Computational Linguistics and Chinese Language Processing
Vol. 1, No. 1, August 1996, pp. 147-170

# A Hybrid Approach to Machine Translation System Design

**Kuang-Hua Chen*, and Hsin-Hsi Chen*+**

## Abstract

It is difficult for pure statistics-based machine translation systems to process long sentences. In addition, the domain dependent problem is a key issue under such a framework. Pure rule-based machine translation systems have many human costs in formulating rules and introduce inconsistencies when the number of rules increases. Integration of these two approaches reduces the difficulties associated with both. In this paper, an integrated model for machine translation system is proposed. A partial parsing method is adopted, and the translation process is performed chunk by chunk. In the synthesis module, the word order is locally rearranged within chunks via the Markov model. Since the length of a chunk is much shorter than that of a sentence, the disadvantage of the Markov model in dealing with long distance phenomena is greatly reduced. Structural transfer is fulfilled using a set of rules; in contrast, lexical transfer is resolved using bilingual constraints. Qualitative and quantitative knowledge is employed interleavingly and cooperatively, so that the advantages of these two approaches can be retained.

**Keywords:** Bigram Language Model, Lexical Selection, Machine Translation System, Probabilistic Chunker, Predicate-Argument Structure, X'-Theory.

## 1. Introduction

Many different approaches to machine translation design have been proposed [Bennett and Slocum 1985; Brown *et al.* 1992; Nagao 1984; Mitamura *et al.* 1991; Baker *et al.*1994].The traditional rule-based machine translation system [Bennett and Slocum 1985] is expensive in terms of formulating rules. It easily introduces inconsistencies, and it is too rigid to be robust. However, rules are usually universal; i.e., they are not domain dependent. In contrast, the statistics-based machine translation system [Brown *et al.* 1992] is based on noise channel model and is robust in processing partial and ill-formed sentences. The difficulty underlying

* Dept. of Computer Science & Information Engineering, National Taiwan University, Taipei, Taiwan.
E-mail: khchen@nlg.csie.ntu.edu.tw ; hh_chen@csie.ntu.edu.tw

+ To whom all the correspondence should be sent.

such systems is the register diversified problem. In addition, the computation time in processing long sentences sharply increases as the number of words increases. The example-based system [Nagao 1984] heavily depends on the quality of collected examples and the similarity measures between examples and input sentences. When the matched units are subsentential structures (e.g., phrase structures), the performance of such a system is better than that of a word-level system. As for the knowledge-based system [Mitamura *et al.* 1991; Baker *et al.* 1994], the difficulties are in how the knowledge is represented, how fine the knowledge is, and what the inference engine is. In addition, the cost of compiling knowledge is expensive. Table 1 summarizes the advantages and disadvantages of these approaches. From the viewpoint of system designers, if we could integrate the advantages of these approaches and get rid of their disadvantages, a hybrid system could perform better than any of the systems. For example, Brown *et al.* [1992] included a small amount of linguistic knowledge (parts of speech and morphological analysis) in their original statistical machine translation system. However, strictly speaking, the modified MT system is not a hybrid system. The knowledge is used to guide the search mechanism and to avoid unnecessary searching.

The structure of this paper is as follows. Section 2 gives an overview of the proposed MT model. Sections 3, 4 and 5 discuss the analysis module, transfer module and synthesis module, respectively. Section 6 presents some experiments which demonstrate the feasibility of this MT model. Section 7 provides some concluding remarks.

***Table 1. Summary of Different Approaches to Machine Translation System***

| | Advantages | Disadvantages |
|---|---|---|
| Rule-Based | 1. easy to build an initial system<br>2. based on linguistic theories<br>3. effective for core phenomena | 1. rules are formulated by experts<br>2. difficult to maintain and extend<br>3. ineffective for marginal phenomena |
| Knowledge-Based | 1. based on taxonomy of knowledge<br>2. contains an inference engine<br>3. interlingual representation | 1. hard to build knowledge hierarchy<br>2. hard to define granularity of knowledge<br>3. hard to represent knowledge |
| Example-Based | 1. extracts knowledge from corpus<br>2. based on translation patterns in corpus<br>3. reduces the human cost | 1. similarity measure is sensitive to system<br>2. search cost is expensive<br>3. knowledge acquisition is still problematic |
| Statistics-Based | 1. numerical knowledge<br>2. extracts knowledge from corpus<br>3. reduces the human cost<br>4. model is mathematically grounded | 1. no linguistic background<br>2. search cost is expensive<br>3. hard to capture long distance phenomena |

## 2. Overview of Our MT Model

In statistics-based MT systems, researchers apply the noisy channel model to compute the probability of a translation. In reality, it is also easy to make use of probability under the traditional transfer-based MT framework. The advantages of the probabilistic transfer-based MT model over noisy channel model are:

- It consists of much finer modules than does the noisy channel model.
- Utilization of linguistic knowledge in the model is more natural than that in noisy channel model.
- It agrees human intuition much more than does the noisy channel model.

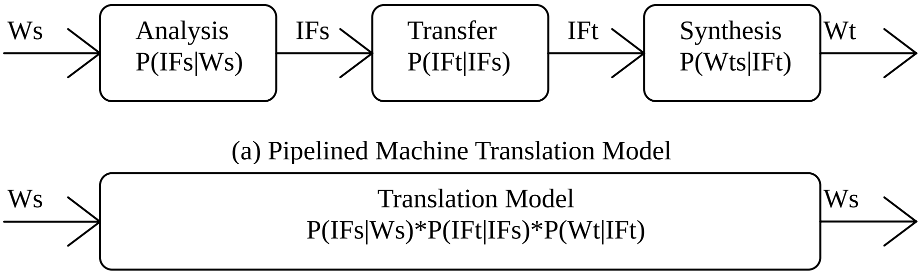
Such a machine translation system is assumed to consist of analysis, transfer and synthesis modules. When it receives an *n*-word source sentence $W_s$, this MT system generates an *l*-word target sentence $W_t$. Let the intermediate forms for the source part and target part be $IF_s$ and $IF_t$, respectively. Equation 1 describes the general model of transfer-based MT systems [Chang and Su 1993]. This equation specifies that each representation is dependent on all its pervious representation(s) based on the traditional three-stage translation model:

$$(1) \quad P(W_t \mid W_s) = \sum_{IF_s, IF_t} P(IF_s \mid W_s) \times P(IF_t \mid IF_s, W_s) \times P(W_t \mid IF_t, IF_s, W_s) \,.$$

However, the number of parameters involved in this model is inevitably too large to put it into practice. Some reduction should be asserted. Assume that each representation only depends on its immediate previous representation. Equation 1 is reformulated as Equation 2:

$$(2) \quad P(W_t \mid W_s) \cong \sum_{IF_s, IF_t} P(IF_s \mid W_s) \times P(IF_t \mid IF_s) \times P(W_t \mid IF_t)$$

In fact, the complexity introduced by searching for possible combinations of these representations is also too high to endure. Further simplification in searching should be made. The simplification is that the best representation chosen in the previous stage is kept as the proper one. We call such a simplified machine translation model a pipelined machine translation model. On the other hand, the original one is called a conglomerated machine translation model. Figure 1 conceptualizes the two different translation models.



