <! "置く" !> ].
```

Below is the implementation of the V_1 of V_1-V_2 with semantically deverbalized V_1 and an example of lexical item of the type.

LEXICAL-VV-PREFIX-V1-STEM-LEX

```
lexical-vv-prefix-v1-stem-lex := lexical_sign-affix &
 [SYNSEM [LOCAL [CAT [HEAD lexical-vv-prefix-v1-stem_head,
                      VAL saturated],
                 ARG-S < >,
                 CONT [RELS <! !>,
                       HCONS <! !>],
                 CTXT.BACKGROUND <! !>,
                 BAR -],
          NON-LOCAL [QUE <! !>,
                     AFFIX <! !>]],
  INFLECTED +].


tori-lex-vv-pref-v1 := lexical-vv-prefix-v1-stem-lex &
 [ORTH <! "取り" !>].
```

Note that it does not inflect and has no semantic content.

The implementation of V_2 of V_1-V_2 with semantically deverbalized V_2 is given below.

LEXICAL-VV-EVENT-EMBEDDING-STEM-LEX

```
lexical-vv-event-embedding-stem-lex := verb-stem-lex &
                                       lexical_sign-word &
 [SYNSEM [LOCAL [CAT [HEAD lexical-vv-event-embedding-stem_head &
                           [H-TENSE #tense],
                      VAL saturated],
                 CTXT.BACKGROUND <! !>,
                 ARG-S <>,
                 BAR -,
                 CONT [HOOK [INDEX #event & [E [TENSE #tense]]],
                       RELS diff-list & <! #key,
                                         message &
                                         [PRED proposition_m_rel] !>]],
          NON-LOCAL [QUE <! !>,
                     AFFIX <! !>],
          LKEYS.KEYREL #key &
                [ARG0 #event]],
  J-NEEDS-AFFIX +].


lex-vv-event-emb-nonagentive-stem-lex
 := lexical-vv-event-embedding-stem-lex &
 [SYNSEM.LOCAL [ARG-ST nonagentive]].


lex-vv-event-emb-agentive-stem-lex
 := lexical-vv-event-embedding-stem-lex &
 [SYNSEM.LOCAL [ARG-ST agentive]].


konasu-hiragana-lex-vv-event-emb-agentive-stem
 := lex-vv-event-emb-agentive-c-stem-lex &
 [SYNSEM.LKEYS [KEYREL [PRED 'konasu_eemb]],
  ORTH <! "こなす" !>].


wataru-lex-vv-event-emb-nonagentive-stem
 := lex-vv-event-emb-nonagentive-c-stem-lex &
 [SYNSEM.LKEYS [KEYREL [PRED 'wataru_eemb]],
  ORTH <! "渡る" !>].
```

Appendix B Additional illustration of the analysis

In this appendix, we look closer into my analysis to help you understand the technical details of it.

B.1 Preliminaries

First of all, let us look at the analysis of (190). The JACY output of the sentence is given in Figure 30.

(190) Ken-ga hon-o yomu
Ken-NOM book-ACC read
'Ken reads a book.'

read(Ken, book)

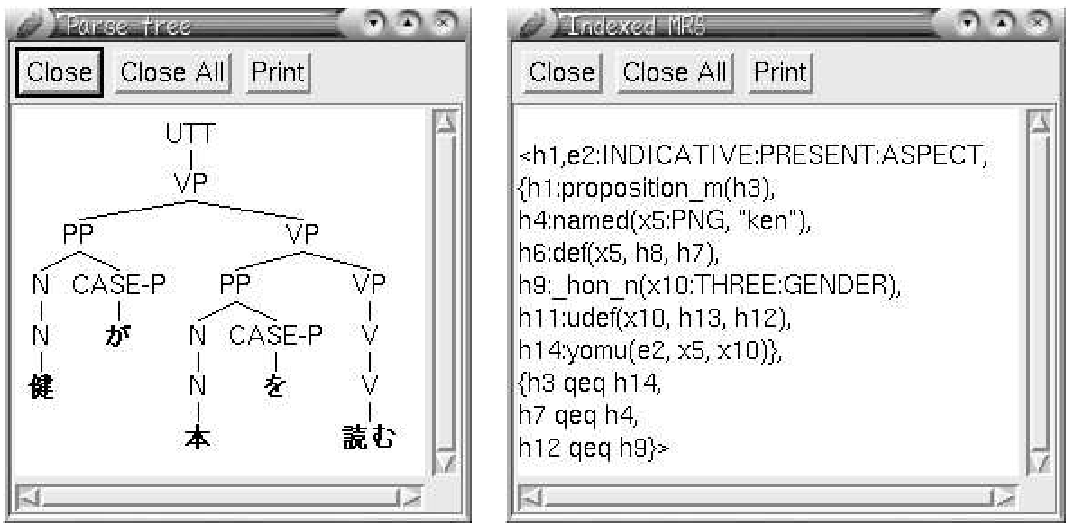
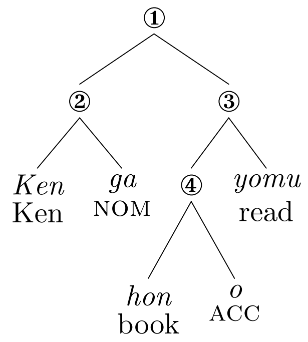


Figure 30: *Ken-ga hon-o yomu*

(191) shows which rule applies to which node.[1]

[1] In (191), I omit the lexical rules and the utterance rule that appear in Figure 30.

(191)



① HEAD-SUBJECT RULE
② HEAD-COMPLEMENT RULE
③ HEAD-COMPLEMENT RULE
④ HEAD-COMPLEMENT RULE

Table 14 shows the Indexed MRSs of *yomu*, *hon-o yomu*, and *Ken-ga hon-o yomu*. The table tells us how the semantic composition proceeds. In the second row, *hon-o*

Table 14: The semantic composition of *Ken-ga hon-o yomu*

yomu read	<h1,e2:INDICATIVE:PRESENT:ASPECT, {h1:proposition_m(h3), h4:yomu(e2, u6, u5)}, {h3 qeq h4}>
hon-o yomu book-ACC read	<h1,e2:INDICATIVE:PRESENT:ASPECT, {h1:proposition_m(h3), h4:_hon_n(x5:THREE:GENDER), h6:udef(x5, h8, h7), h9:yomu(e2, u10, x5)}, {h3 qeq h9, h7 qeq h4}>
Ken-ga hon-o yomu Ken-NOM book-ACC read	<h1,e2:INDICATIVE:PRESENT:ASPECT, {h1:proposition_m(h3), h4:named(x5:PNG, "ken"), h6:def(x5, h8, h7), h9:_hon_n(x10:THREE:GENDER), h11:udef(x10, h13, h12), h14:yomu(e2, x5, x10)}, {h3 qeq h14, h7 qeq h4, h12 qeq h9}>

(book-ACC) is represented by `h4:_hon_n(x5:THREE:GENDER)` and `h6:undef(x5,h8,h7)`.[2]

[2]The `undef` and `def` represent definitness.

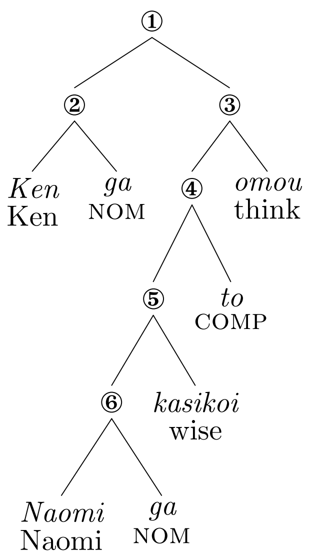
Note that introducing *hon-o* specifies the second semantic argument of *yomu*, which represents what is read, as `x5`[3]. That is, in the second row, it is stated that what is read is the *hon*. Similarly, in the third row, the subject *Ken-ga* is introduced, which is represented by `h4:named(x5:PNG,"ken")` and `h6:def(x5,h8,h7)`. We should notice that, in the third row, the representation of *hon-o* is changed to `h9:_hon_n(x10:THREE:GENDER)` and `h11:undef(x10,h13,h12)`. Also, the third row shows that the first semantic argument of *yomu* is `x5`, which is the *Ken-ga* in the row.

Let us move on to (192), which contains an embedded clause. This example will allow you to understand the semantic embedding structure that I posit for the three syntactic V_1-V_2 compounds and the V_1-V_2 with semantically deverbalized V_2. Figure 31 shows the JACY output of the sentence.

(192) Ken-ga Naomi-ga kasikoi-to omou
Ken-NOM Naomi-NOM wise-COMP think
'Ken thinks that Naomi is wise.'

Evidently, the sentence embeds the clause, *Naomi-ga kasikoi* 'Naomi is wise'. (193) shows the rule applications in the sentence.

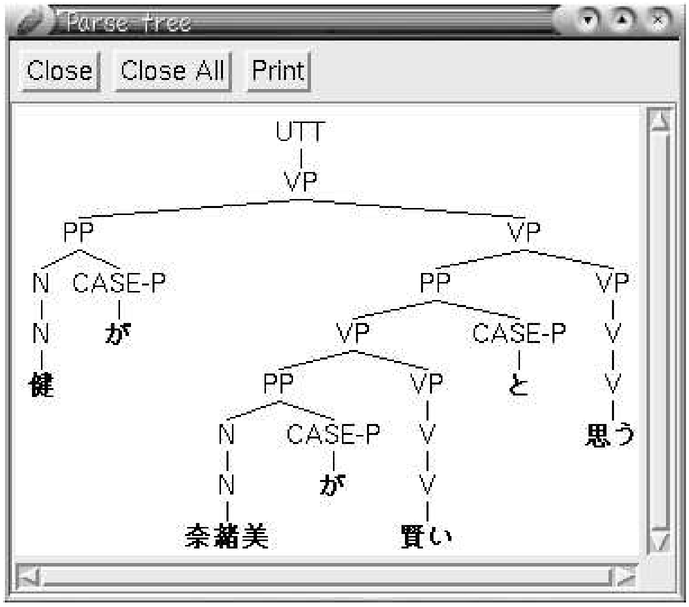
(193)



① HEAD-SUBJECT RULE
② HEAD-COMPLEMENT RULE
③ HEAD-COMPLEMENT RULE
④ HEAD-COMPLEMENT RULE
⑤ HEAD-SUBJECT RULE
⑥ HEAD-COMPLEMENT RULE

Table 15 illustrates the semantic composition of the sentence. The semantics of the embedded sentence, *Naomi-ga kasikoi* 'Naomi is wise' is given in the first row. In the second row is the semantics of *omou* 'think', in which neither the first semantic argument (a person who

[3]The `e2` in `h9:yomu(e2,u10,x5)` is the event variable, and thus the two participants of the reading events, a reader and a thing to read, are represented by the `u10` and the `x5`.



