Chapter 1

Introduction

Cobra Gold Machine Translation Project is a tight collaboration between Communications and Electronics Research, Development, and Engineering Center (CERDEC), US Army and National Electronics and Computer Technology Center (NECTEC), Thailand to develop a rule-based machine translation system between English and Thai. It aims at supporting communication between American and Thai militaries by providing a translation service in both directions. At present, Thai to English MT system was delivered from NECTEC to CERDEC and have been implemented recently.

This project is under contracted research no. PMO/004/2550-N68047-07-P0018. Project's name is Thai to English translation project. The total amount of budget is 170,000 $.

Thai to English MT system consists of three main parts; Thai word-segmentation, lexical bi-lingual resource, and translation system. Since this system is rule-based, linguistic information is applied in this project. This report also provides the details of divergence between Thai and English to explain the differences between Thai and English which concerns problems of translation. This information includes different word order, no plural inflection in Thai, serial noun construction, serial verb construction, etc.

Moreover, this report also explains Thai-English Dictionary construction and its linguistic information. It aims to construct lexical information to serve Thai to English rule-based machine translation system. To gather the lexicon and its information, we estimate the different types of usage and provide sufficient information to each lexical entry. Implicit attributes of function words are also included to the framework to automatically derive grammatical information from Thai sentences.

The rest of this report is constructed as follows. Chapter 2 describes a divergence between Thai and English. Word segmentation system is explained in Chapter 3. In Chapter 4, lexical resource is explained in details. Chapter 5 reports the engine of translation system. Lastly, Chapter 6 concludes this report.

Chapter 2

Divergence between Thai and English

2.1 Characteristics of Thai

Thai belongs to analytic language typology; that is, its syntax and meaning depend on the use of particles and word orders rather than inflection. It bears some resemblances to other analytic languages such as Chinese, Laotian, and Vietnamese. Pronouns and other grammatical information, such as tenses, aspects, numbers, and voices, are expressed by adverbs and adjectives and they are commonly omitted in non-initial sentences. Serial verb construction is commonly used to express consecutive or concurrent events. Word order schema of Thai is summarized below in Table 2.1. The parentheses are used to denote an optional part.

Table 2.1. Word order schema of Thai

Types	Word orders
Sentence	Subject + Verb + Object [rigid order]
Compound noun	Core noun + Attachment
Adjective modification	Noun + Adjective
Article	There is no article in Thai
Determiner	Noun + (Classifier) + Determiner
Numeral expression	Noun + (Modifier) + Number + Classifier + (Modifier) Noun + (Modifier) + Number + Collective + (Modifier)
Adverb modification	Sentence + Adverb
Preposition phrase	Preposition + Complementary
Negation	Subject + Negator + Verb phrase
Passive	Actee + Passive marker + (Actor) + Verb
Ditransitive	Subject + Ditransitive verb + Direct object + Indirect object

In orthographical perspective, Thai script is written consecutively without word boundary markers. Spaces are ambiguously used to separate sentences and phrases for ease of reading.

2.2 Divergence Issues in Thai-English Translation

Translation from Thai to English is a challenging task. There are six issues of translation from Thai to English: (1) different word orders, (2) different verb arguments, (3) implicit relations in Thai serial noun construction, (4) semantic duplication in Thai serial verb construction, (5) no plural inflection in Thai, and (6) no inflection to express voices, tenses, and aspects in Thai.

2.2.1 Different Word Orders

Belonging to quite different typologies, they bear somewhat unlike word orders. The differences are summarized below in Table 2.2.

Table 2.2. The differences of word order of Thai and English

Types	Word Orders of Thai	Word Order of English
Sentence	S + V + O	S + V + O
Compound noun	Core N + Attachment	Core N + Attachment
Adjective modification	N + Adj	Adj + N
Article	There is no article in Thai	Art + N
Determiner	N + (CL) + Det	Det + N
Numeral expression	N + (Mod) + Num + CL + (Mod) N + (Mod) + Num + Col + (Mod)	Num + N Num + Col + of + N
Adverb modification	Sent + Adv	S + (Adv) + V + O + (Adv)
Preposition phrase	Prep + Comp	Prep + Comp
Negation	S + Neg + VP	S + Aux verb + Neg + VP
Passive	Actee + Pass + (Actor) + V	Actee + be + V_3 + (by + Actor)
Ditransitive	S + DV + DO + IO	S + DV + IO + DO

For example, consider the following Thai noun phrase and verb phrase and their translation.

Noun phrase

Thai	เสื้อ	ใหม่	สอง	ตัว	นั้น
IPA	sɯ̂ə	mài	sɔ̌ːŋ	tua	nán
Lit.	shirt	new	two	(CL: body)	that
Translation	‘those two new shirts’				

Verb phrase

Thai	แม่ครัว	เตรียม	อาหาร	อย่าง	รวดเร็ว
IPA	mɛ̂ː·kʰrua	triam	ʔaː·hǎːn	jàːŋ	ruât·reʊ
Lit.	female cook	prepare	meal	(Adv marker)	rapid
Translation	‘The female cook rapidly prepares a meal.’				

2.2.2 Different Verb Arguments

There are slight differences in verb argumentation between Thai and English — motion verbs and complex verbs, in particular.

Thai motion verbs differ from English ones in that the former does not require a preposition phrase while the latter does. In Thai, motion verbs are considered as transitive; i.e. they require the destination of the action as the object. However, this is not the case in English. English motion verbs are considered as intransitive ones that require a preposition phrase designating the destination. Consider the following example.

Thai	ฉัน	ไป	ฝรั่งเศส
IPA	cʰǎn	pai	fá·ràŋ·sèːt
Lit.	I	go	France
Translation	‘I go to France.’		

In the above example, the Thai verb /pai/ ‘go’ requires the destination as the direct object. In English, the verb ‘go’ however requires a preposition phrase that determines the destination.

Complex verbs in Thai differ from English ones in that the former express sequential event by means of serial verb construction, while the latter by means of gerundial phrase. Consider the following example.

Thai	ภารโรง	หยุด	สูบ	บุหรี่
IPA	pʰaːn·roːŋ	jùt	sùːp	bù·rìː

Lit.	janitor	stop	smoke	cigarette
Translation	‘The janitor stops smoking cigarette.’			

In the above example, the Thai verbs /jùt/ ‘stop’ and /sùːp/ ‘smoke’ are serialized, while the English verb ‘smoke’ is gerundialized to form a complement of the verb ‘stop.’

2.3 Implicit Relations in Thai Serial Noun Construction

Serial noun construction (abbreviated SNC henceforward) is a phrasal construction in which two or more nouns combine to build up a noun phrase. Though occurring in English and Thai, Thai’s SNCs are more complicated than that of English’s ones in that Thai exhibits a variety of semantic relations between two consecutive verbs. There are three types of relations in Thai SNCs as exemplified as follows.

Types	Relations	Examples
I	Adjective	châonâathîi$_{N1}$ kamphuuchaa$_{N2}$ Lit: officer$_{N1}$ cambodia$_{N2}$ Trans: cambodian$_{N2}$ officer$_{N1}$
II	Possession	phuumpanyaa$_{N1}$ banphábùrùt$_{N2}$ Lit: intelligence$_{N1}$ ancestor$_{N2}$ Trans: ancestor$_{N2}$’s intelligence$_{N1}$
III	Apposition	phûusàmàk$_{N1}$ khonnôrk$_{N2}$ Lit: applicant$_{N1}$ outsider$_{N2}$ Trans: applicant$_{N1}$, who is an outsider$_{N2}$,

From corpus observation, Type I is the most frequent while Type II and Type III are less frequent.

2.4 Semantic Duplication in Serial Verb Construction

Serial verb construction (abbreviated SVC henceforward) is a clausal construction in which two or more verbs, optionally with syntactic objects, are used to express consequent or complex events. Let us consider the following example.

Thai	สมชาย	เดิน	ลง	บันได	ไป เข้าแถว	ซื้อ	อาหาร
IPA	somchaay	doen	long	bandai	pai khaothaew	sue	ahaan
Lit	Somchai	walk	descend	stairs	go queue	buy	food
Translation	‘Somchai walks down the stairs to queue up for buying some food.’						
	(Concisely ‘Somchai walks down the stairs for food.’)						