(a) Pipelined Machine Translation Model

(b) Conglomerated Machine Translation Model

***Figure 1. Pipelined Model vs. Conglomerated Model***

In using our translation model, one point should be noted. Although many efforts have been devoted to understanding texts, success has not been achieved in practical applications. For example, a knowledge-based system needs much hand-coded knowledge and a complex control component. However, it is not easy to extend application to the other domains. Thus, if a complex mechanism does not underpin good systems, why don't we apply a much simpler mechanism? The proposed MT system is used to demonstrate the feasibility of the following ideas.

- Do not fully parse the texts.

  It is difficult to resolve the attachment problem in the parsing stage.

- Do not devise many rules.

  The great part of the knowledge is extracted from the corpus and expressed in numeric form. Some symbolic rules covering core phenomena are applied for effectiveness.

- Quantitative parameters and qualitative rules are used cooperatively.

  Quantitative parameters are used to get a shallow analysis for sentences, and qualitative rules are used to determine their predicate-argument structures.

- Increase the independence of participating language pairs.

  The synthesis module is based on the target language model.

Our overall translation model is shown in Figure 2. The analysis module is composed of a tagger, a chunker and a detector for predicate-argument structures. The transfer module consists of a lexical selection component and a simple transfer component. The knowledge needed in the module is in a mutual information table, a simple bilingual dictionary, and the mapping rules of predicate-argument structures between language pairs. The third module is the synthesis module. It consists of a generator and a bi-gram table. All the knowledge (whether rules or training tables) will be discussed in the appropriate sections.

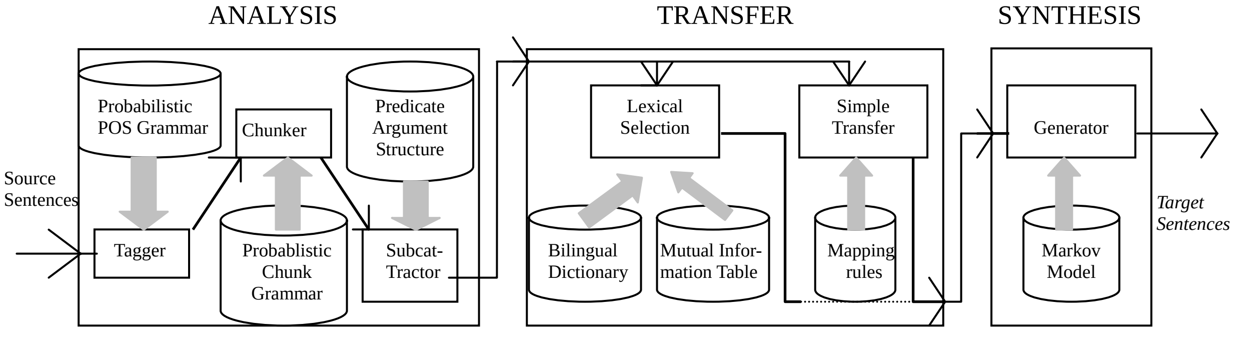


***Figure 2. Pipelined Translation Model***

The whole translation process is briefly introduced in the following. As Figure 1(a) shows, the analysis module is used to find $IF_s$ which has the largest $P(IF_s|W_s)$. IFs consists of a chunk structure ($C_s$) and a predicate-argument structure ($PA_s$). That is,

$$\begin{aligned} P(IF_s\,|\,W_s) &= P(C_s,PA_s\,|\,W_s) = P(C_s\,|\,W_s)\times P(PA_s\,|\,C_s,W_s) \\ &= \sum_{T_s}(P(C_s\,|\,T_s,W_s)\times P(T_s\,|\,W_s))\times P(PA_s\,|\,C_s,W_s)\ . \\ &\cong \sum_{T_s}(P(C_s\,|\,T_s)\times P(T_s\,|\,W_s))\times P(PA_s\,|\,C_s,W_s) \end{aligned} \tag{3}$$

The chunk structure is a linear chunk sequence. In contrast, the predicate-argument structure provides the domination relation. The $P(T_s|W_s)$ part is a probabilistic tagger; $P$ $(C_s|T_s)$ is a probabilistic chunker; $P(PA_s|C_s,W_s)$ is implemented using 23 rules rather than a probabilistic model. Source sentences are first input to a probabilistic tagger, and then a sequence of tags is sent to the chunker. The tagger is trained from the LOB Corpus [Johansson 1986] and has a 95%correct rate. The chunker [Chen and Chen 1993] is used to determine the plausible boundaries of phrasal structures and then to segment the input tags into a series of chunks. The predicate-argument structures are determined by using 23 pre-defined verb patterns. The SUBCAT-TRACTOR is applied to detect the predicate-argument structures [Chen and Chen 1994a].

The transfer module consists of two components shown in the following: lexicon selection (lexical transfer) and simple transfer (structural transfer):

$$\begin{aligned} P(IF_t\,|\,IF_s) &= P(C_t,PA_t\,|\,C_s,PA_s) \\ &= P(C_t\,|\,PA_t,C_s,PA_s)\times P(PA_t\,|\,C_s,PA_s)\ . \\ &\cong P(C_t\,|\,C_s)\times P(PA_t\,|\,PA_s) \end{aligned} \tag{4}$$

The simple transfer mechanism focuses on the predicates and only maps the predicates and arguments of source sentences into the respective constituents of target sentences ($PA_s \rightarrow PA_t$). These predicate-argument structures are regarded as the skeletons of sentences. The simple transfer mechanism transfers the source skeleton to the target skeleton. The remaining "flesh" is adjusted in the synthesis module. Therefore, some overheads are reduced in the traditional transfer stage under our framework. The transfer module looks for the target mapping of the detected structure. This is resolved by a set of transfer rules. For example, (i) shows a mapping of predicate-argument structures between English and Chinese:

(i) Kevin gave John a passkey. [arg0] give [arg1] [arg2]
凱文 把 萬能鑰匙 給了 約翰。 [arg0] 把 [arg2] 給了 [arg1]
Kevin Ba passkey give John .

The transfer module is also responsible for lexicon selection ($C_s \rightarrow C_t$). The lexicon selection algorithm is presented on the basis of source and target word associations [Church and Hanks 1990]. Word association of source and target languages can be trained independently from the corresponding text corpora. The bilingual dictionary sets up the word correspondence. The proposed method chooses the most informative mates in the source language as well as in the target language. On the one hand, bilingual corpora are not needed in our approach; thus, the difficulty of collecting a large number of bilingual texts for reliable probabilities is avoided. On the other hand, the computation of our method is not complex.

Language generation is the task of the synthesis module. The word order of the target sentence is determined by global reordering and local reordering. The reordering tasks are performed by the simple transfer rules R and synthesis module $P(W_t|C_t)$ shown in Figure 3, respectively.

