Automatic Acquisition of Translation Knowledge Using Structural Matching Between Parse Trees

Werner Winiwarter
Department of Scientific Computing, University of Vienna
Universitätsstraße 5, A-1010 Vienna, Austria
Email: werner.winiwarter@univie.ac.at

Abstract— **In this paper we present a rule-based formalism for the representation, acquisition, and application of translation knowledge. The formalism is being used successfully in a Japanese-English machine translation system. The translation knowledge is learnt automatically from a parallel corpus using structural matching between the parse trees of translation examples. We have developed a comfortable user interface, which makes it possible to invoke the translation functionality directly from MS Word. The user can customize the translation knowledge by simply correcting translation results in MS Word. Our system is mainly intended for language students, therefore, we also offer the display of detailed information about linguistic and translation knowledge, in particular token lists, parse trees, translation rules, and a single step trace mode to provide a better understanding of the translation process.**

I. Introduction

Despite the long experience and huge amount of effort invested in the development of *rule-based machine translation* systems, the translation quality that can be achieved is still very disappointing [1], e.g. Fig. 1 shows three examples of poor results from free Web-based translation services for a quite short and simple Japanese sentence.

Japanese sentence:
国際テロは、世界の平和と安全に対する重大な脅威だ。
Roman transcription:
kokusai tero wa, sekai no heiwa to anzen nitaisuru juudai na kyoui da.

Correct translation:
International terrorism is a grave threat to world peace and security.

www.freetranslation.com:
The international terrorism is the important threat that corresponds safely with the peace of the world.
www.worldlingo.com/en/products_services/worldlingo_translator.htm:
International terrorism is peace of the world and the serious threat for safely.
www.brother.co.jp/jp/honyaku/demo/:
International terrorism is the serious menace to face safely with the peace of the world.

Fig. 1. Example of Machine Translation Output

One major reason for this unsatisfying situation is that the translation rules have to be carefully designed by human experts. Each new rule can have negative side effects on other existing rules. Therefore, it is a non-trivial task to keep a translation rule base of reasonable size consistent. Most commercial machine translation products have thus a rather static behavior. The user can only add new words to a customized lexicon or choose between several preferences for the generation of the translation output. This means that a machine translation system cannot learn from its mistakes whereas a human translator improves his skills with experience over time [2].

The most common solution to this knowledge acquisition bottleneck have been *corpus-based machine translation* approaches, which learn the translation knowledge from a large parallel corpus for the language pair [3]. The opposite extreme of rule-based machine translation is *statistical machine translation*. In its purest form it uses no additional linguistic knowledge except for the corpus to train both a statistical translation model and target language model [4]. These models are then used to assign probabilities to translation candidates and to choose the one with the maximum score. For several years the translation model was only built at the word level. As the limitations of this word-based translation approach became apparent, in particular for dissimilar language pairs like Japanese-English, several extensions towards phrase-based translation [5] and syntax-based translation [6] have been developed in the last few years. Although some improvements in the translation quality could be achieved, statistical machine translation suffers from the same ailment as rule-based translation, i.e. an incremental adaptation of the statistical model by the user is impossible. Finally, in contrast to rule-based translation, statistical translation has no easily comprehensible translation knowledge, which makes it unsuitable for language students who want to have an explanation component to get an understanding of how the translation process really works.

Although there exists no clear definition of *example-based machine translation* [7], it can be said that it lies between the two extremes of rule-based and statistical translation by using a parallel corpus to create a database of translation examples for source language fragments. The different approaches may vary in how they represent these fragments in the database: as surface strings, structured representations, generalized templates with variables, etc. [8]. 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 process can only automatically discover translation examples, however, manual crafting or at least reviewing of the fragments in the database is necessary to achieve sufficient coverage and accuracy for a corpus of reasonable size [9]. On the other hand, the representation of the translation knowledge in the database is less readily convertible to a lucid explanation of the translation process.

