Chapter II

A dependency grammar for Thai

In Melčuk's model of grammar (1988), he described the distinction between three major types of syntagmatic dependency that link wordforms of a sentence in any language: morphological dependency, syntactic dependency and semantic dependency. These dependencies are relations of wordforms at different levels of representation. Morphological dependency is not included in this study because Thai, being an isolative language, has almost no morphological process. Therefore, only two types of dependency will be covered: syntactic dependency and semantic dependency, which will be called conceptual dependency in this study.

This chapter describes a dependency grammar for Thai which is used for analyzing syntactic dependency and conceptual dependency. The model of grammar used in this study will be described first. Then the details of each component used in the model will be described later.

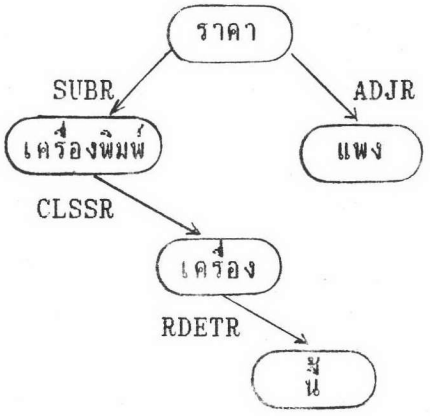
2.1 Model of dependency grammar

The model of grammar adopted consists of three important components: a syntactic component, a conceptual component, and a dictionary.

The syntactic component consists of syntactic cases, which define syntactic dependency relations of various wordforms, and priorities of relations, which dictate the strength of bond between wordforms. A syntactic representation yielded by the syntactic component is a dependency tree, or a D-tree. An example below is the

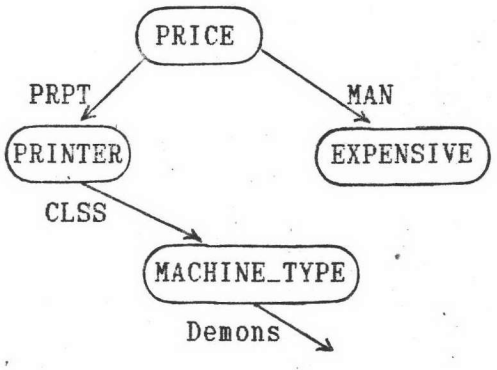
dependency tree of the sentence "เครื่องพิมพ์ เครื่อง นี้ ราคา แพง".

(1) a)



The conceptual component consists of conceptual cases, which define dependency relations of concepts, a conceptual hierarchy, mapping between syntactic cases and conceptual cases, and constraints on conceptual cases. The input of the conceptual component is a D-tree and the output yielded is a conceptual network. The conceptual network yielded from the sentence "เครื่องพิมพ์ เครื่อง นี้ ราคา แพง" is as follows:

(1) b)



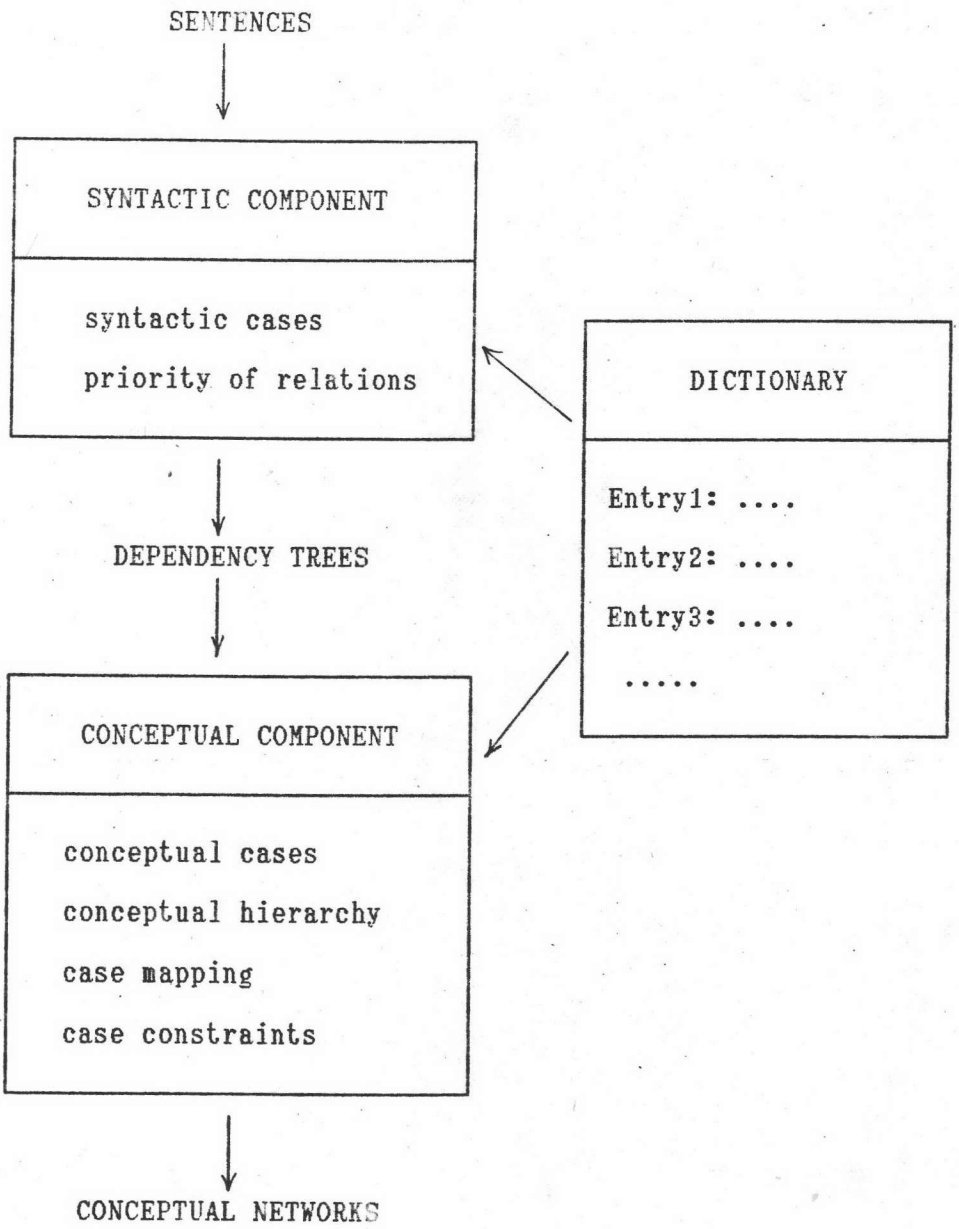
The dictionary or lexicon is an inventory of all the lexemes in the language domain. Each lexeme contains information, such as category, concept, and conceptual attribute, needed for both syntactic and conceptual components. The information is represented as features and values (see 2.4). Each lexeme together with all its features and values is an entry in the dictionary. Following is an example of the entry of wordform "ราคา"

(2) [ราคา]

MORPH:ราคา. MAJCAT:N,V. MINCAT:CMNN,VEQU. CP:PRICE. UPCP:ABST, PRPT. MSUBR:PRPT. MFOBR:CMPL. CSCMPL:MONEY. MPOSSPR:PRPT. CSPRPT: CONC. CSMAN:PRPT_STS.

The model of grammar can be summed up in the following chart.

(3)



2.2 Syntactic component

The main task of the syntactic component is to establish syntactic dependency relations, or syntactic cases, between wordforms in a sentence. Syntactic cases are assigned on the basis of category and word order. The structure that describes syntactic dependency is a D-tree.

Four aspects of syntactic dependency in this component will be discussed: dependency tree, categories for Thai, syntactic cases and criteria for syntactic case assignment, and priorities of syntactic relations.

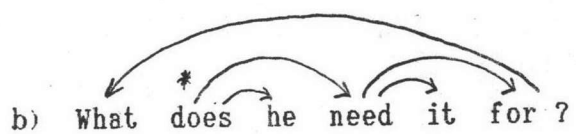
2.2.1 Dependency tree (D-tree)

A D-tree is a syntactic dependency representation. A D-tree consists of nodes and arcs linking a pair of nodes. A node of a D-tree is labeled with the morphological form of a lexeme, which in most cases is the same as its wordform in Thai. A lexeme is defined by Melčuk as a word taken in one well-defined sense or meaning. An arc of a D-tree is labeled with a syntactic case. The words which are used as clues for assigning syntactic cases, namely function words, are not represented as nodes in a D-tree. Only content words, the words representing some concepts or conceptual attributes, will be represented as nodes in a D-tree. Most importantly, a D-tree must contain one node that does not depend on any other node. This node is called the top or root node. Other nodes must depend on another node. No node can simultaneously depend on two or more nodes. Every wordform, except a function word, in a sentence must have a dependency relation to another word and that dependency relation cannot be bilateral in that the one wordform cannot be both the head and depender of another wordform.

In 1970, Robinson (cited by Hudson, 1984: 98) formulated the adjacency principle in dependency theory as follows:

If A depends directly on B (i.e. is a modifier of B), and some other element C intervenes between them (in linear order of strings), then C depends directly on A or on B or some other intervening element.