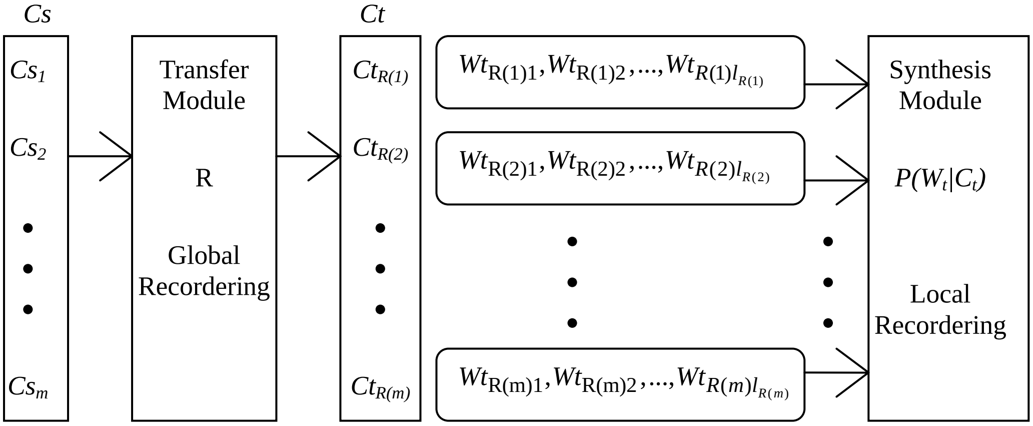

Cs
Ct
$Cs_1$
$Cs_2$
$Cs_m$
Transfer Module
R
Global Recordering
$Ct_{R(1)}$
$Ct_{R(2)}$
$Ct_{R(m)}$
$Wt_{R(1)1}, Wt_{R(1)2}, ..., Wt_{R(1)l_{R(1)}}$
$Wt_{R(2)1}, Wt_{R(2)2}, ..., Wt_{R(2)l_{R(2)}}$
$Wt_{R(m)1}, Wt_{R(m)2}, ..., Wt_{R(m)l_{R(m)}}$
Synthesis Module
$P(W_t|C_t)$
Local Recordering

***Figure 3. Global Reordering and Local Reordering***

Therefore, the model for the synthesis module $P(Wt|IFt)$ can be rewritten as:

$$
\begin{aligned}
P(W_t \mid IF_t) &= P(W_t \mid C_t, PA_t) \cong P(W_t \mid C_t, R) \cong P(W_t \mid Ct_{R(1)}^{R(m)}) \\
&\cong \sum_{i=1}^{m} P(Wt_{R(i)} \mid Ct_{R(i)}) = \sum_{i=1}^{m} P(Wt_{R(i)1}{}^{l_{R(i)}} \mid Ct_{R(i)}) \quad . \qquad (5) \\
&\cong \sum_{i=1}^{m} P(Wt_{R(i)1}) \prod_{j=1}^{l_{R(i)}-1} P(Wt_{R(i)j+1} \mid Wt_{R(i)j})
\end{aligned}
$$

where $l_{R(i)}$ denotes the number of words in target chunk $c_{R(i)}$. The corpus provides a large volume of lively language phenomena. The implicit word order can be trained from the corpus andmeasured in terms of probability. In addition, many marginal phenomena can be covered naturally. This is why the Markov model is used in the synthesis module.

## 3. Analysis Module

Two tasks are fulfilled in this module: one is determination of the chunk sequence, and the other is determination of the predicate-argument structures. The following section describes this module in detail.

### 3.1 Partial Parsing

Different intermediate forms result in different analysis methods. Here, we choose linear analysis for the input sentence. The main component of the analysis module is a partial parser. The partial parser is motivated by an intuition shown below.

(ii) When we read a sentence, we read it chunk by chunk.

Based on this intuition, we call the partial parser *a chunker*. The chunker receives tagged texts and outputs a linear chunk sequence. For example, the sentence in (ii) may be chunked into

[When we] [read a sentence], [we usually read it] [chunk by chunk].

The words between the left square bracket and the right square bracket form a chunk. When reading the sentence, we pause briefly between chunks. In fact, chunks can be regarded as the plausible phrase structure. Abney [1991] has discussed a rule-based chunker. The following will describe a probabilistic chunker. Given an *n*-word sentence, $ws_1$, $ws_2$, ..., $ws_n$ (including punctuation marks), the parsing task is to find the best chunk

sequence, $\hat{Cs}$, such that

(6) $$\hat{Cs} = \arg\max_{Cs_i} P(Cs_i \mid W_s) = \arg\max_{Cs_i} P(Cs_i \mid Ws_1^n) .$$

$Cs_i$ is a possible chunk sequence, $cs_1, cs_2, ..., cs_{m_i}$, where mi is the number of chunks in the chunk sequence. To chunk raw texts without using other information is very difficult since the word patterns are numbered in billions. Therefore, a tagger is applied to preprocess the raw texts, and each word is given a unique part of speech. That is, for an n-word sentence, $ws_1$, $ws_2$, ... , $ws_n$ (including punctuation marks), the parts of speech $ts_1$, $ts_2$, ..., $ts_n$ are assigned to the respective words. Now, the working model is:

$$(7)\quad \begin{aligned} \hat{C}_S &= \arg\max_{Cs_i} P(Cs_i \mid Ws_1^n) \\ &= \arg\max_{Cs_i} \sum_{Ts} P(Cs_i \mid Ts, W_s) \times P(Ts \mid Ws) \\ &= \arg\max_{Cs_i} \sum_{Ts} P(Cs_i \mid ts_1^n, W_{s1}^n) \times P(ts_1^n \mid Ws_1^n) \\ &\cong \arg\max_{Cs_i} P(Cs_i \mid ts_1^n) \times P(ts_1^n \mid Ws_1^n) \end{aligned} .$$

To resolve the optimization problem, various language models may be adopted. Here the bi-gram Markov model is applied. We then reduce $P(Cs_i \mid ts_1^n)$ as shown equation 8:

$$(8)\ P(Cs_i \mid ts_1^n) = P_i(cs_1^{mi} \mid ts_1^n) \cong \prod_{k=1}^{m_i} P_i(cs_k \mid cs_{k-1}, ts_1^n) \times P_i(cs_k \mid ts_1^n) \cong \prod_{k=1}^{m_i} P_i(cs_k \mid cs_{k-1}) \times P_i(cs_k)$$

where $P_i(cs_1^{mi} \mid ts_1^n)$ denotes the probability for the *i*'th chunk sequence. Note that an xtra sentence initial marker denoted by $c_0$ is added. Following equation 8, equation 7 is rewritten as equation 9:

$$(9)\quad \begin{aligned} \hat{C}s &= \arg\max_{Cs_i} P(Cs_i \mid ts_1^n) \cong \arg\max_{Cs_i} \prod_{k=1}^{m_i} P_i(cs_k \mid cs_{k-1}) \times P_i(cs_k) \\ &= \arg\max_{Cs_i} \sum_{k=1}^{m_i} [\log(P_i(cs_k \mid cs_{k-1})) + \log(P_i(cs_k))] \end{aligned} .$$

Dynamic programming shown in Algorithm 1 is used to find the best chunk sequence. *score*[*i*] denotes the score of position *i*. The words between position *pre*[*i*] and position *i* form the best chunk from the viewpoint of position *i*. *dscore*($c_i$) is the score of probability $P(c_i)$, and *cscore*($c_i|c_{i-1}$) is the score of probability $P(c_i \mid c_{i-1})$. These scores are collected from the training corpus, SUSANNE Corpus [Sampson 1995].