In our research, we have developed a Japanese-English rule-based machine translation system, however, we learn all translation rules automatically by using structural matching between the parse trees of translation examples. As training data we use the JENAAD corpus [10], which contains 150,000 Japanese-English sentence pairs from news articles. The current research work is based on a previous project on Japanese-German translation [11].

The system 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 translation rules, and full Unicode support for Japanese characters. Finally, the Amzi! Prolog Logic Server comes with several APIs, in particular the Amzi! Prolog Logic Server Visual Basic Module, which we used to develop a comfortable MS Word user interface. The resulting system is called JETCAT (Japanese-English Translation using Corpus-based Acquisition of Translation rules) and allows the user to invoke the translation functionality directly from MS Word. The users can customize their personal translation rule bases by simply post-editing translation results in the editor window. In addition, it is possible to inspect lexical, syntactic, and translation knowledge, including a single step trace mode for the application of translation rules, which makes JETCAT a very useful tool for language students.

The rest of the paper is organized as follows. In Sect. II we first give an overview of the system architecture, before we describe our formalism for the representation of translation knowledge in more detail in Sect. III. Finally, Sect. IV presents the features included in the implementation of the JETCAT user interface.

II. System Architecture

The JETCAT system architecture is outlined in Fig. 2. 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 three main tasks of the machine translation system are the translation of Japanese input, the acquisition of new rules, and the consolidation of the translation knowledge.

For the *translation* of a Japanese sentence, we first analyze it with the *tagging* module, which uses the morphological analysis system ChaSen [12] to produce the correct segmentation into a list of word tokens annotated with part-of-speech tags. Next, the token list is transformed into a parse tree by the *parsing* module, whose grammar rules are written in Definite Clause Grammar syntax. The parse tree of a sentence is a list of *constituents*, which are modeled as compound terms of arity 1 with the *constituent category* as principal functor. The *argument* of a constituent can be a syntactic feature

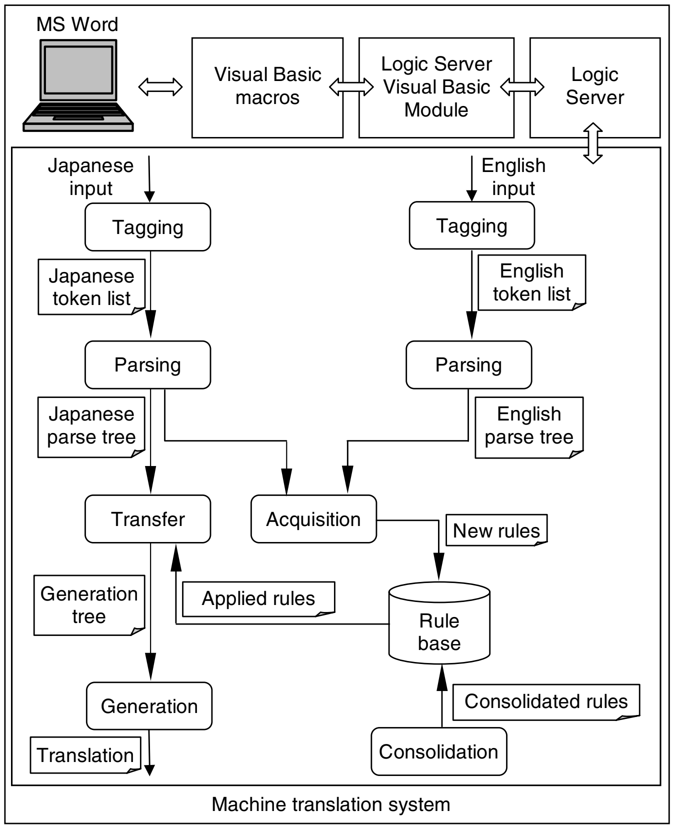


Fig. 2. System Architecture

(atom) or a word with its part-of-speech tag (atom/atom) for a *simple constituent*; it is a phrase (list of subconstituents) for a *complex constituent*. The *transfer* module traverses the Japanese parse tree top-down and applies the translation rules in the rule base to produce an equivalent English generation tree. As last processing step, 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.

