JETCAT – Japanese-English Translation Using Corpus-Based Acquisition of Transfer Rules

Werner Winiwarter
University of Vienna, Department of Scientific Computing, Austria
Email: werner.winiwarter@univie.ac.at

***Abstract*— In this paper we present a rule-based formalism for the acquisition, representation, and application of the transfer knowledge used in a Japanese-English machine translation system. The transfer knowledge is learnt automatically from a parallel corpus by using structural matching between the parse trees of translation pairs. The user can customize the rule base by simply correcting translation results. We have extended the machine translation system with two user-friendly front ends: an MS Word interface and a Web interface. Since our system is mainly intended as a tool for language students to convey a better understanding of Japanese, we also offer the display of detailed information about lexical, syntactic, and transfer knowledge. The system has been implemented in Amzi! Prolog, using the Amzi! Logic Server Visual Basic Module and the Amzi! Logic Server CGI Interface to develop the front ends.**

***Index Terms*— natural language processing, machine translation, linguistic knowledge acquisition, parallel corpora, logic programming**

I. Introduction

Nowadays, a wealth of foreign language texts is readily available via the Web. The study of these online documents is an excellent way to improve language skills because new words and phrases can be learnt in their natural context. However, in particular for Japanese, the self-study of online documents is a cumbersome undertaking because of the following obstacles [1]–[3]:

- the complex writing system comprising a melange of the two syllabaries *hiragana* and *katakana* as well as several thousand Chinese characters called *kanji*,
- the lack of spaces or any other visual indicators for word boundaries,
- the high degree of ambiguity in Japanese grammar, e.g. there exist no articles to indicate gender or definiteness, no declension to mark number or case, etc.,
- the tendency to omit any information that can be inferred implicitly, e.g. the speaker or addressee in dialogs,
- socialinguistic factors, e.g. the avoidance of decisive expressions, instead choosing indirectness for reasons of politeness,
- finally, a confusing system of formality levels including honorific and humble verb forms depending on the social status and relationships between speaker, addressee, and referent.

There are several Web-based tools available to assist the student in comprehending the meaning of a Japanese text. Some Web sites offer pop-up information about kanji, pronunciation data, and word translations. For example, *POPjisho* (www.popjisyo.com) provides English word translations but often produces incorrect results regarding segmentation and tagging of proper names and conjugated word forms. Another popular tool is *Rikai* (www.rikai.com), however, it provides neither segmentation nor information for words written in hiragana, i.e. conjugated forms and function words. A comprehensive list of Japanese online tools, lexica, and other educational resources can be found at www.csse.monash.edu/~jwb/japanese.html.

Although all these tools are certainly of great value for language students, they all suffer from the same shortcoming, i.e. no correct lexical analysis of conjugated forms and function words, which are vital for the understanding of the semantic relations in a sentence. Another problem is that the word translations are just lists of all possible meanings, which can be rather long for common words. Therefore, the task of choosing the correct interpretation for each word in a specific context can become difficult.

Machine translation systems would be an important additional tool for language students. Regrettably, today's commercial programs and Web-based services are still not mature enough, especially regarding the language pair Japanese-English (see Figure 1).

Against this background we have developed *JETCAT* (Japanese-English Translation using Corpus-based Acquisition of Transfer rules). In our machine translation system we learn all transfer rules automatically by using structural matching between the parse trees of translation examples. As training data we use the bilingual data from the JENAAD corpus [4], which contains 150,000 sentence pairs from news articles. The foundations of our rule-based formalism were developed as part of a previous project on Japanese-German translation [5].

JETCAT has been implemented in Amzi! Prolog, which offers an expressive declarative programming language within the Eclipse Platform, powerful unification operations for the efficient application of the transfer rules, and full Unicode support for Japanese characters. In addition,

This paper is based on "Automatic Acquisition of Translation Knowledge Using Structural Matching Between Parse Trees ," by W. Winiwarter, which appeared in the Proceedings of the First International Conference on the Digital Society (ICDS), Guadeloupe, French Caribbean, January 2007.

Japanese sentence:
新しい政治勢力結集の基盤は、政治理念、
基本政策の共有でなくてはなるまい。

Roman transcription:
Atarashii seiji seiryoku kesshū no kiban wa, seiji rinen,
kihon seisaku no kyōyū de naku te wa naru mai.

Correct translation:
A new political force should be based on common political ideas and basic policies.

www.freetranslation.com:
May not become that the base of new politics influence concentration is not the sharing of a politics idea, basis policy.

www.worldlingo.com/en/products_services/worldlingo_translator.htm:
The basis of new political power concentration, not being joint ownership of political belief and basic policy, will not become.

www.excite.co.jp/world/english/:
In case of sharing neither the political belief nor the basic policy, the base of a new political power concentration will not become it.

tool.nifty.com/globalgate/:
If the base of new political influence concentration is not sharing of a political belief and a basic policy, it will not become.

www.brother.co.jp/jp/honyaku/demo/:
The base of the political new power concentration is not the joint ownership of the political belief and the basic policy, and it won't be.

Figure 1. Example output of machine translation systems.

Amzi! Prolog comes with several APIs, in particular the Amzi! Logic Server Visual Basic Module and the Amzi! Logic Server CGI Interface, which we used to develop an MS Word interface and a Web interface so that the user can invoke the translation functionality directly from an editor or browser window. The students can customize their personal transfer rule bases by simply post-editing translation results and resubmitting them to JETCAT. In addition, they can inspect all the intermediate results of a translation process, i.e. token lists, parse trees, and transfer rules. A particularly instructive feature is the single step trace mode, which allows to watch how a Japanese parse tree gradually turns into a fully translated English parse tree.

The rest of the paper is organized as follows. After a brief discussion of related work in Sect. II, we first give an overview of the system architecture of JETCAT in Sect. III. Next, we describe the formalism for the representation of the transfer knowledge in more detail in Sect. IV. Finally, the realization of the MS Word interface and Web interface is illustrated in Sect. V and Sect. VI. We close the paper with some concluding remarks and an outlook on future work in Sect. VII.

II. Related Work

Despite the long history of machine translation research (see [6]–[10]) and the huge amount of effort invested in the development of machine translation systems, the achieved translation quality is still very disappointing [11], [12]. This is true for *transfer-based* systems, which try to find mappings between specific language pairs and even more so for *interlingua-based* approaches aiming to find a language-independent representation that mediates among several languages. The latter often use a semantic formalism as interlingua in which case they are also referred to as *knowledge-based* translation systems [13]–[15]. A well-known representative for an interlingua-based Japanese machine translation system is GAZELLE [16], [17].

