A Compositional Approach toward Dynamic Phrasal Thesaurus

Atsushi Fujita Shuhei Kato Naoki Kato Satoshi Sato
Graduate School of Engineering, Nagoya University
{fujita,ssato}@nuee.nagoya-u.ac.jp
{shuhei,naoki}@sslab.nuee.nagoya-u.ac.jp

Abstract

To enhance the technology for computing semantic equivalence, we introduce the notion of phrasal thesaurus which is a natural extension of conventional word-based thesaurus. Among a variety of phrases that conveys the same meaning, i.e., paraphrases, we focus on syntactic variants that are compositionally explainable using a small number of atomic knowledge, and develop a system which dynamically generates such variants. This paper describes the proposed system and three sorts of knowledge developed for dynamic phrasal thesaurus in Japanese: (i) transformation pattern, (ii) generation function, and (iii) lexical function.

1 Introduction

Linguistic expressions that convey the same meaning are called paraphrases. Handling paraphrases is one of the key issues in a broad range of natural language processing tasks, including machine translation, information retrieval, information extraction, question answering, summarization, text mining, and natural language generation.

Conventional approaches to computing semantic equivalence between two expressions are five-fold. The first approximates it based on the similarities between their constituent words. If two words belong to closer nodes in a thesaurus or semantic network, they are considered more likely to be similar. The second uses the family of tree kernels (Collins and Duffy, 2001; Takahashi, 2005). The degree of equivalence of two trees (sentences) is defined as the number of common subtrees included in both trees. The third estimates the equivalence based on word alignment composed using templates or translation probabilities derived from a set of parallel text (Barzilay and Lee, 2003; Brockett and Dolan, 2005). The fourth espouses the distributional hypothesis (Harris, 1968): given two words are likely to be equivalent if distributions of their surrounding words are similar (Lin and Pantel, 2001; Weeds et al., 2005). The final regards two expressions equivalent if they can be associated by using a set of lexico-syntactic paraphrase patterns (Mel'čuk, 1996; Dras, 1999; Yoshikane et al., 1999; Takahashi, 2005).

Despite the results previous work has achieved, no system that robustly recognizes and generates paraphrases is established. We are not convinced of a hypothesis underlying the word-based approaches because the structure of words also conveys some meaning. Even tree kernels, which take structures into account, do not have a mechanism for identifying typical equivalents: e.g., dative alternation and passivization, and abilities to generate paraphrases. Contrary to the theoretical basis, the two lines of corpus-based approaches have problems in practice, i.e., data sparseness and computation cost. The pattern-based approaches seem steadiest. Yet no complete resource or methodology for handling a wide variety of paraphrases has been developed.

On the basis of this recognition, we introduce the notion of **phrasal thesaurus** to directly compute semantic equivalence of phrases such as follows.

(1) a. be in our favor / be favorable for us
b. its reproducibility / if it is reproducible
c. decrease sharply / show a sharp decrease
d. investigate the cause of a fire /
investigate why there was a fire /
investigate what started a fire /
make an investigation into the cause of a fire

Phrasal thesaurus is a natural extension of conventional word-based thesaurus. It is thus promised that it will bring us the following benefits:

Enhancement of NLP applications: As conventional thesauri, phrasal thesaurus must be useful to handle paraphrases having different structures in a wide range of NLP applications.

Reading and writing aids: Showing more appropriate alternative phrases must be a powerful aid at certain situations such as writing text. Controlling readability of text by altering phrases must also be beneficial to readers.

Our aim is to develop resources and mechanisms for computing semantic equivalence on the working hypothesis that phrase is the appropriate unit for that purpose. This paper describes the first version of our paraphrase generation system and reports on our ongoing work on constructing resources for realizing phrasal thesaurus.

The following sections describe the range of phenomena we treat (Section 2), the overall architecture of our paraphrase generation system which functions as phrasal thesaurus (Section 3), the implementation of knowledge bases (Section 4) followed by discussion (Section 5), and conclusion (Section 6).

2 Dynamic phrasal thesaurus

2.1 Issue

Toward realizing phrasal thesaurus, the following two issues should be discussed.