The tagging and parsing of English sentences is a necessary preprocessing step for the *acquisition* of new translation rules. The *tagging* module is realized with the help of MontyTagger [13], the *parsing* module uses again the Definite Clause Grammar preprocessor of Amzi! Prolog. The *acquisition* module traverses the Japanese and English parse tree and derives new translation rules. The 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 corresponding English subconstituents to derive a translation rule. If a rule derivation was successful, then all Japanese and English subconstituents that are covered by the rule are removed from the input to continue the search. The default mapping of a Japanese subconstituent is to find an English subconstituent with identical category and to continue the matching procedure recursively for the two arguments. Each derived rule is added to the rule base if it is not included yet.

The rules that are learnt by the acquisition procedure are rather specific because they consider context-dependent translation dependencies in full detail to produce accurate translations and to avoid conflicts with other translation rules in the rule base. However, this high specificity badly affects the coverage for new unseen data. Therefore, the *consolidation* module tries to generalize translation rules as long as this rule pruning does not affect the consistency of the rule base.

III. TRANSLATION KNOWLEDGE

In our approach, we have chosen a very flexible and robust formalism to represent the translation knowledge. We model all translation situations with three generic rule types: word, constituent, and phrase translation rules. The translation rules are actually stored as Prolog facts in the rule base. Figure 3 shows an example of the rules learnt from a Japanese-English sentence pair (under the presumption of an empty rule base). Rules 1-7 are derived in that order by the acquisition module. We also indicate the generalizing transformations produced by the consolidation module, resulting in 9 simplified rules.

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

A. Word Translation Rules

Whenever the acquisition procedure reaches two simple constituents with identical category, a new word translation rule (predicate `wtr`) is derived to model a simple context-insensitive translation at the word level. Such a rule changes a Japanese word and part-of-speech tag to the equivalent English word and part-of-speech tag. For example, Rule 1 in Fig. 3 states that the attributive particle `NO/pat` is translated as the preposition `of/in`.

Many word translation rules are generated by the consolidation module as a result of the generalization of phrase translation rules. For example, in Fig. 3, Rule 5a translates the particle `KARA/par` as the preposition `in/in`, Rule 7b the verb `HAJIMERU/ver` as the verb `begin/vb`, and Rule 2a the sahen noun `TOUGOU/nsa` as the noun `integration/nn` (a sahen noun can be used as a verb, e.g. in this case “to integrate”, by adding the verb `SURU`, “to do”).

The transfer module tries to apply a word translation rule, once it reaches the argument of a simple constituent while traversing the parse tree. If the argument of the simple constituent equals the first argument of a word translation rule, it is substituted with the second argument of the rule.

B. Constituent Translation Rules

Constituent translation rules (predicate `ctr`) 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.

For example, in Fig. 3, Rule 6 translates the modifying noun (`mno`) `KEIZAI` as modifying adjective phrase (`maj`) `economic`, and Rule 3 the modifying noun phrase (`mnp`) `JISSAI NO` as modifying adjective phrase `practical` (`JISSAI` is an adverb (`adv`) that can also be used as a noun).

The third argument of a constituent translation rule is used as an index by the transfer module to speed up the access to rule candidates and to reduce the number of rules that have to be examined.

Constituent translation 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 untouched. For example, the following rule changes the modifying noun phrase `X KOTO NO` into the modifying adpositional phrase with gerund (category `mag`) `of X`:

```
ctr(mnp, mag, KOTO/nde,
   [apo(NO/pat), hea(KOTO/nde),
      mcl([vbl([hea(H), hef(_), sac(S)])|R])],
   [apo(of/in),
      vbl([hea(H), sac(S), asp(prg)])|R]).
```

In more detail, the rule states that the argument of the modifying noun phrase must contain `NO/pat` as adposition (`apo`), the dependent noun `KOTO/nde` as head (`hea`), and a modifying clause (`mcl`). The modifying clause must include a verbal (`vbl`) with head, head form (`hef`, encodes the conjugation type and conjugation form produced by ChaSen), and a sahen compound (`sac`, a sahen noun to derive a verb).

