Development of Corpora Tagged with High-Precision Semantic Information

Alastair Butler*† Zhen Zhou‡ Tomoko Hotta‡ Su Zhang‡ Kei Yoshimoto†‡
*PRESTO, Japan Science and Technology Agency
†Center for the Advancement of Higher Education, Tohoku University
‡Graduate School of International Cultural Studies, Tohoku University

ajb129@hotmail.com

Abstract
We report on the initial stages of our research for developing corpora of high-precision meaning representations. So far our focus has been to assemble tools to start building corpora semi-automatically. The central component is a system of evaluation for a small formal language with respect to a structured assignment based on Scope Control Theory (SCT; Butler 2010). The output of the evaluation system is a meaning representation with a model theoretic interpretation. The input is an expression of the SCT language that is generated by the syntactic conversion of existing parsed representations of natural language. One advantage of SCT as the basis for building meaning representations is its ability to accept input with minimal conversion from almost any parsed representation. Currently we utilise existing gold standard parsed data that conforms to the very different annotation schemes of the Kyoto Corpus for Japanese (based on bunsetsu dependencies) and the Penn Treebank for English (based on phrase structure trees). A further advantage of SCT is that while the approach is robust and facilitates very wide coverage it also guarantees the enforcement of required dependencies: when garbage is given as input, only debugging information is received as output.

1 Introduction

This paper describes tools we have assembled for the semi-automatic construction of corpora tagged with deep semantic information with high coverage and precision. The method we employ for arriving at meaning representations involves a procedure of semantic evaluation that is notable for accepting what can essentially be conventional parsed syntactic forms as input. Outputs from evaluation (in essence computed denotations) are returned as formulas of a predicate logic notation with the further option of allowing expressions embedded as arguments to predicates to facilitate a compact readibility.

Output meaning representations, that may be subsequently revised and corrected, are to form the content of the corpora we plan to build. Such representations will make explicit predicate argument information (which may be checked against existing resources like PropBank; Palmer et al., 2005), as well as information previously unavailable from wide scale corpora, notably the scopes of quantifiers (e.g., existential quantification), operators (e.g., negation), connectives (e.g., conjunction) and embedding predicates (e.g., propositional attitudes), while also capturing inter and intra sentential binding dependencies and discourse anaphoric dependencies. In keeping to predicate logic notation the tagging of the corpora will have a standard model theoretic interpretation as well as being appropriate to feed theorem provers and model builders (see e.g., Blackburn and Bos 2003).

The paper is structured as follows. Section 2 briefly sketches the theoretical background that underlies our approach to constructing semantic corpora. Section 3 describes the encoding of predicates and demonstrates when semantic evaluation will either work or fail. Section 4 illustrates how we are employing the approach to take as input parsed bunsetsu dependency annotations and output predicate logic meaning representations. Section 5 concludes.

2 Semantic Theory of Sentence Processing

In the formal study of language, the essence of grammatical structure is regarded as the range of valid dependencies. While standard linguistic theories typically take dependencies to be syntactically determined, Scope Control Theory (SCT) aims to characterise grammatical dependencies in terms of the relationships that can be established when there is evaluation of a structure against an assignment function as introduced by Vermeulen (2000) that stores discourse and intra sentential information as sequence

values.

It has been a formidable problem in the study of natural language meanings that the semantic structure of sentences as grasped by formal languages of meaning representation such as predicate logic has the initial appearance of being inconsistent with the syntactic structures of sentences. In order to solve this problem and analyse various constructions, formal linguistic theories have so far attempted to perform complex manipulations on syntactic structures and features (see for example Combinatory Categorial Grammar in Steedman 1996 and Head-driven Phrase Structure Grammar in Pollard and Sag 1994).

For SCT the well-formedness of sentences, the most important condition that must be satisfied in analysing sentences, is defined as equality between the number of sequence values required during the evaluation of a sentence or discourse and the length of sequence values provided by the assignment. By extending, manipulating, reducing or temporarily making inactive parts of sequence values of the assignment under this condition, the complex formation of natural language sentences is simulated while allowing for the preservation of expected syntactic structures. In addition to providing a mechanism for capturing effects of grammatical dependencies, evaluation can be utilised to return calculated denotations as meaning representations to realise our mechanism for automating the process of building semantic corpora.