All these traditional *rule-based* machine translation systems rely on a careful design of the transfer or interlingual rule base by human experts. Each new rule that is added to the rule base can produce negative side effects on other existing rules. Therefore, it is a difficult and time-consuming task to keep a rule base of reasonable size consistent. Most commercial machine translation products exhibit thus a rather static behavior. The user can only add new words to a custom lexicon or choose between several stylistic preferences for the generation of the translation output. This means that the machine translation system cannot learn from its mistakes whereas a human translator improves his skills with experience over time [18].

As a response to these shortcomings, research interests in machine translation have shifted towards *corpus-based* approaches in the last few years, which try to learn the transfer knowledge from a large parallel corpus for the language pair [19]. The opposite extreme of rule-based systems is *statistical machine translation*, which, in its pure form, uses no additional linguistic knowledge to train both a statistical translation model and target language model [20], [21]. The two models are used to assign probabilities to translation candidates and then to choose the candidate with the maximum score. For the first few years the translation model was built only at the word level, however, as the limitations of these word-based approaches became apparent, several extensions towards phrase-based translation [22] and syntax-based translation [23]–[25] have been proposed, in particular for dissimilar language pairs like Japanese-English. Although some improvements in the translation quality could be achieved, statistical translation has one main disadvantage in common with rule-based translation, i.e. an incremental adaptation of the statistical model by the user is usually impossible. Furthermore, statistical translation, as opposed to rule-based translation, has no easily comprehensible rule base, which makes it unsuitable for language students who want to have an explanation component for a better understanding of the translation process.

Example-based machine translation is a compromise between the two extremes of rule-based and statistical translation [26]–[28]. It uses a parallel corpus to create a database of translation examples for source language fragments. The different approaches vary in how they represent these fragments in the database: as surface strings, structured representations, generalized templates with variables, etc. [29]–[32]. The equivalent target language fragments are retrieved and combined to build the translation. As a hybrid technology, example-based translation inherits some of the weaknesses of both rule-based and statistical translation. On the one hand, the acquisition can discover translation examples automatically, however, manual crafting or at least reviewing of the fragments is mandatory to achieve sufficient accuracy for a corpus of reasonable size [33]. On the other hand, the representation of the translation knowledge in the

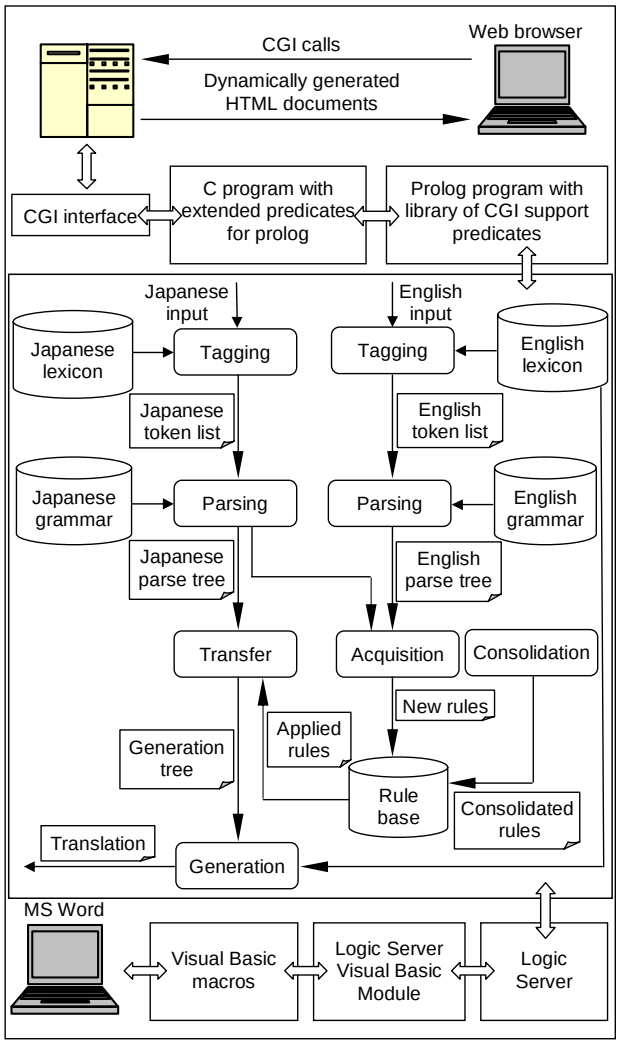

Figure 2. System architecture.

database is less readily convertible to a lucid explanation of the translation process.

III. System Architecture

The JETCAT system architecture is depicted in Figure 2. The three main tasks for the machine translation system are the translation of Japanese input, the acquisition of new transfer rules, and the consolidation of the rule base.

For the *translation* of a Japanese sentence we first analyze it with the *tagging* module, which accesses the Japanese lexicon to produce a list of morphemes with pronunciation, base form, part-of-speech, conjugation type, and conjugation form. The lexicon was compiled automatically by applying the morphological analysis system ChaSen [34] to the JENAAD corpus. Instead of the original numerical ChaSen tags we use more user-friendly three letter acronyms.

Next, the token list is transformed into a parse tree by the *parsing* module with the assistance of the Definite Clause Grammar preprocessor of Amzi! Prolog. The parse tree of a sentence is represented as a list of *constituents*, which are modeled as compound terms of arity 1 with the *constituent category* as principal functor. Regarding the *argument* of a constituent we distinguish between:

- *simple constituent*: word with part-of-speech tag (atom/atom) or syntactic feature (atom), and
- *complex constituent*: phrase (list of subconstituents).

The *transfer* module traverses the parse tree top-down and applies the transfer rules in the rule base to transform the Japanese parse tree into a corresponding English generation tree. We also perform some standard transformations, the two most common ones are:

- the removal of Japanese particles that indicate the relationship of a phrase to the embedding phrase, these particles are often redundant because the relationship is already expressed through the category of the complex constituent,
- the addition of the coordinating conjunction “and”, which is often not explicitly expressed in Japanese.

As last processing step of a translation, the *generation* module produces the final English sentence by traversing the generation tree top-down and computing a nested list of surface forms, which is afterwards flattened and converted into a string. Irregular inflections are produced by accessing the English lexicon, which was also built automatically by applying the MontyTagger [35] to the JENAAD corpus.

The tagging and parsing of English sentences are necessary preprocessing steps for the *acquisition* of new transfer rules. The *tagging* module segments the English input into morphemes, and annotates each morpheme with its base form and part-of-speech tag from the Penn Treebank tagset. For the convenience of the reader we list the textual descriptions of the Japanese and English part-of-speech tags used in this paper in Table I. The *parsing* module applies grammar rules written again in Definite Clause Grammar syntax to the token list to compute the structural representation of the English sentence as parse tree.