The adjacency principle ensures that a depender is never separated from its head by anything other than another depender of the same head or its own depender. This principle corresponds to the discovery by Hays and Lecerf (cited by Melčuk, 1988: 35) of an extremely important property of word order called projectivity. A sentence is called projective if and only if among the arcs of dependency linking its wordforms, no arc crosses another arc and no arc covers the top node. Most sentences in a language are projective. Sentences which are not projective are marked sentences (Melčuk 1988: 37), such as in these examples.



These sentences are non-projective sentences. Sentence (4a) has a crossing arc while sentence (4b) has an arc over the root node, "does".

2.2.2 Categorization

Categorization is the grouping of words into sets or classes according to certain criteria. Different criteria results in different categorization for the same language. The difference

between Panupong's wordclasses (1982) and Bhandhumetha's part of speech (1982) is an example. A problem in categorization lies in the fact that there is no set of criteria that can be used as common property of all members of a category. McCawley (1986: 12, cited by Taylor, 1989: 190) talked about this phenomenon of category as follows:

Parts of speech are much more like biological species than has generally been recognized. Within any part of speech, or any biological species, there is considerable diversity. Parts of speech can be distinguished from one another, just as biological species can be distinguished from one another, in terms of characteristics that are typical for the members of that part of speech (or species), even though none of those properties need be instantiated by all members of the parts of speech (or species).

Moreover, Crystal (1967, cited by Taylor, 1989: 190) stated about the gradience of categories that wordclasses usually consist of central members, which satisfy a maximum number of the criteria, and borderline members, which satisfy fewer criteria. Following Crystal's line of thought, it is possible to find words which are not full members of a category, which means that the words may exhibit membership of more than one category. For example, the word "use" exhibits the membership of both the noun and the verb categories.

2.2.2.1 Categorization for Thai

Various approaches have been adopted by various linguists in the categorization of words in Thai. Phraya Upatiksilapasan (1968), following the traditional grammar, proposed categories using semantic criteria. Panupong (1982) proposed categories using syntactic criteria. Each category is defined as a wordclass that can fill a certain slot in a pattern or frame. Bhandhumetha (1982) proposed categories using semantic criteria. She classified words in Thai into major groups, such as head word, modify

word, and relator, and then gave a semantic as well as syntactic definition for each group. These approaches, varied as they may be, do not succeed in clear cut categories. Some words still fall in the intersection of categories. For example, the word "ราคา" can be defined as noun or verb, the word "ให้" can be defined as verb or preposition, the word "อย่าง" can be defined as preposition or classifier.

In this study, category is defined as a property of a word in terms of feature and values (see 2.4). Different category values describe different syntactic dependency properties. It is possible for a word to have one or more of these category values. The word "หนังสือ" has the property of only one category, noun. The word "ราคา", on the other hand, has the value of both noun and verb. The context in which it occurs determines the value called for. The word "ราคา" in (5a) is used as a verb while in (5b) it is used as a noun.

(5) a) "หนังสือ ราคา 100 บาท" : verb

b) "ราคา หนังสือ เท่ากับ 100 บาท" : noun

Words like "ราคา" are instances of polysemy as defined by Vongvipanond (1983: 312). This differs from homonymy, in which more than one semantically unrelated words happen to have the same wordform, such as "คน" in example (6). Homonymous words are considered different lexemes. In example (6), the word "คน" has two lexemes which refer to the meaning "HUMAN" and "STIR". The first lexeme has two category values, which are noun and classifier, while the latter has only one category value, which is verb.

(6) "คน" HUMAN : noun, classifier.

STIR : verb.

Describing category in terms of features is very useful. Each category value can describe a syntactic property of a lexeme. Lexemes that undergo the same syntactic process, therefore, must have the same category value. For example, the lexemes "ราคา" in (5a) and

"อ่าน" in (7a) have the same category value, verb, because they have a subject relation to the lexeme "หนังสือ". The lexemes "ราคา" in (5b) and "หน้าปก" in (7b) have the same category value, noun, because they are used in the subject position of verb.

(7) a) "หนังสือ เล่ม นี้ อ่าน ยาก"

b) "หน้าปก หนังสือ ขาด แล้ว"

2.2.2.2 Category inventory for Thai

In this study, category represents basic syntactic information. Category information is taken as an important input for determining syntactic dependency. Category is defined in terms of two types of features: major category (MAJCAT) and minor category (MINCAT). Five major categories are proposed for Thai, using semantic criteria as follows:

1. Noun (N)

A noun is a syntactic property of a lexeme which can be used as a reference of entity that has region in some domain. It can refer to concrete or abstract entity.

2. Verb (V)

A verb is a syntactic property of a lexeme which exists only at a certain point of time. It can be used to indicate an action, a state, or a process.

3. Determiner (DET)

A determiner is a syntactic property of a lexeme which adds various aspects of information about the reference of noun. It can occur to the left and to the right of noun.

4. Auxiliary (AUX)

An auxiliary is a syntactic property of a lexeme which adds to a verb the information conveying the speaker's attitude, aspect, or time. It can occur to the left and the right of verb.

5. Relator (REL)

A relator is a syntactic property of a lexeme which is

used as a marker or a function word connecting noun and verb, noun and noun, or verb and verb.

Each of these MAJCATs can be subcategorized into MINCATs, using both syntactic and semantic criteria. The following values are proposed for MINCAT of each MAJCAT:

1. MINCAT of Noun

a. Cardinal noun (CRDN) : CRDN is a syntactic property of a lexeme which is used in counting, such as 1, 2, 3, etc.

b. Proper noun (PRPN) : PRPN is a syntactic property of a lexeme which refers to a specific entity, such as "ประเทศไทย", "สะพานพุทธฯ", etc.

c. Pronoun (PRON) : PRON is a syntactic property of a lexeme which can stand for an entity that is known, such as "เขา", "เธอ", "นี่", "นั่น", etc.

d. Classifier (CLSS) : CLSS is a syntactic property of a lexeme which is used as a unit with which a noun can be counted, such as "คน", "ตัว", "ชิ้น", "อย่าง", etc.

e. Common noun (CMNN) : CMNN is a syntactic property of a lexeme which refers to common entities which can be either concrete or abstract, such as "หนังสือ", "โต๊ะ", "ราคา", "สี", etc.

2. MINCAT of Verb

a. Common verb (VCMN) : VCMN is a syntactic property of a lexeme which is commonly used to indicate an action of an entity, such as "กิน", "นอน", "วิ่ง", etc.

b. Adjective verb (VADJ) : VADJ is a syntactic property of a lexeme which indicates a state of an entity. It usually can occur with the lexeme "กว่า", such as "ดี", "สวย", "สูง", etc.

c. Existential verb (VEXI) : VEXI is a syntactic property of a lexeme which introduces new information on the existence of an entity or event, such as "มี", "ดูเหมือน" etc.

d. Equative verb (VEQU) : VEQU is a syntactic property of a lexeme which shows that the preceding entity and the following entity have some kind of equivalent relationship, such as "เป็น", "คือ", "ราคา", "ชื่อ", etc.

3. MINCAT of Determiner

a. Left determiner (LDET) : LDET is a syntactic property of a determiner lexeme which occurs to the left of a noun, such as "ราว", "แค่", "ทุก", "ทั้ง", "กว่า", etc.

b. Right determiner (RDET) : RDET is a syntactic property of a determiner lexeme which occurs at the right of a noun, such as "นี่", "นั้น", "ก้วน", "กว่า", etc.

4. MINCAT of Auxiliary

a. Left modal (LAMD) : LAMD is a syntactic property of a lexeme which adds to a verb the information conveying the speaker's attitude in respect to anticipation on the possibility that an action or a state indicated by the verb will be carried out, and often used before LAIRL, such as "น่า", "ควร", "คง", "อาจ", etc.

b. Left irrealis (LAIRL) : LAIRL is a syntactic property of a lexeme which adds to a verb the information conveying the realization of an action or a state indicated by the verb, such as "จะ".

c. Left additional modal (LAAMD) : LAAMD is a syntactic property of a lexeme which adds to a verb the information conveying the speaker's attitude in respect to obligation of an action or a state, and often used with LAMD and LAIRL, such as "ต้อง".

d. Left aspect (LAASP) : LAASP is a syntactic property of a lexeme which adds to a verb the information conveying the aspect of an action or a state, such as "เพิ่ง", "ยัง", "กำลัง", etc.

e. Left attribute (LAATT) : LAATT is a syntactic property of a lexeme which adds to a verb the additional information, such as experience "เคย", opportunity "ได้", obligation "ต้อง", etc.

f. Right attribute (RAATT) : RAATT is a syntactic property of a lexeme which adds to a verb the additional information, but occurs at the right of the verb, such as ability "ได้", direction "ไป" "มา", etc.

g. Right aspect (RAASP) : RAASP is a syntactic property of a lexeme which adds to a verb the information conveying the aspect of an action or a state, but occurs at the right of the verb such as "แล้ว", "อยู่", etc.