- What sorts of phrases should be treated
- How to cope with a variety of expressions

Although technologies of shallow parsing have been dramatically improved in the last decade, it is still difficult to represent arbitrary expression in logical form. We therefore think it is reasonable to define the range relying on lexico-syntactic structure instead of using particular semantic representation. According to the work of (Chklovski and Pantel, 2004; Torisawa, 2006), predicate phrase (simple sentence) is a reasonable unit because it approximately corresponds to the meaning of single event.

Combination of words and a variety of construction coerce us into handling an enormous number of expressions than word-based approaches. One may think taking phrase is like treading a thorny path because one of the arguments in Section 1 is about coverage. On this issue, we speculate that one of the feasible approach to realize a robust system is to divide phenomena into compositional and non-compositional (idiosyncratic) ones[1], and separately develop resources to handle them as described in (Fujita and Inui, 2005).

To compute semantic equivalence of idiosyncratic paraphrases, pairs or groups of paraphrases have to be **statically** compiled into a dictionary as word-based thesaurus. The corpus-based approach is valuable for that purpose, although they are not guaranteed to collect all idiosyncratic paraphrases. On the other hand, compositional paraphrases can be captured by a relatively small number of rules. Thus it seems tolerable approach to generate them **dynamically** by applying such rules. Our work is targeted at compositional paraphrases and the system can be called **dynamic phrasal thesaurus**. Hereafter, we refer to paraphrases that are likely to be explained compositionally as **syntactic variants**.

2.2 Target language: Japanese

While the discussion above does not depend on particular language, our implementation of dynamic phrasal thesaurus is targeted at Japanese. Several methods for paraphrasing Japanese predicate phrases have been proposed (Kondo et al., 1999; Kondo et al., 2001; Kaji et al., 2002; Fujita et al., 2005). The range they treat is, however, relatively narrow because they tend to focus on particular paraphrase phenomena or to rely on existing resources. On the other hand, we define the range of phenomena from a top-down viewpoint. As a concrete definition of predicate phrase in Japanese,

noun phrase + case marker + predicate

is employed which is hereafter referred to "phrase."

Noun phrase and predicate in Japanese themselves subcategorize various syntactic variants as shown in Figure 1 and paraphrase phenomena for above phrase also involve those focused on their interaction. Thus the range of phenomena is not so narrow, and intriguing ones, such as shown in examples[2] (2) and (3), are included.

[1] We regard lexical paraphrases (e.g., "scope" ⇔ "range") and idiomatic paraphrases (e.g., "get the sack" ⇔ "be dismissed from employment") as idiosyncratic.

[2] In each example, "s" and "t" denote an original sentence and its paraphrase, respectively. SMALLCAPS strings indicate the syntactic role of their corresponding Japanese expressions. [N] indicates a nominalizer.

(2) Head switching

s. *kakunin-o isogu.*
checking-ACC to hurry-PRES
We hurry checking it.

t. *isoide kakunin-suru.*
in a hurry to check-PRES
We check it in a hurry.

(3) Noun phrase ⇔ sub-clause

s. *kekka-no saigensei-o kenshou-suru.*
result-GEN reproducibility-ACC to validate-PRES
We validate its reproducibility.

t. [*kekka-o saigen-dekiru*]
result-ACC to reproduce-to be able
ka-douka-o kenshou-suru.
[N]-whether-ACC to validate-PRES
We validate whether it is reproducible.

We focus on syntactic variants at least one side of which is subcategorized into the definition of phrase above. For the sake of simplicity, we hereafter represent those expressions using part-of-speech (POS) patterns. For instance, (2s) is called $N : C : V$ type, and (3s) is N_1 : *no* : $N_2 : C : V$ type.

3 Paraphrase generation system

Given a phrase, the proposed system generates its syntactic variants in the following four steps:

1. Morphological analysis
2. Syntactic transformation
3. Surface generation with lexical choice
4. SLM-based filtering

where no particular domain, occasion, and media is assumed[3]. Candidates of syntactic variants are first over-generated in step 2 and then anomalies among them are filtered out in steps 3 and 4 using rule-based lexical choice and statistical language model.