This input is replaced by a modifying adpositional phrase with gerund comprising the adposition `of/in` and a verbal, which contains the Japanese head and sahen compound as well as the aspect (`asp`) progressive (`prg`) to model the gerund. As the Japanese head form is not listed here, it is deleted from the input. All other subconstituents `R` of the modifying clause are appended unchanged.

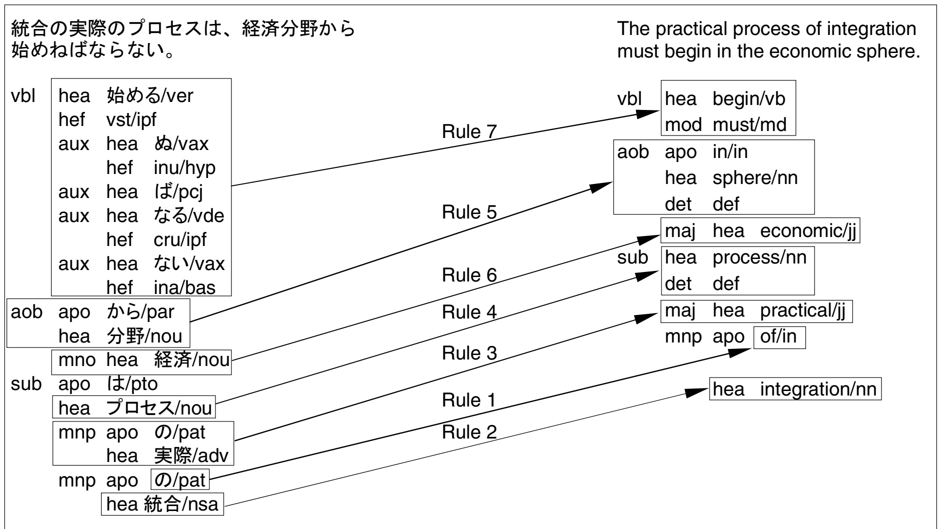
For example, the Japanese input `Y KAIFUKU SURU KOTO NO` (“of restoring `Y`”), i.e. head `SURU/ver`, head form `sur/bas` (base form of irregular verb `SURU`), sahen compound `KAIFUKU/nsa`, and a direct object (`dob`) `Y` (to shorten the example):

```
mnp([apo(NO/pat), hea(KOTO/nde),
   mcl([vbl([hea(SURU/ver), hef(sur/bas),
   sac(KAIFUKU/nsa)]), dob(Y)])])
```

is translated by the above rule as:

```
mag([apo(of/in),
   vbl([hea(SURU/ver), sac(KAIFUKU/nsa),
   asp(prg)]), dob(Y)])
```

If the transfer module arrives at a complex constituent during the traversal of the parse tree, it first tries to apply a constituent translation rule before it continues its search for the argument of the complex constituent. To find suitable rule candidates the transfer module first checks if a rule has the correct category as first argument and the correct head as third argument. If these two conditions are satisfied, it tries to unify the fourth argument with the argument of the complex constituent. If the unification is successful, the second and