3 Encoding predicates

This section sketches the general approach adopted for encoding the contribution of predicates which is the key to constraining SCT for the parsed input of a particular natural language and annotation scheme. The motivating idea is that the presence of a predicate within an expression should conspire to bring about the enforcement of fixed grammatical and contextual roles on binding names with the consequence that: (i) when predicates have specified their argument binding information, evaluation is constrained to accept only the grammatical input of the natural language; and (ii) should argument binding information be absent, evaluation itself becomes the driving force for determining the allocation of binding dependencies.

A binding name has a grammatical role when either: (a) it has a local binding role and so may serve as the bound name of a predicate argument, or (b) it provides a source for fresh bindings, which are bindings that cannot bind the arguments of predicates but which may shift during an evaluation to local binding names. (If a given binding happened to be retained only as a fresh binding then it would bind vacuously.)

Control over binding names to enforce particular roles is gained by making the evaluation sensitive to what should and should not be present as a binding. This is achieved with an operation `check`. For example, (1) will establish sensitivity to the `"arg1"` binding name, such that an evaluation of the expression e against the current assignment g is only possible when the count of the number of instances of operators with the form `Use "arg1"` inside e equals the exact number of bindings open for the `"arg1"` name in g.

(1) `check ["arg1"]` e

In (2) we illustrate examples with the `check` of (1), where $e =$ `Use ("arg1", T "arg1")` in (2a) and $e =$ `T "arg1"` in (2b). `T` is a primitive operation for constructing a bound argument from a binding name. $(.,.)^\circ$ is the SCT evaluation procedure.

(2) a. $\exists g : (g,$ `check ["arg1"] (Use ("arg1", T "arg1"))`$)^\circ = x$

b. $\forall g : (g,$ `check ["arg1"] (T "arg1")`$)^\circ = *$

The result of (2a) shows evaluation is possible, which can be illustrated as in (3). There is no change to an assignment containing a sequence with a single element assigned to `"arg1"`, which serves as the result (computed denotation) returned by evaluating `T "arg1"`.

(3)
$$\left[\ \texttt{"arg1"} \rightarrow [x]\ \right]\ \texttt{check ["arg1"]}$$

.

`Use "arg1"`

$$\left[\ \texttt{"arg1"} \rightarrow [x]\ \right]\ \texttt{T "arg1"}$$

In contrast for (2b) all evaluations fail. To see why consider an evaluation where there is a sequence with a single element assigned to `"arg1"`. As (4) shows, evaluation fails since the assignment does not meet the requirement from `check` of there being zero bindings for the `"arg1"` name. (The returned information "from arg1 shift(snoc) fails" is debugging information.)

(4)
$$\left[\ \texttt{"arg1"} \rightarrow [x]\ \right]\ \texttt{check ["arg1"]}$$

.

from arg1 shift(snoc) fails

In (5) the requirement from `check` is met by there being no `"arg1"` binding (the assignment is empty), but this fails to meet the requirement of `T "arg1"` that there should be an available `"arg1"` binding to return.

(5) empty assignment `check ["arg1"]`

`T "arg1"`: no `"arg1"` binding

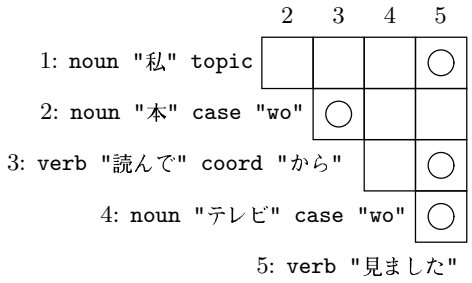
4 Building Corpora

This section illustrates how we apply SCT evaluation to create semantic representations of unrestricted texts by demonstrating the process with sentence (6).

(6) 私 は 本 を 読んでから テレビ
I wa book wo read after television
noun topic noun case verb coord noun
を 見ました。
wo watched
case verb
'After I read a book, I watched television.'

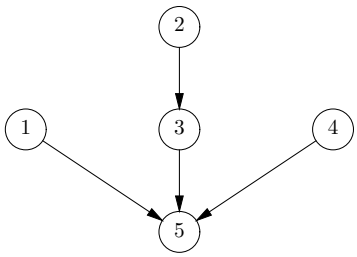
Sentence (6) can be parsed, for example by the KNP parser (Kurohashi and Nagao 1994), to obtain the bunsetsu dependency analysis of (7).