The rest of this section elaborates on each component in turn.

3.1 Morphological analysis

Technologies of morphological analysis in Japanese have matured by introducing machine learning techniques and large-scale annotated corpus, and there are freely available tools. Since the structure of input phrase is assumed to be quite simple, employment of dependency analyzer was put off. We simply use a morphological analyzer MeCab[4].

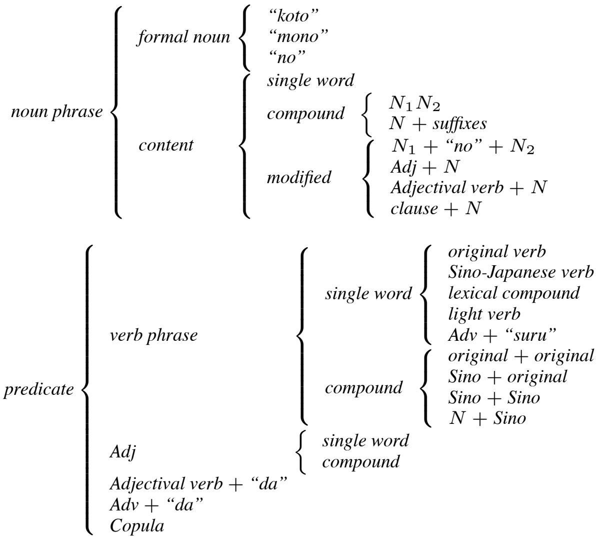

Figure 1: Classification of syntactic variants of noun phrase and predicate in Japanese.

Our system has a post-analysis processing. If either of Sino-Japanese verbal nouns (e.g., "*kenshou* (validation)" and "*kandou* (impression)") or transliteration of verbs in foreign language (e.g., "*doraibu* (to drive)" and "*shifuto* (to shift)") is immediately followed by "*suru* (to do)" or "*dekiru* (to be able)," these adjacent two morphemes are joined into a single morpheme to avoid incorrect transformation.

3.2 Syntactic transformation

The second step over-generates syntactic variants using the following three sorts of knowledge:

(i) Transformation pattern: It gives skeletons of syntactic variants. Each variant is represented by POS symbols designating the input constituents and triggers of the generation function and lexical function below.

(ii) Generation function: It enumerates different expressions that are constituted with the same set of words and subcategorized into the required syntactic category. Some of generation functions handle base phrases, while the rest generates functional words. Base phrases the former generates are smaller than that transformation patterns treat. Since some functional words are disjunctive, the latter generates all candidates with a separator "/" and leaves the selection to the following step.

[3] This corresponds to the linguistic transformation layer of KURA (Takahashi et al., 2001).

[4] http://mecab.sourceforge.net/

Table 1: Grammar in Backus-Naur form, example, and instantiation for each knowledge.

Knowledge type	Grammar / Example / Instantiation

(i) Transformation pattern	`<transformation_pattern> ::= <left_pattern> ⇒ <right_pattern>`
	`<left_pattern> ::= (<POS_symbol>|<word_form>)+`
	`<POS_symbol> ::=` ($N \vert C \vert V \vert Adj \vert Adv$)
	`<word_form> ::= (<hiragana>|<katakana>|<kanji>)+`
	`<right_pattern> ::=`
	`(<POS_symbol>|<word_form>|<function_definition>|<lexical_function>)+`

	(a) $N : C : V \Rightarrow adv(V) : vp(N)$ (b) N_1 : *no* : $N_2 : C : V \Rightarrow N_1$: *genCase*() : *vp*(N_2) : *ka-douka* : $C : V$
	(a) *kakunin* : *o* : *isogu* ⇒ *adv*(*isogu*) : *vp*(*kakunin*) checking ACC to hurry *adv*(to hurry) *vp*(checking) (b) *kekka* : *no* : *saigensei* : *o* : *kenshou-suru* result GEN reproducibility ACC to validate-PRES ⇒ *kekka* : *genCase*() : *vp*(*saigensei*) : *ka-douka* : *o* : *kenshou-suru* result case marker *vp*(reproducibility) [N]-whether ACC to validate-PRES