`Ken(think, is(Naomi, wise))`

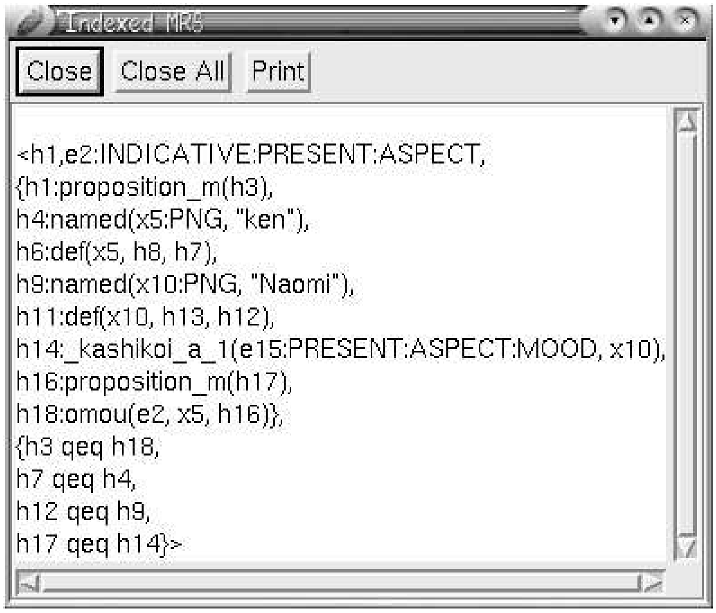


Figure 31: *Ken-ga Naomi-ga kasikoi-to omou*

Table 15: The semantic composition of *Ken-ga Naomi-ga kasikoi-to omou*

Naomi-ga Naomi-NOM kasikoi wise	<h1,e2:PRESENT:ASPECT:MOOD, {h1:proposition_m(h3), h4:named(x5:PNG, "Naomi"), h6:def(x5, h8, h7), h9:_kashikoi_a_1(e2, x5)}, {h3 qeq h9, h7 qeq h4}>
omou think	<h1,e2:INDICATIVE:PRESENT:ASPECT, {h1:proposition_m(h3), h4:omou(e2, u6, h5)}, {h3 qeq h4}>
Naomi-ga Naomi-NOM kasikoi-to omou wise-COMP think	<h1,e2:INDICATIVE:PRESENT:ASPECT, {h1:proposition_m(h3), h4:named(x5:PNG, "Naomi"), h6:def(x5, h8, h7), h9:_kashikoi_a_1(e10:PRESENT:ASPECT:MOOD, x5), h11:proposition_m(h12), h13:omou(e2, u14, h11)}, {h3 qeq h13, h7 qeq h4, h12 qeq h9}>
Ken-ga Ken-NOM Naomi-ga Naomi-NOM kasikoi-to omou wise-COMP think	<h1,e2:INDICATIVE:PRESENT:ASPECT, {h1:proposition_m(h3), h4:named(x5:PNG, "ken"), h6:def(x5, h8, h7), h9:named(x10:PNG, "Naomi"), h11:def(x10, h13, h12), h14:_kashikoi_a_1(e15:PRESENT:ASPECT:MOOD, x10), h16:proposition_m(h17), h18:omou(e2, x5, h16)}, {h3 qeq h18, h7 qeq h4, h12 qeq h9, h17 qeq h14}>

thinks, `u6`) nor the second argument (the content of thought, `h5`) is specified. The semantics of *Naomi-ga kasikoi-to omou* 'think that Naomi is wise' is given in the third row. Notice that the content of thought is specified as the proposition `h11:proposition_m(h12)`. Through `h12 qeq h9`, the proposition refers to `h9:_kashikoi_a_1(e10:PRESENT:ASPECT:MOOD,x5)`, that is, the proposition that Naomi (`x5`) is wise. (194) describes this flow.

(194) `h13:omou(e2,u14,h11)` (*think that* ···)

  ↓ `h11:proposition_m(h12)`

    ↓ `h12 qeq h9`

      ↓ `h9:_kashikoi_a_1(e10:PRESENT:ASPECT:MOOD,x5)` (*Naomi is wise*)

As a result, there are two `propositions` in the third row: one (`h11:proposition_m(h12)`) represents the proposition that Naomi is wise, and the other (`h1:proposition_m(h3)`) refers to the proposition that someone thinks that Naomi is wise. Finally, the person who thinks is specified as *Ken* in the fourth row.

B.2 Syntactic V_1-V_2 compound

I analyzed the three syntactic V_1-V_2 compounds as having different semantic embedding structures. In this section, I present each of them in turn.

B.2.1 A type

Below is the example of the A type, which repeats (85a) on page 58.

(195) Ken-ga hon-o yomi-kakeru
Ken-NOM book-ACC read-be.about.to
'Ken is about to read a book.'

Figure 32 shows the JACY output of the sentence. In (196), it is illustrated which rule constitutes which node.

`be.about.to(read(Ken, book))`

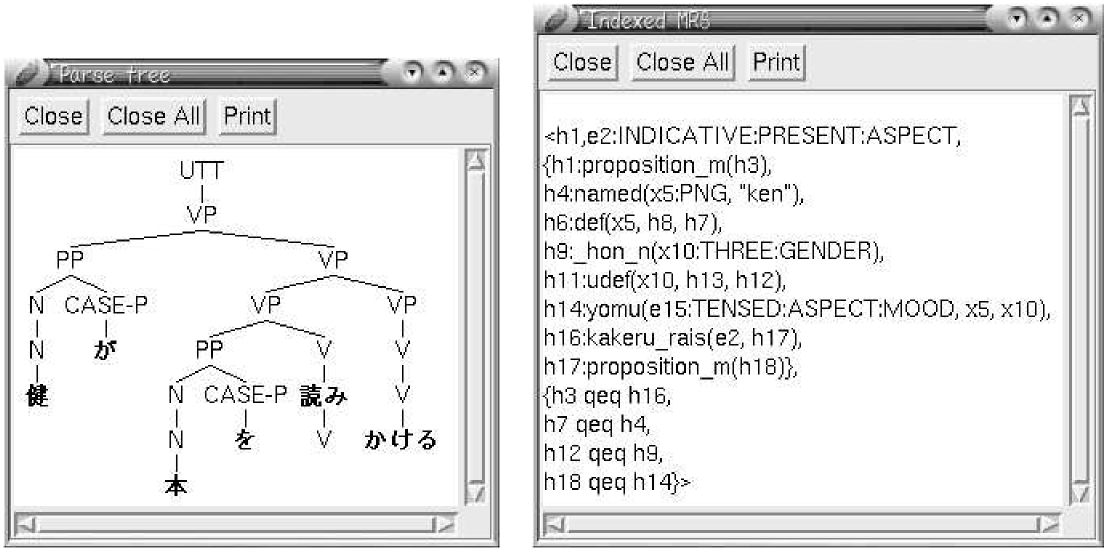
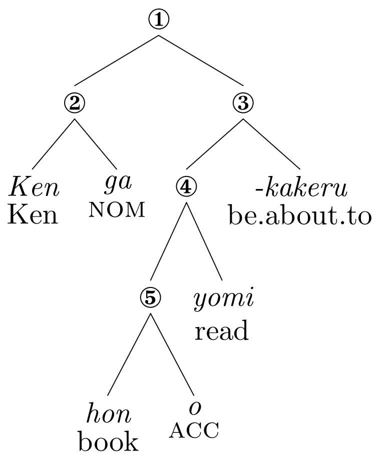


Figure 32: *Ken-ga hon-o yomi-kakeru*

(196)



① HEAD-SUBJECT RULE
② HEAD-COMPLEMENT RULE
③ HEAD-SPECIFIER RULE
④ HEAD-COMPLEMENT RULE
⑤ HEAD-COMPLEMENT RULE

Syntactically speaking, the A type has a VP embedding structure, as discussed in §3.5.1. Regarding its semantic representation, the V_2 embeds the proposition that is represented by the VP headed by the V_1. Look at Table 16. Note that, in the first row, where the semantics of *yomi-kakeru* (read-be.about.to) is shown, there are two propositions: one (`h1:proposition_m(h3)`) refers to the event that *yomi-kakeru* as a whole represents, and the other (`h9:proposition_(h10)`) refers to the event that *yomu* represents. The latter is embedded by the *-kakeru* semantically. (197) describes the whole semantic structure.

Table 16: The semantic composition of *Ken-ga hon-o yomi-kakeru*

yomi-kakeru read-be.about.to	<h1,e2:INDICATIVE:PRESENT:ASPECT, {h1:proposition_m(h3), h4:yomu(e7:TENSED:ASPECT:MOOD, u6, u5), h8:kakeru_rais(e2, h9), h9:proposition_m(h10)}, {h3 qeq h8, h10 qeq h4}>
hon-o book-ACC yomi-kakeru read-be.about.to	<h1,e2:INDICATIVE:PRESENT:ASPECT, {h1:proposition_m(h3), h4:_hon_n(x5:THREE:GENDER), h6:udef(x5, h8, h7), h9:yomu(e11:TENSED:ASPECT:MOOD, u10, x5), h12:kakeru_rais(e2, h13), h13:proposition_m(h14)}, {h3 qeq h12, h7 qeq h4, h14 qeq h9}>
Ken-ga hon-o Ken-NOM book-ACC yomi-kakeru read-be.about.to	<h1,e2:INDICATIVE:PRESENT:ASPECT, {h1:proposition_m(h3), h4:named(x5:PNG, "ken"), h6:def(x5, h8, h7), h9:_hon_n(x10:THREE:GENDER), h11:udef(x10, h13, h12), h14:yomu(e15:TENSED:ASPECT:MOOD, x5, x10), h16:kakeru_rais(e2, h17), h17:proposition_m(h18)}, {h3 qeq h16, h7 qeq h4, h12 qeq h9, h18 qeq h14}>

(197) `h1:proposition_m(h3)`

`h3 qeq h8`

`h8:kakeru_rais(e2,h9)` (V_2)

`h9:proposition_(h10)`

`h10 qeq h4`

`h4:yomu(e7:TENSED:ASPECT:MOOD,u6,u5)` (V_1)

At this point, the first and the second semantic arguments of *yomu* (`u6` and `u5`), corresponding to a reader and a thing to read, remain to be specified, as `h4:yomu(e7:TENSED:ASPECT:MOOD,u6,u5)` indicates. These two semantic arguments are specified in the second and the third rows. Look at the third row, where `x5` represents the reader and `x10` represents the thing to read.