**Algorithm 1: Chunker**

Input: word sequence $w_1, w_2, \ldots, w_n$, and the corresponding POS sequence $t_1, t_2, \ldots, t_n$

Output: a sequence of chunks $c_1, c_2, \ldots, c_m$

```
Method:   (1) score[0] = 0;
                pre[0] = 0;
          (2) for (i = 1; i<n+1; i++) do 3 and 4;
          (3)  $j* = \arg\max_{0 \le j < i}(\text{score[pre[j]]} + \text{dscore}(c_j) + \text{cscore}(c_j|c_{j-1}))$;
                where $t_{j+1}, \ldots, t_i$; $c_{j-1} = t_{pre[j]+1}, \ldots, t_j$;
          (4) score[i]=score[pre[j*]]+dscore($c_{j*}$)+cscore($c_{j*}|c_{j*-1}$); pre[i] = j*;
          (5) for (i=n; i>0; i=pre[i]) do
              output the word $w_{pre[i]+1}, \ldots, w_i$ to form a chunk;
```

## 3.2 Determination of Predicate-Argument Structures

The analysis model not only produces the chunk sequence, but also determines the predicate-argument structure. $P(PA_s|C_s,W_s)$ is used to describe the functionality of this part, but at present it is implemented in rules. It is not difficult to determine these structures using chunk sequences and word information. As shown in the Figure 4, a finite state mechanism, SUBCAT-TRACTOR, selects one out of 23 predefined predicate- argument structures [Chen and Chen 1994a]. These structures are a modified version of definitions in the Oxford Advanced Learner's Dictionary (OALD) [Hornby 1989]. Table 2 summarizes these predicate-argument structures.

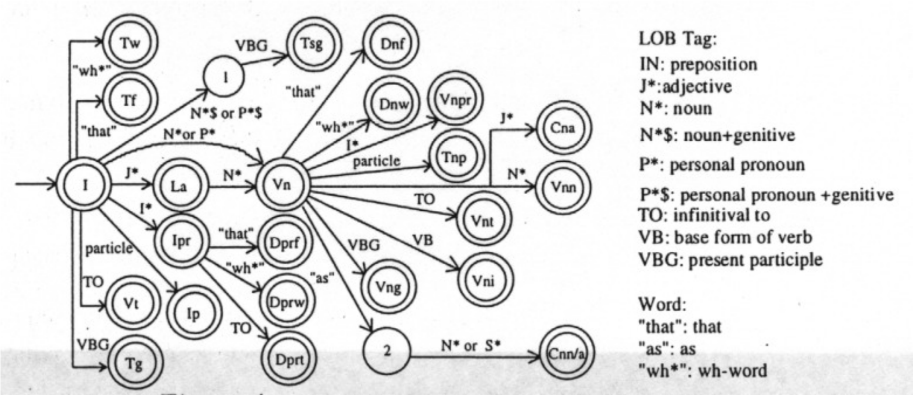


*Figure 4. Finite State Mechanism for Extraction of Predicate-Argument Structures*

*Table 2. Predicate-Argument Structures*

| Types | Predicate-Argument Structure | Types | Predicate-Argument Structure |
|---|---|---|---|
| Cna | Complex-trans. verb + noun + adj. | Tf | Transitive verb + finite that-clause |
| Cnn/a | Complex-trans. verb+noun+as+noun (adj.) | Tg | Transitive verb + -ing form |
| Dnf | Double-trans. verb +noun+finite that-clause | Tnp | Transitive verb + noun + particle |
| Dnw | Double-trans. verb + noun + wh-clause | Tsg | Transitive verb + noun's + -ing form |
| Dprf | Double-trans. verb +prep.+finite that-clause | Tw | Transitive verb + wh-clause |
| Dprt | Double-trans. verb +prep.+ to-infinitive | Vn | Verb + noun |
| Dprw | Double-trans. verb +prep.+wh-clause | Vng | Verb + noun + -ing form |
| I | Intransitive verb | Vni | Verb + noun + infinitive |
| Ip | Intransitive verb + particle | Vnt | Verb + noun + to-infinitive |
| Ipr | Intransitive verb + prep. | Vnn | Verb + noun + noun |
| La | Linking verb + adj. | Vnpr | Verb + noun + prep. |
|  |  | Vt | Verb + to-infinitive |

Since some noun phrases might be moved to other places, e.g., passive sentences, to determine predicate-argument structures requires that this issue be taken into account. Move-a [Sells 1985] formalizes the phenomenon that a constituent is moved to the landing site, and that a trace is left at the empty site. In general, the moved constituent is an NP, and the object position is a possible empty site. Thus, the effect of movement transformation must be considered when predicate-argument structures are formulated. Two rules are shown as follows.

- While it is a passive sentence, the predicate-argument structure of a matrix verb is added an argument, for example, *eat* in the sentence "an apple is eaten by Mary".
- The predicate-argument structure of a verb in a relative clause whose relative pronoun serves as object is added to an argument, for example, *meet* in the sentence "the man whom you met is my brother".

Another problem is induced by compound nouns. For example, a transitive verb may be misrecognized as a ditransitive verb. An NP-TRACTOR [Chen and Chen 1994b] is applied to collecting noun phrases.

## 4. Transfer Module

The previous section specified that the output of the analysis module is a chunk sequence with a head word and the underlying predicate-argument structure. Upon receiving the information, the transfer module has to complete two tasks. One is lexical transfer, and the other is structural transfer. This is described by the mathematical model $P(C_t|C_s) \times P(PA_t|PA_s)$. The lexical transfer, $P(C_t|C_s)$, is intended to select the proper target counterpart for a source word, and the structural transfer, $P(PA_t|PA_s)$ , is intended to change the chunk order according to the predicate-argument structure provided by the analysis module. This kind of transfer module only performs simple transfer, that is, transfers the skeleton of the source language to that of the target language. Transfers within chunks are fulfilled in the synthesis model.

### 4.1 Simple transfer

Simple transfer focuses on the relationship between predicates and arguments. The predicate-argument structure forms the skeleton of the whole sentence, and the other constituents are the modifying components. Consider example (iii). *Gave* is a ditransitive verb, which introduces two internal arguments: one receives the accusative case, and the other dative case:

(iii) I gave Mary a book. 我 把 書 給了 瑪莉。

In English, both "I gave Mary a book" and "I gave a book to Mary" are correct while the corresponding Chinese is " 我把書給了瑪莉 " is the general form. In the above example, the argument structure is formed by *gave, I, Mary*, and *book*. *Gave* is the predicate. *I* plays the role of external argument, and two internal arguments are *Mary* and *book*. Figure 5 shows the transfers between the two different languages. The predicate- argument structures constrain the word order in the coming synthesis module.

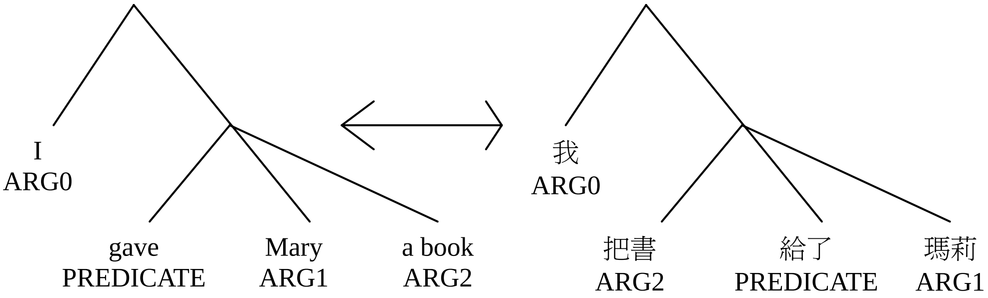


***Figure 5. The Simple Transfer across Languages***

A set of rules is responsible for transfer of predicate-argument structures across languages. Each rule deals with one predicate-argument structure. These structures are determined in the analysis module as discussed in Section 3.2. Some mappings of predicate-argument structures between English and Chinese are direct. These predicate-argument structures are I, Ipr, Ip, La, Vn, Vt, Vnt, Vng, Vni, Tf, Tnp, Tw, Tg, Tsg, Dnf, Dprf, Dnw, Dprw, and Dprt. The mapping rules for the rest of the predicate- argument structures are shown in Table 3.