(ii) Generation function	`<generation_function> ::= <function_definition> ⇒ '{'<right_pattern>+'}'`
	`<function_definition> ::= <syntactic_category>'('<POS_symbol>*')'`
	`<syntactic_category> ::=` (*np* | *vp* | *lvc*)

	(a) *vp*(N) ⇒ {*v*(N) : *genVoice*() : *genTense*()}
	(b) *np*(N_1, N_2) ⇒ {$N_1, N_2, N_1 : N_2, N_1$: *no* : N_2, *vp*(N_1) : N_2, *wh*(N_2) : *vp*(N_1) : *ka*, ...}

	(a) *vp*(*kakunin*) ⇒ { *v*(*kakunin*) : *genVoice*() : *genTense*() } *vp*(verification) *v*(verification) verbal suffix for voice verbal suffix for tense (b) *np*(*shukka*, *gen-in*) *np*(starting fire, reason) ⇒ { *shukka* , *gen-in* , *shukka* : *gen-in* , *shukka* : *no* : *gen-in* , starting fire reason starting fire reason starting fire GEN reason *vp*(*shukka*) : *gen-in* , *wh*(*gen-in*) : *vp*(*shukka*) : *ka* , ... } *vp*(starting fire) reason *wh*(reason) *vp*(starting fire) [N]
(iii) Lexical function	`<lexical_function> ::= <relation>'('<POS_symbol>')'` `<relation> ::=` (*n* | *v* | *adj* | *adjv* | *adv* | *wh*)
	(a) *adv*(V) (b) *wh*(N)
	(a) *adv*(*isogu*) (⇒ *isoide* (given by a verb–adverb dictionary)) *adv*(to hurry) in a hurry (b) *wh*(*gen-in*) (⇒ { *naze* , *doushite* } (given by a noun–interrogative dictionary)) *wh*(reason) why why

(iii) Lexical function: It generates different lexical items in certain semantic relations, such as derivative form, from a given lexical item. The back-end of this knowledge is a set of pre-compiled dictionaries as described in Section 4.2.

Table 1 gives a summary of grammar in Backus-Naur form, examples, and instantiations of each knowledge. Figure 2 illustrates an example of knowledge application flow for transforming (4s) into (4t), where ":" denotes delimiter of constituents.

(4) s. "*kakunin*:*o*:*isogu*"
t. "*isoide*:{*kakunin-suru*:
{ϕ, *reru*/*rareru*, *seru*/*saseru*}:{ϕ, *ta*/*da*}}"

First, transformation patterns that match to the given input are applied. Then, the skeletons of syntactic variants given by the pattern are lexicalized by consecutively invoking generation functions and lexical functions. Plural number of expressions that generation function and lexical function generate are enumerated within curly brackets. Transformation is ended when the skeletons are fully lexicalized.

In fact, knowledge design for realizing the transformation is not really new, because we have been inspired by the previous pattern-based approaches. Transformation pattern is thus alike that in the Meaning-Text Theory (MTT) (Mel'čuk, 1996), Synchronous Tree Adjoining Grammar (STAG) (Dras, 1999), meta-rule for Fastr (Yoshikane et al., 1999),

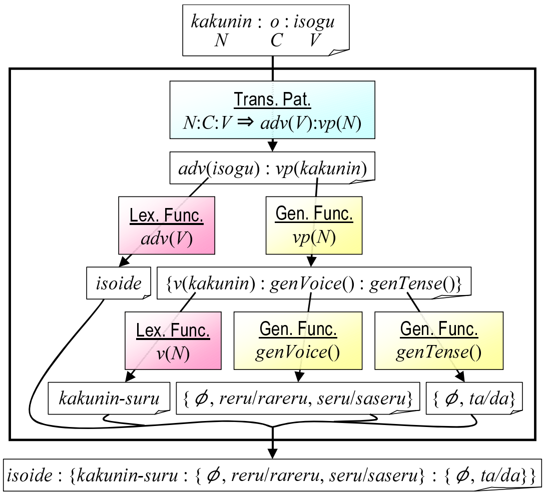

Figure 2: Syntactic transformation (for (2)).

and transfer pattern for KURA (Takahashi et al., 2001). Lexical function is also alike that in MTT. However, our aim in this research is beyond the design. In other words, as described in Section 1, we are aiming at the following two: to develop resources for handling syntactic variants in Japanese, and to confirm if phrasal thesaurus really contribute to computing semantic equivalence.