English translations of Thai's SVCs are sometimes semantically duplicated. Strictly speaking, English verbs convey both an action and the result of the action while Thai verbs are separated to convey each of which. For example, the two Thai verbs '*รายงาน*' /raayngaan/ (the action part — the patient is not necessary to know the matter) and '*ทราบ*' /saab/ (the result part) semantically compose the English verb 'to report.' However, semantic conveyance in Thai verbs is simplified for each of translation resulting in the verb '*รายงาน*' translated directly into 'to report' and this yields semantic duplication in Thai's SVCs. Let us consider some examples of semantic duplication.

	raayngaan$_{V1}$ hâi$_{P}$ sâab$_{V2}$
Lit:	report$_{V1}$ to$_{P}$ know$_{V2}$
Trans:	report$_{V1}$ (to know$_{V2}$)
	khâa$_{V1}$ hâi$_{P}$ taay$_{V2}$
Lit:	kill$_{V1}$ to$_{P}$ die$_{V2}$
Trans:	kill$_{V1}$ (to die$_{V2}$)
	phûut$_{V1}$ hâi$_{P}$ fang$_{V2}$
Lit:	tell$_{V1}$ to$_{P}$ listen$_{V2}$
Trans:	tell$_{V1}$ (to listen$_{V2}$)

2.5 No Plural Inflection in Thai

Thai exploits numeral modifiers to express plurality in contrast to English in which inflection plays an important role. There are four ways to express plurality in Thai: (1) by numeral phrase, (2) by collective phrase, (3) by duplication, and (4) by pluralization marker.

Types	Pluralization Methods	Examples
I	Numeral phrase	sùnák$_{N}$ sãam$_{NUM}$ tua$_{CL}$ Lit: dog$_{N}$ three$_{NUM}$ body$_{CL}$ Trans: three$_{NUM}$ dogs$_{N}$
II	Collective phrase	plaa$_{N}$ sãam$_{NUM}$ fũung$_{COL}$ Lit: fish$_{N}$ three$_{NUM}$ school$_{COL}$ Trans: three$_{NUM}$ schools$_{COL}$ of fish$_{N}$
III	Duplication	dèk$_{N}$-dèk$_{N}$ Lit: child-child$_{N}$ Trans: children$_{N}$
IV	Pluralization marker	phûak$_{PLU}$ nák-rian$_{N}$ Lit: group$_{PLU}$ student$_{N}$ Trans: students$_{N}$

In Type I and Type II, numeral phrases and collective phrases are made use as modifiers to express plurality. Moreover, short nouns are pluralized by duplication in Type III while a pluralization prefix ‘phûak’ is used in Type IV.

2.6 No Inflection to Express Voices, Tenses, and Aspects in Thai

Thai expresses voices, tenses, and aspects by means of adverbs. Consider the following examples.

Tns/Asp/Mod	Examples				
Past tense		chãn$_{PR}$	koey$_{PAST}$	pai$_{V}$	fràngsès$_{N}$
	Lit:	I$_{PR}$	PAST	go$_{V}$	France$_{N}$
	Trans:	I$_{PR}$ went$_{V+PAST}$ to France$_{N}$			
Progressive aspect		chãn$_{PR}$	kamlang$_{PROG}$	dùem$_{V}$	náam$_{N}$
	Lit:	I$_{PR}$	PROG	drink$_{V}$	water$_{N}$
	Trans:	I$_{PR}$ [am drinking]$_{V+PROG}$ water$_{N}$			
Passive voice		chãn$_{PR}$	thùuk$_{PASS}$	khruu$_{N}$	thamthôt$_{V}$
	Lit:	I$_{PR}$	PASS	teacher$_{N}$	punish$_{V}$
	Trans:	I$_{PR}$ [am punished]$_{V+PASS}$ by the teacher$_{N}$			

From the examples above, various auxiliaries are used to express tenses, aspects, and voices. The auxiliary verb ‘koey’ in the first example conveys the past tense. The auxiliary ‘kamlang’ in the second example shows the progressive aspect. Finally, the passive marker ‘thùuk’ is used to passivize the sentence.

Chapter 3

Word Segmentation

Since Thai word meaning relies in a word in a sentence and there is no explicit word boundary for Thai word, automatic word segmentation tool is necessary for separating words in NLP application especially RBMT. In this project, we aim to implement RBMT with word-based translation. An automatic word-segmentation is necessary to separate word in the input sentence before it is taken to MT system.

3.1 System overview

Our word-segmentation, called JWordSeg, is implemented based on a combination of rule-based and statistical method. JWordSeg consists of 5 modules: (1) statistical information generator, (2) word lattice builder, (3) lattice selection algorithm, (4) tuning up process, and (5) string generator. The main process of JWordSeg is illustrated in Figure X below.

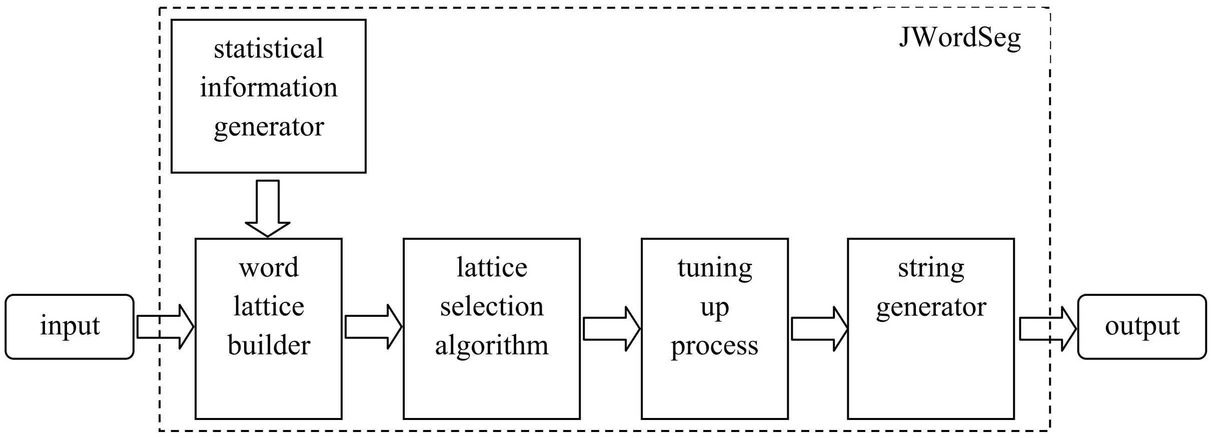


Figure 3.1. System overview

3.2 Generating statistical information module

This module is a pre-processing module for JWordSeg. For statistic information, we choose CMU toolkit to train required information for the further usages. CMU toolkit is used to generate general textual data into word frequency lists and vocabularies, word bigram and trigram counts, vocabulary-specific word bigram and trigram counts, bigram- and trigram-related statistics, and various Backoff bigram and trigram language models.

In this project, inputs for CMU toolkit are Orchid-1 and Orchid-2 which are word-segmented and POS-annotated. In total, these corpora contain 32,609 sentences and 468,833 words. After training, statistical information of the corpora is stored in the file type called *ARPA* file and it will be used as an input in lattice selection algorithm described in Section 3.4. An example of ARPA file is shown below in Figure 3.2.

```
…
-0.6298 คุย กัน -0.1253                                    [bigram example]
-2.2506 คุย กันเอง -0.0802
-0.6465 คุย กับ -0.2212
…
-0.8058 กระจาย อยู่ ใน                                     [trigram example]
-1.1723 กระจาย ออก เป็น
-0.0555 กลับ ถึง บ้าน
-2.1374 กลับ ถึง โรงแรม
…
```

Figure 3.2. An example of ARPA file

3.3 Word lattice builder

This module aims to generate any possible word lattices. A word lattice is a partially ordered set of words in a sequence pattern. An example of Thai word lattice is shown in Figure 3.3.

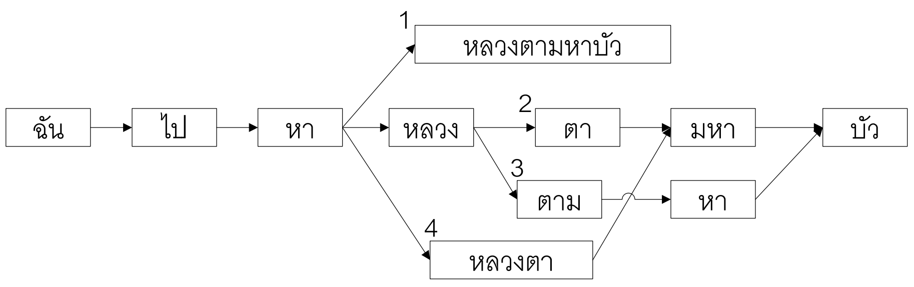


Figure 3.3. An example of Thai word lattice

This module requires three inputs: (1) a non-segmented input text, (2) a word list, and (3). non-segmented restriction rules

- A non-segmented input text is an ordinary raw text that will be word-segmented.
- A word list is a list of Thai words stored in file a word per line. An example of a word list file is illustrated in Figure 3.4.

```
. . .
กระแสจิต
กระแสตรง
กระแสน้ำ
กระแสพระราชดำรัส
```

กระแสรับสั่ง

กระแสลม

. . .