(7)



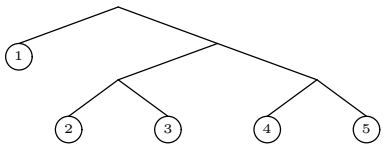
The modifier dependency information of (7) gives the structure of (8).

(8)



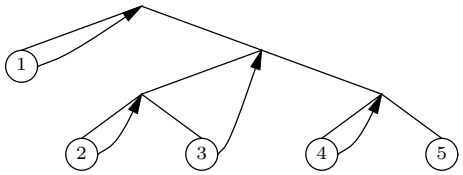
Because Japanese is exclusively head-final the order of numbering of bunsetsu captures the hierarchical scoping information of a constituent tree structure with lower numbered modifier bunsetsu having wider syntactic scoping. Following this scope convention the dependency structure (8) can be interpreted as specifying the constituent tree structure (9).

(9)



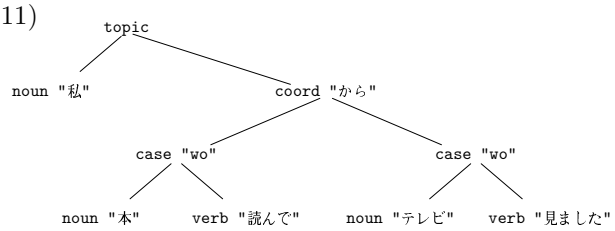
Functional information of a modifier bunsetsu can be used to label the parent tree node that connects the bunsetsu with its associated constituent tree structure to the constituent tree structure associated with its head bunsetsu. The arrows in (10) illustrate such an integration of functional information.

(10)



Combining the lexical information of (7) with the structural information of the tree in (10) results in the constituent parse tree of (11) in which nonterminal nodes are labelled with functional information.

(11)



To evaluate (11) with our implementation of SCT requires one final step of transforming the labelled constituent tree into an expression that consists of the primitive operations of the SCT language. This is accomplished by reformulating (11) as the 'syntactically sugared' SCT representation (12). This maintains the constituency of (11), refines node information with operator information, and adds information about local binding names (`["h", "wo", "ga"]`).

(12)

```
( λlc.
  ( ( r lc fh ["h"] nil "私")
    slash
    ( kp lc fh "ga" "wa")
    rslash
    ( ( ( r lc fh ["h"] nil "本")
        slash
        ( kp lc fh "wo" "e")
        rslash
        ( r lc fh ["ga", "wo"] nil "読んで"))
      slash
      ( coord fh "から"
      rslash
      ( ( r lc fh ["h"] nil "テレビ")
        slash
        ( kp lc fh "wo" "e")
        rslash
        ( r lc fh ["ga", "wo"] nil "見ました")))))
["h", "wo", "ga"]
```

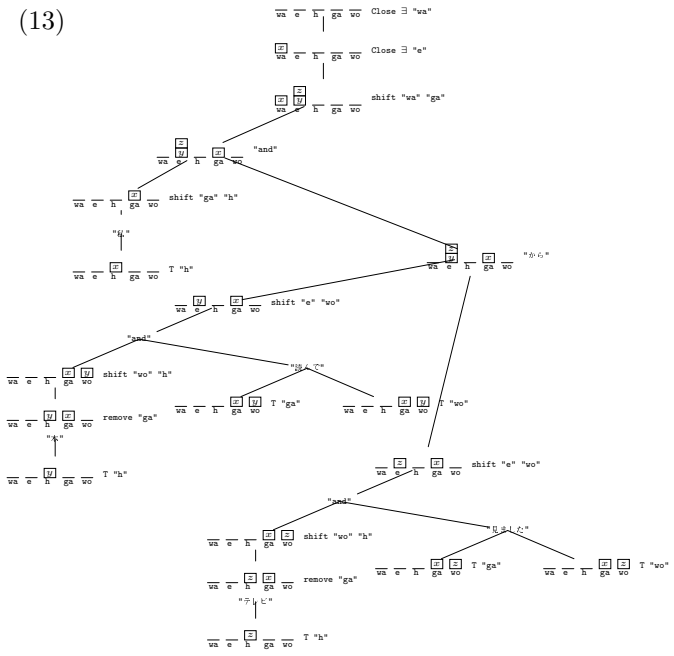
The representation of (12) reduces to an evaluable expression with definitions for `fh` (binding names providing sources for fresh bindings), `coord` (used to create a coordinating relation with the semantic content of から 'after'), `r` (used to create predicates), `kp` (used to create a case phrase), and `slash` and `rslash` (guidance for function application to take an argument from the left and right, respectively).