B.2.2 B type

The B type has the syntactic and semantic structures that are very similar to those of the A type. But the former constitutes a control construction, while the latter is analyzed as having a raising structure. We take a close look at the analysis of (198), which repeats (85b), so as to illustrate the difference in more detail.

(198) Ken-ga hon-o yomi-sokoneru
Ken-NOM book-ACC read-fail
'Ken fails to read a book.'

Figure 33 shows the JACY output of the sentence. (199) illustrates the rule applications, which are identical to those of the A type.

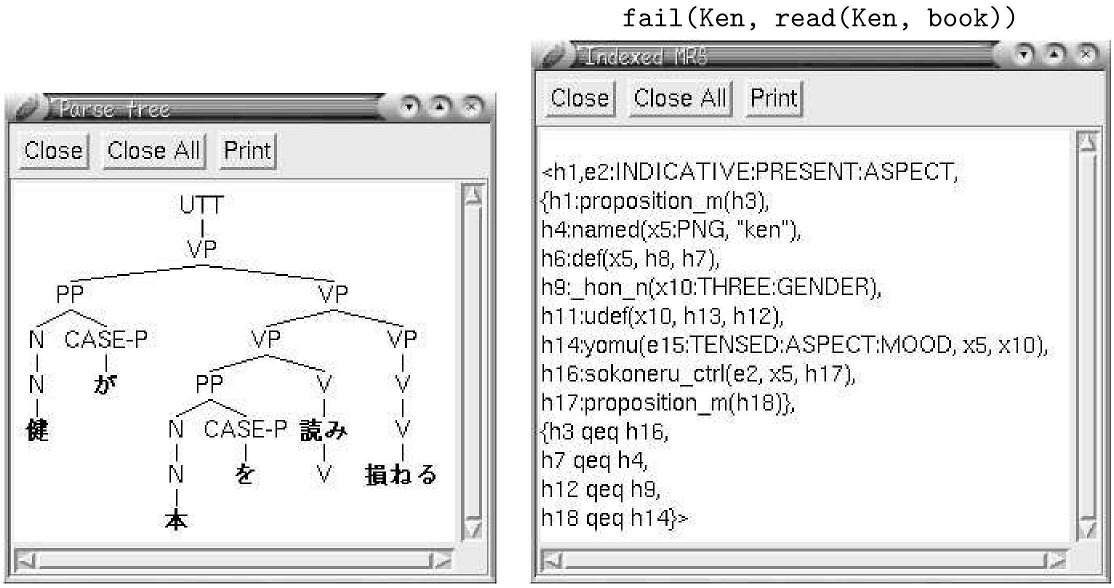
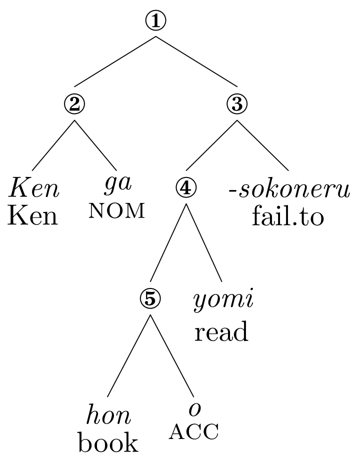


Figure 33: *Ken-ga hon-o yomi-sokoneru*

(199)



① HEAD-SUBJECT RULE
② HEAD-COMPLEMENT RULE
③ HEAD-SPECIFIER RULE
④ HEAD-COMPLEMENT RULE
⑤ HEAD-COMPLEMENT RULE

However, the semantic structure of the B type is a bit different from that of the A type. Look at the first row of Table 17. Notice that the B type verb, *sokoneru*, has two semantic arguments (`u6` and `h9`), unlike the A type verb, *kakeru*, and that, as a control structure, the first semantic argument of *sokoneru* controls the first semantic argument of *yomu*, as indicated in (200).

(200) `h8:sokoneru_ctrl(e2,u6,h9)` (V_2)

CONTROL

`h4:yomu(e7,TENSED:ASPECT:MOOD,u6,u5)` (V_1)

Table 17: The semantic composition of *Ken-ga hon-o yomi-sokoneru*

yomi-sokoneru read-fail.to	<h1,e2:INDICATIVE:PRESENT:ASPECT, {h1:proposition_m(h3), h4:yomu(e7:TENSED:ASPECT:MOOD, u6, u5), h8:sokoneru_ctrl(e2, u6, h9), h9:proposition_m(h10)}, {h3 qeq h8, h10 qeq h4}>
hon-o book-ACC yomi-sokoneru read-fail.to	<h1,e2:INDICATIVE:PRESENT:ASPECT, {h1:proposition_m(h3), h4:_hon_n(x5:THREE:GENDER), h6:udef(x5, h8, h7), h9:yomu(e11:TENSED:ASPECT:MOOD, u10, x5), h12:sokoneru_ctrl(e2, u10, h13), h13:proposition_m(h14)}, {h3 qeq h12, h7 qeq h4, h14 qeq h9}>
Ken-ga hon-o Ken-NOM book-ACC yomi-sokoneru read-fail.to	<h1,e2:INDICATIVE:PRESENT:ASPECT, {h1:proposition_m(h3), h4:named(x5:PNG, "ken"), h6:def(x5, h8, h7), h9:_hon_n(x10:THREE:GENDER), h11:udef(x10, h13, h12), h14:yomu(e15:TENSED:ASPECT:MOOD, x5, x10), h16:sokoneru_ctrl(e2, x5, h17), h17:proposition_m(h18)}, {h3 qeq h16, h7 qeq h4, h12 qeq h9, h18 qeq h14}>

In the second row, the second semantic argument of *yomu* is specified as *hon-o* that is represented by `x5`. In the third row, the subject *Ken-ga* `x5` is introduced, which fills the first argument positions of *yomu* and *sokoneru*. At this point, *hon-o* is represented by `x10`.

B.2.3 C type

The C type is different from the previous two types in that it constitutes a V embedding structure, rather than a VP embedding and that both the V_1 and the V_2 put thematic restrictions on a subject and an object. Let's examine the analysis of (201).

(201) Ken-ga hon-o yomi-naosu
Ken-NOM book-ACC read-do.again
'Ken reads a book again.'

The JACY output of the sentence is given in Figure 34. (202) shows which node is licensed

`do.again(Ken, book, read(Ken, book))`

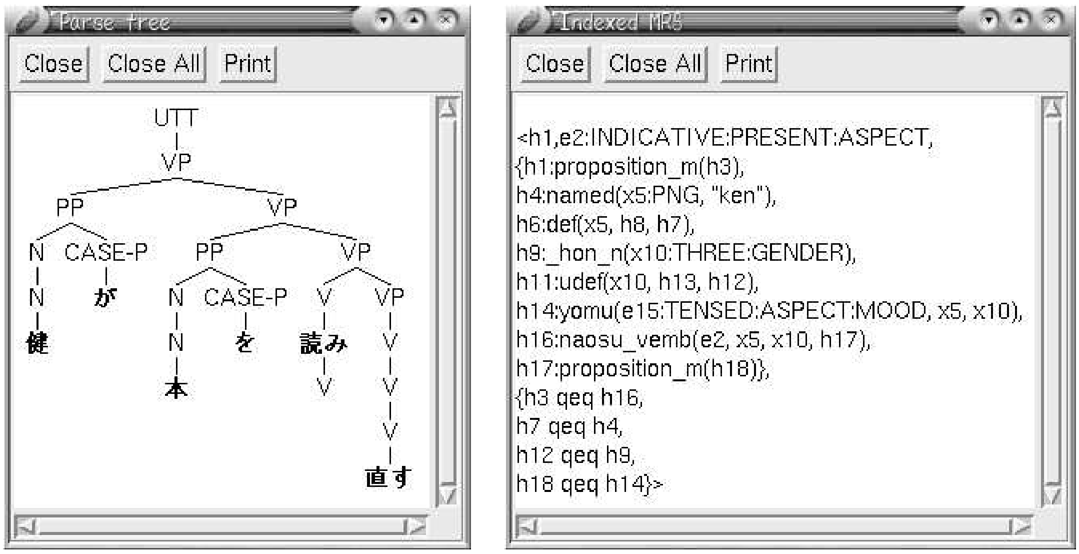
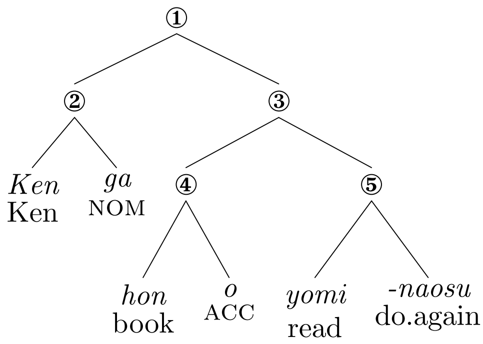


Figure 34: *Ken-ga hon-o yomi-naosu*

by which rule.

(202)



① HEAD-SUBJECT RULE
② HEAD-COMPLEMENT RULE
③ HEAD-COMPLEMENT RULE
④ HEAD-COMPLEMENT RULE
⑤ HEAD-SPECIFIER RULE

It is shown in (202) that the C type has a V embedding structure. Table 18 presents the semantic composition of the sentence, *Ken-ga hon-o yomi-naosu.* In the first row, there are two propositions, `h1:proposition_m(h3)` and `h9:proposition_m(h10)`, which is the same as the previous two types. Note, however, that, unlike the previous two types, two control relations hold between the V_1, *yomu*, and the V_2, *naosu*. Look at (203).

(203) `h8:naosu_vemb(e2,u6,u5,h9)` (V_2)

CONTROL

`h4:yomu(e7,TENSED:ASPECT:MOOD,u6,u5,)` (V_1)

As I discussed in §3.5.1, both the V_1 and the V_2 assign thematic roles to both a subject and an object (the `u6` and the `u5` in the case of (203)), and the first and the second semantic arguments of the V_2 control those of the V_1.

B.3 Lexical V_1-V_2 compound

As described in §3.6, there were five types in the lexical V_1-V_2 compounding, four of which were (partially) compositional. I describe those four types in more detail in this section.

B.3.1 Right headed V_1-V_2

First, I describe the analysis of the Right headed V_1-V_2 through the example below.

(204) huku-ga ki-kuzureru
clothes-NOM wear-get.out.of.shape
'The clothes get out of the shape due to (someone's) wearing it.'