The *acquisition* module traverses the Japanese and English parse tree and derives new transfer rules. The

TABLE I.
PART-OF-SPEECH TAGS

Japanese	adn	adverbial dependent noun
	axv	auxiliary verb
	cma	comma
	cou	country
	cno	copular noun
	fna	family name
	mdp	modifying particle
	nou	noun
	par	particle
	per	period
	pno	predicative noun
	pnp	prenominal particle
	ver	verb
	vsu	verbal suffix
English	in	preposition
	jj	adjective
	nn	noun
	nnp	proper noun
	vb	verb

search for new rules starts at the sentence level by recursively mapping the individual subconstituents of the Japanese sentence. There exists a specific rule for each Japanese constituent category to match a subconstituent of this category (and potentially other subconstituents) with English subconstituents to derive a transfer rule. Each derived rule is added to the rule base if it is not included yet. All Japanese and English subconstituents that are covered by the derived rule are removed from the input before continuing the search for new rules. The default mapping of a Japanese subconstituent is to find an English subconstituent with identical constituent category and to continue the matching procedure recursively for the arguments of the two constituents.

The rules that are learnt by the acquisition procedure are rather specific because they consider contextual translation dependencies in full detail to produce accurate translations and to avoid any conflict with other transfer rules in the rule base. However, this high degree of specificity badly affects the coverage for new unseen data. Therefore, the task of the *consolidation* module is to generalize transfer rules by relaxing their condition part as long as this does not introduce a conflict with another rule in the rule base.

For the moment, JETCAT comes with two user-friendly front ends. The first option is an MS Word interface. The user invokes *Visual Basic macros*, which call procedures declared in the *Logic Server Visual Basic Module* to communicate with the *Logic Server*. The Logic Server is the Prolog runtime engine packaged as DLL. It has a number of public methods to implement the Logic Server API, and it loads and runs the compiled Prolog code for the machine translation system.

The second possibility is to access JETCAT via a Web interface. The user's Web browser sends CGI calls to the Web server, which calls the *CGI interface* to return dynamically generated HTML documents. The CGI application consists of a *C program* responsible for starting the Amzi! Logic Server and loading the Prolog CGI script. All user input and CGI variables are asserted as facts to the Prolog logicbase before calling the Prolog part of the CGI Amzi! interface. This *Prolog wrapper* performs the necessary CGI bookkeeping functions and calls predicates defined in the Prolog script implementing the machine translation system.

IV. Transfer Knowledge

In our approach, we have chosen a very flexible and robust formalism to represent the transfer knowledge. We model all translation situations with just three generic rule types:

- a *word transfer rule* translates the argument of a simple constituent,
- a *constituent transfer rule* translates both the category and the argument of a complex constituent,
- a *phrase transfer rule* allows to define elaborate conditions and substitutions on the argument of a complex constituent.

All the transfer rules are actually stored as Prolog facts in the rule base. Figure 3 shows an example of the rules learnt from a sentence pair for an empty rule base. The 9 rules are learnt in that order by the acquisition module. We also indicate the generalizing transformations produced by the consolidation module resulting in 10 simplified rules.

In the following subsections, we explain the three different rule types in more detail by using the rules in Figure 3 as well as one additional illustrative example. For the ease of the reader, we use Roman transcriptions of Japanese characters.

A. Word Transfer Rules

For simple context-insensitive translations at the word level, the argument of a simple constituent in the input a_i is changed to its translation t_a by applying the following predicate, i.e. if a_i is equal to the *argument condition* in the transfer rule a_r, it is replaced by t_a:

$$wtr(a_r, t_a).$$

Such a rule changes a Japanese word and its part-of-speech tag to the equivalent English word and part-of-speech tag.

Example 1. Rule 3a: *wtr*(*shidō/pno, leadership/nn*). This states that the predicative noun *shidō/pno* is translated as noun *leadership/nn*. A predicative noun can be used as a verb, e.g. in this case "to lead", by adding the verb *suru*, "to do".

Example 2. Rule 8: *wtr*(*kaikaku/pno, reform/nn*). This rule translates the predicative noun *kaikaku/pno* as noun *reform/nn*.

Example 3. Rule 9b: *wtr*(*noridasu/ver, embark/vb*). The application of this rule changes the verb *noridasu/ver* into the verb *embark/vb*.

Acquisition. Whenever the acquisition procedure reaches two simple constituents with identical categories, a new word transfer rule is derived. Many word transfer rules (e.g. Rule 3a and Rule 9b) are generated by the consolidation module as a result of generalizing phrase transfer rules.

Therefore, word transfer rules should be interpreted with caution, i.e. they are only valid as long as no conflicting translation exists in the training data that would require a more contextualized rule. Furthermore, the rules have to be understood as unidirectional translations from Japanese into English, e.g. in Example 1 "leadership" would be rather translated as "shidōryoku" than "shidō".

Transfer. The transfer module tries to apply a word transfer rule, once it reaches the argument of a simple constituent during the traversal of the parse tree. If the argument of the simple constituent a_i equals a_r, it is substituted with t_a.