3.3 Surface generation with lexical choice

The input of the third component is a bunch of candidate phrases such as shown in (4t). This component does the following three processes in turn:

Step 1. Unfolding: All word sequences are generated by removing curly brackets one by one.

Step 2. Lexical choice: Disjunctive words are concatenated with "/" (e.g., "*reru/rareru*" in (4t)). One of them is selected based on POS and conjugation types of the preceding word.

Step 3. Conjugation: In the transformation step, conjugative words are moved to different positions and some of them are newly generated. Inappropriate conjugation forms are corrected.

3.4 SLM-based filtering

In the final step, we assess the correctness of each candidate of syntactic variants using a statistical language model. Our model simply rejects candidate phrases that never appear in a large size of raw text corpus consisting of 15 years of newspaper articles (Mainichi 1991–2005, approximately 1.8GB). Although it is said that Japanese language has a degree of freedom in word ordering, current implementation does not yet employ structured language models because phrases we handle are simple.

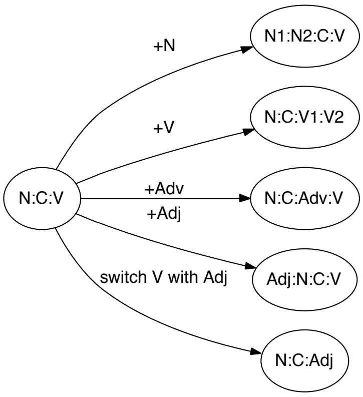

Figure 3: Derivations of phrase types.

4 Knowledge implementation

4.1 Transformation patterns and generation functions

An issue of developing resources is how to ensure their coverage. Our approach to this issue is to describe transformation patterns by extending those for simpler phrases. We first described following three patterns for $N : C : V$ type phrases which we consider the simplest according to Figure 1.

(5) a. $N : C : V \Rightarrow vp(N)$
b. $N : C : V \Rightarrow N : genCase() : lvc(V)$
c. $N : C : V \Rightarrow adv(V) : vp(N)$

While the pattern (5c) is induced from example (2), the patterns (5a-b) are derived from examples (6) and (7), respectively.

(6) s. *shigeki-o* *ukeru*
inspiration-ACC to receive
to receive an inspiration
t. *shigeki-sareru*
to inspire-PASS
to be inspired

(7) s. *hada-o* *shigeki-suru*
skin-ACC to stimulate
to stimulate skin
t. *hada-ni* *shigeki-o* *ataeru*
skin-DAT stimulus-ACC to give
to give skin a stimulus

Regarding the patterns in (8) as the entire set of compositional paraphrases for $N : C : V$ type phrases, we then extended them to a bit more complex phrases as in Figure 3. For instance, 10 patterns

Table 2: Transformation patterns.

Target phrase	# of patterns
$N : C : V$	3
$N_1 : N_2 : C : V$	10
$N : C : V_1 : V_2$	10
$N : C : Adv : V$	7
$Adj : N : C : V$	4
$N : C : Adj$	3
Total	37

Table 3: Generation functions.

Definition	Syntactic category	# of returned value
$np(N_1, N_2)$	noun phrase	9
$vp(N)$	verb phrase	1
$vp(N_1, N_2)$	verb phrase	2
$vp(V_1, V_2)$	verb phrase	3
$lvc(V)$	light verb construction	1
$genCase()$	case marker	4
$genVoice()$	verbal suffix for voice	3
$genTense()$	verbal suffix for tense	2
$genAspect()$	verbal suffix for aspect	2

for $N_1 : N_2 : C : V$ type phrases shown in (8) have been described based on patterns in (5), mainly focusing on interactions between newly introduced N_1 and other constituents.