1. wtr(の/pat, of/in).
2. ptr(np, 統合/nsa, [hea(統合/nsa)], [hea(integration/nn)]) .
   ⇒ 2a. wtr(統合/nsa, integration/nn).
3. ctr(mnp, maj, 実際/adv, [apo(の/pat), hea(実際/adv)], [hea(practical/jj)]).
4. ptr(np, プロセス/nou, [hea(プロセス/nou)], [hea(process/nn), det(def)]).
5. ptr(cl, 始める/ver, [aob([apo(から/par), hea(分野/nou) | X])],
   [aob([apo(in/in), hea(sphere/nn), det(def) | X])]).
   ⇒ 5a. wtr(から/par, in/in).
   ⇒ 5b. ptr(np, 分野/nou, [hea(分野/nou)], [hea(sphere/nn), det(def)]).
6. ctr(mno, maj, 経済/nou, [hea(経済/nou)], [hea(economic/jj)]).
7. ptr(vbl, 始める/ver, [hea(始める/ver), hef(vst/ipf), aux([hea(ぬ/vax), hef(inu/hyp)]),
   aux([hea(ば/pcj)]), aux([hea(なる/vde), hef(cru/ipf)]), aux([hea(ない/vax), hef(ina/bas)])],
   [hea(begin/vb), mod(must/md)]).
   ⇒ 7a. ptr(vbl, nil, [hef(vst/ipf), aux([hea(ぬ/vax), hef(inu/hyp)]), aux([hea(ば/pcj)]),
   aux([hea(なる/vde), hef(cru/ipf)]), aux([hea(ない/vax), hef(ina/bas)])], [mod(must/md)]).
   ⇒ 7b. wtr(始める/ver, begin/vb).

Fig. 3. Example of Translation Rules

fifth argument are used to build the English equivalent of the complex constituent by binding any shared variables as shown in the previous example.

C. Phrase Translation Rules

The most expressive rule type are phrase translation rules (predicate `ptr`), they allow to define elaborate conditions and substitutions on phrases, i.e. on arguments of complex constituents. Phrase translation 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 consider contextual translation dependencies.

Phrase translation rules translate the argument a_I of a complex constituent with category c_I in the Japanese input; they have four arguments:

1) *category* c_R: must subsume c_I, i.e. $c_\mathrm{I} \sqsubseteq c_\mathrm{R}$,
2) *head* h_R: a_I must contain h_R as head element,
3) *argument* a_R: contains the required elements in a_I,
4) *translation* t_R: translation of a_R.

One important requirement for the efficient and robust application of phrase translation rules by the transfer module is that the condition expressed by a_R has to be understood as subset condition, i.e. $a_\mathrm{R} \subseteq a_\mathrm{I}$. All additional subconstituents $a_\mathrm{I} \setminus a_\mathrm{R}$ are appended to t_R unchanged. That way one phrase translation rule may change only certain elements of a phrase whereas all other elements are translated later on by other rules.

For example, in Fig. 3, Rule 2 states that if a noun phrase contains the sahen noun `TOUGOU/nsa` as head, it is replaced by the noun `integration/nn`. In the same way, Rule 4 substitutes the noun `PUROSESU/nou` with the noun `process/nn` and a definite determiner (syntactic feature `det(def)`). Rule 7 translates the verbal of the Japanese sentence, which uses a rather complex sequence of auxiliaries to express obligation. a_R includes the imperfective form `(ipf)` of the vowel-stem verb `(vst)` `HAJIMERU/ver` as head and the following auxiliaries `(aux)`: the hypothetical form `(hyp)` of the irregular auxiliary verb `NU/vax` (conjugation type `inu`), the conjunctive particle `BA/pcj`, the imperfective form of the consonant-stem dependent verb `NARU/vde` (conjugation type `cru`), and, finally, the base form of the irregular auxiliary verb `NAI/vax` (conjugation type `ina`). This expression is translated as `must begin`, i.e. head verb `begin/vb` and modal `(mod)` `must/md`. The consolidation module splits Rule 7 into two separate more general rules, Rule 7a translates the modal and Rule 7b the head.

If the applicability of a phrase translation rule does not depend on the head of the phrase (as in Rule 7a), then the special constant `nil` can be used as second argument h_R. Finally, the special constant `notex` can be used for h_R to indicate that a_I must not contain a head element. In a similar way, `notex` can be used as argument of a subconstituent in a_R, e.g. `sub(notex)`, with the meaning that a_I must not include a subconstituent of that category, e.g. no subject.