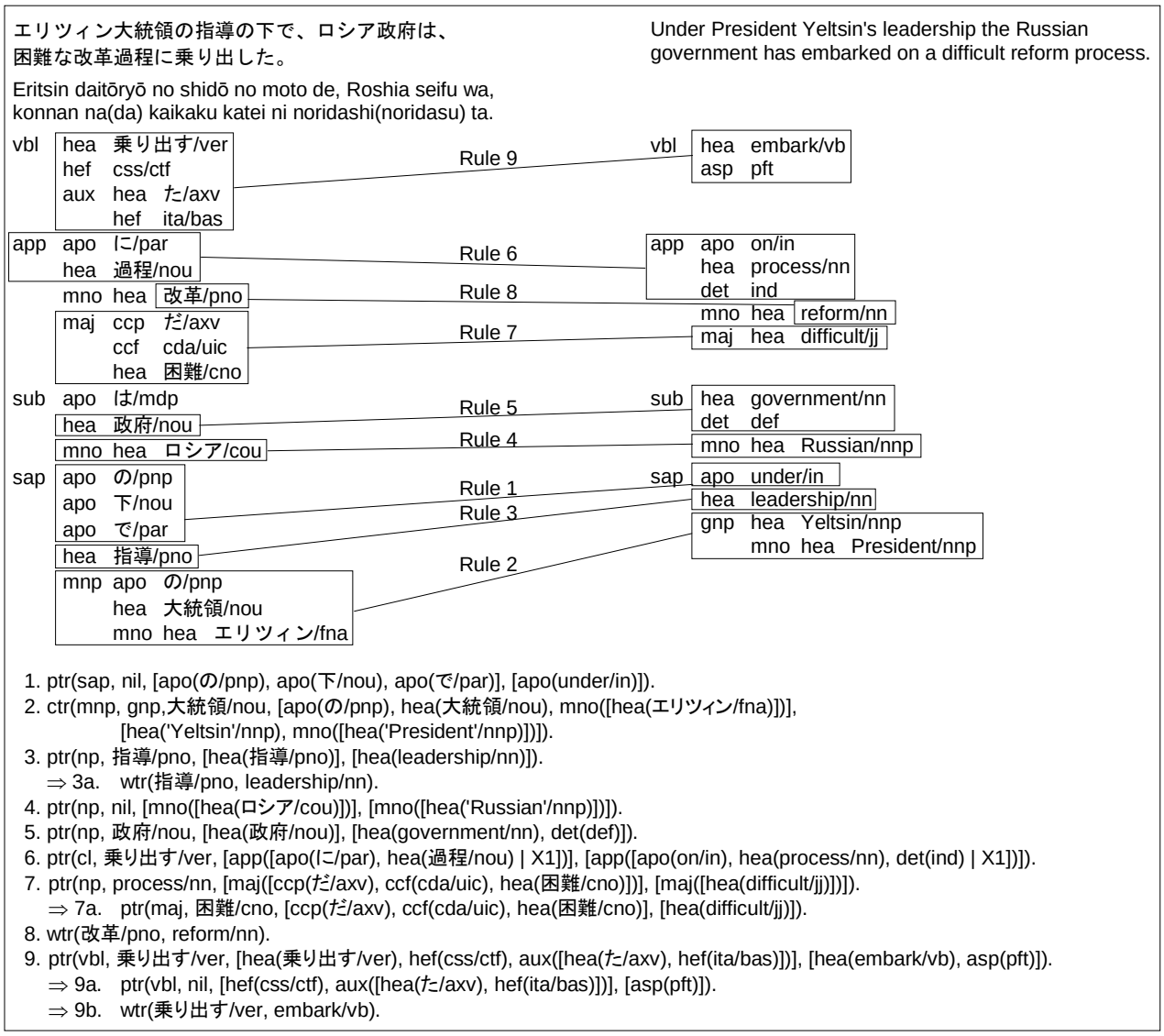

Figure 3. Example of transfer rules.

B. Constituent Transfer Rules

The second rule type concerns the translation of complex constituents to cover cases where both the category and the argument of a constituent have to be altered:

$$ctr(c_r, t_c, h_r, a_r, t_a).$$

This changes a complex constituent $c_i(a_i)$ to $t_c(t_a)$ if the category c_i is equal to *category condition* c_r, the head h_i is equal to *head condition* h_r, and the argument a_i is unifiable with a_r.

The head condition serves as index for the fast retrieval of matching facts during the translation of a sentence and significantly reduces the number of facts for which the argument condition has to be tested. For clauses, h_i is retrieved from the head of the verbal of the clause.

Constituent transfer rules may contain *shared variables for unification*. They make it possible to translate only certain subconstituents of the complex constituent whereas the rest of the argument remains intact.

Example 4. Rule 2 translates the modifying noun phrase (*mnp*) "*Eritsin daitōryō no*" as genitive noun phrase (*gnp*) "*President Yeltsin's*". In more detail, the modifying noun phrase contains the prenominal particle *no/pnp* as adposition (*apo*), the noun *daitōryō/nou* as head (*hea*), and the family name *Eritsin/fna* as head of a modifying noun (*mno*):

ctr(*mnp, gnp, daitōryō/nou,* [*apo*(*no/pnp*), *hea*(*daitōryō/nou*), *mno*([*hea*(*'Eritsin'/fna*)])], [*hea*(*'Yeltsin'/nnp*), *mno*([*hea*(*'President'/nnp*)])]).

Example 5. The following constituent transfer rule is an example of the usage of shared variables for unification:

ctr(*mnp, mtc, tame/adn,* [*apo*(*no/pnp*), *hea*(*tame/adn*), *mcl*($X1$)], $X1$).

The rule changes a modifying noun phrase "$X1$ *tame no*" with *no/pnp* as adposition, the adverbial dependent noun *tame/adn* as head, and $X1$ as modifying clause (*mcl*) into a modifying to-infinitive clause (*mtc*) $X1$.

For example, the application of the rule to the Japanese input "Y *chōwa sa seru tame no*" ("*to harmonize Y*") leads to:

$$\begin{aligned}
&c_i = \textit{mnp},\\
&h_i = \textit{tame/adn},\\
&a_i = [\textit{apo}(\textit{no/pnp}), \textit{hea}(\textit{tame/adn}),\\
&\quad \textit{mcl}([\textit{vbl}([\textit{hea}(\textit{suru/ver}), \textit{hef}(\textit{isu/icr}),\\
&\quad \textit{aux}([\textit{hea}(\textit{seru/vsu}), \textit{hef}(\textit{vsv/bas})]),\\
&\quad \textit{prn}(\textit{chōwa/pno})]), \textit{dob}(Y)])],\\
&t_c = \textit{mtc},\\
&t_a = [\textit{vbl}([\textit{hea}(\textit{suru/ver}), \textit{hef}(\textit{isu/icr}),\\
&\quad \textit{aux}([\textit{hea}(\textit{seru/vsu}), \textit{hef}(\textit{vsv/bas})]),\\
&\quad \textit{prn}(\textit{chōwa/pno})]), \textit{dob}(Y)].
\end{aligned}$$

This means that the argument of the modifying clause is transformed into t_a whereas the individual subconstituents are left unchanged, i.e. a verbal (*vbl*) and a direct object Y (to shorten the example). The verbal consists of the head verb *suru/ver*, the head form *(hef)* indicating the conjugation type (*isu*: irregular verb 'suru') and the conjugation form (*icr*: imperfective connection with 'reru'), the verbal suffix *seru/vsu* (head form *vsv*: vowel-stem verb / *bas*: base form) as auxiliary (*aux*), and the predicative noun (*prn*) *chōwa/pno*. As mentioned before, the verb *suru* changes the predicative noun *chōwa* (*harmony*) into a verb, the verbal suffix *seru* is used to derive the causative form *saseru* of the verb *suru*.

Acquisition. Constituent transfer rules are learnt by the acquisition module if it encounters a situation where a complex constituent in the Japanese parse tree corresponds to a complex constituent with a different category in the English parse tree.