5. MINCAT of Relator

a. Preposition (PREP) : PREP is a syntactic property of a lexeme which is used to relate a verb with a noun or a noun with another noun, such as "จาก", "เพื่อ", "บน", "ถึง", etc. A PREP which is always used after N is called PREPN, such as "ของ", while a PREP which is always used after V is called PREPV, such as "แก่", "แด่".

b. Complementizer (COMP) : COMP is a syntactic property of a lexeme which occurs in front of a relative or subordinate clause in a complex sentence, such as "ที่", "ซึ่ง", etc.

c. Connector (CONN) : CONN is a syntactic property of a lexeme which is used to combine two simple constructions into one, such as "และ", "หรือ", "แต่", "เพราะ", "ดังนั้น", etc.

2.2.3 Syntactic cases

Syntactic case defines dependency relation in a D-tree. It describes a syntactic relation between lexemes in a sentence. A syntactic case is assigned on the basis of category features and word order. This section discusses criteria for syntactic case analysis, syntactic case analysis, and syntactic case inventory for Thai.

2.2.3.1 Criteria for syntactic case analysis

Assigning a syntactic case between lexemes in a sentence is the main task in the syntactic analysis phase. To do this, three important questions need to be answered. First, how do we know which

lexeme is related to another lexeme? Second, how do we decide which lexeme is head or depender in the relation? Lastly, how do we determine whether those relations are similar or different? The answers to these questions are different sets of criteria to be used in the analysis.

Many set of criteria have been proposed by Melčuk (1988) and Hudson (1984). Some are in conflict with others. However, they provide practical criteria which can be used to analyze any language. The following sets of criteria have been adapted from those presented by Melčuk and Hudson.

1. Criteria for determining a relation

If A and B are closely related and co-occur in a sentence, and the linear position of one of them cannot be determined without reference to the other, then, A has relation to B. For example, in the sentence "the horse runs", "the" is more closely related to "horse" rather than "runs", and "the" cannot co-occur with "runs", and the position of "the" cannot be specified without referring to "horse". Therefore, "the" has a relation to "horse" while "horse" has a relation to "runs".

2. Criteria for determining head and depender

To determine the head of a relation, the following of five criteria have been proposed.

a. Inflectional form.

The head always decides the inflectional form of the depender. This criterion is very useful only in an inflectional language; however, Thai has no inflection.

b. Meaning.

The depender is less significant in meaning than the head, such as in the phrase "brown eyes", "brown" defines the color of the eyes rather than "eyes" defining the application-range of "brown"; therefore, "eyes" is the head of "brown".

c. Collocation.

The head selects the depender that can co-occur with it, such as "depend" selects to occur with "on" rather than "on" selects to occur with "depend".

d. Omissibility.

The depender is omissible whereas the head is not, such as in the noun phrase "brown eyes", "brown" is omissible whereas "eyes" is not.

e. Passive SS-valency.

Melčuk (1988: 115) defined passive SS-valency as the capacity of a surface-syntactic unit to depend syntactically on certain types of lexemes. The head is the lexeme that determines the passive SS-valency of the phrase to a greater degree than other lexeme. If the passive SS-valency of the phrase A B intersected with the passive SS-valency of A is greater than the passive SS-valency of the phrase A B intersected with the passive SS-valency of B, then, A is the head of B. Moreover, in case that passive SS-valency of any lexeme could not be determined, since it cannot stand alone, the passive SS-valency of that lexeme is determined to be equal to the passive SS-valency of the whole phrase and that lexeme is determined as the head. For example, in the phrase "at home", the passive SS-valency of "at home" is different from "home", and "at" is not used without the noun; therefore, the passive SS-valency of the prepositional phrase "at home" will be attributed to the preposition "at", and then, we can conclude that "at" is the head of "home".

3. Criteria for determining if relations are different

If A has a relation R1 to B, and C has a relation R2 to D, the question is whether or not R1 is different from R2. Three criteria have been proposed by Melčuk to answer this question:

a. Semantic contrast in minimal pairs

Let G1 and G2 be the phrases consisting of relations between A-B and C-D respectively. If A is the same lexeme as C, and B is the same lexeme as D, and the relations A-B and C-D are in the same direction, such as ¦A->B¦ ¦C->D¦ or ¦A<-B¦ ¦C<-D¦, and G1 and G2 are syntactically different, then R1 will be different from R2 if and only if there is at least one sentence pair, S1 and S2, that S1 contains G1 and S2 contains G2, and there is no other difference between S1 and S2, but are semantically distinct. The following example of Thai sentences illustrates this criterion.

(8) a) "เขา ทำ ได้" He can do.

b) "เขา ได้ ทำ" He has an opportunity to do.

The only observable difference between "ทำ ได้" and "ได้ ทำ" is in word order, which results in the two sentences having different semantic meaning. The conclusion then is that the relations between "ทำ" and "ได้" in these two sentences are different.

b. Reciprocal substitutability of subtrees

Let $C_{\blacktriangle}$ and $D_{\blacktriangle}$ be the complete terminal subtrees (subtrees having as their top nodes lexemes of C and D). R1 is the same relation as R2 and they have the Kunze property if and only if, for any pair of D-trees that contain $A->B_{\blacktriangle}$ and $C->D_{\blacktriangle}$, respectively, replacing $B_{\blacktriangle}$ by $D_{\blacktriangle}$ (or inversely) does not affect the correctness, otherwise R1 is different from R2. For example, in the phrases "the dog" and "small dog", the relation the<-dog is different from the relation small<-dog, because in the phrase "the small dog" we cannot replace "small" with "the" as *"the the dog".

c. Repeatability of a relation

Any relation must be either repeatable or non-repeatable with any dependent elements. If a structure ¦Y<-Ri-X-Ri->Z¦ is correct, the relation Ri is repeatable. If this is not the case, that relation then must be split into two different relations. For

example, if ¦Y<-Ri-X-Ri->Z¦ is correct for some Ys and Zs but incorrect for other Ys and Zs, this relation is split into two different relations.

2.2.3.2 Syntactic case analysis

The criteria discussed above are used, when applicable, to identify possible syntactic cases for Thai. Discussion on syntactic cases will be divided into three sections: syntactic cases within a noun phrase, syntactic cases between verb and auxiliary, and syntactic cases between verb and noun.

1. Syntactic cases within a noun phrase

Our analysis of syntactic cases covers the following noun phrase constructions:

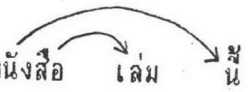
(9)	a) หนังสือ	CMNN
	b) หนังสือ นี้	CMNN RDET
	c) หนังสือ เล่ม นี้	CMNN CLSS RDET
	d) เล่ม นี้	CLSS RDET
	e) หนังสือ 3 เล่ม นี้	CMNN CRDN CLSS RDET
	f) 3 เล่ม นี้	CRDN CLSS RDET
	g) 3 เล่ม	CRDN CLSS
	h) กว่า 3 เล่ม	LDET CRDN CLSS
	i) กว่า 3	LDET CRDN
	j) กว่า เล่ม	LDET CLSS
	k) หนังสือ คอมพิวเตอร์ เล่ม นี้	CMNN CMNN CLSS RDET
	l) หนังสือ คอมพิวเตอร์	CMNN CMNN

In (9b), since "นี้" is omissible and less significant in meaning than "หนังสือ", therefore, "หนังสือ" is the head of "นี้".

In (9d), although neither "เล่ม" nor "นี้" is omissible, "เล่ม" is more significant in meaning and its passive SS-valency is equal to the passive SS-valency of the noun phrase, (since "เล่ม" cannot stand alone without any lexeme modified it); therefore, "เล่ม" is the head of "นี้".

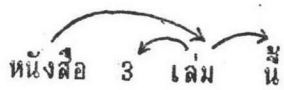
In (9c), both "หนังสือ" and "เล่ม" could be the head of "นี้" but only one of them will be. There are two possibilities as follows:

(10) a) 

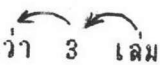
b) 

(10b) is incorrect because we cannot find the relation "หนังสือ->เล่ม". The lexeme "เล่ม" has a relation to "หนังสือ" if and only if "เล่ม" has a relation to another lexeme, such as "นี้". It seems to be the case that "เล่ม" is an incomplete noun (this is the reason why it cannot stand alone), it must have a modifying lexeme and the relation between "หนังสือ" and "เล่ม" is, in fact, the relation between the first noun "หนังสือ" and the second noun phrase "เล่ม นี้". It, therefore, is reasonable to choose the analysis (10a).

In (9e), we find that "3" has a relation to "เล่ม" and "3 เล่ม" can stand alone as a noun phrase. For the same reason in the phrase "เล่ม นี้", "เล่ม" is the head of "3", and in this phrase the dependency relations are as follows:

(10) c) 

In (9i) and (9j), we find that "กว่า" has a relation to "3" and "เล่ม" respectively. "กว่า" is less significant than "3" and "เล่ม"; therefore, it is the depender in the relation. In (9h), "กว่า" can have a relation to either "3" or "เล่ม", but a relation to the nearer lexeme which is "3" is preferred because "กว่า" is more closely related to the number noun rather than the classifier noun; therefore, the relations in (9h) are as follows:

(10) d) 

In (9k), we find that "หนังสือ" has a relation to "คอมพิวเตอร์" and "หนังสือ" is the head of "คอมพิวเตอร์" because "หนังสือ" is more significant than "คอมพิวเตอร์". In addition, "เล่ม" has a relation to "หนังสือ" rather than "คอมพิวเตอร์" because "เล่ม" is the classifier of "หนังสือ".