The JACY output of the sentence is shown in Figure 35. The V_1-V_2 compound, *ki-kuzureru*, is licensed by the *monotrans-monounac* rule, which is one of the 29 Right headed V_1-V_2 rules. (205) illustrates this.

Table 18: The semantic composition of *Ken-ga hon-o yomi-naosu*

yomi-naosu read-do.again	<h1,e2:INDICATIVE:PRESENT:ASPECT, {h1:proposition_m(h3), h4:yomu(e7:TENSED:ASPECT:MOOD, u6, u5), h8:naosu_vemb(e2, u6, u5, h9), h9:proposition_m(h10)}, {h3 qeq h8, h10 qeq h4}>
hon-o yomi-naosu book-ACC read-do.again	<h1,e2:INDICATIVE:PRESENT:ASPECT, {h1:proposition_m(h3), h4:_hon_n(x5:THREE:GENDER), h6:udef(x5, h8, h7), h9:yomu(e11:TENSED:ASPECT:MOOD, u10, x5), h12:naosu_vemb(e2, u10, x5, h13), h13:proposition_m(h14)}, {h3 qeq h12, h7 qeq h4, h14 qeq h9}>
Ken-ga hon-o Ken-NOM book-ACC yomi-naosu read-do.again	<h1,e2:INDICATIVE:PRESENT:ASPECT, {h1:proposition_m(h3), h4:named(x5:PNG, "ken"), h6:def(x5, h8, h7), h9:_hon_n(x10:THREE:GENDER), h11:udef(x10, h13, h12), h14:yomu(e15:TENSED:ASPECT:MOOD, x5, x10), h16:naosu_vemb(e2, x5, x10, h17), h17:proposition_m(h18)}, {h3 qeq h16, h7 qeq h4, h12 qeq h9, h18 qeq h14}>

```
relation(∃u wear(u, clothes),
         get.out.of.shape(clothes))
```

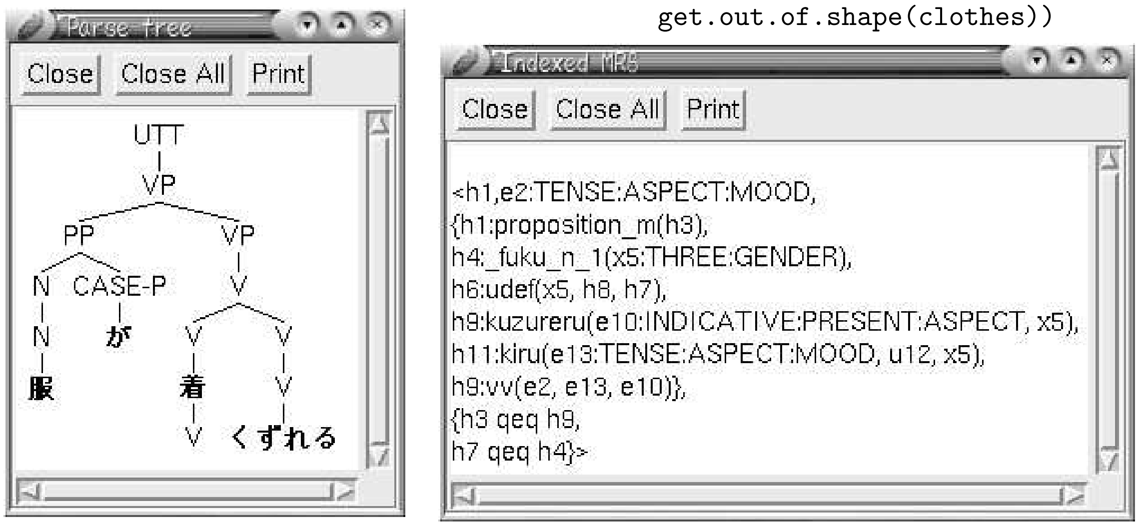
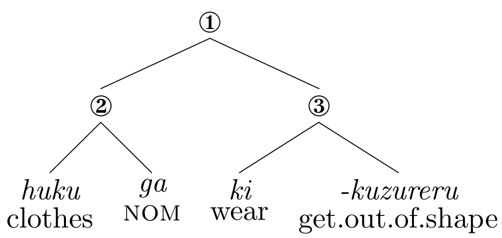


Figure 35: *huku-ga ki-kuzureru*

(205)



① HEAD-SUBJECT RULE
② HEAD-COMPLEMENT RULE
③ MONOTRANS-MONOUNAC RULE

Table 19 shows the semantic composition of the sentence. Unlike the syntactic V_1-V_2 compounds, there is only one proposition in the semantic representation: `h1:proposition_m(h3)`.

(206) `h1:proposition_m(h3)`

```
h3 qeq h4

h4:vv(e2,e9,e6)

{ h4:kuzureru(e6:INDICATIVE:PRESENT:ASPECT,u5)   (V2)
{ h7:kiru(e9:TENSE:ASPECT:MOOD,u8,u5)   (V1)
```

The proposition refers to the event described by the V_1 (`kiru`), the V_2 (`kuzureru`), and the semantic relation (`vv`) between the V_1 and the V_2. As discussed in §3.6, I underspecified the

Table 19: The semantic composition of *huku-ga ki-kuzureru*

yomi-naosu read-do.again	<h1,e2:TENSE:ASPECT:MOOD, {h1:proposition_m(h3), h4:kuzureru(e6:INDICATIVE:PRESENT:ASPECT, u5), h7:kiru(e9:TENSE:ASPECT:MOOD, u8, u5), h4:vv(e2, e9, e6)}, {h3 qeq h4}>
huku-ga clothes-NOM ki-kuzureru wear-get.out.of.shape	<h1,e2:TENSE:ASPECT:MOOD, {h1:proposition_m(h3), h4:_fuku_n_1(x5:THREE:GENDER), h6:udef(x5, h8, h7), h9:kuzureru(e10:INDICATIVE:PRESENT:ASPECT, x5), h11:kiru(e13:TENSE:ASPECT:MOOD, u12, x5), h9:vv(e2, e13, e10)}, {h3 qeq h9, h7 qeq h4}>

semantic relation (`vv`) of the Right headed V_1-V_2 that might be specified, by a pragmatic component, as one of the four relations: `pair`, `cause`, `manner`, and `means`. Notice that the first semantic argument of the V_2, *kuzureru*, is co-indexed with the V_1's (*kiru*) second semantic argument, rather than the first argument, because of the *monotrans-monounac* rule. (207) describes this.

(207) `h4:kuzureru(e6:INDICATIVE:PRESENT:ASPECT,u5)` (V_2)

× ○ CO-INDEXED

`h7:kiru(e9:TENSE:ASPECT:MOOD,u8,u5)` (V_1)

B.3.2 Argument mixing V_1-V_2

I described the Argument mixing V_1-V_2 in §3.6.3, where it was shown that the compound shows ambiguity as to which component verb contributes an object argument. This is depicted in (208), which repeats (139) on page 92.

(208) *nomi-aruku* (drink-walk) 'drink something while walking'

a. ×

Ken-ga sake-o nomi-aruku

Ken-NOM sake-ACC drink-walk

○

'Ken drinks sake around somewhere.' (*sake* is V_1's argument.)

b.

○		
Ken-ga	Tokyo-o	nomi·aruku
Ken-NOM	Tokyo-ACC	drink-walk
	×	

'Ken drinks around Tokyo.' (*Tokyo* is V_2's argument.)

Figures 36 and 37 show the JACY output of (208a). There are two semantics outputs

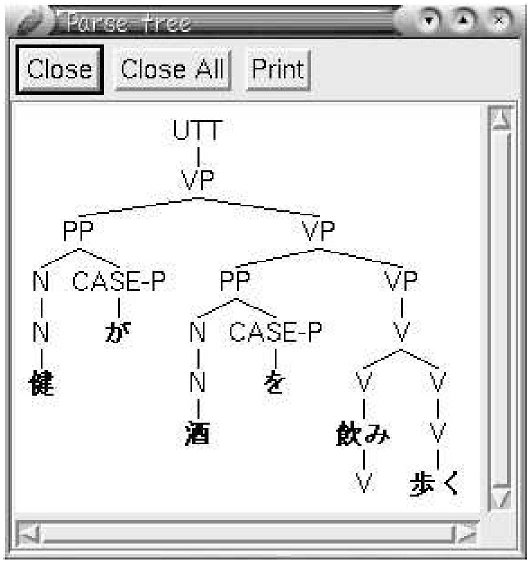
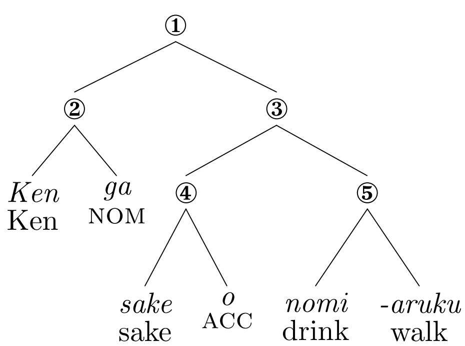


Figure 36: *Ken-ga sake-o nomi-aruku*: syntax

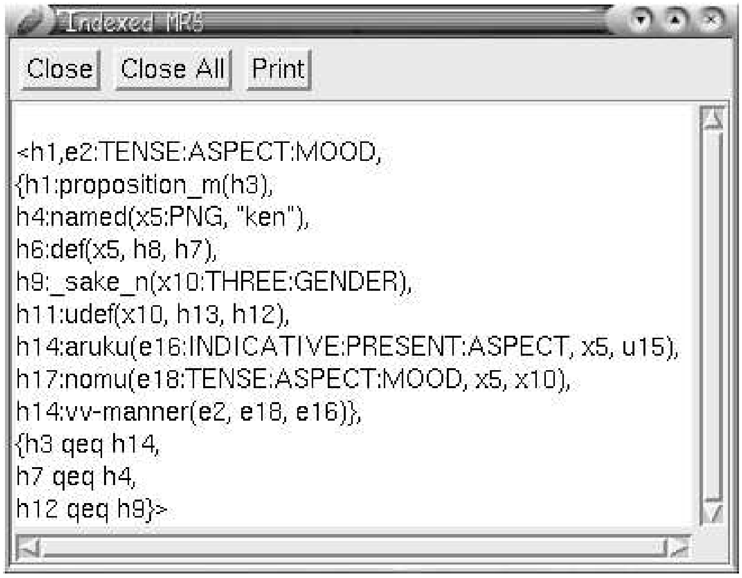
in Figure 37 because of the ambiguity of the Argument mixing V_1-V_2. The upper side corresponds to the semantics of (208a), where the V_1 contributes the object, while, in the figure on the lower side, it is the V_2 that contributes the object. I call the former "am-V_1" and the latter "am-V_2". Next, let us look at (209), which shows the rule applications of the sentence.