Rule 5 is an example of a phrase translation rule with a shared variable for unification. For a clause (`cl`) with head `HAJIMERU/ver` (for clauses, h_R is tested on the head of the verbal), the adpositional object (`aob`) with adposition `KARA/par` and head noun `BUNYA/nou` is translated as an adpositional object with adposition `in/in`, head noun `sphere/nn`, and a definite determiner. All other subconstituents are left untouched. As mentioned, the order of the subconstituents is of no importance for the unifiability of a_R. For example, for the sentence in Fig. 3, the input a_I could be (using variables `V` and `S` to shorten the example):

```
[vbl(V), aob([hea(BUNYA/nou), mno([hea(KEIZAI/nou)]),
    apo(KARA/par)]), sub(S)]
```

The resulting translation would be:

```
[aob([apo(in/in), hea(sphere/nn), det(def),
    mno([hea(KEIZAI/nou)])]), vbl(V), sub(S)]
```

Again, the consolidation module splits the phrase translation rule in two more general rules, Rule 5a and Rule 5b, which separately translate the adposition and the head noun.

The transfer module starts at the top level of the Japanese parse tree and tries to apply phrase translation rules. To apply a phrase translation rule, 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. In addition, rules are assigned a higher score if $h_\mathrm{R} \neq$ `nil`, a_R does not contain the head of the phrase, or if `notex` is used in a_R.

The verification of the condition in a_R is a quite complex task because it involves testing for set inclusion ($a_\mathrm{R} \subseteq a_\mathrm{I}$) at the top level as well as recursively testing for set equality 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_R through unification, and returning the remaining constituents from the input as list of additional elements $a_\mathrm{I} \setminus a_\mathrm{R}$ to be appended later on to t_R. A constituent from a_R can be removed from the input if the two constituents can be directly unified or if their categories are identical and their arguments a_CR and a_CI are unifiable sets. The latter condition is tested by again removing each subconstituent in a_CR from a_CI until either a free variable as tail of a_CR (i.e. `|X]`) or the end of both lists is reached. In addition, any `notex` condition in a_R 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 separately. We first search for constituent translation rules before we perform a transfer of the argument. The latter involves the application of word translation rules for simple constituents, whereas the top-level procedure is repeated recursively for complex constituents including some cleanup actions, e.g. removing the topic particle `WA/pto` from the subject (`sub`) in Fig. 3.