Figure 3.4. An example of a word list for word lattice builder module

Non-segmented restriction rules are Thai grammatical rules for a system to point out the point where cannot absolutely be segmented. There are three main circumstances that are not allowed to be word-segmented in any case; 1. after Thai_Preceeding character, 2. before Thai_succeeding character, and 3.before Thai_Zero_Length character. The details of non-segmented-restriction-rules are listed with an in Table 3.1. The item in the table is referred to Thai Unicode standard.

Table 3.1. A list of character using in non-segmented-restriction-rules

Thai Characters Set Name	Set Members
THAI_PRECEEDING	0E40 - THAI_CHARACTER_SARA_E, 0E41 - THAI_CHARACTER_SARA_AE, 0E42 - THAI_CHARACTER_SARA_O, 0E43 - THAI_CHARACTER_SARA_AI_MAIMUAN, 0E44 - THAI_CHARACTER_SARA_AI_MAIMALAI
THAI_SUCCEEDING	0E30 - THAI_CHARACTER_SARA_A, 0E31 - THAI_CHARACTER_MAI_HAN_AKAT, 0E32 - THAI_CHARACTER_SARA_AA, 0E33 - THAI_CHARACTER_SARA_AM, 0E34 - THAI_CHARACTER_SARA_I, 0E35 - THAI_CHARACTER_SARA_II, 0E36 - THAI_CHARACTER_SARA_UE, 0E37 - THAI_CHARACTER_SARA_UEE, 0E38 - THAI_CHARACTER_SARA_U, 0E39 - THAI_CHARACTER_SARA_UU, 0E3A - THAI_CHARACTER_PHINTHU
THAI_ZERO_LENGTH	0E31 - THAI_CHARACTER_MAI_HAN_AKAT, 0E34 - THAI_CHARACTER_SARA_I, 0E35 - THAI_CHARACTER_SARA_II, 0E36 - THAI_CHARACTER_SARA_UE, 0E37 - THAI_CHARACTER_SARA_UEE, 0E38 - THAI_CHARACTER_SARA_U, 0E39 - THAI_CHARACTER_SARA_UU, 0E3A - THAI_CHARACTER_PHINTHU, 0E47 - THAI_CHARACTER_MAITAIKHU, 0E48 - THAI_CHARACTER_MAI_EK, 0E49 - THAI_CHARACTER_MAI_THO, 0E4A - THAI_CHARACTER_MAI_TRI, 0E4B - THAI_CHARACTER_MAI_CHATTAWA, 0E4C - THAI_CHARACTER_THANTHAKHAT, 0E4D - THAI_CHARACTER_NIKHAHIT, 0E4E - THAI_CHARACTER_YAMAKKAN

A process of this module is described by algorithm below.

```
algorithm lattice-builder (wordList : list of word, input : input
stream)
returns Lattice
    let lattice as Lattice;
    let minLen = minimum word length in wordList
    let maxLen = maximum word length in wordList
    let buffer = []

    while(true) do
        let ch = next char in input
        case ch of
            EOS:    exit while // End Of Stream
            CR, LF: clear buffer
            others: append ch to buffer
        end case

        for i = minLen to maxLen do
            let subFrom = length of buffer minus by i
            if(subFrom < 0) then
                subFrom = 0
            end if
            let subTo = length of buffer
            let substr = substring of buffer from subFrom to subTo

            if(test-non-segment-restriction-rules(buffer, substr,
subFrom, subTo) == false) then
                if(substr found in wordList) then
                    add substr with subFrom and subTo to lattice
                end if
            end if
        end for
    end while
    clear buffer
    return lattice
end algorithm
```

The process of an algorithm `lattice-builder` begins with the system takes a raw-text and a word list as an input. Then the system examines the minimum size and maximum character's size of the word in the word list for adjusting window size. The system reads a character of raw-text input from the left. The process then checks the character by the minimum to maximum window size respectively. However, the process ignores the window size if the character in the window size fails `test-non-segment-restriction-rules`. If the sequence of character by a scope of window size matches the existing word in the word list, it will be put in the storage. After that, the system focuses on the next character and does the process again until the end of an input.

```
algorithm test-non-segment-restriction-rules(
    buffer : list of char,
    substr : substring of buffer from subForm to subTo
    subForm : start index of substr in buffer
    subTo : end index of substr in buffer)
returns Boolean

comment:
    return true if substr at subForm and subTo in buffer are invalid
rule in Thai word segmentation, otherwise return false
end comment

    let lastSubstrCh = last character in substr
    if(lastSubstrCh in THAI_PRECEDING_SET) then
        return true
    end if

    let firstSubstrCh = first character in substr
    if(firstSubstrCh in THAI_SUCCEEDING_SET) then
        retrun true
    end if

    if(firstSubstrCh in THAI_ZERO_LENGTH_CHAR_SET) then
        retrun true
    end if

    return false
end algorithm
```

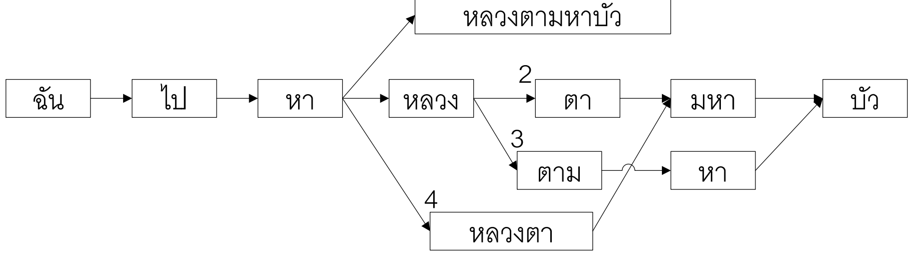
The process of an algorithm `test-non-segment-restriction-rules` is to test the characters in the window size. If the characters match the given rules in Table 3.2, the sequence of the character can ignore matching a word in the word list since it cannot be segmented.

This process returns all possible lattices which are an input to following process.

3.4 Lattice selection algorithm

The purpose of this module is to choose the highest probability of all word-lattices given by the previous module. This module takes all possible word lattices and statistic information from ARPA file as inputs. In the word lattice, it can be many possible word sequence. For example, Thai sentence "*ฉันไปหาหลวงตามหาบัว*" has 4 alternative lattices shown in Figure 3.5 below.

Figure 3.5. An example of Thai word lattice



Let exemplify as statistic resource form ARPA file shows that sequence of n-gram probability of the number 1 path is the highest comparing to the other, then the number 1 path will be selected from the 4 alternatives.

3.5 tuning up process

This module takes a selected lattice as an input. It is possible to have unknown word, such as proper name, abbreviation, etc., in a sentence that cannot be matched by any words in a word list or can partially be matched. With this case, there are some unwanted single characters or symbols, which definitely are not a word, existing in the selected lattice. This problem can be solved by combining those characters to preceding word or succeeding word by using combining rules. Currently there are 9 combining rules. The details of the combining rules are described below.

1. If it is a dot "." and it succeeds any Thai word, it will combine together.
   - Example: |ก|.| --> |ก.|
2. If there are blanks surrounding a single character which is not "ธ" or "ฌ" or "บ่", the blanks are combined with the character as a word.
3. If there is a single character with a blank before it and a word after it, the character is combined with the following word.
4. If the word contains one of "()[]{}<>|\" at the front or last character, these symbol is split to another word.
   - Example: |(นก| --> |(|นก|, |นก)| --> |นก|)|
5. If there is a single consonant with THAI_CHARACTER_THANTAKAT after any word, combine those together.
   - Example: |กา|ร์| --> |การ์|
6. If there is single consonant and last character of the previous word is not THAI_CHARACTER_SARA_A or blank, this single consonant will be combined with previous word.
   - Example: |มา|ร| --> |มาร|
7. If there is any Thai word or character and the last character of the previous word is THAI_ CHARACTER_MAITAIKHU but is not a word "ก็", these words are combined together.
   - Example: |เก็|บ| --> |เก็บ|
8. If there is any Thai word or character and the last character of the previous word is THAI_ CHARACTER_ SARA_UEE, these words are combined together.
   - Example: |เกื|อบ| --> |เกือบ|
9. If there is any Thai word or character and the last character of the previous word is THAI_ CHARACTER_ SARA_UE but is not a word "รึ", these words are combined together.
   - Example: |บึ|ง| --> |บึง|

These rules tune up an output of the system to become more accurate.