***Table 3. The Mapping Rules for Predicate-Argument Structures***

| | English Predicate-Argument Structure | Chinese Predicate-Argument Structure |
|---|---|---|
| Vnpr | arg0 verb arg1 preposition arg2 | arg0 使 (把, 將) arg1 verb arg2 |
| Cna | arg0 verb arg1 adjective | arg0 使 (把, 將) arg1 verb adjective |
| Cnn/a | arg0 verb arg1 as arg2 | arg0 verb arg1 爲 arg2 |
| | arg0 verb arg1 as adjective | arg0 verb arg1 爲 adjective |
| Vnn | arg0 verb arg1 arg2 | arg0 使 (把, 將) arg2 verb arg1 |

The main difference between English and Chinese is in the positions of prepositional phrases. In Chinese, they are in pre-modifying positions. Therefore, resolving the problem of prepositional phrase attachment is indispensable. Some approaches to determination of PPs have been reported in the literature [Kimball 1973; Frazier 1978; Ford *et al.* 1982; Shieber 1983; Wilks *et al.* 1985; Liu *et al.* 1990; Chen and Chen 1992; Hindle and Rooth 1993; Brill and Resnik 1994]. The possible kinds of attachment they consider are NOUN attachment and VERB attachment. These resolutions fall into three categories: syntax-based, semantics-based and corpus-based approaches. These approaches resolve PP attachment via only one language consideration. In contrast, because PP-attachment is a part of our machine translation system, we investigate this problem from the viewpoint of machine translation and do not restrict ourselves in the two possible attachment choices.

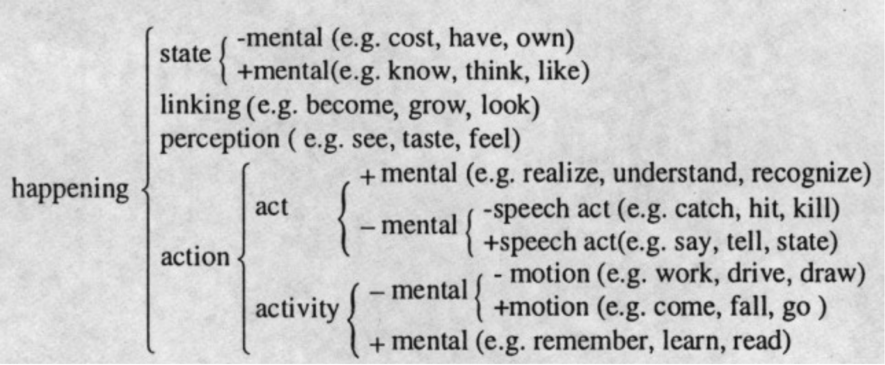

happening
state
-mental (e.g. cost, have, own)
+mental(e.g. know, think, like)
linking (e.g. become, grow, look)
perception ( e.g. see, taste, feel)
action
act
+ mental (e.g. realize, understand, recognize)
– mental
-speech act (e.g. catch, hit, kill)
+speech act(e.g. say, tell, state)
activity
– mental
- motion (e.g. work, drive, draw)
+motion (e.g. come, fall, go )
+ mental (e.g. remember, learn, read)

***Figure 6. Semantic Tags for Verbs***

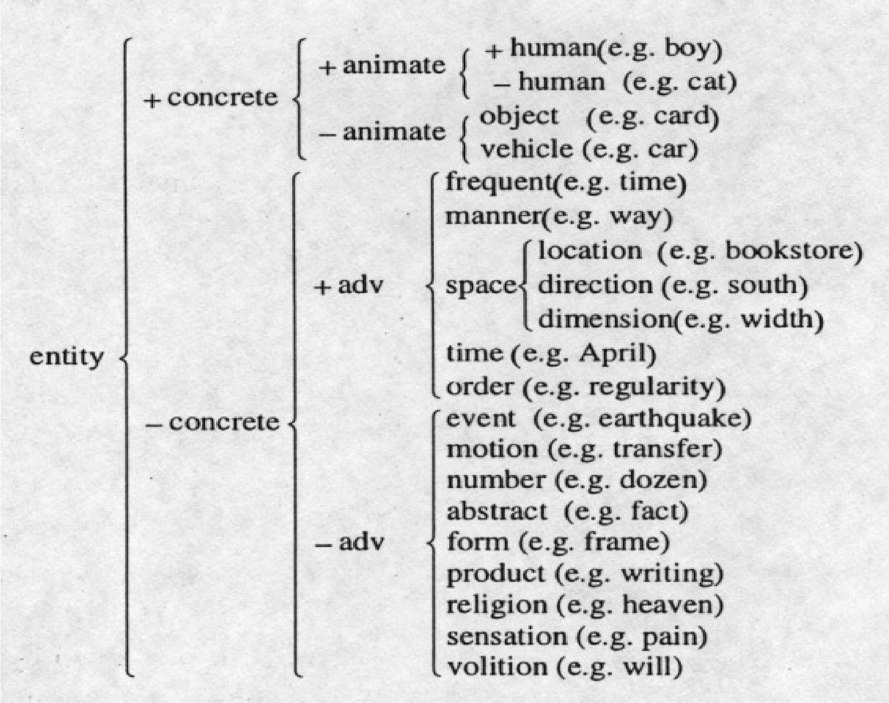


***Figure 7. Semantic Tags for Nouns***

In general, four factors influence the determination of PP-attachment: 1) verbs, 2) accusative nouns, 3) prepositions, and 4) oblique nouns. A total of 68 rule-templates in the form of 4-tuple <$V$, $N$ 1, $P$, $N$ 2> are used to decide on the attachment point, where $V$ denotes semantic tag of the verb, $N$ 1 denotes the semantic tag of accusative noun, $P$ denotes the preposition and $N$ 2 denotes the semantic tag of oblique noun. These semantic tags for verbs and nouns are shown in Figure 6 and Figure 7. In addition, we distinguish four kinds of prepositional phrases.

- Predicative Prepositional Phrases (PPP): PPs that serve as predicates.

  He is at home. He found a lion in the net.

- Sentential Prepositional Phrases (SPP): PPs that serve as functions of time and location.

  There is no parking along the street. We had a good time in Paris.

- Prepositional Phrases Modifying Verbs (VPP)

  I went to a movie with Mary. I bought a book for Mary.

- Prepositional Phrases Modifying Nouns (NPP)

  The man with a hat is my brother. Give me the book on the desk.

When a prepositional phrase occurs, we check whether it is PPP, SPP, or VPP. Otherwise, it is an NPP. The procedure for resolving PP attachment is shown as follows.

**Algorithm 2: Resolution to PP-Attachment**

```
(1)   Check if it is a PPP according to the predicate-argument structure.
(2)   Check if it is an SPP according to 21 rule-templates for SPP.
(3)   Check if it is a VPP according to 47 rule-templates for VPP.
(4)   Otherwise, it is an NPP.
```

Appendix A lists 21 rule-templates for SPP and 47 rule-templates for VPP.