(209)



① HEAD-SUBJECT RULE
② HEAD-COMPLEMENT RULE
③ HEAD-COMPLEMENT RULE
④ HEAD-COMPLEMENT RULE
⑤ ARGUMENT-MIXING RULE

am-V_1: `relation(drink(Ken, sake), ∃u walk(Ken, u))`



am-V_2: `relation(∃u drink(Ken, u), walk(Ken, sake))`

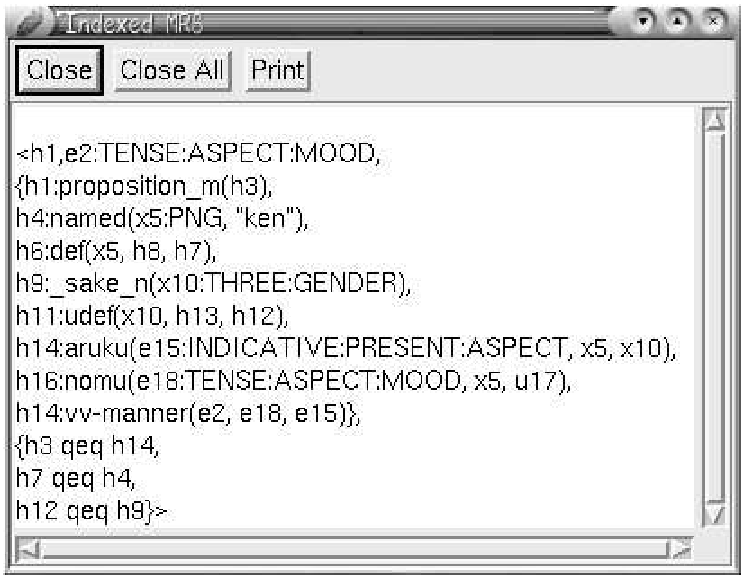


Figure 37: *Ken-ga sake-o nomi-aruku*: semantics

It is shown that the argument-mixing V_1-V_2 rule combines the V_1, *nomu*, and the V_2, *aruku*. Next we move on to the semantic composition of the sentence. Look at Table 20 and Table 21. Table 20 corresponds to the am-V_1, and Table 21 represents the am-V_2. There

Table 20: The semantic composition of *Ken-ga sake-o nomi-aruku*: am-V_1

nomi-aruku drink-walk	<h1,e2:TENSE:ASPECT:MOOD, {h1:proposition_m(h3), h4:aruku(e7:INDICATIVE:PRESENT:ASPECT, u6, u5), h8:nomu(e10:TENSE:ASPECT:MOOD, u6, u9), h4:vv-manner(e2, e10, e7)}, {h3 qeq h4}>
sake-o nomi-aruku sake-ACC drink-walk	<h1,e2:TENSE:ASPECT:MOOD, {h1:proposition_m(h3), h4:_sake_n(x5:THREE:GENDER), h6:udef(x5, h8, h7), h9:aruku(e12:INDICATIVE:PRESENT:ASPECT, u11, u10), h13:nomu(e14:TENSE:ASPECT:MOOD, u11, x5), h9:vv-manner(e2, e14, e12)}, {h3 qeq h9, h7 qeq h4}>
Ken-ga sake-o Ken-NOM sake-ACC nomi-aruku drink-walk	<h1,e2:TENSE:ASPECT:MOOD, {h1:proposition_m(h3), h4:named(x5:PNG, "ken"), h6:def(x5, h8, h7), h9:_sake_n(x10:THREE:GENDER), h11:udef(x10, h13, h12), h14:aruku(e16:INDICATIVE:PRESENT:ASPECT, x5, u15), h17:nomu(e18:TENSE:ASPECT:MOOD, x5, x10), h14:vv-manner(e2, e18, e16)}, {h3 qeq h14, h7 qeq h4, h12 qeq h9}>

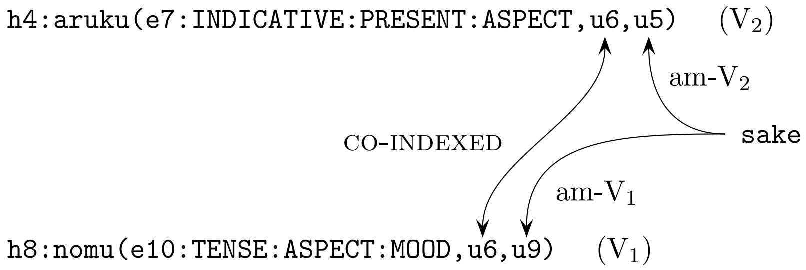
is no difference between the first rows of the two tables. But it differs from that of the Right headed V_1-V_2 in that the semantic relation between the V_1 and the V_2 is specified as `manner`, as indicated by `h4:vv-manner(e2,e10,e7)`, in accord with the analysis I gave in §3.6.3. Note also that the compounding gives rise to co-indexing only between the two first semantic arguments between the V_1 and the V_2, with the two second semantic arguments left separated. In the am-V_1, the *sake* goes into the V_1's second semantic argument position,

Table 21: The semantic composition of *Ken-ga sake-o nomi-aruku*: am-V_2

nomi-aruku drink-walk	<h1,e2:TENSE:ASPECT:MOOD, {h1:proposition_m(h3), h4:aruku(e7:INDICATIVE:PRESENT:ASPECT, u6, u5), h8:nomu(e10:TENSE:ASPECT:MOOD, u6, u9), h4:vv-manner(e2, e10, e7)}, {h3 qeq h4}>
sake-o nomi-aruku sake-ACC drink-walk	<h1,e2:TENSE:ASPECT:MOOD, {h1:proposition_m(h3), h4:_sake_n(x5:THREE:GENDER), h6:udef(x5, h8, h7), h9:aruku(e11:INDICATIVE:PRESENT:ASPECT, u10, x5), h12:nomu(e14:TENSE:ASPECT:MOOD, u10, u13), h9:vv-manner(e2, e14, e11)}, {h3 qeq h9, h7 qeq h4}>
Ken-ga sake-o Ken-NOM sake-ACC nomi-aruku drink-walk	<h1,e2:TENSE:ASPECT:MOOD, {h1:proposition_m(h3), h4:named(x5:PNG, "ken"), h6:def(x5, h8, h7), h9:_sake_n(x10:THREE:GENDER), h11:udef(x10, h13, h12), h14:aruku(e15:INDICATIVE:PRESENT:ASPECT, x5, x10), h16:nomu(e18:TENSE:ASPECT:MOOD, x5, u17), h14:vv-manner(e2, e18, e15)}, {h3 qeq h14, h7 qeq h4, h12 qeq h9}>

while it goes into the V_2's second semantic argument position in the am-V_2. (210) describes this.

(210)



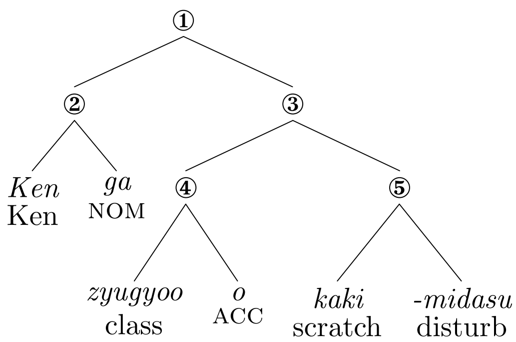
B.3.3 V_1-V_2 with semantically deverbalized V_1

This section presents the detailed analysis of the V_1-V_2 with semantically deverbalized V_1, the example of which is given in (211).

(211) Ken-ga zyugyoo-o kaki-midasu
Ken-NOM class-ACC scratch-disturb
'Ken disturbs the class.'

Figure 38 shows the JACY output of the sentence. (212) shows which rule licenses which node. Note that the V_1-V_2 compound, *kaki-midasu*, is licensed by the PREFIX-V1-ATTACH RULE.

(212)



① HEAD-SUBJECT RULE
② HEAD-COMPLEMENT RULE
③ HEAD-COMPLEMENT RULE
④ HEAD-COMPLEMENT RULE
⑤ PREFIX-V1-ATTACH RULE

Table 22 illustrates how the semantic composition of the example sentence proceeds. In §3.6.4, I analyzed the V_1 of the compound as a prefix and assumed that the V_1 does not contribute to the meaning of the V_1-V_2. Actually, there is no formula representing the semantics of the V_1, *kaku*, in any row. Instead, the PREFIX-V1-ATTACH RULE introduces `vv-prefix-v1`, which informs us that the semantics represents the meaning of the V_1-V_2 with semantically deverbalized V_1.[4]

[4]There are additional propositions in the second and the third rows: `h6:proposition_m(h7)` (in the second row) and `h11:proposition_m(h12)` (in the third row). They are introduced by the verbal noun

Table 22: The semantic composition of *Ken-ga zyugyoo-o kaki-midasu*

kaki-midasu scratch-disturb	<h1,e2:PRESENT:ASPECT:INDICATIVE, {h1:proposition_m(h3), h4:midasu_1(e2, u6, u5), h7:vv-prefix-v1(e2, h8)}, {h3 qeq h7, h8 qeq h4}>
zyugyoo-o class-ACC kaki-midasu scratch-disturb	<h1,e2:PRESENT:ASPECT:INDICATIVE, {h1:proposition_m(h3), h4:noun-relation(x5:PNG, h6), h6:proposition_m(h7), h8:_jugyou_vn_1(x5, u9), h10:udef(x5, h12, h11), h13:midasu_1(e2, u14, x5), h15:vv-prefix-v1(e2, h16)}, {h3 qeq h15, h7 qeq h8, h11 qeq h4, h16 qeq h13}>
Ken-ga zyugyoo-o Ken-NOM class-ACC kaki-midasu scratch-disturb	<h1,e2:PRESENT:ASPECT:INDICATIVE, {h1:proposition_m(h3), h4:named(x5:PNG, "ken"), h6:def(x5, h8, h7), h9:noun-relation(x10:PNG, h11), h11:proposition_m(h12), h13:_jugyou_vn_1(x10, u14), h15:udef(x10, h17, h16), h18:midasu_1(e2, x5, x10), h19:vv-prefix-v1(e2, h20)}, {h3 qeq h19, h7 qeq h4, h12 qeq h13, h16 qeq h9, h20 qeq h18}>

disturb(Ken, class)

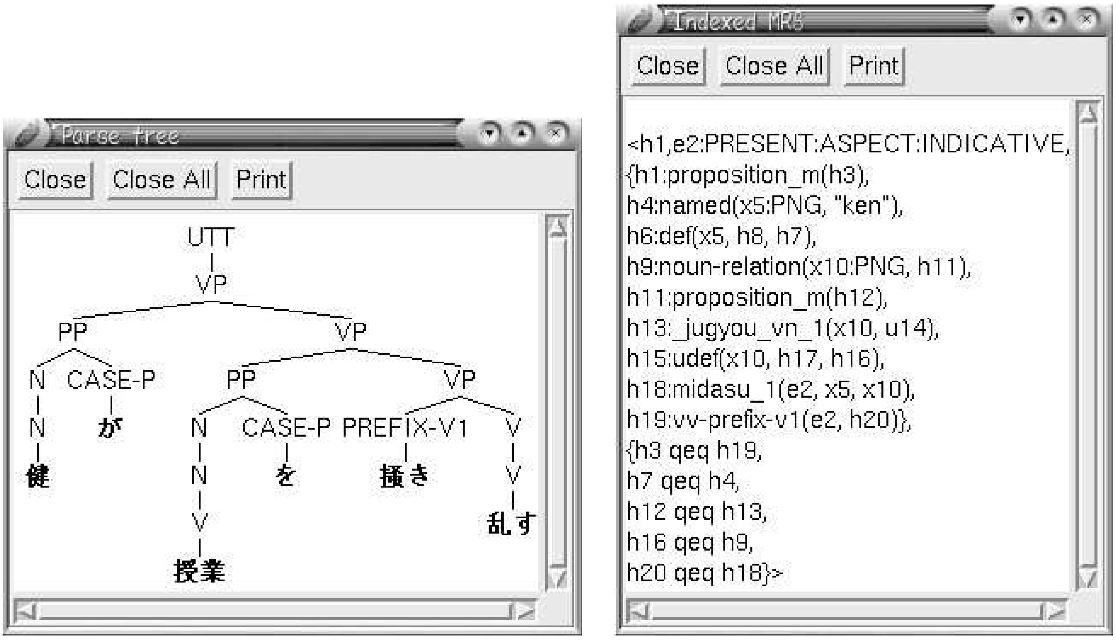


Figure 38: *Ken-ga zyugyoo-o kaki-midasu*

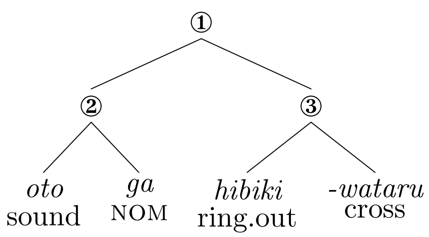
B.3.4 V_1-V_2 with semantically deverbalized V_2

As described in §3.6.5, the V_1-V_2 with semantically deverbalized V_2 has a semantic embedding structure. Let us examine the characteristic through the example in (213).