3.6 Sting Generation Module

An output of the tuning up process is word-segmented; however, it is in lattice form. To show an output, system requires post-processing module to transform a lattice into a sting. With a sting, the system can display a word-segmented output or transfer the output to MT system to translate it.

Chapter 4

Thai-English Dictionary

In this chapter, we explain how we manage our Thai-English parallel lexicons and their format to be a source-data of machine translation system as dictionary. Dictionary is necessary data for rule-based MT.

Since Thai and English, as well as other language pairs, belong to different language typologies, translation rules for rule-based MT require fine-grained linguistic information This makes translation rule formation for translating Thai to English hard to describe accurately, especially the special cases in translation such as different orderings, different verb arguments, no plural inflection, no inflection to express voices, tenses, and aspects, etc. However, these issues can be resolved by providing sufficient information in the lexical entries in a dictionary. To resolve the translation issues, lexicons are necessary to be annotated both syntactically and semantically to provide sufficient information for English generation.

We aim to construct lexical information to serve Thai to English rule-based machine translation. To gather the lexicon and its information, we estimate the different types of usage and provide sufficient information to each lexical entry. Implicit attributes of function words are also included to the framework to automatically derive grammatical information from Thai sentences.

According to major purpose, we classify linguistic information into three types as semantic concept, syntactic information, and lexical attribute. Each of linguistic information has different purposes for MT. First, semantic concept is the meaning of word which assists MT to select appropriate equivalent translation. Second, grammatical expression, annotated with Categorial Grammar (CG), helps MT to reduce misusing grammatical usage and illustrates lexicon's syntactic behavior. Last, lexical attribute is a property belonging to or characteristic of a word. Lexical attribute indicates essential syntactic feature information for each word, especially function words which are able to convert attributes of other content words into their own attribute. This tunes the translated result to become more accurate in specific cases.

To be an input of MT, we split up files according to different MT system utilization. The files are separated into very different types of their behaviors and attributes. There are 15 main lexical types listed below with their sub-types.

1. noun
    1.1. ordinary noun
    1.2. military domain noun
2. pronoun
    2.1. personal pronoun
    2.2. relative pronoun
3. verb
    3.1. verb
    3.2. attributive-verb

4. adjective
5. adverb
    5.1. adverb
    5.2. sentential adverb
6. preposition
7. conjunction
8. comparer
9. classifier
    9.1. unit classifier
    9.2. collective classifier
10. auxiliary verb
    10.1. future tense auxiliary verb
    10.2. pass tense auxiliary verb
    10.3. progressive aspect auxiliary verb
    10.4. passive voice auxiliary verb
    10.5. perfect tense auxiliary verb
    10.6. pass tense auxiliary verb
    10.7. modality auxiliary verb
11. determiner
12. particle
13. question marker
    13.1. yes-no question marker
    13.2. wh-question marker
    13.3. quantitative question marker
14. negator
15. special type

Totally, we have 28 files, and each of them contains different role of usage differently. In the next section, semantic concept information will be explained in depth.

4.1 Semantic concept

Semantic concept is an intermediate meaning representation for MT to select equivalent target word.

We group Thai equivalent words into a set of synonym (aka. synonym-set or synset) stored into line with the parallel English word(s). Therefore, each line entry in the dictionary contains the synonym of the word and it has a different conceptual meaning to the other line.

For example, Thai words "*ใบอนุญาตขับรถ*", "*ใบขับขี่*", and "*ใบอนุญาตขับขี่*" have the same conceptual definition as "a license authorizing the bearer to drive a motor vehicle" and these words are stored together with their parallel English translation as "driving license". We carefully group Thai synonym-set by using our semi-automatic Thai WordNet program with linguists' approval. Moreover, our linguists select English equivalent word by choosing a translation from Gister dictionary and our electronic dictionary Lexitron. Additionally, we use comma "," to bound each word and English words are stored in root form. The examples of synonym-set are shown in Table 4.1.

Table 4.1. Examples of Thai and English Noun synonym-set

Thai Word	**English Word**
ใบกำกับสินค้า,ใบส่งของ,ใบสั่งของ	invoice
อิสตรี,ผู้หญิง,หญิง,วนิดา,หญิงสาว,นารี,นาง,สตรี	woman,adult_female
น้องชาย	little_brother
น้องสาว	little_sister
ตลาด,ท้องตลาด	market
ใบขับขี่,ใบอนุญาตขับขี่,ใบอนุญาตขับรถ	driver's_licence
ผู้กำกับเส้น	linesman

Currently, our dictionary has 193,918 Thai lexicons and 156,201 concepts. By combining Gister dictionary and Lexitron, our dictionary acquires sufficient lexicon to cover daily used in general domain. Furthermore, we add military domain dictionary to cover up a military domain translation.

After adding translation pair into the dictionary, next needful information in the dictionary that we will describe is syntactic information of the lexicon.

4.2 Syntactic information

For syntactic information annotating, we apply Categorial Grammar (CG) to express Thai lexical grammatical expression. In CG, All syntactic categories in categorial grammar are distinguished by a syntactic category identifying them as one of the following three types.

1. Argument: this type is a basic category, such as S and NP.

2. Functor (or functor category): this type is a function from arguments to one type to result another, such as S\NP and NP/NP.

3. Conjunction (or conjunctive category): this type is a conjunction that coordinates two similar categories surrounding it into one.).

CG captures the same information by associating a functional type or category with all grammatical entities. With CG, our lexical entries can be annotated with its own syntactic category. However, a lexicon could have more than one syntactic category if it is able to be used in different appearances. Conclusively, each word is assigned with a syntactic category that controls its linguistic behaviors

Let us exemplify the CG by the following simplified example.

"กิน / kin/ (to eat)" : (s\np)/np

"ช้าง / c^{h}áːŋ/ (elephant)" : np

"กล้วย / kluâj/ (banana)" : np

The notation α/β is a rightward-combining functor over a domain of α into a range of β. The notation α\β is a leftward-combining functor over β into α. α and β are both syntactic categories. In example, three words are defined as syntactic categories. The words "ช้าง (elephant)" and "กล้วย (banana)" are defined as a primitive category "np". The word "กิน (to eat)" is de-fined as "(s\np)/np", a function that takes two arguments which both of them are np's; the first one from the right side, and another one from the left side. The dependency analysis of the sentence "ช้าง กิน กล้วย (elephant eats banana)" is illustrated in Figure 4.1. First, the derivation begins by forward combination from (s\np)/np to np yielding the syntactic category s\np. Then, backward combination from s\np to np is applied yielding the syntactic category s.

```
'elephant'      'to eat'       'banana'
  ช้าง            กิน            กล้วย
----------   ------------   -----------
    np        (s\np)/np          np
             --------------------------
                        s\np
---------------------------------------
                  s
```

Figure 4.1. Example of CG derivation

Since our main objective is to translate Thai to English, we though stored each line entry with Thai CG. Now, we have six primitive syntactic categories for Thai: `np`, `s`, `num`, `spnum`, `ws`, and `X`. The ***3*** primitive `np`, `num`, and `spnum` can be tagged to lexical entry directly, but the ***3*** other basic categories produce other function categories. Their examples and descriptions are shown below in Table 4.2.

Table 4.2. Thai primitive syntactic categories

Syntactic category	**Definition**
	Examples
`np`	A word that serves as the subject or object
	ช้าง 'elephant', ผม 'I'
`s`	A sentence
	ช้าง กิน กล้วย 'elephant eats banana'
`num`	A number
	สิบ 'ten', 11 'eleven'
`spnum`	A number which is succeeding to classifier instead of proceeding classifier as ordinary number
	นึง 'one', เดียว 'one'
`ws`	A specific category for Thai. It is assigned to a sentence that begins with Thai word ว่า 'that' (sub-ordinate clause marker).
	ว่า 'that' (sub-ordinate clause marker)
`X`	An undefined category that takes the same categories from the left or right sides and produces the taken category.

Form above primitive categories, Thai words' syntactic information is generated accordingly, and Thai grammatical expressions are stored with the word. Their examples are shown below in Table 4.3.