## 4.2 Lexical Selection

Assume that the probability of a source and target language pair ($C_s$,$C_t$) is $P(C_s,C_t)$. By the definition of conditional probability, $P(C_s,C_t) = P(C_s) \times P(C_t|C_s) = P(C_t) \times P(C_s|C_t)$. The co-occurrence of a source and target pair could be explained by either side. These probabilities are the implicit bilingual constraints, and this implies that the source language and target language can disambiguate each other [Alshawi 1994]. As the result, many approaches based on bilingual corpora or bilingual machine readable dictionaries have also been proposed which select the right senses of words. Brown *et al.* [1992] applied a flip-flop algorithm to decide on the informants of a word and to use the mutual information of informant and the word to choose the sense. However, the training data used in their approach is huge (12,028,485 connections extracted from 1,002,165 pairs of short French and English), and aligned bilingual texts are needed. Firth [1968] pointed out that "You shall know a word by the company it keeps." We thus follow this concept and propose an algorithm to disambiguate word sense. Doi and Muraki [1993] have used a similar idea in choosing a Japanese lexicon from its English counterpart. The most informative word is used as a constraint to select the proper target word. Here, "the most informative" means that two words with the highest mutual information (*MI*) will disambiguate each other. The mutual information is

$$(10)\qquad MI(x,y)=\log\frac{P(x,y)}{P(x)\times P(y)}=\log\frac{\frac{f(x,y)}{N}}{\frac{f(x)}{N}\times\frac{f(y)}{N}}=\log\frac{f(x,y)\times N}{f(x)\times f(y)},$$

where *f(x)* and *f(y)* are the frequencies of x and y in the training corpus, respectively. *f(x,y)* is the frequency of *x* and *y* co-occurring in the training corpus, and *N* is the number of words in the training corpus. Figure 8 shows how bilingual constraints operate.

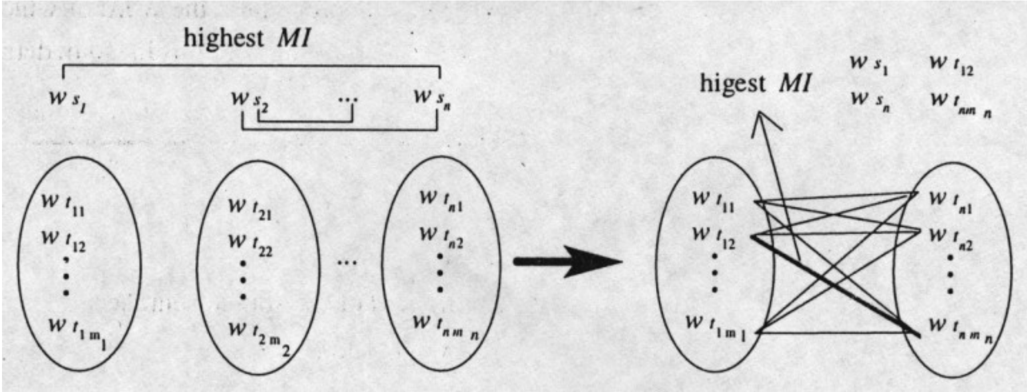


***Figure 8. Bilingual Constraints***

Assume that a source sentence contains n words, and that the word pair ($ws_1$,$ws_n$) has the highest *MI*. In our model, the two senses which have the maximal *MI* are selected from {$wt_{11}$, ..., $wt_{1m1}$} and {$wt_{n1}$, ..., $wt_{nm_n}$ }, respectively, and are regarded as the translation of $ws_1$ and $ws_n$. For example, the right Chinese counterpart of the sentence "flying plane makes her duck" is "正在飛的飛機使她迅速低頭". "Fly" has many senses in Chinese such as "飛 ", "逃出", *etc*. "Duck" has four readings in Chinese: "鴨子", "迅速低頭", "暫時沒入水中", and "水陸兩用車". The possible readings of each word are listed in Table 4.

***Table 4. The Possible Chinese Translations of Various English Words***

| Words | fly | plane | make | her | duck |
|---|---|---|---|---|---|
| Sense 1 | 飛 | 平面 | 吃 | 她 | 鴨子 |
| Sense 2 | 逃出 | 飛機 | 使 | 她的 | 迅速低頭 |
| Sense 3 | | | 規定 | | 水陸兩用車 |
| Sense 4 | | | 製造 | | 暫時沒入水中 |

Word pairs (make,duck), (her,duck), (make,her), (plane,her), (plane,make), and (fly,plane) are with the value of *MI* in descending order. The word pair (make,duck) has the highest *MI*, so both words are the most informative to each other. Then, the senses of the two words are determined by *MI* of the target words. Therefore, the right senses, "使" and "迅速低頭", are selected, and the senses of the two words are fixed. The second highest *MI* is the word pair (her,duck). Since the sense of duck is fixed, the sense of her is the one which has the highest *MI* with "迅速低頭". Using the same procedure, we can determine the sense of each word. After using the procedure, the word of which the sense is not fixed is assigned the most frequent sense. The algorithm is listed in detail as follows.

**Algorithm 3: Lexical Selection**

```
Notation:  $S_i$: the i'th word of the source sentence
           $Ti_k$: the k'th target translation of the i'th word of the source sentence
           $MI(E_i, E_j)$: mutual information of two expressions
Input:     A sentence consists of n words, $S_1, S_2, ..., S_n$.
Output:    A target word sequence.
Method:    (1) Select the source word pair $S_i$ and $S_j$ that has the largest $MI(S_i, S_j)$,
               where at least one of their senses is not fixed.
           (2) Select the target word pair $T_{i_p}$ and $T_{j_q}$ that has the largest $MI(T_{i_p}, T_{j_q})$.
           (3) Fix the target translation of $S_i$ and $S_j$ as $T_{i_p}$ and $T_{j_q}$, respectively.
           (4) Repeat (1) to (3) until every target translation of $S_i$ ($i = 1, 2, ..., n$) is fixed.
```

## 5. Synthesis Module

In general, the synthesis module is responsible for producing natural language from a special structure. For example, the synthesis module in a knowledge-based system generates natural language form concept structures [Baker *et al.* 1994]. Some complicated language generation systems involve taking syntactic knowledge, semantic knowledge, pragmatic knowledge and world knowledge into consideration [McKeown and Swartout 1987]. However, to generate a "natural" sentence is not easy. It raised several questions. How do we choose the appropriate words, how do we arrange the word order, and where do we insert extra words? Since pragmatic knowledge or world knowledge is very subtle, to effectively use it is very difficult. McKeown [1987] has discussed two kinds of generation: deep generation and surface generation. Deep generation is responsible for determining the text content; surface generation is responsible for selecting lexical items. Usually, the two generation mechanisms are concatenated as pipelined modules. Such a complicated text generation system, in fact, covers many tasks performed by the analysis module and the transfer module in a machine translation system. For instance, surface generation could be handled in the transfer module (lexical transfer as discussed in a previous section), and deep generation could be determined in the analysis module as a knowledge-based MT system does.