(213) oto-ga hibiki-wataru
sound-NOM ring.out-cross
'The sound echoes.'

Its JACY output is given in Figure 39. (214) shows the rule applications.

(214)



① HEAD-SUBJECT RULE
② HEAD-COMPLEMENT RULE
③ EVENT-EMBEDDING-NON-AGENTIVE RULE

It is shown that the EVENT-EMBEDDING-NON-AGENTIVE RULE licenses the V_1-V_2 compound. Table 23 shows the semantic composition of the sentence, *oto-ga hibiki-wataru*. We

zyugyoo 'class' (jugyou_vn_1 in the semantic representations), since a verbal noun can mean a proposition. But we can leave the propositions out of consideration in this illustration.

cross(ring.out(sound))

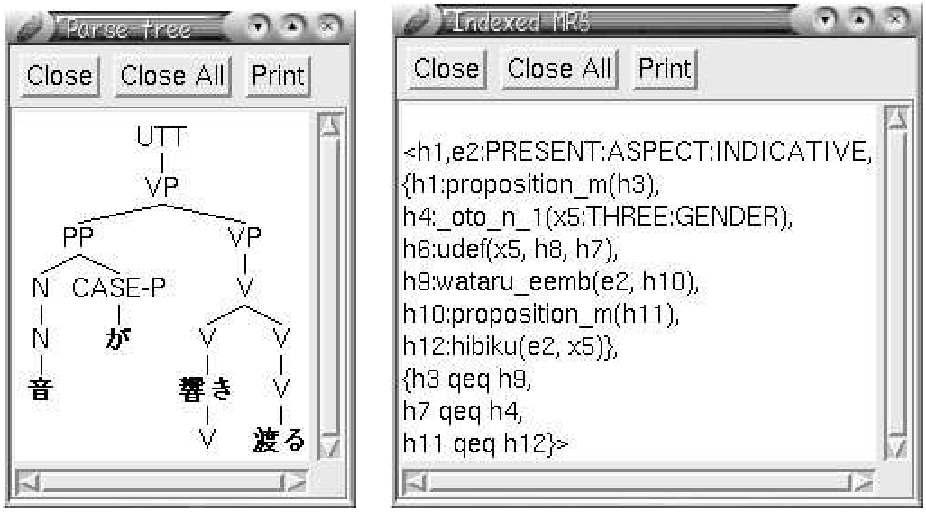


Figure 39: The JACY output of *oto-ga hibiki-wataru*

Table 23: The semantic composition of *oto-ga hibiki-wataru*

hibiki-wataru ring.out-cross	<h1,e2:PRESENT:ASPECT:INDICATIVE, {h1:proposition_m(h3), h4:wataru_eemb(e2, h5), h5:proposition_m(h6), h7:hibiku(e2, u8)}, {h3 qeq h4, h6 qeq h7}>
oto-ga sound-NOM hibiki-wataru ring.out-cross	<h1,e2:PRESENT:ASPECT:INDICATIVE, {h1:proposition_m(h3), h4:_oto_n_1(x5:THREE:GENDER), h6:udef(x5, h8, h7), h9:wataru_eemb(e2, h10), h10:proposition_m(h11), h12:hibiku(e2, x5)}, {h3 qeq h9, h7 qeq h4, h11 qeq h12}>

should notice that, in the first row, the semantic representation contains two propositions: `h1:proposition_m(h3)` and `h5:proposition_m(h6)`. The former stands for the meaning of the sentence as a whole, while the latter refers to the event represented by the V_1, *hibiku*. The V_2, *wataru*, semantically embeds the latter proposition, which is illustrated in (215).

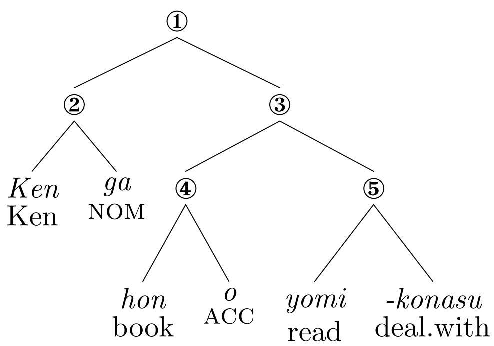
(215)
```
h1:proposition_m(h3)
  h3 qeq h4
    h4:wataru_eemb(e2,h5)    (V2)
      h5:proposition_m(h6)
        h6 qeq h7
          h7:hibiku(e2,u8)    (V1)
```

hibiki-wataru consists of the two non agentive verbs. Following is another example of the V_1-V_2 with semantically deverbalized V_2, which consists of two agentive verbs.

(216) Ken-ga hon-o yomi-konasu
Ken-NOM book-ACC read-deal.with
'Ken reads a book competently.'

Figure 40 is the JACY output of the sentence. Notice that, in (217), the V_1-V_2, *yomi-konasu*, is licensed by the EVENT-EMBEDDING-AGENTIVE RULE, rather than the EVENT-EMBEDDING-NON-AGENTIVE RULE.

(217)



① HEAD-SUBJECT RULE
② HEAD-COMPLEMENT RULE
③ HEAD-COMPLEMENT RULE
④ HEAD-COMPLEMENT RULE
⑤ EVENT-EMBEDDING-AGENTIVE RULE

The semantic composition of *Ken-ga hon-o yomi-konasu* is depicted in Table 24. In the first row, both the semantics of the V_1, `yomu`, and that of the V_2, `konasu`, have an agentive argument, `u6`. The two agentive arguments are co-indexed, as described in (218).

Table 24: The semantic composition of *Ken-ga hon-o yomi-konasu*

yomi-konasu read-deal.with	<h1,e2:PRESENT:ASPECT:INDICATIVE, {h1:proposition_m(h3), h4:konasu_eemb(e2, u6, h5), h5:proposition_m(h7), h8:yomu(e2, u6, u9)}, {h3 qeq h4, h7 qeq h8}>
hon-o book-ACC yomi-konasu read-deal.with	<h1,e2:PRESENT:ASPECT:INDICATIVE, {h1:proposition_m(h3), h4:_hon_n(x5:THREE:GENDER), h6:udef(x5, h8, h7), h9:konasu_eemb(e2, u11, h10), h10:proposition_m(h12), h13:yomu(e2, u11, x5)}, {h3 qeq h9, h7 qeq h4, h12 qeq h13}>
Ken-ga hon-o Ken-NOM book-ACC yomi-konasu read-deal.with	<h1,e2:PRESENT:ASPECT:INDICATIVE, {h1:proposition_m(h3), h4:named(x5:PNG, "ken"), h6:def(x5, h8, h7), h9:_hon_n(x10:THREE:GENDER), h11:udef(x10, h13, h12), h14:konasu_eemb(e2, x5, h15), h15:proposition_m(h16), h17:yomu(e2, x5, x10)}, {h3 qeq h14, h7 qeq h4, h12 qeq h9, h16 qeq h17}>

deal.with(Ken, read(Ken, book))

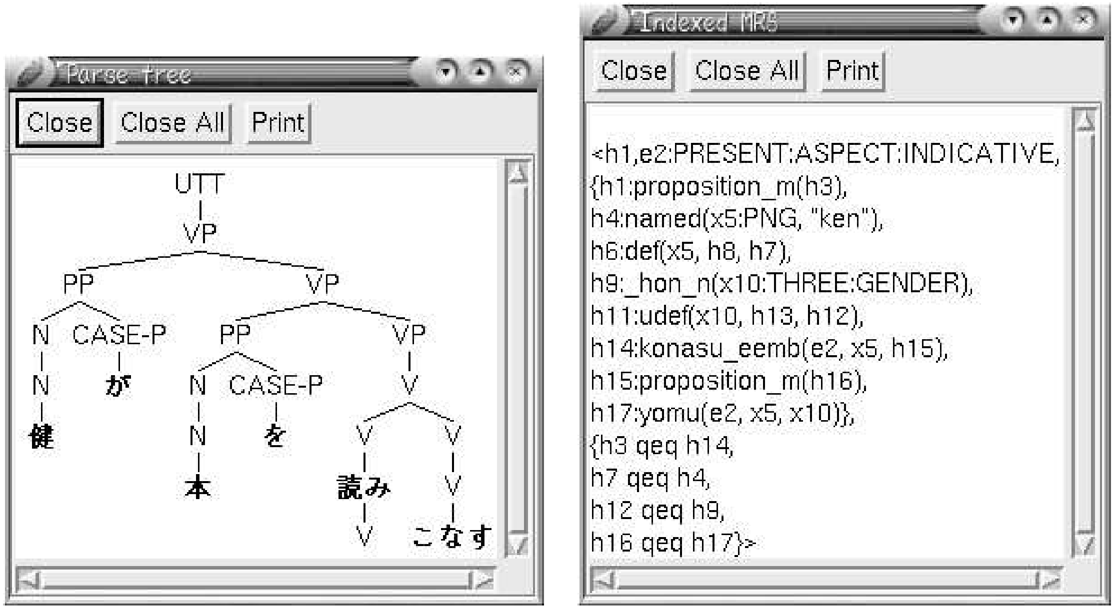


Figure 40: *Ken-ga hon-o yomi-konasu*

(218) h1:proposition_m(h3)

h3 qeq h4

h4:konasu_eemb(e2,u6 ,h5) (V_2)

h5:proposition_m(h7)

h7 qeq h8

h8:yomu(e2,u6 ,u9) (V_1)

CO-INDEXED AGENTIVE ARGUMENTS

Bibliography

Allen, J. (1994). *Natural Language Understanding (2nd ed.)*. Benjamin/Cummings Publishing Company.