Transfer. If the transfer module arrives at a complex constituent during the traversal of the parse tree, it first tries to apply a constituent transfer rule before it continues its search for the argument of the complex constituent. To find suitable rule candidates the transfer module first checks if c_r equals c_i and h_r equals h_i. If the category and head condition are satisfied, it tries to unify a_r with a_i. If the unification is successful, t_c and t_a are used to build the English equivalent $t_c(t_a)$ of the complex constituent by binding any shared variables as shown in Example 5.

C. Phrase Transfer Rules

The most common and most versatile type of transfer rules are phrase transfer rules, which translate the argument a_i of a complex constituent $c_i(a_i)$:

$$ptr(c_r, h_r, a_r, t_a).$$

In addition to an exact match, the generalized categories *cl* (clause) and *np* (noun phrase) can be used for the category condition c_r. c_r must then subsume c_i, i.e. $c_i \sqsubseteq c_r$. The head condition is defined in the same way as for constituent transfer rules, i.e. h_i must equal h_r. If the applicability of a phrase transfer rule does not depend on h_i, then the special constant *nil* can be used for h_r. In addition, h_r can be set to the special constant *notex* to indicate that a_i must not contain a head, i.e. $\nexists h_i$.

One important precondition for the efficient and robust application of phrase transfer rules by the transfer module is that the condition expressed by a_r is interpreted as subset condition, i.e. $a_r \subseteq a_i$. All additional subconstituents $a_i \backslash a_r$ are appended to t_a unchanged. That way one phrase transfer rule may change only certain elements of a phrase whereas all other elements are translated later on by other transfer rules. The order of the subconstituents does not affect the satisfiability of the argument condition. This set property does not only apply to the top level of a_r but extends recursively to any level of detail specified in a_r. It is also possible to use the special constant *notex* as argument of a subconstituent in a_r, e.g. *sub*(*notex*). In that case the rule can only be applied if no subconstituent of this category is included in a_i, e.g. if a_i does not include any subject: $\textit{sub}(S) \notin a_i$.

Just as in the case of constituent transfer rules, also the expressiveness of phrase transfer rules can be increased significantly by using shared variables for unification.

Example 6. Rule 1 states that the sentence-initial phrase (*sap*) "X *no moto de*" with the three adpositions *no/pnp, moto/nou,* and particle *de/par* is translated as "*under* X":

$$\begin{aligned}
&\textit{ptr}(\textit{sap}, \textit{nil}, [\textit{apo}(\textit{no/pnp}), \textit{apo}(\textit{moto/nou}), \textit{apo}(\textit{de/par})],\\
&\quad [\textit{apo}(\textit{under/in})]).
\end{aligned}$$

Example 7. Rule 3 transforms the head *shidō/pno* of a noun phrase into *leadership/nn*:

$$\begin{aligned}
&\textit{ptr}(\textit{np, shidō/pno,} [\textit{hea}(\textit{shidō/pno})],\\
&\quad [\textit{hea}(\textit{leadership/nn})]).
\end{aligned}$$

Example 8. Rule 4 changes the country *'Roshia'/cou*, when used as a modifying noun, into *'Russian'/nnp*:

$$\begin{aligned}
&\textit{ptr}(\textit{np}, \textit{nil}, [\textit{mno}([\textit{hea}(\textit{'Roshia'/cou})])],\\
&\quad [\textit{mno}([\textit{hea}(\textit{'Russian'/nnp})])]).
\end{aligned}$$

Example 9. Rule 5 translates *seifu/nou* as the noun *government/nn* with definite determiner *the*, expressed as syntactic feature *det*(*def*):

$$\begin{aligned}
&\textit{ptr}(\textit{np}, \textit{seifu/nou,} [\textit{hea}(\textit{seifu/nou})],\\
&\quad [\textit{hea}(\textit{government/nn}), \textit{det}(\textit{def})]).
\end{aligned}$$

Example 10. Rule 7 can be applied to a noun phrase with head *process/nn*. It replaces the modifying adjective phrase (*maj*) "*konnan na*" with *difficult*:

$$\begin{aligned}
&\textit{ptr}(\textit{np, process/nn,} [\textit{maj}([\textit{ccp}(\textit{da/axv}), \textit{ccf}(\textit{cda/uic}),\\
&\quad \textit{hea}(\textit{konnan/cno})])], [\textit{maj}([\textit{hea}(\textit{difficult/jj})])]).
\end{aligned}$$

The individual elements of the Japanese modifying adjective phrase are the auxiliary verb *da/axv* used as connective copula (*ccp*), the connective copula form (*ccf*): uninflected connection (*uic*) of the copula 'da' (*cda*), and the copular noun *konnan/cno*. A copular noun is a noun that can be used as an adjective in such a context. Rule 7 is generalized to Rule 7a by the consolidation module by removing the head condition for *process/nn* and moving

the rule context one level down to the modifying adjective phrase:

ptr(*maj*, *konnan/cno,* [*ccp*(*da/axv*), *ccf*(*cda/uic*), *hea*(*konnan/cno*)], [*hea*(*difficult/jj*)]).

Example 11. Rule 9 deals with the translation of the verbal "*noridashi ta*", which consists of the continuative form (*ctf*) of the consonant-stem verb with ending 'su' (*css*) *noridasu/ver* as head verb, and the base form of the irregular verb 'ta' (*ita*) as auxiliary verb *ta/axv*. The head verb is translated as *embark/vb*; the auxiliary verb *ta* indicates perfect aspect, which is expressed by the syntactic feature *asp*(*pft*):

ptr(*vbl, noridasu/ver,* [*hea*(*noridasu/ver*), *hef*(*css/ctf*), *aux*([*hea*(*ta/axv*), *hef*(*ita/bas*)])], [*hea*(*embark/vb*), *asp*(*pft*)]).

Rule 9 is split into two more general rules, Rule 9b for the translation of the head verb (see Example 3) and Rule 9a for the translation of the aspect:

ptr(*vbl, nil,* [*hef*(*css/ctf*), *aux*([*hea*(*ta/axv*), *hef*(*ita/bas*)])], [*asp*(*pft*)]).

Example 12. Finally, Rule 6 is an example of the use of shared variables for unification. It states that for a clause with head verb *noridasu*, the adpositional phrase (*app*) "$X1$ *katei ni*" is substituted with "*on a* $X1$ *process*":

ptr(*cl, noridasu/ver,* [*app*([*apo*(*ni/par*), *hea*(*katei/nou*)|$X1$])], [*app*([*apo*(*on/in*), *hea*(*process/nn*), *det*(*ind*)|$X1$])]).