Table 4.3. Examples of Thai categorial grammar

Thai Word	**English Word**	**CG**
ใบกำกับสินค้า,ใบส่งของ,ใบสั่งของ	invoice	`np`
ผู้หญิง,อิสตรี,หญิง,วนิดา,หญิงสาว,นารี, นาง,สตรี	woman,adult_female	`np`
ใบขับขี่,ใบอนุญาตขับขี่,ใบอนุญาตขับรถ	driver's_licence	`np`
ตาย,เสียชีวิต,สิ้นชีพ,ถึงแก่กรรม,ถึงฆาต,สิ้นอายุขัย, สวรรคต,สิ้นชีพิตักษัย,ถึงอนิจกรรม,ถึงแก่อนิจกรรม,ถึงอสัญกรรม,ถึงแก่อสัญกรรม,สิ้นลม,สิ้นอายุขัย,สิ้นใจ,...	die, pass_away, kick_the_bucket,cash_in_one's_chips, drop_dead,decease,perish, …	`s\np`
ข่มขวัญ,ข่มขู่,ขู่,ขู่เข็ญ	threaten	`s\np,s\np/np, s\np/ws/np, s\np/ws`
จะ,จัก	will	`(s\np)/(s\np)`
ใน,ภายใน	in	`np\np/np`

และ	and	X\X/X

In conclusion, we currently obtain 27 syntactic categories annotating with the word in total. The last information that we provide in the dictionary is lexical attribute. This is very unique information of the lexicon that will change the translation output to become more sound and smooth. Its details will be revealed in the next section.

4.3 Lexical attribute

Lexical attribute represents specific properties which are externalized from any words. For example, Thai word *เสื้อผ้า* /sɯâ˙p^{h}âː/ ‘cloth’ has a plural sense underlying in its lexicon without any plural marker. Lexical attributes have a major role for function word since they can be embedded into its content word. For example, Thai function word *ทั้งหลาย* /t^{h}áŋ˙lăːj/ (plural marker) has a plural attribute. When this word combines with Thai content word *ทหาร* /t^{h}á˙hâːn/ ‘soldier’, the plural attribute is embedded from function word into content word and generates *ทหาร ทั้งหลาย* /t^{h}á˙hâːn/ /t^{h}áŋ˙lăːj/ to ‘soldier**s**’.

We design different attribute set to serve different English generation as each word class requires alternate attributes. These attributes help MT’s transfer rule to fill up implicit information to English translation such as verb arguments, plural inflection, inflection to express voices, tenses, and aspects, etc. In this project, we classify Thai lexical attributes on POS of English word since their major task is to tune the translated result to become more accurate in specific cases. However, the content words mostly have their attribute value for each attribute set as “unknown”

To obtain and assign different attribute to lexicon, we categorized all words, based on semantic criteria which is generalized and analyses from corpus. We then found the attribute sets and their values and carefully tagged to each lexical entry.

For content word noun and function word that integrates noun or refers to noun, there are 5 unique attribute sets as: 1.number 2.person 3.gender and 4.definiteness**.** Each of attribute set and its own value are shown below in the table 4.4.

Table 4.4. Attribute sets and their values with description

Attribute set	**Attribute value**	**Description**
number	unknown	This value is marked for usual content noun. Its default value when there is no function word integration is singular.
	singular	This value is marked to show singularity of the word.
	plural	This value is marked to show plurality of the word. This will help the system to not add article “a/an”.
person	unknown	This value is marked for usual content noun. Its default value when there is no function word integration is third person.
	first person	This value is marked to show that the word is first person.
	second person	This value is marked to show that the word is second person.
	third person	This value is marked to show that the word is third person.
gender	unknown	This value is marked for usual content noun. Its default value when there is no function word integration is neuter.
	masculine	This value is marked to show that the word is masculine.
	feminine	This value is marked to show that the word is feminine.

	neuter	This value is marked to show that the word is neuter.
definiteness	unknown	This value is marked for usual content noun. Its default value when there is no function word integration is indefiniteness.
	definiteness	This value is marked to show that the word is definiteness. This will help the system to add article “the”.
	indefiniteness	This value is marked to show that the word is indefiniteness. This will help the system to add article “a/an”.

For content word verb, attributive verb, adjective, and function word that integrates verb, attributive verb, and adjective, there are 3 different attribute sets as: 1.tense 2.direction 3.modality type and 4.causative**.** Each of attribute set and its own value are shown below in the table 4.5.

Table 4.5. Attribute sets and their values with description

Attribute set	**Attribute value**	**Description**
tense	unknown	This value is marked for usual content word. Its default value when there is no function word integration is present tense.
	present	This value is marked to show that the word or the integrated word is inflected by present tense form.
	past	This value is marked to show that the word or the integrated word is inflected by past tense form.
	future	This value is marked to show that the word or the integrated word is inflected by future tense form.
direction	unknown	This value is marked for usual content word. Its default value when there is no function word integration is no direction.
	inward	This value is marked to show that the word or the integrated word is inflected by inward direction to speaker.
	outward	This value is marked to show that the word or the integrated word is inflected by outward direction from speaker.
	X	This value is marked to show that the word or the integrated word is not inflected by direction.
modality type	unknown	This value is marked for usual content word. the default value when there is no function word integration is none.
	progressive	This value is marked to show that the word or the integrated word is inflected by progressive aspect.
	perfect	This value is marked to show that the word or the integrated word is inflected by perfect aspect.
	can	This value is marked to show that the word or the integrated word is previously inserted by can.
	could	This value is marked to show that the word or the integrated word is previously inserted by could.
	have to	This value is marked to show that the word or the integrated word is previously inserted by have to.
	may	This value is marked to show that the word or the integrated word is previously inserted by may.
	must	This value is marked to show that the word or the integrated word is previously inserted by must.

	should	This value is marked to show that the word or the integrated word is previously inserted by should.
	would	This value is marked to show that the word or the integrated word is previously inserted by would.
	passive	This value is marked to show that the word or the integrated word is changed to passive voice form.
	X	This value is marked to show that the word or the integrated word is not inflected by modality.
causative	Yes	This value is marked to show that the word is causative and is able to cause following action
	X	This value is marked to show that the word is not causative and is not able to cause following action. This value is also a default value for most words.

However, adjective and attributive verb have another attribute set as same as adverb which is comparative set**.** This attribute set and its own value are shown below in the table 4.6.

Table 4.6. Attribute sets and their values with description

Attribute set	**Attribute value**	**Description**
comparative	yes	This value is marked to show comparative function. With this attribute marking, adjective, adverb, and attributive verb will transform to comparative form.
	no	This value is marked to show non-comparative function. It is also a default value to adjective , adverb, and attributive verb.
superlative	yes	This value is marked to show comparative function. With this attribute marking, adjective, adverb, and attributive verb will transform to superlative form.
	no	This value is marked to show non-superlative function. It is also a default value to adjective , adverb, and attributive verb.
simile	yes	This value is marked to show simile function. With this attribute marking, adjective, adverb, and attributive verb will be translated to simile form such as “as high as”, “as fast as”.
	no	This value is marked to show non-simile function. It is also a default value to adjective , adverb, and attributive verb.

With the attribute value marking to lexicons, the system will use it to tune up the translation word and affect the translation result to become more accurate

4.4 Limitation and Solution

With the main idea in the chapter 2, we construct lexicons and their information to be a dictionary. However, the dictionary has limitation in utilization.

4.4.1 Multi-sense word

As other languages, there are many multi-sense words in Thai. These words have the same surface form but they have different meanings and different usages.

Currently, the different usages can be separated because the mark of syntactic information. For example, Thai word "*ขโมย*" can be can be used to refer to noun as a **thief** and it is marked as **np**, and this word can also be denoted an action which means **to steal** and it is marked as **s\np/np**.

For the multi-sense words that have only different meaning (homograph), we can separate the words into different synonym-set. For example, Thai word "*เงิน*" can be used to refer to 3 senses which are **money**, **silver** (metallic element), and **silver** (color). However, The MT system cannot decide the right meaning for different usages in the input sentence. Therefore, when the input sentence contains the word "*เงิน*", the MT system will choose randomly one of them and the output is possibly incorrect.

To resolve this issue, we pretentiously choose the most frequent used meaning to be the major translation of the word by marking the "#" symbol in the front of word. This will help MT to choose often used translation. However, this method would unfortunately disregard appropriate translation for not widely distributed and uncommon cases.