Looking at these relations in a noun phrase, we can see that the relation CMNN->$CLSS_{\Delta}$ is different from the relation CMNN->CMNN because in the phrase "หนังสือ คอมพิวเตอร์ เล่ม นี้", we cannot replace CMNN with $CLSS_{\Delta}$ and get *"หนังสือ เล่ม นี้ เล่ม นี้". The relation CMNN->$CLSS_{\Delta}$ is also different from the relation CLSS->RDET because in the phrase "เล่ม นี้", replacing RDET with $CLSS_{\Delta}$ results in an ungrammatical phrase *"เล่ม เล่ม นี้"; therefore, the first set of syntactic cases in a noun phrase are as follows:

(11) Compound_noun relation (CMPNR) : CMNN->CMNN

Classifier relation (CLSSR) : CMNN->$CLSS_{\Delta}$

Right determiner relation (RDETR) : CMNN->RDET, CLSS->RDET

In addition, the relation between $CRDN_{\Delta}$ and CLSS in (9h) is also different from the relation LDET<-CRDN in (9i) because in the phrase "กว่า 3" replacing LDET with $CRDN_{\Delta}$ as *"กว่า 3 3" results in ungrammaticality. We, therefore, can establish the following additional syntactic cases:

(12) Number relation (NUMR) : CRDN<-CLSS

Left determiner (LDETR) : LDET<-CRDN, LDET<-CLSS

In a noun phrase, the head noun can have both left and right determiner as dependers as follows:

(13) กว่า สิบ ตัว LDET CRDN CLSS

สิบ กว่า ตัว CRDN RDET CLSS

"กว่า" is the lexeme that has two categories: LDET and RDET. Since the relations between "สิบ" and "กว่า" in these two phrases are in the same direction, "สิบ" is the head of "กว่า". These two phrases are semantically different. This semantic difference is

caused by only one syntactic factor, namely word order. We, therefore, assign different syntactic cases for these relations as follows:

(14) Left determiner relation (LDETR) : LDET<-CRDN

Right determiner relation (RDETR) : CRDN->RDET

2. Syntactic cases between V and AUX

In determining a syntactic case between V and AUX, we find that V is syntactically more significant than AUX because AUX cannot stand alone. If AUX were the head of V, we would have relations LAUX->V and V<-RAUX, which is impossible because V cannot be a depender of both LAUX and RAUX simultaneously; therefore it is reasonable to treat V as the head of AUX. However, we also find that V can co-occur with more than one AUXs, such as "คง เคย ไป". We make both the first and the second AUX dependers of V. This is not a final decision. We acknowledge that it may be possible to consider the first AUX the depender of the second AUX.

Like in a noun phrase, V can have relations to both left and right auxiliaries. Let us consider these phrases:

(15) ไป ได้ VCMN RAATT

ได้ ไป . LAATT VCMN

This is a similar case to "กว่า สิบ ตัว" and "สิบ กว่า ตัว". So we propose the following syntactic cases:

(16) Right attribute relation (RATTR) : VCMN->RATT

Left attribute relation (LATTR) : LATT<-VCMN

Moreover, we cannot replace "เคย" with "คง" in the phrase "คง เคย ไป" as *"คง คง ไป"; therefore the relation "เคย<-ไป" and "คง<-ไป" are different relations. There are a maximum of five positions of LAUX as follows:

(17) คง จะ ต้อง เพิ่ง เคย ไป

LAMD LAIRL LAAMD LAASP LAATT V

The position of LAIRL is after LAMD, or between LAASP and LAATT. The relation of V and LAATT is repeatable. Since

reordering and replacement results in an ungrammatical string, we propose five different syntactic cases for AUX's to the left of V. To the right of V, there are only two minor categories, RAATT and RAASP. Altogether we recognized seven different syntactic cases between V and AUX as follows:

(18) Left modal relation (LMDR) : LAMD<-V (คง<-ไป)

Left additional modal relation (LAMDR) : LAAMD<-V (ต้อง<-ไป)

Left tense relation (LTNSR) : LAIRL<-V (จะ<-ไป)

Left aspect relation (LASPR) : LAASP<-V (เพิ่ง<-ไป)

Left attribute relation (LATTR) : LAATT<-V (เคย<-ไป)

Right aspect relation (RASPR) : V->RASPR (ไป->แล้ว)

Right attribute relation (RATTR) : V->RATT (ไป->ได้)

Note that although a particular word sequence may seem syntactically correct, it can be semantically incorrect, as in these examples.

(19) a) *"กำลัง เคย" LAASP LAATT

b) *"ยัง จะ เคย" LAASP LAIRL LAATT

c) "กำลัง จะ เคย" LAASP LAIRL LAATT

d) "ยัง เคย" LAASP LAATT

The sequence (19a) is incorrect because "กำลัง" is a progressive but "เคย" tells about an experience in the past; therefore, this sequence is a semantic conflict, but the sequence (19c) is correct since "จะ" tells that the experience has not occurred yet. The sequence (19b) is also incorrect because "ยัง" tells that the event is still going and "เคย" tells that the event did occur but "จะ" tells that the experience has not occurred yet; therefore, this sequence creates a semantic conflict. To obtain a correct string, "จะ" must be omitted.

3. Syntactic cases between V and N

Relation between V and N can be exhibited overtly with a relator. A relator is a lexeme with the category REL and it is

treated as a case marker; therefore, the phrase V REL N, will be analyzed as V->N rather than V->REL and REL->N.

(20) a) "นั่ง บน ต้นไม้" (sit on a tree)

b) "นั่ง ใต้ ต้นไม้" (sit under a tree).

In two phrases in (20), "นั่ง" is head and "ต้นไม้" is depender, the only difference lies in the choice of relator, "บน" for (20a) and "ใต้" for (20b), but these two phrases are semantically different. This syntactic difference motivates the assignment of different syntactic cases as follows:

(21) Locative-on prep relation : (LONPR) V->N

Locative-under prep relation : (LUNPR) V->N

These examples show the relations between N and V with case markers. Each case marker signifies a different syntactic case. When no REL occurs between N and V, four syntactic cases are proposed corresponding to the positions in relation to V as follows:

(22) N1 N2 V N3 N4

The positions N1, N2, N3, and N4, signify syntactic cases Topic, Subject, First_object and Second_object respectively. Examples of these four syntactic cases are shown as follows:

	N1	N2	V	N3	N4	
(23) a)		เขา	ไป	โรงเรียน		
b)		เขา	ให้	เงิน	เด็ก	
c)		หนังสือ	เขียน			เสร็จ แล้ว
d)	พรุ่งนี้	เขา	จะ ไป	โรงเรียน		
e)		พรุ่งนี้	ไป	โรงเรียน		

The syntactic case TOPR relates "พรุ่งนี้" and "ไป" in (23d). The syntactic case SUBR relates "เขา" and "ไป" in (23a) and (23d), "เขา" and "ให้" in (23b), "หนังสือ" and "เขียน" in (23c), and "พรุ่งนี้" and "ไป" in (23e). The syntactic case FOBR relates "ไป" and "โรงเรียน" in (23a), (23d), and (23e), and "ให้" and "เงิน" in (23b). The syntactic case SOBR relates "ให้" and "เด็ก" in (23b).

In most cases V is the head of N. However, N can be the head of V in a relative clause (24a) and in an adjective modified noun (24b).

(24) a) "เขา เป็น คน ที่ ดี"

b) "เขา เป็น คน ดี"

The relations between "คน" and "ดี" in these sentences are reciprocally substituable (see 2.2.3.1); therefore, they are analyzed to be the same syntactic case as follow:

(25) Complementizer relation (COMPR) : N-COMP->V, N->VADJ

2.2.3.3 Syntactic case inventory for Thai

The syntactic cases for the Thai language proposed for the analysis are as follows:

1. Compound_noun relation (CMPNR). This is a syntactic case between noun modifying noun, such as "หนังสือ-CMPNR->คอมพิวเตอร์".

2. Classifier relation (CLSSR). This is a syntactic case between noun and classifier, such as "หนังสือ-CLSSR->เล่ม", "โต๊ะ-CLSSR->ตัว".

3. Number relation (NUMR). This is a syntactic case between cardinal number and classifier, such as "10<-NUMR-คน"

4. Right determiner relation (RDETR). This is a syntactic case between noun and right determiner such as, "คน-RDETR->นี้", "เล่ม-RDETR->นี้", "สิบ-RDETR->กว่า".

5. Left determiner relation (LDETR). This is a syntactic case between noun and left determiner such as, "ทั้ง<-LDETR-10", "ทุก<-LDETR-คน".

6. Adjective relation (ADJR). This is a syntactic case between verb and adjective verb such as, "วิ่ง-ADJR->เร็ว", "ราคา-ADJR->แพง".

7. Left modal relation (LMDR). This is a syntactic case between verb and left modal, such as "คง<-LMDR-ไป", "อาจ<-LMDR-ไป", "ควร<-LMDR-ไป".

8. Left additional modal relation (LAMDR). This is a syntactic case between verb and left modal, "ต้อง", that usually follows "คง จะ" or "น่า จะ".