The indefinite article is indicated as syntactic feature *det*(*ind*). For example, for the sentence in Figure 3 the application of the rule could look as follows (using ... and variables *N*, *A*, *Sub*, and *Sap* to shorten the example):

c_i = *cl*,
h_i = *noridasu/ver*,
a_i = [*vbl*([*hea*(*noridasu/ver*), ...]), *app*([*hea*(*katei/nou*), *mno*(N), *apo*(*ni/par*), *maj*(A)]), *sub*(*Sub*), *sap*(*Sap*)],
t_a = [*app*([*apo*(*on/in*), *hea*(*process/nn*), *det*(*ind*), *mno*(N), *maj*(A)]), *vbl*([*hea*(*noridasu/ver*), ...]), *sub*(*Sub*), *sap*(*Sap*)].

Acquisition. Phrase transfer rules are used by the acquisition module to account for all situations that cannot be handled by the other two rule types, in particular to model contextual translation dependencies.

Transfer. The transfer module starts at the top level of the Japanese parse tree and tries to apply phrase transfer rules. For a successful rule application, we first collect all rule candidates that satisfy the conditions in c_r, h_r, and a_r. Then we rate each rule and choose the rule with the highest score. The score is calculated based on the complexity of a_r, i.e. it is recursively computed from the number of subconstituents in a_r. In addition, rules are assigned a higher score, if: $h_r \neq$ *nil*, a_r does not contain the head of the phrase, or if *notex* is used in a_r.

The verification of the argument condition a_r is a quite complex task because it requires testing for set inclusion at the top level ($a_r \subseteq a_i$) as well as recursively testing for set unifiability of arguments of subconstituents. We solve this problem by removing each constituent in a_r from a_i, at the same time binding free variables in a_r and t_a through unification. The remaining constituents from the input $a_i \setminus a_r$ are returned as a list of additional elements to be appended to t_a. A constituent in a_r can be removed from a_i if the two constituents co_r and co_i can be directly unified, or if their categories are identical and their arguments a_{co_r} and a_{co_i} are unifiable sets. The latter condition is verified by again removing each subconstituent in a_{co_r} from a_{co_i} until either a free variable as tail of a_{co_r} (i.e. $|X1]$) or the end of both lists has been reached. In addition, any *notex* condition has to be verified by the satisfiability test.

If no more rules can be applied at the sentence level, each constituent in the sentence is examined individually. We first search for constituent transfer rules before we perform a transfer of the argument. The latter involves the application of word transfer rules for simple constituents, whereas the top-level procedure is repeated recursively for complex constituents.

V. MS Word Interface

We have developed a user-friendly MS Word interface so that the translation functionality is directly available from any editor window, see Figure 4 for a screenshot. All tasks can be invoked via two toolbars. The commands in the first toolbar concern Japanese sentences. The user can click anywhere in a Japanese document and select a command. This results in the automatic extraction of the sentence at the cursor position, the execution of the task by JETCAT, and the insertion of the formatted output with borders after the analyzed sentence.

The user can retrieve the English translation of a Japanese sentence by clicking on "Translation". In addition, it is possible to inspect all the intermediate results of the translation process via the commands "Japanese Token List", "Japanese Parse Tree", and "Generation Tree". The language student can also select "Applied Rules" to receive an enumerated list of the transfer rules used by the transfer module in the correct order of their application.

A particularly instructive feature is the single step trace mode. Starting from the original Japanese parse tree, the user can watch how the tree gradually turns into the completely translated generation tree. If the user clicks on "Single Step Trace", the first transfer rule applied by the transfer module is displayed together with its effects on the parse tree. By clicking repeatedly on "Single Step Trace", the users can follow the progress of the transfer module and get a better understanding of the translation process. For example, Figure 4 depicts the output after clicking on "Single Step Trace" for the third time, i.e.