Amano, S., & Kondo, T. (1999). *Lexical properties of Japanese* (in Japanese). Sanseido.

Aronoff, M. (1976). *Word Formation in Generative Grammar*. Linguistic Inquiry Monograph 1. MIT Press, Cambridge.

Baker, M. (1989). *Incorporation: A Theory of Grammatical Function Changing*. University of Chicago Press.

Baldwin, T., Bannard, C., Tanaka, T., & Widdows, D. (2003). An Empirical Model of Multiword Expression Decomposability. In *Proceedings of the ACL-2003 Workshop on Multiword Expressions: Analysis, Acquisition and Treatment*, pp. 89–96 Sapporo, Japan.

Baldwin, T., & Bond, F. (2002). Multiword Expressions: Some Problems for Japanese NLP. In *Proceedings of the Eighth Annual Meeting of the Association of Natural Language Processing, Japan*, pp. 379–382 Japan, Keihanna, Japan.

Bannard, C., Baldwin, T., & Lascarides, A. (2003). A Statistical Approach to the Semantics of Verb-Particles. In *Proceedings of the ACL-2003 Workshop on Multiword Expressions: Analysis, Acquisition and Treatment*, pp. 65–72 Sapporo, Japan.

Bender, E. M., Flickinger, D. P., & Oepen, S. (2002). The Grammar Matrix: An Open-Source Starter-Kit for the Rapid Development of Cross-Linguistically Consistent Broad-Coverage Precision Grammars. In Carroll, J., Oostdijk, N., & Sutcliffe, R. (Eds.), *Proceedings of the Workshop on Grammar Engineering and Evaluation at the 19th International Conference on Computational Linguistics*, pp. 8–14 Taipei, Taiwan.

Bond, F., Fujita, S., Hashimoto, C., Nariyama, S., Nichols, E., Ohtani, A., Tanaka, T., & Amano, S. (2004a). The Hinoki Treebank — A Treebank for Text Understanding. In *Proceedings of the First International Joint Conference of Natural Language Processing*, pp. 554–559.

Bond, F., Fujita, S., Hashimoto, C., Nariyama, S., Nichols, E., Ohtani, A., Tanaka, T., & Amano, S. (2004b). The Hinoki Treebank — Toward Text Understanding. In *Proceedings of the 5th International Workshop on Linguistically Interpreted Corpora (LINK-04)*, pp. 7–10 Geneva.

Bond, F., Nichols, E., & Tanaka, S. F. T. (2004c). Acquiring an Ontology for a Fundamental Vocabulary. In *20th International Conference onh Computational Linguistics (COLING-2004)*, pp. 1319–1325 Geneva.

Bresnan, J., & Kaplan, R. M. (1982). Lexical-Functional Grammar: A formal system for grammatical representation. In Bresnan, J. (Ed.), *The Mental Representation of Grammatical Relations*, pp. 173–281. The MIT Press.

Briscoe, T., & Copestake, A. (1999). Lexical rules in constraint-based grammar. *Computational Linguistics 25:4*, 487–526.

Briscoe, T. J., Copestake, A., & Lascarides, A. (1995). Blocking. In Saint-Dizier, P., & Viegas, E. (Eds.), *Computational Lexical Semantics*, pp. 273–302. Cambridge University Press.

Burzio, L. (1986). *Italian Syntax*. D. Reidel.

Butt, M., Dyvik, H., King, T. H., Masuichi, H., & Rohrer, C. (2002). The Parallel Grammar Project. In *Proceedings of COLING 2002 Workshop on Grammar Engineering and Evaluation*, pp. 1–7.

Callmeier, U. (2000). PET — A Platform for Experimenting with Efficient HPSG Processing Techniques. *Journal of Natural Language Engineering. Special Issue on Efficient Processing with HPSG: Methods, Systems, Evaluation 6(1)*, 99–108.

Callmeier, U., Eisele, A., Schäfer, U., & Siegel, M. (2004). The DeepThought Core Architecture Framework. In *Proceedings of LREC 04*, pp. 1205–1208 Lisbon, Portugal.

Carpenter, B. (1992). *The Logic of Typed Feature Structures*. Cambridge University Press.

Charniak, E. (2001). Immediate-head parsing for language models. In *Proceedings of the 39th Annual Meeting of the ACL and 10th Conference of the EACL*, pp. 116–123.

Chomsky, N. (1957). *Syntactic Structures*. Mouton, The Hague.

Chomsky, N. (1965). *Aspects of the Theory of Syntax*. MIT Press.

Chomsky, N. (1981). *Lectures on Government and Binding: The Pisa Lectures.* Foris, Dordrecht.

Chomsky, N. (1995). *The Minimalist Program.* MIT Press.

Clark, S., Hockenmaier, J., & Steedman, M. (2002). Building Deep Dependency Structures with a Wide-Coverage CCG Parser. In *the 40th Annual Meeting of the Association for Computational Linguistics*, pp. 327–334.

Copestake, A. (1992). The Representation of Lexical Semantic Information. Cognitive science research papers 280, University of Sussex.

Copestake, A. (2002). *Implementing Typed Feature Structure Grammars.* CSLI Publications.

Copestake, A., Flickinger, D., Malouf, R., Riehemann, S., & Sag, I. (1995). Translation using Minimal Recursion Semantics. In *Proceedings of the 6th. International Conference on Theoretical and Methodological Issues in Machine Translation (TMI-95)*, pp. 15–32 Leuven. Pages.

Copestake, A., Flickinger, D. P., & Sag, I. A. (1999). Minimal Recursion Semantics: An Introduction. Manuscript, Stanford University: CSLI.

Copestake, A., Lascarides, A., & Flickinger, D. (2001). An Algebra for Semantic Construction in Constraint-based Grammars. In *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics (ACL 2001)*, pp. 132–139 Toulouse, France.

Daniels, M. W., & Meurers, W. D. (2004). A Grammar Formalism and Parser for Linearization-based HPSG. In *Proceedings of Coling 2004*, pp. 169–175 Geneva, Switzerland. COLING.

Dorr, B. J. (2001). LCS Verb Database, Online Software Database of Lexical Conceptual Structures and Documentation. Online HTML document, *http://www.umiacs.umd.edu/~bonnie/LCS_Database_Documentation.html*, UMCP.

Dowty, D. R. (1991). Thematic Proto-Roles and Argument Selection. *Language 67, 3*, 547–619.

Dowty, D. R., Wall, R. E., & Peters, S. (1981). *Introduction to Montague Semantics.* D. Reidel, Dordrecht.

Fillmore, C. (1968). The Case for Case. In Bach, E., & Harms, R. (Eds.), *Universals in Linguistic Theory*, pp. 1–88. New York: Holt, Rinehart and Winston.

Fillmore, C. J. (1982). Frame semantics. *Linguistics in the Morning Calm*, 111–137.

Flickinger, D. (2000). On building a more efficient grammar by exploiting types. *Natural Language Engineering, 6 (1) (Special Issue on Efficient Proceeding with HPSG*, 15–28.

Flickinger, D. P. (1987). *Lexical Rules in the Hierarchical Lexicon*. Ph.D. thesis, Stanford University.

Flickinger, D. P., & Bender, E. M. (2003). Compositional Semantics in a Multilingual Grammar Resource. In Bender, E. M., Flickinger, D. P., Fouvry, F., & Siegel, M. (Eds.), *Proceedings of the Workshop on Ideas and Strategies for Multilingual Grammar Devel, ESSLLI 2003opment*, pp. 33–42.

Fukushima, K. (2003). A Neo Lexical Account for the Compounding Complexities of V-V Compounds in Japanese. draft.

Grimshaw, J. (1990). *Argument Structure*. MIT Press.

Gunji, T. (1987). *Japanese Phrase Structure Grammar*. D. Reidel Publishing Company.

Gunji, T. (1999). On Lexicalist Treatments of Japanese Causatives. In Levine, R. D., & Green, G. M. (Eds.), *Studies in Contemporary Phrase Structure Grammar*, pp. 119–160. Cambridge University Press.

Hale, K., & Keyser, S. J. (1987). A View from the Middle. *Lexicon Working Papers 10, MIT*.

Harada, S. (1973). Counter-Equi NP Deletion. In *Annual Bulletin*, No. 7, pp. 113–148. Research Institute of Logopaedics and Phoniatrics, Tokyo University.

Hasegawa, N. (1980). The VP constituent in Japanese..

Hashimoto, C. (2003a). Construction of a Working Japanese HPSG Grammar on Computers (in Japanese). In *Proceedings of the 9th annual meeting of the Association for Natural Language Processing*, pp. 469–472 Yokohama, Japan.

Hashimoto, C. (2003b). HPSG analysis of Long Distance Passives (in Japanese). In *Proceedings of the 126th annual meeting of Linguistic Society of Japan*, pp. 256–261 Tokyo, Japan.