9. Left tense relation (LTNSR). This is a syntactic case between verb and "จะ".

10. Left aspect relation (LASPR). This is a syntactic case between verb and aspect lexeme, such as "เพิ่ง<-LASPR-ไป", "กำลัง<-LASPR-ไป", "ยัง<-LASPR-ไป".

11. Left attribute relation (LATTR). This is a syntactic case between verb and left attribute, such as "เคย<-LATTR-ไป", "ได้<-LATTR-ไป"

12. Right aspect relation (RASPR). This is a syntactic case between verb and right aspect, such as "ทำ-RASPR->อยู่", "ทำ-RASPR->แล้ว".

13. Right attribute relation (RATTR). This is a syntactic case between verb and right attribute, such as "ไป->ได้".

14. Topic relation (TOPR). This is a syntactic case between verb and the second noun at the left without any case marker.

15. Subject relation (SUBR). This is a syntactic case between verb and the first noun at the left without any case marker.

16. First_object relation (FOBR). This is a syntactic case between verb and the first noun at the right without any case marker.

17. Second_object relation (SOBR). This is a syntactic case between verb and the second noun at the right without any case marker.

18. Complementizer relation (COMPR). This is a syntactic case between noun and adjective, or between noun and relative clause with or without complementizer.

19. Locative on prep relation (LONPR). This is a syntactic case between verb and noun, or noun and noun, with a case marker "บน".

20. Locative-under prep relation (LUDRPR). This is a syntactic case between verb and noun, or noun and noun, with a case marker "ใต้".

21. Locative-at prep relation (LATPR). This is a syntactic case between verb and noun, or noun and noun, with a case marker "ที่".

22. Locative-in prep relation (LINPR). This is a syntactic case between verb and noun, or noun and noun, with a case marker "ใน".

23. Locative-front prep relation (LFRNTPR). This is a syntactic case between verb and noun, or noun and noun, with a case marker "หน้า".

24. About prep relation (ABOUTPR). This is a syntactic case between verb and noun, or noun and noun, with a case marker "เกี่ยวกับ".

25. Benefactive prep relation (BENPR). This is a syntactic case between verb and noun with a case marker "แก่".

26. Possessive prep relation (POSSPR). This is a syntactic case between noun and noun, with a case marker "ของ".

27. Mean-with prep relation (MWITHPR). This is a syntactic case between verb and noun with a case marker "ด้วย".

28. Partner prep relation (PARTNPR). This is a syntactic case between verb and noun, or noun and noun, with a case marker "กับ".

29. Range-from prep relation (RFROMPR). This is a syntactic case between verb and noun, or noun and noun, with a case marker "ตั้งแต่".

30. Range-to prep relation (RTOPR). This is a syntactic case between verb and noun, or noun and noun, with a case marker "ถึง".

31. Compare prep relation (COMPPR). This is a syntactic case between noun and noun, with a case marker "กว่า".

2.2.4 Priority of syntactic relation

In the analysis to convert a linear sequence of lexemes into a D-tree, it is necessary to set up criteria which determine the priority of dependency relations to be constructed at a stage in the analysis process.

In this study, the basic assumption in constructing relations in a D-tree is that a relation can be constructed for a pair of adjacent lexemes if the relation is possible in the language. In another word, a head can have a relation, or bond, to its depender only when they are adjacent in a lexeme sequence. Once the relation is constructed, the depender is moved out of the lexeme sequence and the next lexeme becomes adjacent to the head. This assumption follows from Robinson's adjacency principle (cited by Hudson, 1984: 98), which was later elaborated and broken down into two simpler principles: simple adjacency principle and priority-to-the-bottom principle (Hudson 1984: 99) as follows:

Adjacency Principle:

If A depends directly on B (i.e. is a modifier of B), and some other element C intervenes between them (in linear order of strings), then C depends directly on A or on B or some other intervening element.

Simple Adjacency Principle :

A modifier must not be separated from its head by anything except other modifiers of the same head.

Priority-to-the-Bottom Principle :

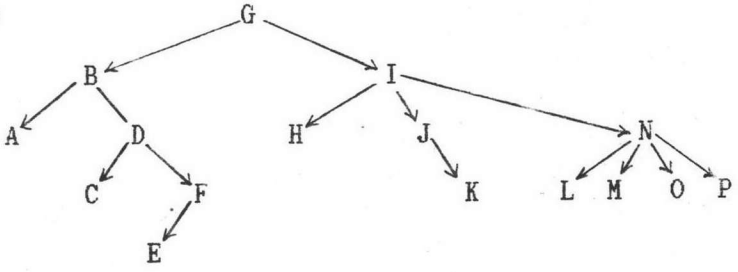
The adjacency requirements of a word A take priority over those of any other word which is higher than A in the same dependency chain.

This basic assumption alone is not enough to derive a D-tree, which is a hierarchical construction. Criteria are needed to determine which relations should be constructed when, i.e. priority of

relations. This priority indicates the comparative strength of the bond between lexemes. The study of dependencies in the D-trees of sentences, which are manually constructed, reveals three types of priority: bottom-up priority (BPT), immediacy priority (IPT), and probability priority (PPT).

The following is a description in detail of these three priorities and how they are obtained from a D-tree. Suppose that we have an input the list of lexemes A B C D E F G H I J K L M N O and P, and the desired output D-tree of this sentence is (26).

(26)



2.2.4.1 Bottom-up priority (BPT)

BPT requires that relations at a lower level should be constructed before relations at a higher level. For example, if J is attached to I before K is attached to J, then K will lose the opportunity to be attached to J. BPT operates in a reverse direction of a path from root to leaf nodes. The relation at the lower level has higher BPT than the relation at the higher level in that path. The sets of ordering in terms of BPT for the relations in sentence (26) will then be as follows:

(27) A<-B > B<-G

C<-D > B->D > B<-G

E<-F > D->F > B->D > B<-G

H<-I > G->I

J->K > I->J > G->I

L<-N > I->N > G->I

M<-N > I->N > G->I

N->O > I->N > G->I

N->P > I->N > G->I

The definition of BPT can be summed up as follows:

For any D-tree from an input sequence N0 N1 N2 N3 N4...,

if |Ni->Nj|, |Nj->Nk|, |Nk->Nl|,..., |Ny->Nz|,

Ni is the root, and Nz is the leaf node,

then |Ni->Nj| < |Nj->Nk| < |Nk->Nl| <....< |Ny->Nz|.

It is clear that BPT follows from Hudson's priority-to-the-bottom principle.

2.2.4.2 Immediacy priority (IPT)

A head can have more than one depender to its left or its right in a sequence. For example, in (26) N has M and L attached to its left and has O and P attached to its right. Of all the dependers on the same side, the depender immediately adjacent to the head has a higher priority to be attached to the head than the next depender, which will then become its immediately adjacent depender. We can, therefore, conclude that by looking at the same level of any subtrees, we are able to decide which relation has higher IPT than others. The sets of ordering in terms of IPT for the relations in sentence (26) will then be as follows:

(28) I->J > I->M

M<-N > L<-N

N->O > N->P

The definition of IPT can be summed up as follows:

For any D-tree from an input sequence N0 N1 N2 N3 N4..,

if Nx->Ni, Nx->Nj, Nx->Nk, Nx->Nl, ..., and x<i<j<k<l<...,

then Nx->Ni > Nx->Nj > Nx->Nk > Nx->Nl >....,

if Ni<-Nx, Nj<-Nx, Nk<-Nx, Nl<-Nx,..., and i<j<k<l<...<x,

then Ni<-Nx < Nj<-Nx < Nk<-Nx < Nl<-Nx <....

2.2.4.3 Probability priority (PPT)

In any three adjacent lexemes, like X Y Z, four possible patterns of relations can be constructed.

1. X<-Y is constructed and

a. If Y can be the head of Z (X<-Y->Z), then X<-Y and Y->Z do not affect each other, since both relations are on different sides.

b. If Z can be the head of Y (X<-Y<-Z), then X<-Y already has higher BPT than Y<-Z and it does not affect Y<-Z anymore.

2. Y->Z is constructed and

a. If X can be the head of Y (X->Y->Z), then the case will be the same as that in (1.b). Y->Z does not affect X->Y anymore.

b. if Y can be the head of X (X<-Y->Z), then the case will be the same as that in (1.a). Both relations are on different sides.

3. X->Y is constructed and

a. If Y can be the head of Z (X->Y->Z), then X->Y violates BPT because Y->Z should have higher BPT than X->Y. We can, therefore, claim that X->Y has higher PPT than Y->Z since it does not allow the occurrence of Y->Z.

b. If Z can be the head of Y (X->Y<-Z), then both X and Z can possible have Y as a depender but Y had been attached to X. So we can conclude that X->Y has higher PPT than Y<-Z.

4. Y<-Z is constructed and

a. If Y can be the head of X (X<-Y<-Z), then the case is the same as (3.a) in that Y<-Z violates BPT. So we can conclude that Y<-Z has higher PPT than X<-Y.

b. If can be the head of Y (X->Y<-Z), then the case is the same as that in (3.b). So we can conclude that Y<-Z has higher PPT than X->Y.