(8) a. $N_1 : N_2 : C : V \Rightarrow vp(N_1, N_2)$ (5a)
b. $N_1 : N_2 : C : V \Rightarrow N_1 : genCase() : vp(N_2)$ (5a)
c. $N_1 : N_2 : C : V \Rightarrow N_2 : genCase() : vp(N_1)$ (5a)
d. $N_1 : N_2 : C : V \Rightarrow np(N_1, N_2) : genCase() : lvc(V)$ (5b)
e. $N_1 : N_2 : C : V \Rightarrow N_1 : genCase() : N_2 : genCase() : lvc(V)$ (5b)
f. $N_1 : N_2 : C : V \Rightarrow N_2 : genCase() : N_1 : genCase() : lvc(V)$ (5b)
g. $N_1 : N_2 : C : V \Rightarrow adv(V) : vp(N_1, N_2)$ (5c)
h. $N_1 : N_2 : C : V \Rightarrow adv(V) : N_1 : genCase() : vp(N_2)$ (5c)
i. $N_1 : N_2 : C : V \Rightarrow adv(V) : N_2 : genCase() : vp(N_1)$ (5c)
j. $N_1 : N_2 : C : V \Rightarrow np(N_1, N_2) : C : V$ (new)

The number of transformation patterns we have so far developed is shown in Table 2.

Generation functions shown in Table 3 are developed along with creating transformation patterns. Although this is the heart of the proposed model, two problems are remained: (i) the granularity of each generation function is determined according to our linguistic intuition, and (ii) they do not ensure of generating all possible phrases. Therefore, we have to establish the methodology to create this knowledge more precisely.

Table 4: Dictionaries for lexical functions.

ID	POS-pair	$\|D\|$	$\|C\|$	$\|D \cup C\|$	$\|J\|$
(a)	noun–verb	3,431	-	3,431	3,431
(b)	noun–adjective	308	667	906	475 †
(c)	noun–adjectival verb	1,579	-	1,579	1,579
(d)	noun–adverb	271	-	271	271
(e)	verb–adjective	252	-	252	192 †
(f)	verb–adjectival verb	74	-	74	68 †
(g)	verb–adverb	74	-	74	64 †
(h)	adjective–adjectival verb	66	95	159	146 †
(i)	adjective–adverb	33	-	33	26 †
(j)	adjectival verb–adverb	70	-	70	70
Total		6,158	762	6,849	6,322

4.2 Lexical functions

Except $wh(N)$, which generates interrogatives as shown in the bottom line of Table 1, the relations we have so far implemented are lexical derivations. These roughly correspond to **S**, **V**, **A**, and **Adv** in MTT. The back-end of these lexical functions is a set of dictionaries built by the following two steps:

Step 1. Automatic candidate collection: Most derivatives in Japanese share the beginning of words and are characterized by the correspondences of their suffixes. For example, "*amai* (be sweet)" and "*amami* (sweetness)" has a typical suffix correspondence "∗-i:∗-mi" of adjective–noun derivation. Using this clue, candidates are collected by two methods.

- *From dictionary*: Retrieve all word pairs from the given set of words those satisfying the following four conditions: (i) beginning with *kanji* character, (ii) having different POSs, (iii) sharing at least the first character and the first sound, and (iv) having a suffix pattern which corresponds to at least two pairs.
- *Using dictionary and corpus*: Generate candidates from a set of words by applying a set of typical suffix patterns, and then check if each candidate is an actual word using corpus. This is based on (Langkilde and Knight, 1998).

Step 2. Manual selection: The set of word pairs collected in the previous step includes those do not have particular semantic relationship. This step involves human to discard noises.

Table 4 shows the size of 10 dictionaries, where each column denotes the number of word pairs retrieved from IPADIC[5] ($|D|$), those using IPADIC, seven patterns and the same corpus as in Section 3.4 ($|C|$), their union ($|D \cup C|$), and those manually judged correct ($|J|$), respectively. The sets of word pairs J are used as bi-directional lexical functions, although manual screening for four dictionaries without dagger (†) are still in process.

5 Discussion

5.1 Unit of processing

The working hypothesis underlying our work is that phrase is the appropriate unit for computing semantic equivalence. In addition to the arguments in Section 1, the hypothesis is supported by what is done in practice. Let us see two related fields.