Since many tasks are performed in other modules in our MT model, the synthesis module is only responsible for reordering words in chunks. Chunk reordering, discussed in a previous section, focuses on global reordering; in contrast, word reordering in chunks is regarded as local reordering. Equation 5 is presented again in the following:

$$\begin{aligned} P(W_t \mid IF_t) &= P(W_t \mid C_t, PA_t) \cong P(W_t \mid C_t, R) \cong P(W_t \mid Ct_{R(1)}^{R(m)}) \\ &\cong \sum_{i=1}^{m} P(Wt_{R(i)} \mid Ct_{R(i)}) = \sum_{i=1}^{m} P({Wt_{R(i)1}}^{l_{R(i)}} \mid Ct_{R(i)}) \\ &\cong \sum_{i=1}^{m} P(Wt_{R(i)1}) \prod_{j=1}^{l_{R(i)}-1} P(Wt_{R(i)\,j+1} \mid Wt_{R(i)\,j}) \end{aligned} \quad . \qquad (5)$$

In Equation 5, R denotes the selected simple transfer rules or PP-attachment rules, i.e., PPP, NPP, VPP, or SPP. Therefore, R functions as a permutation function, that is *R*: {1, 2, ..., m}→{1, 2, ..., m}. $l_{R(i)}$ denotes the number of words in target chunk $c_{R(i)}$. The target word order is captured by the Markov model as equation 5 shows. The disadvantage of the Markov model is its lack of capability in capturing long distance phenomena. However, this disadvantage is reduced in our model, since long distance phenomena are less in chunks. The knowledge of word order can be trained from large corpora. The Newspaper Corpus, a segmented Chinese corpus composed of texts from three major newspapers, was used as the training corpus. Table 5 lists the extracted statistic information.

***Table 5. Statistic Information of the Newspaper Corpus***

| Corpus | Total Words | Different Words | Word Bi-Gram |
|---|---|---|---|
| Newspaper Corpus | 2,636,793 | 43,262 | 921,633 |

## 6. Experiments

A total of 670 sentences are used to test this proposed MT model. These testing sentences contain intransitive verbs, transitive verbs, ditransitive verbs, prepositional phrases and some common constituents in English. Basically, the testing sentences could be partitioned into three parts: NP + VP + PP. The PP part might modify verbs in VP, nouns in VP or the whole sentence. Thus, these testing suite represented general phenomena in language. The results were evaluated using two factors. One is word sense; the other is word order. For word sense, four grades are considered: *A*, *B*, *C* and *don't care*. By human intuition, a target word is recognized as grade *A (C)* which depends on whether the meaning of the corresponding source word is expressed correctly (wrong) exactly. A target word is marked as *don't care*, if the corresponding source word is not found in the dictionary, or if it is an idiom which cannot be translated directly. The remaining words are regarded as grade *B*. The score for word sense, *SWS*, is defined as

$$(11) \qquad SWS = \frac{S_a \times \#\text{of A} + S_b \times \#\text{of B} + S_c \times \#\text{of C}}{n - \#\text{of don't care}},$$

where $S_a = 1$, $S_b = 0.5$, $S_c = 0$ and $n$ is the number of words.

For word order, the difference between exact word position (*EWP*) and generated word position (*GWP*) for each word is considered. Therefore, the difference of word order, *DWO*, is defined as

$$(12) \qquad DWO = (\sum_{i=1}^{n} abs(EWP_i - GWP_i)) / n,$$

where $n$ is the number of words.

Note that a high *SWS* means good results. On the contrary, a high *DWO* denotes bad results. These two factors together reflect the performance of the MT system. Based on these definitions, the results of the evaluation are shown in Table 6.

***Table 6. Experimental Results***

| *SWS* | Number | *DWO* | Number |
|---|---|---|---|
| $0.9 \le SGS \le 1.0$ | 145 | $0.0 \le DWO < 1.0$ | 457 |
| $0.8 \le SGS < 0.9$ | 224 | $1.0 \le DWO < 2.0$ | 113 |
| $0.6 \le SGS < 0.8$ | 242 | $2.0 \le DWO < 3.0$ | 56 |
| $0.0 \le SGS < 0.6$ | 59 | $3.0 \le \mathrm{DWO}$ | 44 |

From Table 6, the performance of the testing on sentences is promising. The number of sentences with *SWS* higher than 0.6 is 611 (91%), and the number of *DWO* values less than 1.0 dominates the entire distribution (68%). The efforts to recover the original word order from the generated word order are also considered. They are measured according to the number of "key strokes" for recovering. Typically, to move a word needs a key stroke (the actions of cut and paste are regarded as one key stroke). This measure is important for MT systems in the post-editing phase. The distribution of key strokes is listed in Table 7. The average number of key strokes for recovering a sentence is 0.5.

***Table 7. The Distribution of Key Strokes***

| Key Strokes | Number |
|---|---|
| 0 | 447 |
| 1 | 142 |
| 2 | 53 |
| 3 | 25 |
| 4 | 3 |

## 7. Conclusion

In this paper, an integrated approach to machine translation design has been proposed. It not only has advantages of the qualitative approach in coverage of core linguistic phenomena, but also keeps the advantages of the quantitative approach in dealing with marginal linguistic phenomena. The statistics-based approach and linguistic theory are complementary. The statistics-based approach is robust. It provides simple language models for analyzing unrestricted texts. However, it may need a large completely-annotated corpus to deal with complex linguistic phenomena. Linguistic theory gives such a supplement. Well-formed patterns can be explained properly by means of universal principles, so that they can be formulated in terms of rules easily.

Since fully understanding sentences will not be possible in the near future, the proposed MT system does not completely parse input sentences. A partial parsing method is adopted, and the translation process is performed chunk by chunk. In the synthesis module, the word order is locally rearranged in chunks via the Markov model. Since the length of a chunk is much shorter than that of a sentence, the disadvantage of the Markov model in dealing with long distance phenomena is greatly reduced. Structural transfer is fulfilled using a set of rules; in contrast, lexical transfer is resolved using mutual information which is trained from the text corpora. Qualitative and quantitative knowledge is used interleavingly and cooperatively in the proposed MT system.

A testing suite containing general phenomena in language usage has been used to evaluate the feasibility of the proposed MT system. The performance measures are based on word sense and word order. The experimental results show that the integrated approach to the MT system has good performance in both measures. The post-editing efforts needed in this MT system is also small in the testing suite.

## References

Abney, S., "Parsing by Chunks," *in Principle-Based Parsing, Berwick*, Abney and Tenny (Eds.), Kluwer Academic Publishers, 1991, pp. 257-278.

Alshawi, H., "Qualitative and Quantitative Models of Speech Translation," *Proceedings of Workshop on the Balancing Act: Combining Symbolic and Statistical Approaches to Language*, Las Cruces, USA, 1994, pp. 1-10.

Baker, K. *et al.*, "Coping with Ambiguity in a Large-Scale Machine Translation System," *Proceedings of COLING-9*4, Kyoto, Japan, 1994, pp. 90-94.

Bennett, W. and J. Slocum , "The LRC Machine Translation System," *Computational Linguistics*, vol. 11, no. 2-3, 1985, pp. 111-119.

Brill, E. and P. Resnik, "A Rule-Based Approach to Automated Prepositional Phrase Attachment Disambiguation," P*roceedings of COLING-94*, 1994, pp. 1198-1204.

Brown, P. *et al.*, "Analysis, Statistical Transfer, and Synthesis in Machine Translation," *Proceedings of TMI-9*2, 1992, pp. 83-100.

Chang, J.S. and K.Y. Su, "A Corpus-Based Statistics-Oriented Transfer and Generation Model for Machine Translation," *Proceedings of the Fifth International Conference on Theoretical and Methodological Issues in Machine Translation (TMI93)*, Kyoto, Japan, 1993, pp. 3-14.

Chen, K.H. and H.H.Chen, "A Probabilistic Chunker," *Proceedings of R.O.C. Computational Linguistics Conference VI*, 1993, pp. 99-117.