4.4.2 Long distance and discontinuous pattern

Long distance and discontinuous pattern in Thai is a major issue for Thai NLP. The example is shown below in Figure 4.2.

input:	เขา	ซื้อ	เสื้อ	ให้	แม่	3	ตัว
tran:	he:pron	buy:verb	shirt:noun	to:prep	mother:noun	3:num	classifier(shirt)
lit:	He buys 3 shirts to mother						

Figure 4.2. Example of *long distance and discontinuous pattern*

From the above example, the number and classifier of the noun “shirt” are very far away from their core noun. With the current CG formalism, syntactic annotation only shows the surrounding requirement. Therefore, in this case, the number and classifier can only be treated as a modifier of the previous noun. Unfortunately, the translation of this sentence will be “He buys shirt to 3 mothers”.

With the CG formalism, the long distance and discontinuous pattern is still remained as a limitation of the function-argument relationship. This phenomenon cannot be solved with the current dictionary version.

Our dictionary contains three major types of linguistic information for MT resource. First, semantic information, equivalent Thai-English translation, will be used as word replacement in MT. Second, syntactic information will be used as data for parsing input sentence into a tree for MT. Last, lexical attribute, specific property relying in the word, will help MT tuning translation output to become much more accurate and grammatically correct. The mentioned information will separately be used in the MT system which will be explained in details in the following chapter.

Chapter 5

Machine translation system

Cobra Gold MT System is a rule-based machine translation using dependency structure as the syntactic representation. It is underlain by three rudimentary operations: analysis, transfer, and generation, respectively. The overview of the system is illustrated in Figure 5.1. The system consists of three major modules: Thai analyzer module, Thai-English transfer module, and English generator module.

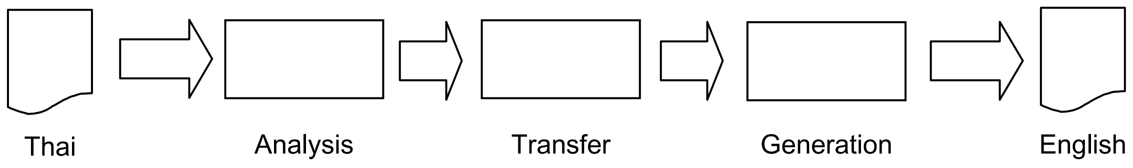


Figure 5.1. System overview

The Thai analyzer module interprets Thai raw text into the syntactic representation. First, an input sentence is required to have been word-segmented earlier according to Chapter 2. The input sentence is recognized for Thai number expressions that can be either written numerically or spelt out. The sentence is then parsed from left to right by grammar rules. The grammar used here is based on categorial grammar explained in Section 4.2. The output of this module is a set of dependency structures that reveal head-modifier relationship among words.

The Thai-English transfer module converts a Thai dependency structure into an equivalent English one using the semantic concept pair describe in Section 3.1. This module commences by taking an input dependency structure. In the system, the structure is analyzed and each relationship is taken into account to annotate each word with syntactic features collecting as lexical attributes showing in Section 3.3 such as tense, number, and person. The lexical attributes are necessary for English generation of agreements. Then, some nodes are reordered, inserted, or deleted. Finally, all leaf nodes of the structure are extracted to form a list of English root forms. The list does not contain only surface forms of the node but also lexical attributes that will be playing a crucial role in the next module.

The English generator module synthesizes an English sentence from a list of English root forms. This module begins by taking an input list of root forms. It consults the inflection table for each root form and its lexical attributes to find the right inflection. Then the sentence is analyzed for article insertion. Each noun phrase is memorized in the discourse stack. The final output of this module is an English sentence with correct articles.

To cover two perspectives of use, the system was designed to work in two environments. The system can run as a single independent process without standing by for user interaction, called standalone mode. In this mode the system takes an input file in raw text format with sentences separated by new line, and then translates each sentence. In the other mode called server mode, the system stands by waiting for incoming connections from the designated port and responses each incoming message with appropriate manner. A specialized protocol was designed for ease of manipulating the system online.

In this section, the anatomy of the system will be described in order. First the parser module will be described in Section 5.1. Then the transfer module will be explained in Section 5.2. The generation module will be described in Section 5.3. The post-processing module will be described in Section 5.4.

5.1 Parser Module

The parser module analyzes an input sentence into dependency structure with a specified grammar. In this section the insight of the parser module is explained. It begins the overview of the module. Then it introduces the formal grammar used in the system, or categorial grammar, and the resulted syntactic representation, or dependency structure. Finally, parsing techniques of the module are described.

5.1.1 Overview of The Parser

The parser module follows rule-based approach. Categorial grammar was selected as the formal grammar that produces a dependency-based syntactic representation. The overview of the parser module is illustrated in Figure 5.2. The parser module takes a list of words as its input. It consists of three sub-modules: a numeral expression chunker, a GLR parser, and a chart parser. Finally it produces a dependency structure as an output.

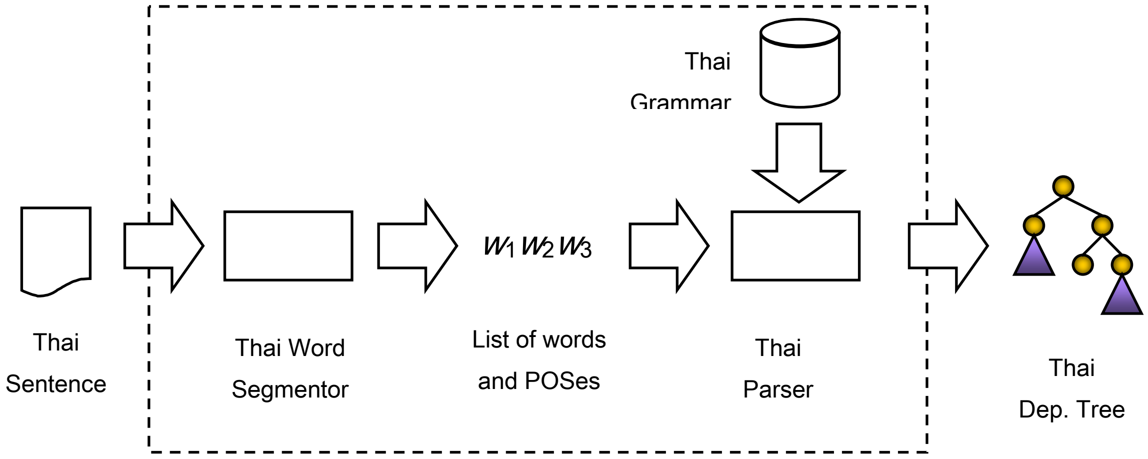


Figure 5.2. Analysis Module

The numeral expression chunker, as self-expressed, recognizes expressions of numbers, either written numerically or spelt out. It makes use of a finite-state automaton to detect a fragment of text as a number. Since the patterns of Thai numbers are quite deterministic, they were therefore manually transformed into an automaton, coded, and compiled. Once a numeral expression is detected, it is then assessed to its value and its grammatical plurality (singular or plural) for ease of further computation. The result of the chunker is still a list of words with numeral expressions denoted by an ordered pair (v, f), where v is its value and f is its syntactic feature structure.

Once numeral expressions are entirely detected, the parser then detects unknown words ac-cording to its lexicons. It then determines the parsing technique to be used for parsing the input sentence. If the sentence contains no unknown words, or noises, the GLR parsing technique is applied for speed purpose. Otherwise, the chart parsing technique is used for noise tolerance purpose. In case that the GLR technique fails to parse the sentence, the chart technique is then switched to continue the parse.

Throughout the parsing process, the CG is utilized to control syntactic behaviors of each lexicon. The next section will describe how to parse this formal grammar.

5.1.2 Parsing Techniques

Two different parsing techniques are used for different perspectives of processing. As stated in Section 3.1.1, the techniques are selected by considering the amount of unknown words in an input sentence. If there are no unknown words, the GLR technique is selected first; otherwise, the chart technique is selected. In case that the GLR technique fails to parse, the system switches to the chart technique instead. This section will compare the two methods in terms of efficiency and noise tolerance.

GLR (abbreviation of ‘Generalized Left-to-Right’) is an almost nondeterministic parsing technique extended from LR algorithm to handle syntactic ambiguity. The technique is based on shift-reduce parser in which the parser is a stack-oriented pushdown automaton. As an input symbol is consumed, the parser moves from a state to another and performs a stack operation. The stack operations of the GLR algorithm are classified into two operations: shift and reduce. A shift operation saves the current state to the stack and then moves to a new state. A reduce operation pops away topmost stack symbols and pushes a new one with respect to the production rules. These operations are described in the parsing action table, which is generated from the state transition analysis of the production rules (Aho 1970). The term ‘almost nondeterministic’ accounts for the fact that the parser deals with syntactic ambiguity, called action conflict, by means of pseudo-backtracking as old states are saved in the stack. This parsing algorithm is speed-efficient and suitable for parsing long sentences, because most of ambiguities in state transition are eradicated in the state transition analysis resulting in the speed of O(n2) in the best case and O(n3) in average and the worst cases. However this method cannot handle unknown words and this leads us to the awkwardness of informing failure after the parser attempts to parse in any alternatives.