From these four possible situations, we can come up with the conclusion that in any three adjacent lexemes like X Y Z, if Y is attached to one lexeme, either to the left or the right, then this relation has higher PPT than any relations between Y and the other lexeme if and only if these relations exist.

(29) case : X->Y implies that X->Y > [Y<-Z]

case : Y<-Z implies that Y<-Z > [X<-Y]

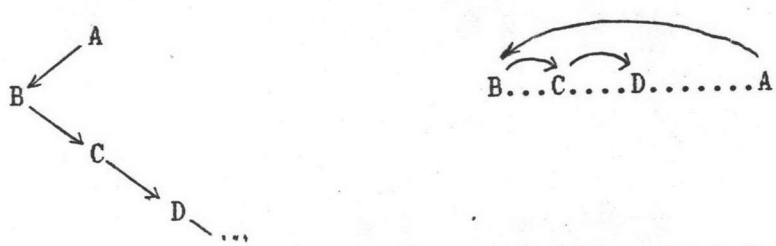
Note that [Y<->Z] refers to Y<-Z, if this relation exists and also refers to Y->Z, if this relation exists.

From this conclusion, we can see that for every three adjacent lexemes in an input sequence we can infer some sets of PPT from them. However, while the construction is in progress the non-adjacent lexemes may have an opportunity to become adjacent lexemes as well, like D and G in the sequence of lexemes in (26). At first, D is not adjacent to G but after E is attached to F and F is attached to D, D will be automatically adjacent to G. It, therefore, is not simple to infer PPT from an input lexemes sequence. Rather PPT has to be obtained from the lexeme sequence all through the process of construction.

In general, there are four types of subtree constructions in which PPT determines the order of relations to be constructed.

1. Left-Right Chain (LRC)

(30)



For any node A, if these relations, B<-A,B->C,C->D,...., exist, and B is any left child of A, C is the right most child of B, and D is the right most child of C, and so on, we can infer these PPTs:

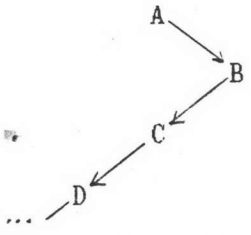
(31) B->C > [C<->A]

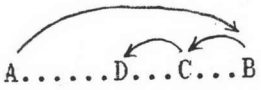
C->D > [D<->A]

... ...

2. Right-Left Chain (RLC)

(32)





For any node A, if these relations, A->B,...D<-C,C<-B, exist, and B is any right child of A, C is the left most child of B, and D is the left most child of C, and so on, we can infer these PPTs:

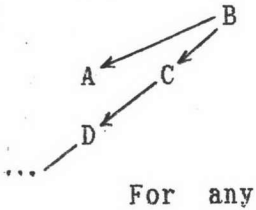
(33) C<-B > [A<->C]

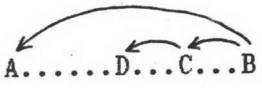
D<-C > [A<->D]

... ...

3. All Left Chain (ALC)

(34)





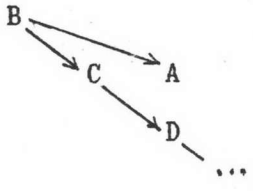
For any node A, if these relations, A<-B,..D<-C,C<-B, exist, and A is the left child of B, C is the left child of B next to A, and D is the left most child of C, and so on, we can infer these PPTs:

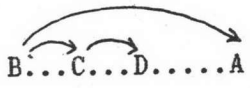
(35) C<-B > [A<->C]

D<-C > [A<->D]

4. All Right Chain (ARC)

(36)





For any node A, if these relations, B->A,B->C,C->D..., exist, and A is the right child of B, A is the right child of B

next to C, and D is the right most child of C, and so on, we can infer these PPTs:

(37) B->C > [C<-A]

C->D > [D<-A]

... ...

For sentence (26), following are the sets of ordering in terms of PPT for the relations in this sentence.

(38) B->D > [D<-G] (a case of LRC)

D->F > [F<-G]

C<-D > [B<-C] (a case of RLC)

E<-F > [D<-E]

H<-I > [G<-H]

L<-N > [I<-L]

J->K > [K<-N] (a case of ARC)

I->J > [J<-N]

N->O > [O<-P]

M<-N > [L<-N] (a case of ALC)

The definition of PPT can be summed up as follows:

For any D-tree from an input sequence N0 N1 N2 N3 N4...,

if Ni<-Nx, Ni->Nj,Nj->Nk,Nk->Nl,Nl->Nm,..., exist,

Nj,Nk,Nl,Nm,... is the right most child of Ni,Nj, Nk, Nl,... respectively,

and i<j<k<l<m<....<x,

then Ni->Nj > [Nj<->Nx],

Nj->Nk > [Nk<->Nx],

Nk->Nl > [Nl<->Nx],

....,

if Nx->Ni, ...Nm<-Nl,Nl<-Nk,Nk<-Nj,Nj<-Ni, exist

Nj,Nk,Nl,Nm,... is the left most child of Ni,Nj,Nk,Nl,.. respectively,

and x<...<m<l<k<j<i,

then Nj<-Ni > [Nx<->Nj],
Nk<-Nj > [Nx<->Nk],
Nl<-Nk > [Nx<->Nl],
....,

if Nx<-Ni, ...Nm<-Nl,Nl<-Nk,Nk<-Nj,Nj<-Ni, exist
Nj is the left child of Ni next to Nx,
Nk,Nl,Nm,... is the left most child of Nj,Nk,Nl,...
respectively,
and x<....<m<l<k<j<i,

then Nj<-Ni > [Nx<->Nj],
Nk<-Nj > [Nx<->Nk],
Nl<-Nk > [Nx<->Nl],
....,

if Ni->Nx, Ni->Nj,Nj->Nk,Nk->Nl,Nl->Nm,....., exist
Nx is the right child of Ni next to Nj,
Nk,Nl,Nm,... is the right most child of Nj,Nk,Nl,...
respectively,
and i<j<k<l<m<....<x,

then Ni->Nj > [Nj<->Nx],
Nj->Nk > [Nk<->Nx],
Nk->Nl > [Nl<->Nx],
....

2.3 Conceptual component

Meaning or signification has been defined as the relationship between concept and form in a language (Ullman 1957). De Saussure (1959) used the terms "signifie" and "signifiant" for concept and form respectively. Meaning in linguistics refers to sense or concept, which is an abstraction of reality observed by human beings. It does not refer directly to real objects or reality. For example, the

concept "TREE" can refer to not only trees that exist, but also trees that existed some time in the past, and trees that exist only in the imagination. Besides "concept", others terms are also used, such as mental object (Baldinger 1980), signifie (Saussure 1959), and sememe (Pottier, cited by Baldinger, 1980).

The main task of the conceptual component is to produce a meaning representation, which is a conceptual network in this study, by establishing conceptual dependency relations, or conceptual cases, between concepts in the conceptual network. A conceptual network consists of nodes and arcs linking nodes. A node labeled with a word-concept is called a concept node. A node labeled with no concept is called an empty node. Each arc is labeled with either a conceptual case or a conceptual attribute. An arc linking concept nodes is labeled with a conceptual case while an arc linking a concept node and empty node is labeled with a conceptual attribute. Unlike a D-tree, a conceptual network does not have the restriction that one node must have only one head. Multiple heads are allowed. A head node is taken to be more significant than its depender. A verbal concept is always the head of a nominal concept. A determiner and an auxiliary are always the depender of a nominal and verbal concept respectively. A modal auxiliary is an exception. Since a modal tells of speaker's attitude, in this study it is qualified to be head of verbal concept. In addition, the head noun in compound nominal concept is considered head concept.

This section covers five aspects of conceptual dependency in the conceptual component: conceptual cases; conceptual attributes; correspondence between conceptual cases and syntactic cases; constraints on the conceptual case between a head and a depender; and grouping of concepts as a hierarchy.

2.3.1 Conceptual cases

A conceptual case is a relation between concepts in a conceptual network. Unlike a syntactic case, the criteria for assigning a conceptual case cannot be explicitly formulated since it is usually based on a researcher's subjective view. Directionality for a conceptual case may not be the same as that of the corresponding syntactic case. The conceptual case inventory proposed in this study is as follows.