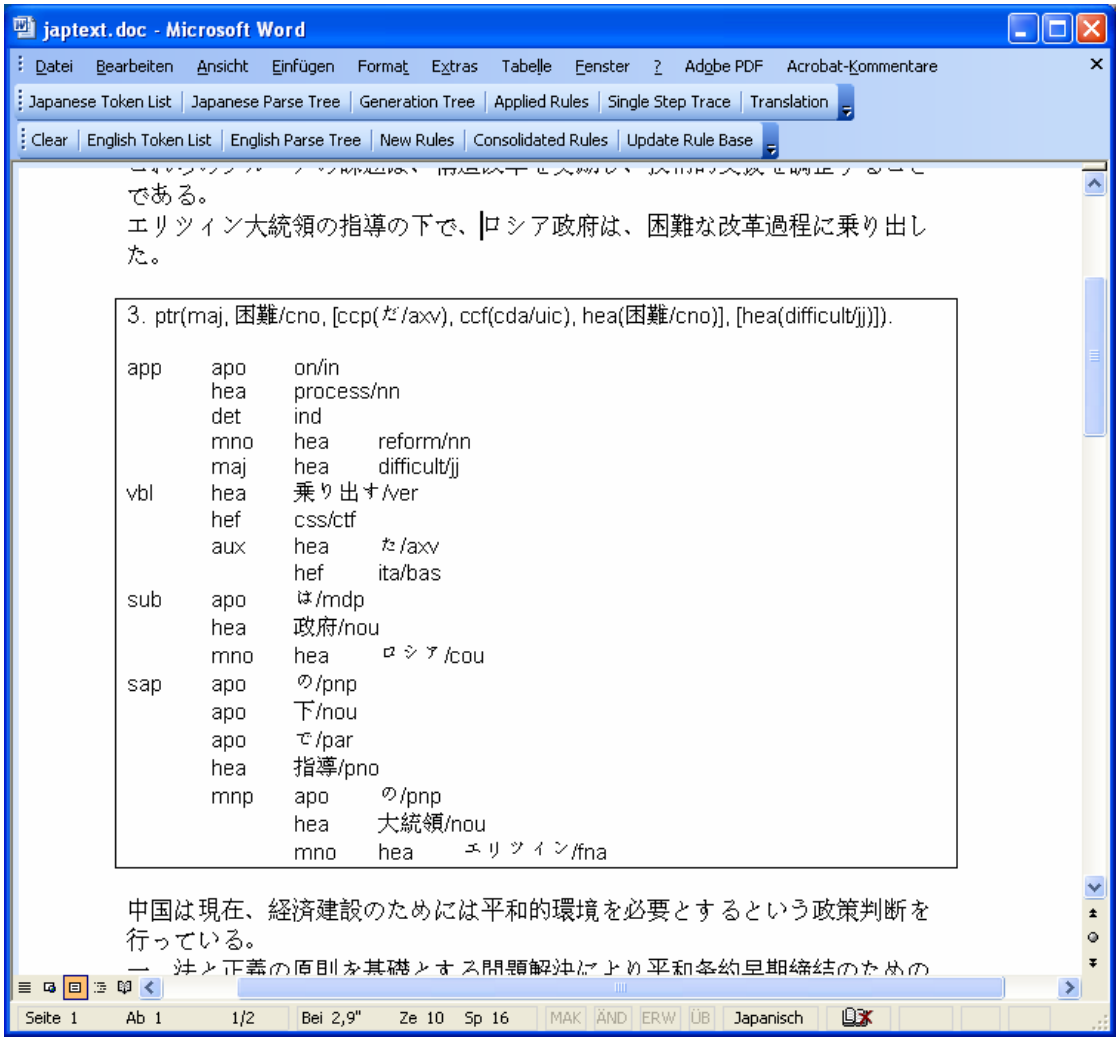

Figure 4. Screenshot of MS Word interface.

after applying Rule 7a from Figure 3 to the example sentence.

The first command of the second toolbar, “Clear”, is used to delete the last output produced by JETCAT. All the other commands of the second toolbar concern English sentences or Japanese-English sentence pairs. The user can view the intermediate results of the linguistic analysis for English sentences by selecting “English Token List” and “English Parse Tree”. The English sentence is again automatically extracted from the current cursor position.

To better comprehend the acquisition task, the language student can click on “New Rules” for a Japanese-English sentence pair. JETCAT returns an enumerated list of transfer rules that could be learnt from this translation example in the correct order of their derivation. Furthermore, by choosing “Consolidated Rules”, the user can scrutinize the generalizing transformations that would be performed for this sentence pair by the consolidation module.

Finally, one important functionality of JETCAT is the possibility to customize translation results by simply correcting them in the editor window and updating the rule base with the command “Update Rule Base”. Before this, the user can verify the consequences of the changes on the acquisition procedure with “New Rules” and “Consolidated Rules”. As soon as the revised translation has been committed with “Update Rule Base”, the sentence will be always translated that way.

VI. Web Interface

In addition to the MS Word interface described in the previous section we also provide the possibility to access JETCAT through a Web browser. This means that the user does not have to install the machine translation system on his local computer, instead he only has to connect to the Web server hosting the JETCAT system. The user can either directly input a Japanese sentence into the Web interface (see Figure 5) or use a Visual Basic macro to open a browser window from MS Word. In the latter case, the macro extracts the Japanese sentence at the cursor position and calls the Web server via the GET method by adding the sentence as query string. The Web server responds by returning the Web form with the Japanese input sentence.

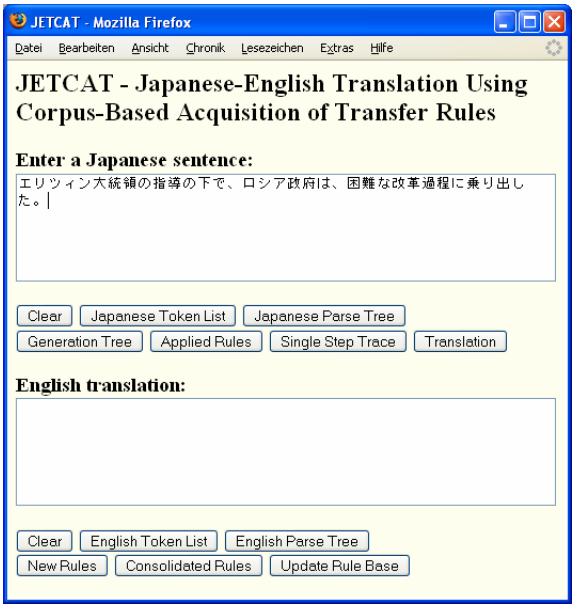

Figure 5. Screenshot of Web interface.

If the user clicks on the "Japanese Token List" button, the Japanese sentence is sent to the Web server via the POST method, which returns a tabular display of the lexical data as shown in Figure 6. The three columns contain the following information about the individual word tokens:

- *surface form*: if the inflected form differs from the base form, the latter is appended in parentheses,
- *Roman transcription* of surface form (and base form),
- *part-of-speech tag* (and tags for conjugation type / conjugation form): with "Display Legend", a list of textual descriptions can be displayed.

For the Roman transcription we use the Hepburn Romanization system. We retrieve the pronunciation data stored as katakana from the Japanese lexicon and map the katakana syllables to their Romanized versions. In a second step we deal with morphological alternations for certain syllable combinations, the capitalization of proper nouns, and other special cases.

The language student can also click on "Japanese Parse Tree" to receive an HTML table with a nicely formatted representation of the sentence structure. In the same way, the user can select "Generation Tree", "Applied Rules", and "Single Step Trace" to inspect the other intermediate results of the translation process. When clicking on "Translation", the English translation of the Japanese input sentence is directly inserted into the second text area of the Web form (see Figure 5). This way, the student can look at the result of the target language analysis ("English Token List" and "English Parse Tree") for the original translation as well as for any corrections suggested by the student. The remaining buttons at the end of the Web form concern both text areas, i.e. the Japanese-English sentence pair. The student can view details about the acquisition task by selecting "New Rules" and "Consolidated Rules".

Figure 6. Screenshot of token list.

Finally, he can resubmit any post-edited translation to JETCAT via "Update Rule Base" so that the sentence will be translated that way in the future. For that purpose, we keep copies of the default rule base derived from JENAAD for the individual users of our system so that each user can have his own customized JETCAT version.

VII. Conclusion

In this paper, we have presented JETCAT, a rule-based Japanese-English machine translation system based on the automatic acquisition of the transfer knowledge from a parallel corpus. We have finished the implementation of the system for a subset of the JENAAD corpus including an MS Word interface and a first local prototype configuration of the Web application to demonstrate the feasibilty of our approach.

Future work will focus on extending the coverage of the system to the complete corpus and on performing a quantitative evaluation of the translation quality by using ten-fold cross-validation on the JENAAD corpus. Moreover, we are working on additional features, e.g. the display of the different readings for individual kanji, to make the system more user friendly.

We intend to let students of Japanese studies at our university use JETCAT to receive valuable feedback. We will design questionnaires with questions regarding usability issues and the comprehensibility and usefulness of the linguistic information provided by JETCAT. We also plan to make a demo version of the Web application publicly available in the near future.

Finally, we are also working on a Web-based language learning tool that randomly chooses translation examples from JENAAD and presents them to the students using JavaScript to dynamically open pop-up windows with additional color-coded information derived from the linguistic knowledge computed by JETCAT.