In defining `kp` we provide the role of noun phrases with case markers. A noun phrase needs the ability to support potentially arbitrary restriction material while placing no requirements on its containing clause, except the need for the containing clause to

support the binding that the noun phrase itself exists to contribute.

Having (12) we are in a position to undertake an evaluation. To get an idea of how evaluation works, we can provide the picture of (13) which offers a (simplified) illustration of the states of an assignment that occur during an evaluation and so reveals the scope manipulations that take place starting from an initially empty assignment state.

(13)



Working through the pictured evaluation of (13) we see that: (i) there are two distinct instances of existential closure, the first of which introduces one sequence value into the assignment as a `"wa"` binding, and the second of which introduces two sequence values into the assignment as `"e"` bindings; (ii) the contribution of an instance of `kp` is encountered that (a) shifts the `"wa"` binding to an `"h"` binding where it is able to serve as the binding value for the nominal predicate `"私"` 'I', and (b) shifts the `"wa"` binding to a `"ga"` binding from where it is able to serve the subject role for both the main predicate of the sentence `"見ました"` 'watched' and the main predicate of the subordinate clause `"読んで"` 'read'; and so on. What is of special interest to note is that by looking at the terminal nodes we can see that only the correct bindings for the relevant predicates survive.

Following from (13) an overall denotation is derived by the evaluation returning the meaing representation of (14).

(14) $\exists x$(私(x) $\wedge$ $\exists yz$(テレビ(z) $\wedge$ 本(y) $\wedge$ から(読んで(x, y), 見ました(x, z))))

5 Conclusion

We have just started to produce deep semantic representations with high coverage and precison by inputing the result of surface parsing of Japanese and English sentences into an implementation of the SCT system. We are finding that this method works robustly and can be rapidly scaled up, e.g., to cover data from the Penn Treebank and Kyoto Corpus. A notable benefit of the approach is that it requires no rich lexicon owing to its ability to adjust/build the contribution of lexical entries depending on the makeup of the (changing) assignment during the runtime of evaluation together with information about available local and fresh binding names (values for `lc` and `fh` in (12); set by default on a language basis and supplemented with a scan of the input tree).

This work has been supported by JST PRESTO program.

References

Blackburn, P. and J. Bos (2003). Computational semantics. *Theoria 13*, 27–45.

Butler, A. (2010) *The Semantics of Grammatical Dependencies*, Emerald.

Butler, A., Y. Miayo, K. Yoshimoto and J. Tsujii (2010) A Constrained Semantics for Parsed English Sentences. 『言語処理学会　第16回年次大会発表論文集』

Kurohashi, S. and M. Nagao (1994). KN Parser: Japanese Dependency/Case Structure Analyzer. In *Proceedings of the Workshop on Sharable Natural Language Resources*, pp. 48–55

Marcus, M., B. Santorini, and M. A. Marcinkiewicz. (1994) Building a Large Annotated Corpus of English: The Penn Treebank. *Computational Linguistics* 19(2), 313-330.

Miyao, Y., A. Butler, K. Yoshimoto and J. Tsujii. A Modular Architecture for the Wide-Coverage Translation of Natural Language Texts into Predicate Logic Formulas. Ryo Otoguro, et al., eds. (2010) *Proceedings of the 24th Pacific Asia Conference on Language, Information and Computation.*

Palmer, P. and D. Gildea and P. Kingsbury (2005). The Proposition Bank: A Corpus Annotated with Semantic Roles, *Computational Linguistics Journal*, 31:1.

Pollard, C. and I. Sag. (1994) *Head-Driven Phrase Structure Grammar.* The University of Chicago Press.

Steedman, M. (1996) *Surface Structure and Interpretation.* The MIT Press.

Vermeulen, C. F. M. (2000) Variables as Stacks: A Case Study in Dynamic Model Theory. *Journal of Logic, Language and Information* 9, 143-167.