Chen, K.H. and H.H. Chen, "Acquiring Verb Subcategorization Frames," *Proceedings of the Second Conference for Natural Language Processing*, Vienna, Austria, 1994a, pp. 407-410.

Chen, K.H. and H.H. Chen, "Extracting Noun Phrases from Large-Scale Texts: A Hybrid Approach and Its Automatic Evaluation," *Proceedings of the 32nd Annual Meeting of ACL*, 1994b, pp. 234-241.

Chen, K.H. and H.H. Chen, "Attachment and Transfer of Prepositional Phrases with Constraint Propogation," *Computer Processing of Chinese and Oriental Languages: An International Journal of the Chinese Language Computer Society*, 6(2), 1992, pp. 123-142.

Church, K. and Hanks, P., "Word Association Norms, Mutual Information and Lexicography," *Computational Linguistics*, 1990, pp. 22-29.

Doi, S. and K. Muraki, "Evaluation of DMAX Criteria for Selecting Equivalent Translation based on Dual Corpora Statistics," *Proceedings of the Fifth International Conference on Theoretical and Methodological Issues in Machine Translation with special emphasis on: MT in Next Generation*, Kyoto, Japan, 1993, pp. 302-311.

Firth, J., "A Synopsis of Linguistic Theory 1930-1955," *Studies in Linguistic Analysis*, Philological Society, Oxford, 1957, reprinted in *Selected Papers of J. R. Firth* (Eds. F. Palmer), Longman, 1968.

Ford, M., J. Bresnan and R. Kaplan, "A Competence-Based Theory of Syntactic Closure," The Mental Representation of Grammatical Relations (Bresnan, J. Eds.), MIT Press, 1982, pp. 727-796.

Frazier, L., *On Comprehending Sentences: Syntactic Parsing Strategies*, Doctoral Dissertation, University of Connecticut, 1978.

Hindle, D. and M. Rooth, "Structural Ambiguity and Lexical Relations," *Computational Linguistics*, 19(1), 1993, pp. 103-120.

Hornby, A.S., *Oxford Advanced Learner's Dictionary*, Oxford University Press, 1989. Johansson, S., *The Tagged LOB Corpus: Users' Manual*, Bergen: Norwegian Computing Centre for the Humanities, 1986.

Kimball, J., "Seven Principles of Surface Structure Parsing in Natural Language," *Cognition*, 2, 1973, pp. 15-47.

Liu, C.L., J.S. Chang and K.Y. Su, "The Semantic Score Approach to the Disambiguation of PP Attachment Problem," *Proceedings of ROCLING-90*, Taiwan, R.O.C., 1990, pp. 253-270.

McKeown, K. and W. Swartout, "Language Generation and Explanation," The Annual Review of Computer Science, 2, 1987, pp. 401-449.

Mitamura, T., Nyberg, E. and Carbonell, J., "An Efficient Interlingua Translation System for Multilingual Document Production," *Proceedings of Machine Translation Summit III*, Washington, DC, 1991.

Nagao, M., "A Framework of Mechanical Translation between Japanese and English by Analogy Principle," *Artificial and Human Intelligence* (A. Elithorn Eds.), 1984, pp. 173-180.

Sampson, G., *English for the Computer*, Oxford University Press, 1995.

Sells, P., *Lectures on Contemporary Syntactic Theories*, Lecture Notes, No. 3, CSLI, 1985.

Shieber, S., "Sentence Disambiguation by a Shift-Reduced Parsing Technique," *Proceedings of IJCAI-83*, Kahlsruhe, Germany, 1983, pp. 699-703.

Wilks, Y., X.H. Huang and D. Fass, "Syntax, Preference and Right Attachment," *Proceedings of IJCAI-85*, Los Angeles, CA, 1985, pp. 779-784.

## Appendix A. Rule Templates for PP-Attachment

The following lists rule-templates for PP-attachment. Every template consists of four elements <*V*, *N*1, *P*, *N*2>. The curl bracket pair denotes *OR*, the underline denotes *DON'T CARE* and ~ denotes *NOT*.

### I. Rule-templates for SPP

1. <___, ___, about, time>
2. <___, ___, across, location>
3. <___, ___, after, time>
4. <___, ___, along, location>
5. <___, ___, among, location>
6. <___, ___, at, {location, time}>
7. <___, ___, before, time>
8. <___, ___, between, {location, time}>
9. <___, ___, by, time>
10. <___, ___, during, time>
11. <___, ___, in, {location, time}>
12. <___, ___, in_front_of, location>
13. <___, ___, near, location>
14. <___, ___, next_to, location>
15. <___, ___, on, time>
16. <___, ___, out_of, {abstract, location}>
17. <___, ___, over, {location, time}>
18. <___, ___, through, {abstract, event, time}>
19. <___, ___, under, time>
20. <___, ___, with, abstract>
21. <___, ___, without, abstract>

### II. Rule-templates for VPP

1. <motion, ___, about, {object, location}>
2. <at_ment, ___, about, object>
3. <action, animate, about, ___>
4. <action, event, after, concrete>

5. <at_ment, {abstract, event}, after, {event, no, time}>
6. <motion, ___, across, {location, object}>
7. <{at_nonmen, ai_nonmen}, ___, along, {location, object}>
8. <~motion, ~{concrete, location}, among, {concrete, location}>
9. <~{at_nonmen, ai_nonmen}, ___, at, {animate, object}>
10. <{at_nonmen, ai_nonmen}, ___, at, {location, object}>
11. <action, event, after, concrete>
12. <at_ment, {abstract, event}, after, {event, no, time}>
13. <{at_nonmen, ai_nonmen}, {event, object}, between, {abstract, concrete, location}>
14. <{at_nonmen, ai_nonmen}, {event, object}, between, time>
15. <motion, ___, by, {location, object}>
16. <___, ___, by, manner>
17. <~motion, ___, by, {location, object}>
18. <___, ___, by, {abstract, event, object, vehicle}>
19. <___, ___, by, animate> passive voice
20. <___, ___, for, time>
21. <motion, ___, for, location>
22. <~linking, ___, for, {abstract, concrete, event}>
23. <___, ___, for, {abstract, event, object}>
24. <___, ___, for, animate}>
25. <{motion, speech_act}, ___, from, entity>
26. <motion, ___, in, {location, object}>
27. <___, ___, in, time>
28. <___, ___, in, vehicle>
29. <ai_nonmen, ___, in_front_of, {concrete, location}>
30. <ai_nonmen, ___, inside, {concrete, location}>
31. <___, ___, into, {abstract, concrete, location}>
32. <act, ___, like, ___>
33. <___, ~{location, object}, near, {location, object}>
34. <___, ~{location, object}, next_to, {location, object}>
35. <{at_nonmen, ai_nonmen}, {event, object}, on, {concrete, location, object}>

36. <___, ___, on, event>
37. <___, ___, on_to, {location, object}>
38. <motion, ___, out_of, {concrete, location}>
39. <___, ___, over, {abstract, event}>
40. <motion, ___, over, {concrete, location}>
41. <ai_nonmen, ___, through, {location, object}>
42. <{ai_nonmen, at_nonmen}, ___, under, {abstract, object}>
43. <___, ___, until, time}
44. <at_nonmen, ___, with, object>
45. <at_nonmen, ___, with, animate>
46. <at_nonmen, ___, without, object>
47. <at_nonmen, ___, without, animate>