REFERENCES

[1] Y. M. McClain, *Handbook of Modern Japanese Grammar*. The Hokuseido Press, 1981.

[2] S. Makino and M. Tsutsui, *A Dictionary of Basic Japanese Grammar*. The Japan Times, 1986.

[3] ——, *A Dictionary of Intermediate Japanese Grammar*. The Japan Times, 1995.

[4] M. Utiyama and H. Isahara, "Reliable measures for aligning Japanese-English news articles and sentences," in *Proceedings of the 41st Annual Meeting of the ACL*, Sapporo, Japan, 2003, pp. 72–79.

[5] W. Winiwarter, "Incremental learning of transfer rules for customized machine translation," in *Applications of Declarative Programming and Knowledge Management*, ser. Lecture Notes in Artificial Intelligence, U. Seipel *et al.*, Eds. Springer-Verlag, 2005, vol. 3392, pp. 47–64.

[6] J. Hutchins, *Machine Translation: Past, Present, Future*. Ellis Horwood, 1986.

[7] ——, "Machine translation over 50 years," *Histoire epistémologie langage*, vol. 23, no. 1, pp. 7–31, 2001.

[8] ——, "Has machine translation improved? Some historical comparisons," in *Proceedings of the 9th MT Summit*, New Orleans, USA, 2003, pp. 181–188.

[9] J. Hutchins and H. Somers, *An Introduction to Machine Translation*. Academic Press, 1992.

[10] J. Newton, Ed., *Computers in Translation: A Practical Appraisal*. Routledge, 1992.

[11] H. Somers, Ed., *Computers and Translation: A Translator's Guide*. John Benjamins, 2003.

[12] J. Hutchins, "Current commercial machine translation systems and computer-based translation tools," *International Journal of Translation*, vol. 17, no. 1-2, 2005.

[13] J. R. R. Leavitt, D. W. Lonsdale, and A. M. Franz, "A reasoned interlingua for knowledge-based machine translation," in *Proceedings of the 10th Canadian Conference on Artificial Intelligence*, Banff, Canada, 1994.

[14] S. Nirenberg *et al.*, Eds., *Machine Translation: A Knowledge-Based Approach*. Morgan Kaufmann Publishers, 1992.

[15] B. Onyshkevych and S. Nirenberg, "A lexicon for knowledge-based MT," *Machine Translation*, vol. 10, no. 1-2, pp. 5–57, 1995.

[16] K. Knight and S. Luk, "Building a large-scale knowledge base for machine translation," in *Proceedings of the American Association of Artificial Intelligence*, Seattle, USA, 1994.

[17] U. Germann, "Making semantic interpretation parser-independent," in *Proceedings of the 3rd AMTA Conference*, Longhorne, USA, 1998, pp. 286–299.

[18] J. Hutchins, "Machine translation and computer-based translation tools: What's available and how it's used," in *A New Spectrum of Translation Studies*, J. M. Bravo, Ed. Univ. Valladolid, 2004, pp. 13–48.

[19] M. Carl, "Towards a model of competence for corpus-based machine translation," in *Hybrid Approaches to Machine Translation*, ser. IAI Working Papers, O. Streiter, M. Carl, and J. Haller, Eds. IAI, 1999, vol. 36.

[20] P. Brown, "A statistical approach to machine translation," *Computational Linguistics*, vol. 16, no. 2, pp. 79–85, 1990.

[21] ——, "The mathematics of statistical machine translation: Parameter estimation," *Computational Linguistics*, vol. 19, no. 2, pp. 263–311, 1993.

[22] P. Koehn, F. J. Och, and D. Marcu, "Statistical phrase-based translation," in *Proceedings of HLT-NAALT*, Edmonton, Canada, 2003, pp. 48–54.

[23] K. Yamada and K. Knight, "A syntax-based statistical translation model," in *Proceedings of the 39th Annual Meeting of the ACL*, Toulouse, France, 2001, pp. 523–530.

[24] K. Imamura, "Hierarchical phrase alignment harmonized with parsing," in *Proceedings of the 6th Natural Language Processing Pacific Rim Symposium*, Tokyo, Japan, 2001.

[25] T. Watanabe, K. Imamura, and E. Sumita, "Statistical machine translation based on hierarchical phrase alignment," in *Proceedings of the 9th International Conference on Theoretical and Methodological Issues in Machine Translation*, Keihanna, Japan, 2002, pp. 188–198.

[26] M. Nagao, "A framework of a mechanical translation between Japanese and English," in *Artificial and Human Intelligence*, A. Elithorn and R. Banerji, Eds. North Holland, 1984, pp. 173–180.

[27] S. Sato, "Example-based machine translation," Ph.D. dissertation, Kyoto University, 1991.

[28] J. Hutchins, "Towards a definition of example-based machine translation," in *Proceedings of the 2nd Workshop on Example-Based Machine Translation at MT Summit X*, Phuket, Thailand, 2005, pp. 63–70.

[29] H. Kaji, Y. Kida, and Y. Matsumoto, "Learning translation examples from bilingual text," in *Proceedings of the 14th International Conference on Computational Linguistics*, Nantes, France, 1992, pp. 672–678.

[30] O. Furuse and H. Iida, "Cooperation between transfer and analysis in example-based framework," in *Proceedings of the 14th International Conference on Computational Linguistics*, Nantes, France, 1992, pp. 645–651.

[31] C. Brockett *et al.*, "English-Japanese example-based machine translation using abstract linguistic representations," in *Proceedings of the COLING-2002 Workshop on Machine Translation in Asia*, Taipei, Japan, 2002.

[32] M. Carl and A. Way, Eds., *Recent Advances in Example-Based Machine Translation*. Kluwer, 2003.

[33] S. Richardson *et al.*, "Overcoming the customization bottleneck using example-based MT," in *Proceedings of the ACL Workshop on Data-driven Machine Translation*, Toulouse, France, 2001, pp. 9–16.

[34] Y. Matsumoto *et al.*, "Japanese morphological analysis system ChaSen version 2.0 manual," NAIST, Tech. Rep. NAIST-IS-TR99009, 1999.

[35] H. Liu, "MontyLingua: An end-to-end natural language processor with common sense," MIT Media Lab, Tech. Rep., 2004.

Werner Winiwarter is the Vice Head of the Department of Scientific Computing, University of Vienna, Austria. He received his MS degree in 1990, his MA degree in 1992, and his PhD degree in 1995, all from the University of Vienna, Austria. The main research interest of Prof. Winiwarter is human language technology. In addition, he also works on data mining and machine learning, Semantic Web, information retrieval, electronic business, and education systems.