1. Verbal-Nominal cases

a. Agent (AGT) : indicates that the depender nominal concept causes the event or action to occur. For example, "เขา กิน ข้าว", "เขา วิ่ง".

b. Object (OBJ) : indicates that the depender nominal concept is the center of action, or directly affected by the event or exhibit such property. For example, "เขา กิน ข้าว", "เสื้อสวย", "เขา ให้ เงิน น้อง", "เขา ถูก ตี".

c. Affected (AFF) : indicates that the depender nominal concept is indirectly affected by the event. For example, "เขา ให้ เงิน น้อง", "เขา ขโมย เงิน น้อง".

d. Comitative (COMT) : indicates that the depender nominal concept signals a co-agent or co-object of an event. For example, "เขา ไป ดู หนัง กับ สมปอง".

e. Partner (PARTN) : indicates that the depender nominal concept is a partner in an event. For example, "เขา คุย กับ เพื่อน", "เขา ซื้อ หนังสือ กับ สมปอง", "เขา พูด กับ สมปอง".

f. Complement (CMPL) : indicates that the depender nominal concept serves as a complement of another noun. For example, such as "เขา เป็น ครู", "เขา ชื่อ สมปอง", "หนังสือ ราคา 100 บาท".

g. Mean (MNS) : indicates that the depender nominal concept is a means by which an action occurs or is accomplished. For example, "เขา หา รายได้ ด้วย การ ขาย หนังสือพิมพ์".

h. Instrument (INS) : indicates that the depender nominal concept signals an instrument involved in an event. For example, "เขา เปิด ประตู ด้วย กุญแจ", "กุญแจ เปิด ประตู นี้ ไม่ ได้".

i. Comparison (COMP) : indicates that the depender nominal concept is used as an object to which a comparison is made of another noun. For example, "รถยนต์ วิ่ง เร็ว กว่า รถจักรยาน".

j. Time (TIM) : indicates that the depender nominal concept gives information on time when an event occurs. For example, "พรุ่งนี้ เขา จะ ไป โรงเรียน".

k. Time_Begin (TIM_B) : indicates that the depender nominal concept tells the time when an event begins. For example, "เขา ทำงาน ตั้งแต่ เช้า ถึง เย็น".

l. Time_End (TIM_E) : indicates that the depender nominal concept tells the time when an event stops. For example, "เขา ทำงาน ถึง เย็น".

m. Location (LOC) : indicates that the depender nominal concept is the location of an event. For example, "เขา ทำงาน ใน ห้องสมุด", "เขา นั่ง กับ พื้น".

n. Source (SOR) : indicates that the depender nominal concept is the source from where there is a movement or transfer. For example, "เขา ออก จาก บ้าน".

o. Target (TAR) : indicates that the depender nominal concept is the target to which a movement or a transfer is made. For example, "เขา ไป โรงเรียน".

p. Quantity (QUAT) : indicates that the depender nominal concept is the quantity of the head. For example, "วิ่ง 3 รอบ".

q. Property (PRPT) : indicates that the depender nominal concept is the property of the head, such as "คอมพิวเตอร์ ราคา แพง".

2. Nominal-Nominal cases

a. Comitative (COMT) : indicates that the depender nominal concept signals a co-agent or co-object of an event. For example, "เขา ซื้อ ข้าว และ ปลา".

b. Quantity (QUAT) : indicates that the depender nominal concept is the quantity of the head. For example, "หนังสือ 3 เล่ม".

c. Number (NUMB) : indicates that the depender nominal concept tells the number of the head classifier noun, such as "หนังสือ 3 เล่ม", "วิ่ง 3 รอบ".

d. Part_of (PRTOF) : indicates that the depender nominal concept is the part of the head. For example, "ประตู รถ".

e. About (ABOUT) : indicates that the head nominal concept supplies detail about the depender. For example, "เรื่อง เกี่ยวกับ คอมพิวเตอร์".

f. Possessor (POSS) : indicates that the depender nominal concept is the possessor of the head. For example, "หนังสือ ของ สมปอง".

g. Label (LABEL) : indicates that the depender nominal concept is the label of the head. For example, "หนังสือ ราชาธิราช".

h. Classified (CLSS) : indicates that the depender nominal concept is a unit or a group of the head. For example, "คอมพิวเตอร์ รุ่น นี้", "คอมพิวเตอร์ เครื่องนี้".

i. Property (PRPT) : indicates that the head nominal concept is the property of the depender. For example, "ราคา ของ คอมพิวเตอร์".

3. Verbal-Verbal case

a. Manner (MAN) : indicates that the depender verbal concept tells the manner of an event. For example, "เขา วิ่ง เร็ว", "หนังสือ ราคา แพง".

4. Modal-Verbal case

a. Modal (MODAL) : indicates that the head modal concept expresses the speaker's attitude about an event. For example, "เขา ควร ไป โรงเรียน", "เขา คง ไป โรงเรียน".

2.3.2 Conceptual attributes

Conceptual attributes are the information, conveying by determiner or auxiliary lexemes, added to nominal or verbal concepts. They are represented here as labels of the arcs linking concept nodes and empty nodes in a conceptual network. The information of these attributes are kept as features left attribute (LATT) and right attribute (RATT) in the dictionary. The conceptual attributes proposed in this study are as follows:

1. Verbal attribute

a. Irrealis : is an attribute which adds to the head verbal concept the information on the realization of an event. For example "เขา จะ ไป โรงเรียน".

b. Opportunity (Opport) : is an attribute which adds to the head verbal concept the information on the opportunity for an event to take place or be performed. For example, "เขา ได้ ไป โรงเรียน".

c. Obligation (Oblige) : is an attribute which adds to the head verbal concept the information on the obligation to perform an event. For example, "เขา ต้อง ไป โรงเรียน".

d. Just : is an attribute which adds to the head verbal concept the information that an event has just occurred. For example, "เขา เพิ่ง ไป โรงเรียน".

e. Still : is an attribute which adds to the head verbal concept the information that an event is still in progress. For example, "เขา ยัง ไป โรงเรียน".

f. Dynamic_progressive (Dyn_prog) : is an attribute which adds to the head verbal concept the information that an event is a dynamic-progressive at that moment. For example "เขา กำลัง ทำงาน".

g. Static_progressive (Stat_prog) : is an attribute which adds to the head verbal concept the information that an event is a static-progressive at that moment. For example, "เขา ทำงาน อยู่".

h. Ability : is an attribute which adds to the head verbal concept the information in respect to ability of the performer. For example, "เขา ไป โรงเรียน ได้".

i. Finish : is an attribute which adds to the head verbal concept the information on the completeness of an event. For example, "เขา ไป โรงเรียน แล้ว".

j. Up_Direction (Up) : is an attribute which adds to the head verbal concept the information that the state or process is increasing. For example, "เขา สวย ขึ้น".

k. Down_Direction (Down) : is an attribute which adds to the head verbal concept the information that the state or process is decreasing. For example, "เขา ผอม ลง".

l. Go_Direction (Go) : is an attribute which adds to the head verbal concept the information that the action is moving away from the focused location. For example, "เขา หยิบ หนังสือ ไป".

m. Come_Direction (Come) : is an attribute which adds to the head verbal concept the information that the action is moving toward the focused location. For example "เขา หยิบ หนังสือ มา".

n. Keep_state (Keep) : is an attribute which adds to the head verbal concept the information on the resultant state of an event. For example, "เขา เก็บ หนังสือ ไว้".

2. Nominal attribute

a. Demonstrative (Demons) : is an attribute which adds to the head nominal concept the information on demonstrativeness. For example, "คน นี้".

b. Approximate (Approx) : is an attribute which adds to the head nominal concept the information on approximation of the number. For example "หนังสือ ราว 3 เล่ม".

c. Part_whole : is an attribute which adds to the head nominal concept the information on the comprehensive view of a quantity. For example, "ทั้ง 3 เล่ม".

d. Every : is an attribute which adds to the head nominal concept the information on equal distribution within a set. For example, "ทุก คน".

e. More_than : is an attribute which adds to the head nominal concept the information that the actual number can exceed the specified number. For example, "กว่า 10 ตัว".

f. More_than_in_range : is an attribute which adds to the head nominal concept the information that the actual number can exceed a specified number in a certain range. For example, "10 กว่า ตัว", "10 ตัว กว่า".

g. Exact : is an attribute which adds to the head nominal concept the information on the exactness of the number. For example, "10 ตัว ถ้วน".

h. Non_Specific (Non_Spec) : is an attribute which adds to nominal concept the information on non-demonstrativeness. For example, "คน คน หนึ่ง".

i. Single : is an attribute which limits the number down to one. For example, "คน คน เดียว".

2.3.3 Case mapping

Case mapping is a correspondence between syntactic cases and conceptual cases. Case mapping defines a certain range of conceptual cases that can correspon to certain syntactic cases. One syntactic case can map onto many conceptual cases and vice versa. The example (38a) shows that the syntactic case SUBR for the verb "ไป" can map

onto different conceptual cases such as AGT, OBJ, and TIM. On the other hand, the example (38b) shows that the conceptual case TIM can be derived from different syntactic cases such as SUBR, TOPR, and LINPR.

(38) a) "เขา ไป โรงเรียน" SUBR maps onto AGT
"โรงเรียน นี้ ไป ลำบาก" SUBR maps onto OBJ
"วันพรุ่งนี้ จะ ไป โรงเรียน" SUBR maps onto TIM
b) "วันพรุ่งนี้ จะ ไป โรงเรียน" SUBR maps onto TIM
"วันพรุ่งนี้ เขา จะ ไป โรงเรียน" TOPR maps onto TIM
"เขา จะ ไป โรงเรียน ใน วันพรุ่งนี้" LINPR maps onto TIM

Likewise, mapping of the same syntactic case onto many different conceptual cases of a different verb is also possible. For example, the relation PARTNPR will map onto conceptual case COMT for the verb "ไป" whereas it will map onto conceptual case PARTN for the verb "พูด".