The other parsing method used in the system is the chart parsing. This technique is based on Dynamic Programming, which considers parsing as optimization problem and sub-trees as overlapping sub-problems and optimal sub-solutions. The algorithm makes use of graph-like data structure called chart to represent all possible constituencies constructed from an input sentence. In the chart each word of the input sentence is represented as edges connected from the start node to the final node. Then each word is attached with its all-possible parts of speech, again as edges. Each fragment of consecutive edges is merged into a new edge covering them with respect to the production rules. This parsing algorithm is noise-tolerant because it can skip unknown words resulting in partial parsing of the sentence. The speed of the algorithm is constantly O(n3) both in the best and the worst cases.

In order to deal with natural language input sentences that are lengthy and perhaps contains unknown words, a hybridization of the two methods becomes necessary. This results in the following criteria. If an input sentence does not contain unknown words, the GLR parser will first try parsing this sentence. If successful, the parser yields out the parse trees. Otherwise the chart parsing will be used. In case that the sentence contains unknown words, the chart parsing will take care of it from the beginning.

In translation from Thai into English, a Thai sentence is given to the parser to produce a set of Thai dependency structures. However, only the first element of the set will be used for translation. The next section will discuss the transfer module, which converts a Thai dependency structure into an English-like one.

5.2 Transfer Module

The transfer module converts a Thai dependency structure into an English one. It takes an input dependency structure and then annotates syntactic attributes to each word according to analysis of head-modifier relationship. Some nodes are then reordered, inserted, or deleted from the tree to imitate English grammar. Finally, all leaf nodes of the structure are extracted to form a list of English root forms. This section will give some important details of the module focusing on transferring Thai tree into English one. The process of the transfer module is illustrated in Figure 5.3.

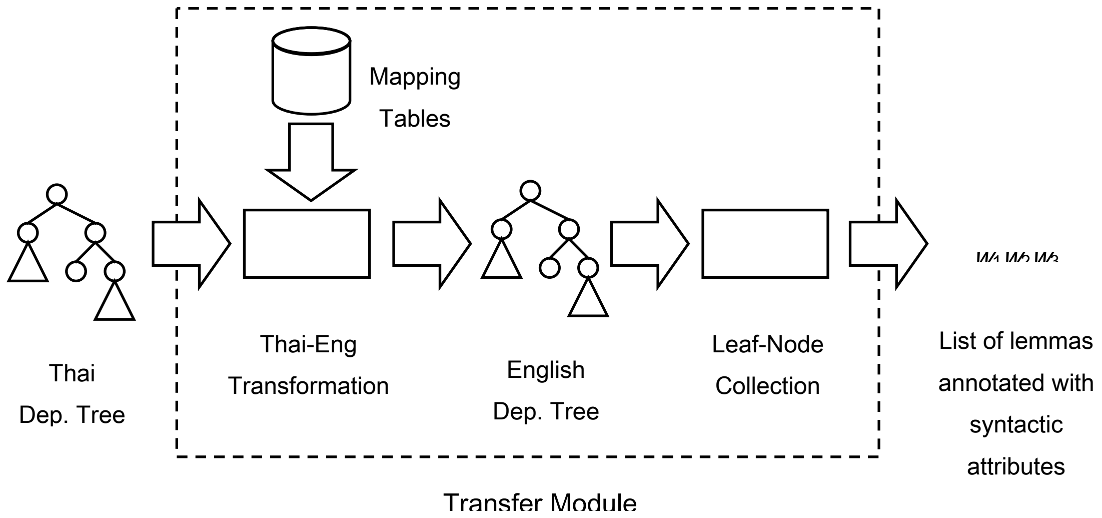


Figure 5.3. The process of the transfer module

5.2.1 Tree Pattern

Rules for the transfer module are defined over tree patterns. A tree pattern is denoted by a constituency of words or variables for matching with a given tree. If the words or the variables are annotated with the star notation to denote that they match only to a leaf node. Otherwise they will be matched up with the head of the tree at any depth. For example the tree pattern W1 < *W2 is matched with the following tree in Figure 5.4.

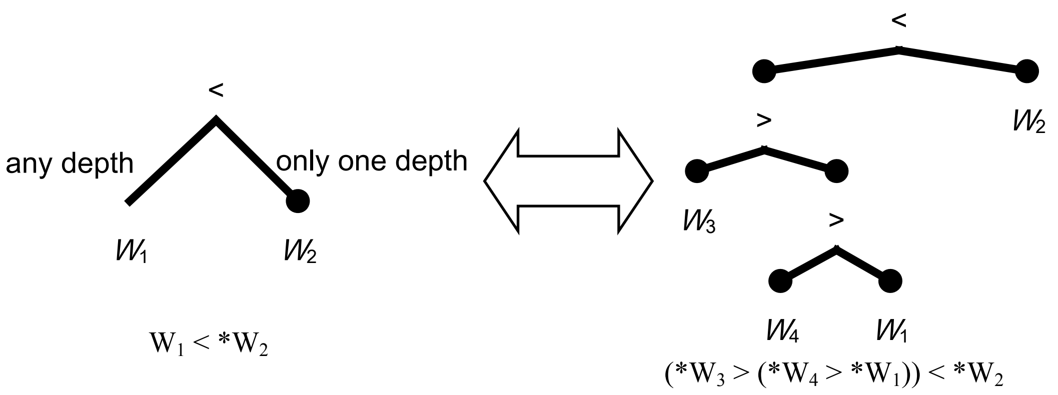


Figure 5.4. Matching of tree pattern to a dependency tree

Besides the star notation, syntactic attributes can be also assigned to each word by the bracket form W[feat1 : val1, feat2 : val2, …]. All four types of transfer rules are written in terms of tree patterns TreePat1 → TreePat2, where a tree pattern TreePat1 matches a given tree into variables and transforms it into the other side TreePat2.

5.2.2 Transfer Rules

Transfer rule plays an important role in the transfer module as a means to control conversion of dependency structure. Transfer rules can be classified into four types as follows.

1. **Reordering:** this type of rules relocates constituents in the Thai tree to imitate word order of English as closest as possible.
2. **Attribute assignment:** this type of rules assigns English syntactic attributes to the reordered Thai tree. This is very fruitful for the generation module in that grammatical agreements can be easily provided by consulting the grammatical features assigned to each word.
3. **Insertion:** this type of rules inserts some constituents into the tree to complete the English meaning.
4. **Deletion:** this type of rules deletes some constituents from the tree since they are unnecessary to translate into English.

The linguistic diversity between Thai and English can be narrowed down by each type of transfer rules as tabulated as follows.

Table 5.1. List of type of transfer rule

Types of transfer rules	Linguistic operations
Reordering	Constituent reordering, collective restructuring, VP structure selection, sentence structure selection

Attribute assignment	Number assignment, tense/aspect assignment, adjectivization, possessivization, participialization, adverbialization
Insertion	Possessive insertion (-'s), 'of' insertion, appositivization, infinitivization
Deletion	Classifier dropping, serial verb fusion

The steps of Thai-English transfer are as follows. First a given dependency structure is reordered by reordering rules. Then syntactic attributes are assigned to each word by means of attribute assignment. Then insertion and deletion rules take the responsibility to insert necessary parts to complete English meaning and delete unnecessary parts in English translation. Finally all leaf nodes are extracted to form a list of English root forms with syntactic attributes.

5.3 Generation Module

The generation module synthesizes an English sentence from a list of English root forms. Each word, which has been already annotated with syntactic features, is inflected by consulting the inflection table. Then the list, which each of the words is inflected, is inserted articles into each noun phrase by consulting the discourse stack. The final output of this module is an English sentence with correct articles. The process of the generation is illustrated as follows.