IV. User Interface

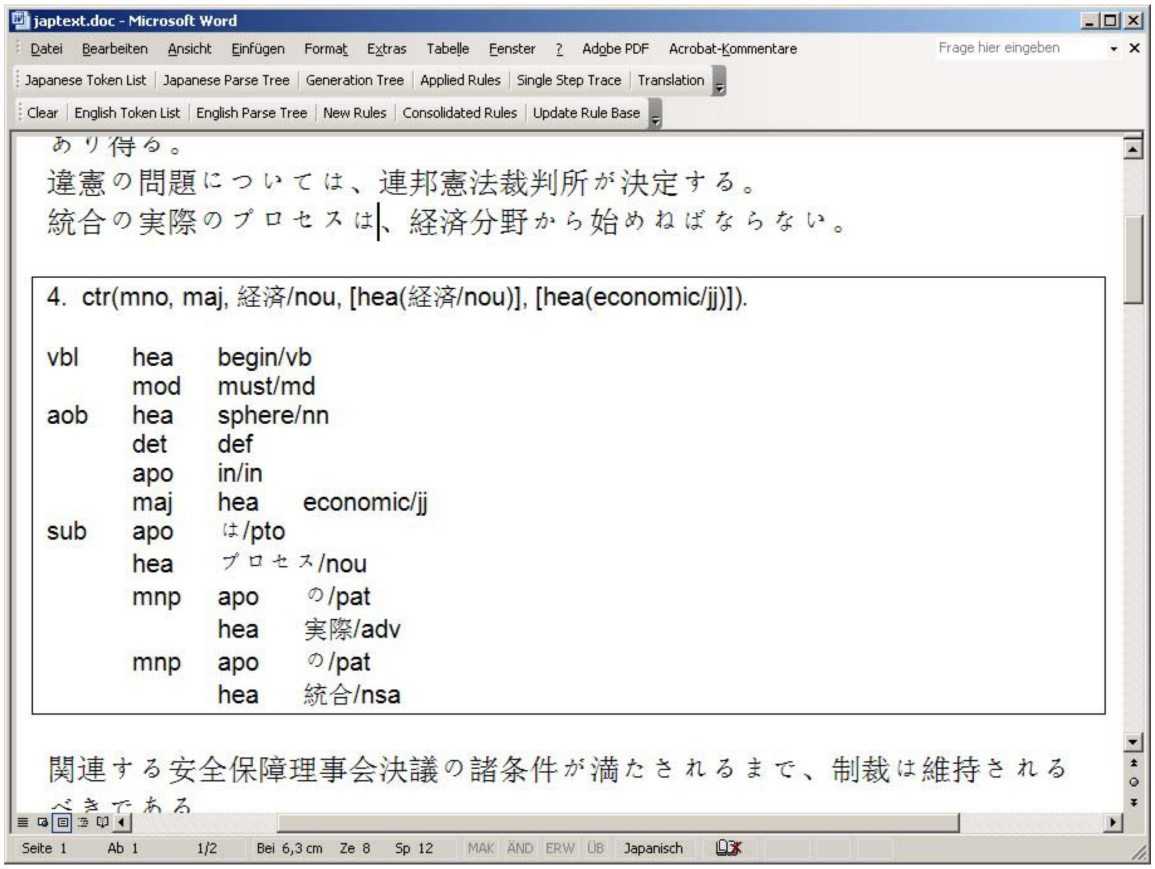
We have developed a user-friendly MS Word interface so that the translation functionality is directly accessible from any editor window. Figure 4 shows a screenshot of the user interface; all tasks can be invoked via the two toolbars. The commands in the first toolbar concern Japanese sentences. The user can click anywhere in a Japanese document and select a command. This results in the extraction of the sentence around the cursor position, the execution of the task by JETCAT, and the insertion of the formatted result with borders after the Japanese sentence. The user can translate a Japanese sentence by clicking on "Translation" and inspect all the intermediate results produced during a translation, i.e. the Japanese token list, the Japanese parse tree, and the generation tree. In addition, the user can click on "Applied Rules" to receive an enumerated list of the rules used by the transfer module in the correct order of their application.

As a special feature we provide a single step trace mode, i.e. starting from the original Japanese parse tree the user can see how the tree gradually changes into the generation tree by the application of individual rules. If the user clicks on "Single Step Trace", the first translation rule applied by the transfer module is displayed together with its effects on the parse tree. By clicking repeatedly on "Single Step Trace", the user can follow the progress of the transfer module and get a better understanding of the translation process. For example, Fig. 4 is the output after clicking on "Single Step Trace" for the fourth time, i.e. after applying Rule 6 from Fig. 3 to the example sentence.

With "Clear", the user can 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 have a look at the English token list or the English parse tree, which are computed for the sentence surrounding the current cursor position.

For a better understanding of the acquisition task, the user can click on "New Rules" for a Japanese-English sentence pair. JETCAT pretends to have an empty rule base and returns a list of rules that would be learnt from this translation example in the correct order of their derivation. In addition, by using "Consolidated Rules" the user can scrutinize the generalizing transformations that would be performed for this sentence pair.