(39) "เขา ไป โรงเรียน กับ น้อง" PARTNPR maps onto COMT
"เขา พูด กับ น้อง" PARTNPR maps onto PARTN

The relation POSSPR can also map onto different conceptual cases in different noun phrases as follows:

(40) "หนังสือ ของ เขา" POSSPR maps onto POSS
"ปีก ของ นก" POSSPR maps onto PRTOF
"ราคา ของ เครื่องพิมพ์" POSSPR maps onto PRPT

Examples (39) and (40) show that case mapping depends directly on the head concept. It, therefore, should be kept in the dictionary as the property, or feature, of the head concept. Case mapping can be divided into two types: default case mapping and individual case mapping. Default mapping need not be specified in a lexeme since it is predictable which conceptual cases correspond to syntactic cases. For example, syntactic case POSSPR corresponds to conceptual case POSS. This mapping information is left unspecified in the head concept but is kept as a part of the conceptual component.

Case mapping, however, alone is not adequate as a means for assigning a conceptual case because case mapping usually defines a certain range of possible conceptual cases rather than gives one single conceptual case. Conceptual case constraint, therefore, is needed to determine what a particular case conceptual case should be related for in a D-tree.

2.3.4 Conceptual case constraints

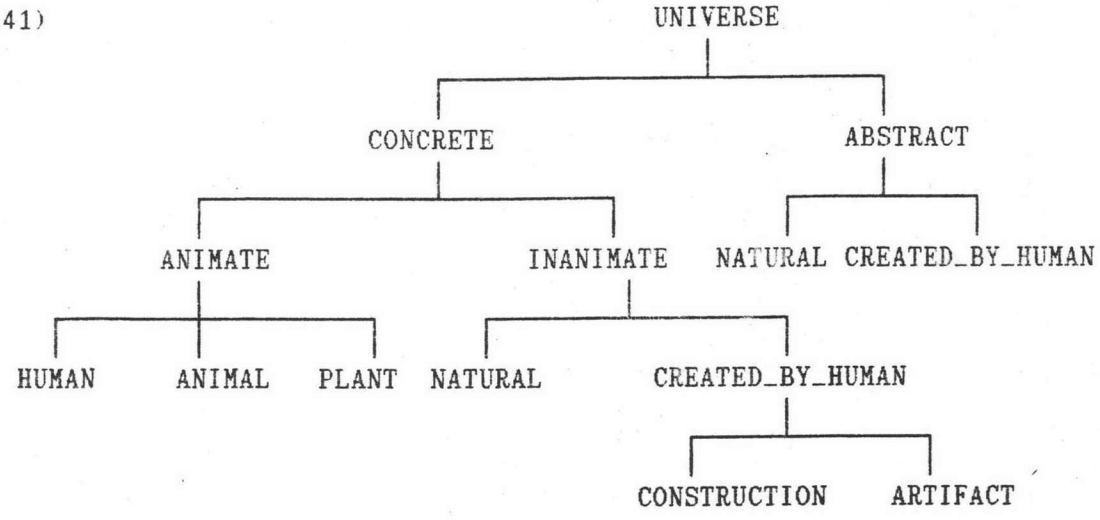
Conceptual case constraints determine what possible conceptual cases exist between a particular head concept and a particular kind of depender concept. For example, the conceptual case, PRPT, can relate the head concept "LENGTH" to the depender concept such as "BOOK", "TABLE", or "ROOM", which all belong to the class "CONCRETE", but it cannot relate "LENGTH" with the depender concept such as "LOVE", "POOR", or "RED", which all belong to the class "ABSTRACT". These conceptual case constraints are stated also in the dictionary as features of the head concept. As in case mapping, these constraints are divided into two types: default constraints and individual constraints. Default constraints are regarded as general constraints since they are predictable, such as the concept "TIME" is regarded as the default constraint on the conceptual case TIM. They are kept as a part of the conceptual component. On the other hand, individual constraints are properties or features of individual head concepts, such as constraints on conceptual cases OBJ and INS.

2.3.5 Conceptual hierarchy

Grouping of concepts has been done as a conceptual hierarchy. A conceptual hierarchy has to correspond to the constraints of concept. A hierarchy of concept can be viewed as a IS_A conceptual network, in which the lower concept is a subset of the upper concept. For example,

the concept "ANIMATE" is lower in the hierarchy than the concept "CONCRETE" and is a subset of "CONCRETE". The concepts "HUMAN", "ANIMAL", and "PLANT" are all subsets of "ANIMATE". The lower concepts by default inherit the properties of the upper concept. If weight is a property of "CONCRETE", it is also a property of "HUMAN", "ANIMAL", and "PLANT" through this inheritance principle.

(41)



One of the weak points of the conceptual hierarchy is that it cannot show the relation between the concepts on different branches. A remedy is to allow the same concept to be placed at more than one location in the hierarchy. For example, "CREATED_BY_HUMAN" can be under both "CONCRETE" and "ABSTRACT". Baldinger (1980: 117) expressed his view on hierarchy as follows:

If we try to arrange the whole language in such a hierarchic order, we shall come up against great difficulties. Neither in reality nor in language is there an absolute hierarchical division: thus, there can be no objective conceptual system which is generally and absolutely valid. ..... But to renounce conceptual classifications for this reason would be foolish. This is not the only compromise which we find essential in linguistics.

2.4 Dictionary

A dictionary supplies all information about a lexeme which is needed for the analysis process. The information is represented as features and values of a lexeme as shown in (42). A lexeme, which is defined by Melčuk as a word taken in one well-defined sense or meaning, is treated as the basic unit in the dictionary. Each lexeme together with all its features and values is an entry in the dictionary. Dictionary can then be defined as a collection of entries, each of which is a lexeme equipped with all the syntactic and semantic information needed for the understanding of a sentence.

(42) [wordform]
feature_A: value, value.....
feature_B: value, value,....
....

A wordform conveying more than one meaning will be kept in the dictionary as a number of lexemes to account for all the meanings of that word. Examples in (43) are entries for "กึ๋ง" and "ที่". The wordform "กึ๋ง" in (43a) has one lexeme and four features. The wordform "ที่" in (43b) has two lexemes which have the same wordform but different features and values. The former is the lexeme of the function word while the latter is the lexeme of the content word. The lexeme of the content word represents either a concept or conceptual attribute.

(43) a) [กึ๋ง]
MORPH:กึ๋ง. MAJCAT:REL. MINCAT:PREP. SYNTC:RTOPR.

b) [ที่]
MORPH:ที่. MAJCAT:REL. MINCAT:PREP,COMP. SYNTC:LATPR.

[ที่]
MORPH:ที่. MAJCAT:N. MINCAT:CMNN. CP:PLACE. UPCP:place,space,conc.

The following section gives details of the features supplyed for syntactic and conceptual analysis.

MORPH : This feature keeps the morphological form of the lexeme, such as "ถึง", "ที่".

MAJCAT : This feature keeps the major category values, such as N, V. It can have more than one value.

MINCAT : This feature keeps the minor category values, such as CMNN, VCMN. It can have more than one value.

CP : This feature keeps the word-concept of the lexeme, such as "USE", "COMPUTER".

UPCP : This feature keeps all upper concepts of the lexeme. This information can be retrieved from the hierarchy of concepts (see 2.3.5) such as "ABST", "CONC".

CLSSG : This feature keeps the morphological forms of the classifier used with the lexeme, such as the lexeme "คอมพิวเตอร์" has "เครื่อง" and "รุ่น" as its classifier.

LATT : This feature keeps the attributes which the lexeme represented, such as Ability, Irrealis. This feature is used for the lexemes LAUX and LDET.

RATT : This feature keeps the attributes which the lexeme represented, such as Opport, Demons. This feature is used for the lexemes RAUX and RDET.

MSUBR : This feature keeps the conceptual cases which can be derived from the syntactic case SUBR for the lexeme.

MFOBR : This feature keeps the conceptual cases which can be derived from the syntactic case FOBR for the lexeme.

MTOPR : This feature keeps the conceptual cases which can be derived from the syntactic case TOPR for the lexeme.

MPOSSPR : This feature keeps the conceptual cases which can be derived from the syntactic case POSSPR for the lexeme.

CSAGT : This feature keeps the conceptual constraint information of the lexeme which indicates that the case must be AGT.

CSOBJ : This feature keeps the conceptual constraint information of the lexeme which indicates that the case must be OBJ.

CSINS : This feature keeps the conceptual constraint information of the lexeme which indicates that the case must be INS.

CSMNS : This feature keeps the conceptual constraint information of the lexeme which indicates that the case must be MNS.

CSAFF : This feature keeps the conceptual constraint information of the lexeme which indicates that the case must be AFF.

CSCMPL : This feature keeps the conceptual constraint information of the lexeme which indicates that the case must be CMPL.

CSPRPT : This feature keeps the conceptual constraint information of the lexeme which indicates that the case must be PRPT.

CFRM : This feature keeps the possible case frames of the verb lexeme, such as AO,O,TO.

SYNTC : This feature keeps the syntactic case between two lexemes.

CONCC : This feature keeps the conceptual case between two lexemes.

INH : This feature keeps the information inherited from the depender lexeme.

PATT : This feature keeps the syntactic cases between the lexeme and its dependers, such as TOPR, SUBR.

RELMS : This feature keeps the syntactic case which is missing in the relative clause. It is either SUBR or FOBR.