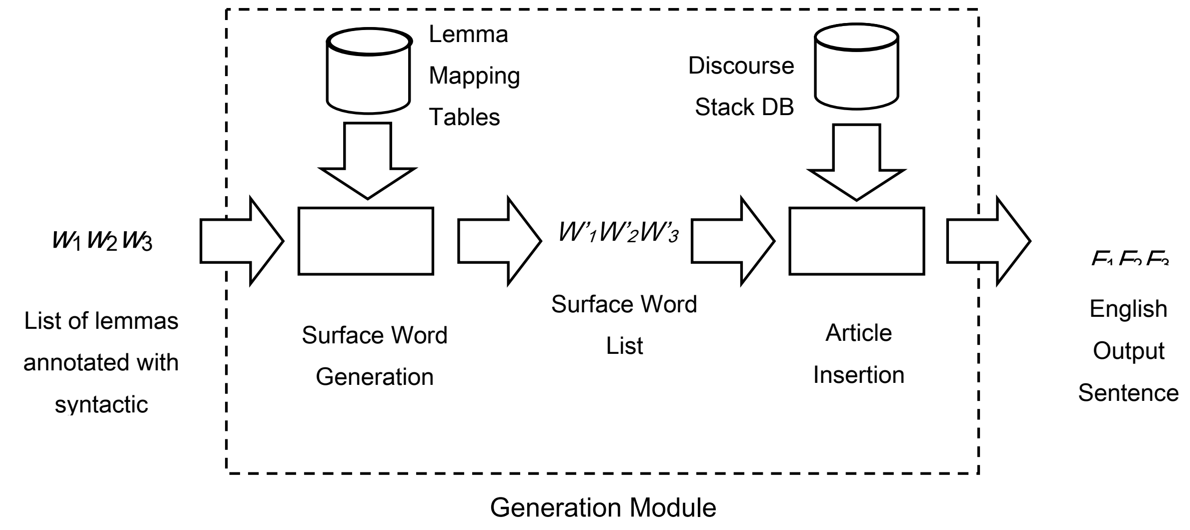


Figure 5.5. The process of generation module

The syntactic attributes used for word inflection are as follows.

Table 5.2. List of syntactic attributes used for word inflection

Parts of Speech	Inflectional Attributes
Adjective	simile, comparative, superlative
Adverb	simile, comparative, superlative
Conjunction	-

Determiner	number, case
Interjection	-
Noun	number
Particle	-
Prefix	-
Preposition	-
Pronoun	person, number, case
Verb	person, tense, number, finiteness

5.4 Post-processing Module

As soon as a Thai parse tree is transferred into a quasi-English one, the article insertion algorithm is activated to accommodate articles in the output English sentence. The key concept of this algorithm is to find ranges of noun phrases (abbreviated NP) and insert an appropriate article in front of them. The main algorithm is given below. The algorithm of article insertion is described as follows.

```
algorithm insert_article(L : list of words): list of words
begin
    let R <- []      { result }
    let i <- |L|     { word index }
    while i > 0 do
        if L[i] is a noun then
            let (def, num) <- the number and the definiteness of
L[i]
            let (np, i') <- recognize_np(L, i)
            if L[i'] is not a determiner then
                let first_word <- L[i']
                let art <- generate_article(first_word, def, num)
                append concatenate(art, np) to R
            else
                append np to R
            end if
            let i <- i'
        else
            append L[i] to R
            let i <- i - 1
        end if
    end while
    return R
end algorithm
```

The algorithm insert_article, the core of the article insertion procedure, can be described as follows. For a list of words L, there is a loop that seek for NPs contained in L from the right side. If it finds a noun, it then expands the range of this NP to the left by the algorithm "recognize_np" covering all nouns, adjectives, numbers, and adverbs, except the determiners. Then it checks if the word preceding the NP is a determiner: if so then it is unnecessary to generate an article for this NP; otherwise, generate one for this NP by the algorithm "generate_article". The loop continues until it reaches the beginning of L.

In order to expand the range of a NP, the algorithm recognize_np takes this responsibility. For this algorithm, a list of words and the index pointing to the last noun are given. The procedure of recognize_np is provided below.

```
algorithm recognize_np(L, i): list of words * int
begin
    let R <- []      { result }
    let i' <- i      { the end of this NP }
    while i' > 0 do
        if L[i'] is either a noun, an adjective, a number, or an
adverb then
            append L[i'] to R
            let i' <- i' - 1
        else
            break
        end if
    end while
    return (R, i')
end algorithm
```

The algorithm recognize_np can be explained as follows. From the given index i, as long as the algorithm finds the word at i is either a noun, an adjective, a number, or an adverb, the algorithm continues to expand the range to the left. Otherwise the algorithm terminates and returns the range of the NP and the index pointing toward the word preceding this NP.

The algorithm generate_article generates an article for a NP by taking the first word --- as well as the definiteness and the number of the core noun --- into account. The procedure of generate_article is listed below.

```
algorithm generate_article(first_word, def, num)
begin
    if def is definite then
        return 'the'
    else if def is indefinite then
        if num is singular then
            if the first character of first_word is a vowel then
                return 'an'
            else
                return 'a'
            end if
        else
```

```
            return ''
        end if
    end if
end algorithm
```

The procedure of generate_article can be described as follows. If the core noun exhibits its definiteness, the article 'the' is generated. Otherwise if the number is singular then generate the article 'a' or 'an' regarding the first character of the first word. Otherwise, it does not generate any article.

Chapter 6

Conclusion

Cobragold machine translation system is a Thai to English automatic rule-based machine translation for military purpose. The system consists of three main parts; Thai word-segmentation, lexical bi-lingual resource, and translation system.

The system takes Thai text as an input. Firstly, Thai text must be word-segmented since Thai has no real word boundary and is ambiguous for machine. Then the word-segmented output is sent to translation system. The system begins with parser and it will parse a word-segmented text into a grammatical tree using syntactic information from lexical dictionary. With tree representation, each Thai content word is transformed to equivalent English one, and Thai grammatical word is transformed to English inflectional usage. Then the system will generate all items according to English grammar into English sentence including articles, plural inflection, etc.

This system is capable to be improved by adding more resources. In this version, there are 193,918 lexical entries as a resource. However, word segmentation system's outputs are not matched to a word in a bilingual dictionary and that causes MT system unable to translate Thai word and marking it as unknown word. In the future, we plan to add more words and synchronize the learning set of word segmentation system with data in a bilingual dictionary which can help to increase accuracy of the translation system. Moreover, we plan to research Thai usage to add rules to effectively parse complex sequence of Thai word-order such as a combination of auxiliary verbs usage, serial verb construction, etc.

Previously, CERDEC's old translation system used a word translation system that translates Thai to English by matching words in a sentence directly. Cobragold machine translation system satisfies them since its translation result is obviously more natural and correct. Comparing to the old system, the new system has more features:

- Article insertion
- Plurality inflection
- Tense inflection
- Word ordering correction

With these new features, translation result is naturally improved therefore CERDEC satisfies and accepts the system. For stability, this system is built and provided on web service. Thus, this system is sufficiently stable.

We test the improvement of the system by sending 200 Thai input sentences to the system. For examining the better translation result, we compare our translation result with the old version system, i.e., direct word by word translation system. The comparison result is shown in Table 6.1 below.

Table 6.1. comparison result for translatable sentence

	new system		old system	
	amount	percentage	amount	percentage
translatable	192	96	180	90
untranslatable	8	4	20	10

From above table, our new system can translate 96 percents of input sentences and the old system can translate 90 percents. The 4 percents of untranslatable sentence come from an unknown word which system never recognizes previously such as name entity.

For better translation comparison, we compare 180 outputs that both systems can translate and find out that all 180 translatable sentences have a better translation result for 4 cases; article insertion, plurality inflection, tense inflection, and word ordering correction. The details of better translation are shown in Table 6.2.

Table 6.2. better translation for each case

Better case	Sentence	Percentage
Article insertion	172	95.56
Tense inflection	152	84.44
Plurality inflection	47	26.11
Word ordering	165	91.67

The example of two systems result is shown below in Table 6.3.

Table 6.3. the example of the translation result comparing with the old version

Input1	[เด็ก, ขโมย, สมุด, จาก, โรงเรียน]
Machine translation	the child steals the book from the school.
Word translation	child(ren), thief, notebook, from, school
Input2	[นักเรียน, ชาย, ชอบ, เล่น, ฟุตบอล]
Machine translation	a male student likes to play a soccer.
Word translation	student, man, like, play/gamble, football
Input3	[เขา, จุด, เทียน, 2, เล่ม, และ, ธูป, 3, ดอก]
Machine translation	he lights 2 candles and 3 joss sticks.
Word translation	he/him/his, (a) point/light, candle, 2, unit, and, joss stick, 3, flower
Input4	[หนังสือ, ที่, อยู่, บน, โต๊ะ, เปียก]
Machine translation	the book that is on a table is wet.
Word translation	book, that/at/place, am/is/are (in/at), on, table/desk, wet

In conclusion, the new translation system can translate better for all translatable sentences. The translation result is more natural and correct.