Finally, one essential functionality of JETCAT is the ability to customize translation results by simply correcting them in the editor window and updating the rule base via "Update Rule Base". Before this, the user can check 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 in the future.


japtext.doc - Microsoft Word
Datei Bearbeiten Ansicht Einfügen Format Extras Tabelle Fenster ? Adobe PDF Acrobat-Kommentare
Frage hier eingeben
Japanese Token List | Japanese Parse Tree | Generation Tree | Applied Rules | Single Step Trace | Translation
Clear | English Token List | English Parse Tree | New Rules | Consolidated Rules | Update Rule Base
Seite 1 Ab 1 1/2 Bei 6,3 cm Ze 8 Sp 12 MAK ÄND ERW ÜB Japanisch

あり得る。
違憲の問題については、連邦憲法裁判所が決定する。
統合の実際のプロセスは、経済分野から始めねばならない。

```
4.  ctr(mno, maj, 経済/nou, [hea(経済/nou)], [hea(economic/jj)]).

vbl    hea    begin/vb
       mod    must/md
aob    hea    sphere/nn
       det    def
       apo    in/in
       maj    hea    economic/jj
sub    apo    は/pto
       hea    プロセス/nou
       mnp    apo    の/pat
              hea    実際/adv
       mnp    apo    の/pat
              hea    統合/nsa
```

関連する安全保障理事会決議の諸条件が満たされるまで、制裁は維持される
べきである

Fig. 4. Screenshot of User Interface

V. Conclusion

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

Future work will focus on extending the coverage of the system so that we can process the complete JENAAD corpus and perform a thorough evaluation of the translation quality using tenfold crossvalidation. We intend to let students of Japanese studies at our university use JETCAT to receive valuable feedback from practical use. In addition, we are currently working on a Web interface to JETCAT and are planning to make a demo version publicly available in the near future.

Acknowledgment

We are deeply grateful to Amzi! inc. for supporting our research with a complimentary Professional Edition of Amzi! Prolog + Logic Server.

References

[1] H. Somers, Ed., *Computers and Translation: A Translator's Guide*. Amsterdam: John Benjamins, 2003.
[2] 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. Valladolid: Univ. Valladolid, 2004, pp. 13–48.
[3] 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. Saarbrücken: IAI, 1999, vol. 36.
[4] P. Brown, "A statistical approach to machine translation," *Computational Linguistics*, vol. 16, no. 2, pp. 79–85, 1990.
[5] P. Koehn, F. J. Och, and D. Marcu, "Statistical phrase-based translation," in *Proc. Human Language Technology Conf. of the North American Chapter of the ACL*, Edmonton, Canada, 2003, pp. 48–54.
[6] K. Yamada, "A syntax-based statistical translation model," Ph.D. dissertation, Kyoto University, 1999.
[7] J. Hutchins, "Towards a definition of example-based machine translation," in *Proc. Second Workshop on Example-Based Machine Translation at MT Summit X*, Phuket, Thailand, 2005, pp. 63–70.
[8] M. Carl and A. Way, Eds., *Recent Advances in Example-Based Machine Translation*. Dordrecht: Kluwer, 2003.
[9] S. Richardson *et al.*, "Overcoming the customization bottleneck using example-based MT," in *Proc. ACL Workshop on Data-Driven Machine Translation*, Toulouse, France, 2001, pp. 9–16.
[10] M. Utiyama and H. Isahara, "Reliable measures for aligning Japanese-English news articles and sentences," in *Proc. 41st Annual Meeting of the ACL*, Sapporo, Japan, 2003, pp. 72–79.
[11] W. Winiwarter, "Incremental learning of transfer rules for customized machine translation," in *Applications of Declarative Programming and Knowledge Management*, ser. LNAI, U. Seipel *et al.*, Eds. Berlin: Springer-Verlag, 2005, vol. 3392, pp. 47–64.
[12] Y. Matsumoto *et al.*, "Japanese morphological analysis system ChaSen version 2.0 manual," NAIST, Tech. Rep. NAIST-IS-TR99009, 1999.
[13] H. Liu, "MontyLingua: An end-to-end natural language processor with common sense," MIT Media Lab, Tech. Rep., 2004.