The first is the task of word sense disambiguation (WSD). State-of-the-art WSD techniques refer to context as a clue. However, the range of context is usually not so wide: words and their POSs within small window centered the target word and content words within the same sentence of the target word. The task therefore can be viewed as determining the meaning of phrase based on its constituent words and surrounding content words.

Statistical language model (SLM) is another field. SLMs usually deal with various things within word sequence (or structure) at the same time. However, relations within a phrase should be differentiated from that between phrases, because checking the former is for grammaticality, while the latter for cohesion. We think SLMs should take the phrase to determine boundaries for assessing the correctness of generated expressions more accurately.

5.2 Compositionality

We examined how large part of manually created paraphrases could be generated in our compositional approach. First, a set of paraphrase examples were created in the following procedure:

Step 1. Most frequent 400 phrases typed $N_1 : N_2 : C : V$ were sampled from one year of newspaper articles (Mainichi 1991).

Step 2. An annotator produced paraphrases for each phrase. We allowed to record more than one paraphrase for a given phrase and to give up producing paraphrases. As a result, we obtained 211 paraphrases for 170 input phrases.

Manual classification revealed that 42% (88 / 211) of paraphrases could be compositionally explainable, and the (theoretical) coverage increases to 86% (182 / 211) if we have a synonym dictionary. This ratio is enough high to give these phenomena preference as the research target, although we cannot reject a possibility that data has been biased.

5.3 Sufficient condition of equivalence

In our system, transformation patterns and generation functions offer necessary conditions for generating syntactic variants for given input. However, we have no sufficient condition to control the application of such a knowledge.

It has not been thoroughly clarified what clue can be sufficient condition to ensure semantic equivalence, even in a number of previous work. Though, at least, roles of participants in the event have to be preserved by some means, such as the way presented in (Pantel et al., 2007). Kaji et al. (2002) introduced a method of case frame alignment in paraphrase generation. In the model, arguments of main verb in the source are taken over by that of the target according to the similarities between arguments of the source and target. Fujita et al. (2005) employed a semantic representation of verb to realize the alignment of the role of participants governed by the source and target verbs. According to an empirical experiment in (Fujita et al., 2005), statistical language models do not contribute to calculating semantic equivalence, but to filtering out anomalies. We therefore plan to incorporate above alignment-based models into our system, for example, within or after the syntactic transformation step (Figure 2).

5.4 Ideas for improvement

The knowledge and system presented in Section 3 are quite simple. Thus the following features should be incorporated to improve the system in addition to the one described in Section 5.3.

- *Dependency structure*: To enable flexible matching, such as $Adv : N : C : V$ type input and transformation pattern for $N : C : Adv : V$ type phrases.
- *Sophisticated SLM*: The generation phase should also take the structure into account to

[5] http://mecab.sourceforge.jp/

evaluate generated expressions flexibly.

- *Knowledge development*: Although we have not done intrinsic evaluation of knowledge, we are aware of its incompleteness. We are continuing manual screening for the dictionaries and planning to enhance the methodology of knowledge development.

6 Conclusion

To enhance the technology for computing semantic equivalence, we have introduced the notion of phrasal thesaurus, which is a natural extension of conventional word-based thesaurus. Plausibility of taking phrase as the unit of processing has been discussed from several viewpoints. On the basis of that, we have been developing a system to dynamically generate syntactic variants in Japanese predicate phrases which utilizes three sorts of knowledge that are inspired by MTT, STAG, Fastr, and KURA.

Future work includes implementing more precise features and larger resources to compute semantic equivalence. We also plan to conduct an empirical evaluation of the resources and the overall system.

Acknowledgments

This work was supported in part by MEXT Grants-in-Aid for Young Scientists (B) 18700143, and for Scientific Research (A) 16200009, Japan.

References

Regina Barzilay and Lillian Lee. 2003. Learning to paraphrase: an unsupervised approach using multiple-sequence alignment. In *Proceedings of the 2003 Human Language Technology Conference and the North American Chapter of the Association for Computational Linguistics* (*HLT-NAACL*), pages 16–23.