Hashimoto, C. (2003c). Japanese HPSG: Treatment of syntax and semantics of syntactically complex verbs (in Japanese). In *Information Processing Society of Japan, 2003-NL-156*, pp. 31–38 Yamagata, Japan.

Hasida, K. (1997). Information Science Approach to Language. In Ôtsu, Y., Gunji, T., Takubo, Y., Nagao, M., Hasida, K., Masuoka, T., & Matsumoto, Y. (Eds.), *An Introduction to Language Science* (in Japanese), chap. 3. Iwanami.

Himeno, M. (1999). *The structure and semantics of compond verbs* (in Japanese). Hitsuji Shobou.

Hinds, J. (1973). On the status of the VP node in Japanese. *Language Research (Seoul)*, 44–57.

Hinrichs, E., & Nakazawa, T. (1994). Linearizing AUXs in German Verbal Complexes. In Nerbonne, J., Netter, K., & Pollard, C. J. (Eds.), *German in Head-Driven Phrase Structure Grammar*, No. 46 in CSLI Lecture Notes, pp. 11–37. CSLI Publications, Stanford University.

Hoji, H. (1985). *Logical form constraints and configurational structures in Japanese*. Ph.D. dissertation, University of Washington.

Hopcroft, J. E., Motwani, R., & Ullman, J. D. (2001). *Introduction to Automata Theory, Languages, and Computation, Second Edition*. Addison-Wesley.

Imaizumi, S., & Gunji, T. (2000). Complex Events in Lexical Compounds. In Itou, T., & Yatabe, S. (Eds.), *Lexicon and Syntax* (in Japanese), pp. 33–59. Hitsuji Shobou.

Inoue, K. (1976). *Transformational Grammar and Japanese* (in Japanese). Taisyûkan.

Kageyama, T. (1993). *Grammar and Word Formation* (in Japanese). Hitsuji Shobou.

Kageyama, T. (1999). Word Formation. In Tsujimura, N. (Ed.), *Japanese Linguistics*, pp. 297–325. Blackwell.

Kanasugi, Y., Kasahara, K., Inago, N., & Amano, S. (2002). Selection of a basic vocabulary based on word familiarity ratings. In *IEICE Technical Report NLC2002*, No. 27, pp. 21–26.

Kaplan, R., & Newman, P. (1997). Lexical resource reconciliation in the Xerox Linguistic Environment. In Netter, K., Estival, D., Lavelli, A., & Pianesi, F. (Eds.), *Proceedings* *of the ACL Workshop on Computational Environments for Grammar Development and Linguistic Engineering*, pp. 54–61 Madrid.

Kasahara, K., Sato, H., Bond, F., Tanaka, T., Fujita, S., Kanasugi, Y., & Amano, S. (2004). Construction of a Japanese Semantic Lexicon: Lexeed. In *Information Processing Society of Japan, 2004-NL-159*, pp. 75–82 Tokyo, Japan.

Krieger, H.-U., & Schafer, U. (1994). $\mathcal{TDL}$ — A type description language for constraint-based grammars. In *Proceedings of the 15th International Conference on Computational Linguistics*.

Lin, D. (1999). Automatic identification of non-compositional phrases. In *Proceedings of the ACL-1999*, pp. 317–324 College Park, Maryland.

Masuichi, H., & Ôkuma, T. (2003). Practical Analysis of Japanese Based on Lexical Functional Grammar (in Japanese). *Journal of NLP Vol.10 No.2*, 79–109.

Matsumoto, Y. (1996). *Complex Predicates in Japanese: A Syntactic and Semantic Study of the Notion 'Word'*. CSLI Publications.

Matsumoto, Y., Kitauchi, A., Yamashita, T., Hirano, Y., Matsuda, H., Takaoka, K., & Asahara, M. (2000). *Morphological Analysis System ChaSen version 2.2.1 Manual*. Nara Institute of Science and Technology.

Matsumoto, Y., & Utsuro, T. (2000). Lexical Knowledge Acquisition. In Dale, R., Moisl, H., & Somers, H. (Eds.), *Handbook of Natural Language Processing*, pp. 563–610. New York: Marcel Dekker, Inc.

Maxwell, J. T., & Kaplan, R. (1993). The interface between phrasal and functional constraints. *Computational Linguistics*.

McCarthy, D., Keller, B., & Carroll, J. (2003). Detecting a continuum of compositionality in phrasal verbs. In *Proceedings of the ACL-2003 Workshop on Multiword Expressions: Analysis, Acquisition and Treatment*, pp. 73–80 Sapporo, Japan.

Meurers, W. D. (2001). On Expressing Lexical Generalizations in HPSG. *Nordic Journal of Linguistics*, *24*(2).

Mitsuishi, Y., Torisawa, K., & ichi Tsujii, J. (1998). HPSG-Style Underspecified Japanese Grammar with Wide Coverage. In *COLING-ACL*, pp. 876–880.

Miyamoto, K., Yasui, T., Ikehara, S., & Murakami, J. (2000). Structure of Compound Verbs and their Translation Rules. In *The 61th Conference of Information Processing Society of Japan*, pp. 125–126.

Oepen, S., Bender, E. M., Callmeier, U., Flickinger, D., & Siegel, M. (2002). Parallel distributed grammar engineering for practical applications. In *Proceedings of COLING 2002 Workshop on Grammar Engineering and Evaluation*, pp. 15–21 Taipei, Taiwan.

Oepen, S., & Carroll, J. (2000). Performance profiling for grammar engineering. *Natural Language Engineering*, 81–97.

Ohtani, A., Miyata, T., & Matsumoto, Y. (2000). On Japanese Grammar Based on HPSG — Refinement and Extension Toward Computational Implementation (in Japanese). *Journal of NLP Vol.7 No.5*, 19–49.

Pollard, C. J., & Sag, I. A. (1987). *Information-based Syntax and Semantics, Vol. 1*. No. 13 in CSLI Lecture Notes. CSLI Publications, Stanford University. Distributed by University of Chicago Press.

Pollard, C. J., & Sag, I. A. (1994). *Head-Driven Phrase Structure Grammar*. University of Chicago Press, Chicago.

Rappaport, M., & Levin, B. (1988). What to Do with θ-Roles. In Wilkins, W. (Ed.), *Thematic Relations (Syntax and Semantics 21)*, pp. 7–36. Academic Press.

Raskin, V., & Nirenburg, S. (1998). An Applied Ontological Semantic Microtheory of Adjective Meaning for Natural Language Processing. *MT*, *13*(2–3), 135–227.

Rizzi, L. (1990). *Relativized Minimality*. MIT Press.

Sag, I. A., Baldwin, T., Bond, F., Copestake, A., & Flickinger, D. (2002). Multiword expressions: A pain in the neck for NLP. In *Computational Linguistics and Intelligent Text Processing: Third International Conference*, pp. 1–15 Mexico City, Mexico.

Sag, I. A., & Wasow, T. (1999). *Syntactic Theory: A Formal Introduction*. Center for the Study of Language and Information, Stanford. Japanese edition (two volumes) – translated and edited by Takao Gunji and Yasunari Harada, appeared in 2001.

Saito, M. (1985). *Some asymmetries in Japanese and their theoretical implications*. Ph.D. dissertation, Massachusetts Institute of Technology.

Siegel, M. (1998). Japanese Particles in an HPSG Grammar. Tech. rep., Verbmobil.

Siegel, M. (1999). The Syntactic Processing of Participles in Japanese Spoken Language. In Wang, J.-F., & Wu, C.-H. (Eds.), *Proceedings of the 13th Pacific Asia Conference on Language, Information and Computation (PACLIC 13), February 10-12* Taipei, Taiwan.

Siegel, M. (2000a). HPSG Analysis of Japanese. In Wahlster, W. (Ed.), *Verbmobil. Foundations of Speech-to-Speech Translation* (Artificial Intelligence edition)., pp. 265 − 280. Springer, Berlin, Germany.

Siegel, M. (2000b). Japanese Honorification in an HPSG Framework. In Ikeya, A., & Kawamori, M. (Eds.), *Proceedings of the 14th Pacific Asia Conference on Language, Information and Computation, February 15-17*, pp. 289–300 Tokyo, Japan. Waseda University International Conference Center.

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

Sperber, D., & Wilson, D. (1986). *Relevance — Communication and Cognition*. Blackwell.

Sugioka, Y. (1984). *Interaction of Derivational Morphology and Syntax in Japanese and English*. Ph.D. thesis, University of Chicago.

Tagashira, Y., & Hoff, J. (1986). *Handbook of Japanese Compound Verbs*. The Hokuseido Press.

Takeuchi, K., Kageura, K., & Koyama, T. (2003). Deverbal Compound Noun Analysis Based on Lexical Conceptual Structure. In *Proceedings of Poster/Demo session in 41st Annual Meeting of the Association for Computational Linguistics (ACL03)*, pp. 181–184 Sapporo, Japan.

Teramura, H. (1969). Endings, auxiliary verbs, subsidiary verbs, and aspect — No.1 —. In *Japanese language and Japanese culture 1* (in Japanese). Osaka University of Foreign Studies.

The XTAG Research Group (1995). A lexicalized tree adjoining grammar for English. Tech. rep., University of Pennsylvania.

Toutanova, K., Manning, C. D., Shieber, S. M., Flickinger, D., & Oepen, S. (2002). Parse Disambiguation for a Rich HPSG Grammar. In *First Workshop on Treebanks and Linguistic Theories*, pp. 253–263 Sozopol, Bulgaria.

Uchiyama, K., & Baldwin, T. (2003). A Disambiguation Method for Japanese Compound Verbs. In *Proceedings of ACL-2003 Workshop on Multiword Expressions: Analysis, Aquisition and Treatment*, pp. 81–88.

Verlinden, M. (1999). *A constraint-based grammar for dialogue utterance.* Unpublished Ph.D. dissertation, University of Tilburg.

Villavicencio, A., & Copestake, A. (2002). Phrasal Verbs and the LinGO-ERG. *LinGO Working Paper No. 2002-01.*

Wahlster, W. (2000). *Verbmobil: Foundations of Speech-to-Speech Translation.* Springer Verlag.

Wood, M. M. (1993). *Categorial Grammar.* Routledge, London and New York.

Yamamoto, K. (1983). *The structure and syntax of compounds: A study of Japanese processing for software documents* (in Japanese). Information-technology Promotion Agency.

Yarowsky, D. (1992). Word-Sense Disambiguation Using Statistical Models of Roget's Categories Trained on Large Corpora. In *Proceedings of the 14th COLING*, pp. 454–460.

Yatabe, S. (1996). Long-distance scrambling via partial compaction. *MIT Working Papers in Linguistics 29.*