Chris Brockett and William B. Dolan. 2005. Support Vector Machines for paraphrase identification and corpus construction. In *Proceedings of the 3rd International Workshop on Paraphrasing* (*IWP*), pages 1–8.

Timothy Chklovski and Patrick Pantel. 2004. VerbOcean: mining the Web for fine-grained semantic verb relations. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing* (*EMNLP*), pages 33–40.

Michael Collins and Nigel Duffy. 2001. Convolution kernels for natural language. In *Advances in Neural Information Processing Systems 14: Proceedings of the 2001 Conference*, pages 625–632.

Mark Dras. 1999. *Tree adjoining grammar and the reluctant paraphrasing of text*. Ph.D. thesis, Division of Information and Communication Science, Macquarie University.

Atsushi Fujita, Kentaro Inui, and Yuji Matsumoto. 2005. Exploiting Lexical Conceptual Structure for paraphrase generation. In *Proceedings of the 2nd International Joint Conference on Natural Language Processing* (*IJCNLP*), pages 908–919.

Atsushi Fujita and Kentaro Inui. 2005. A class-oriented approach to building a paraphrase corpus. In *Proceedings of the 3rd International Workshop on Paraphrasing* (*IWP*), pages 25–32.

Zellig Harris. 1968. *Mathematical structures of language*. New York: Interscience Publishers.

Nobuhiro Kaji, Daisuke Kawahara, Sadao Kurohashi, and Satoshi Sato. 2002. Verb paraphrase based on case frame alignment. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics* (*ACL*), pages 215–222.

Keiko Kondo, Satoshi Sato, and Manabu Okumura. 1999. Paraphrasing of "sahen-noun + suru". *IPSJ Journal*, 40(11):4064–4074. (in Japanese).

Keiko Kondo, Satoshi Sato, and Manabu Okumura. 2001. Paraphrasing by case alternation. *IPSJ Journal*, 42(3):465–477. (in Japanese).

Irene Langkilde and Kevin Knight. 1998. Generation that exploits corpus-based statistical knowledge. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and the 17th International Conference on Computational Linguistics* (*COLING-ACL*), pages 704–710.

Dekang Lin and Patrick Pantel. 2001. Discovery of inference rules for question answering. *Natural Language Engineering*, 7(4):343–360.

Igor Mel'čuk. 1996. Lexical functions: a tool for the description of lexical relations in a lexicon. In Leo Wanner, editor, *Lexical Functions in Lexicography and Natural Language Processing*, pages 37–102. John Benjamin Publishing Company.

Patrick Pantel, Rahul Bhagat, Bonaventura Coppola, Timothy Chklovski, and Eduard Hovy. 2007. Isp: Learning inferential selectional preferences. In *Proceedings of Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics* (*NAACL-HLT*), pages 564–571.

Tetsuro Takahashi, Tomoya Iwakura, Ryu Iida, Atsushi Fujita, and Kentaro Inui. 2001. KURA: a transfer-based lexico-structural paraphrasing engine. In *Proceedings of the 6th Natural Language Processing Pacific Rim Symposium* (*NLPRS*) *Workshop on Automatic Paraphrasing: Theories and Applications*, pages 37–46.

Tetsuro Takahashi. 2005. *Computation of semantic equivalence for question answering*. Ph.D. thesis, Graduate School of Information Science, Nara Institute of Science and Technology.

Kentaro Torisawa. 2006. Acquiring inference rules with temporal constraints by using Japanese coordinated sentences and noun-verb co-occurrences. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics* (*HLT-NAACL*), pages 57–64.

Julie Weeds, David Weir, and Bill Keller. 2005. The distributional similarity of sub-parses. In *Proceedings of the ACL Workshop on Empirical Modeling of Semantic Equivalence and Entailment*, pages 7–12.

Fuyuki Yoshikane, Keita Tsuji, Kyo Kageura, and Christian Jacquemin. 1999. Detecting Japanese term variation in textual corpus. In *Proceedings of the 4th International Workshop on Information Retrieval with Asian Languages*, pages 97–108.