Chapter 2

Pattern Matching Analysis in a Nutshell

2.0 Introductory Notes

The proposed framework of pattern matching analysis has a number of interesting features. I summarize below its notable appeals.

- It provides a **foolproof method** for the analysis/description of natural language syntax.
- The descriptions provided by it are based on **surface-true generalizations** alone, rather than UG-based highly abstract (and sometimes absurd) constructs.
- It provides a new perspective on **representation of linguistic units**, and provides some insights into a theory of "emergent" grammar, which is presumably learnable.
- It is **compatible with connectionist research of the mind**, not "terminologically" but "substantially".

To be faithful, it is necessary to substantiate each of those claims, but there are many difficulties in doing so. First of all, it would be inappropriate to launch into details of the proposed framework, without familiarizing readers with its basics. This is because the proposed framework assumes drastically different things about linguistic structure, especially the representation of linguistic units. Thus, it is more reasonable and practical to provide readers with an intuitive grasp of how pattern matching analysis goes and works. For this reason, I decided to make discussions in this chapter as basic as possible.

2.1 How Pattern Matching Analysis Goes

In this section, sample analyses are performed to familiarize readers with the proposed framework.

2.1.1 Analysis of *Ann asked (Bill) the way*

It is a theoretical claim of PMA that the necessary and sufficient condition for the formation of a surface form *F* from a set of units $\{u_1, \ldots, u_n\}$ is a set of **position-sensitive dependencies** for all component units in *u*. For illustration, take (1) for example.

(1) *Ann asked (him) the way.*

Call (1) form *F*. PMA assumes that *F* is a composition from the set:

(2) *U* = {*Ann*, *asked*, *(him)*, *the way*}

For the purposes of syntax, the necessary and sufficient condition for (1) is the set of conditions:

(3) On a certain scale, there is pattern S V (X) O such that:

i. *Ann* matches S
ii. *asked* matches V
iii. *(him)* matches (X)
iv. *the way* matches O

S, V, and O roughly encode “subject”, “verb”, and “object”. I assume here and elsewhere that (X) is a special kind of O, distinguished from O, both syntactically and semantically.

Under this condition, a pattern matching analysis of (1) is given as (4).

(4)

0.	***Ann***	***asked***	***(him)***	***the way***
1.	***Ann***	*V*		*(O)*
2.	*S*	***asked***	*(X)*	*O*
3.	*S*	*V*	***(him)***	*O*
4.	*S*	*V*	*(X)*	***the way***

An encoding tables like (4) is called a **composition/decomposition table** (C/D table for short). I claim that they encode necessary and sufficient information to specify syntactic structures of surface formations. Details of a C/D table will be discussed in Chapter 3 and Appendix A.

2.1.2 Remarks

Encoding tables (4) are proposed to replace tree diagrams. The conception of a C/D table is innovative, and requires some remarks on it.

Let me establish the terminology first. In (4), the topmost row, with index 0, is called the **base pattern**. Other rows, with indices 1, 2, 3, and 4 are called **subpatterns.** Subpatterns encode structures of words.

My interpretation of tables like (4) is that subpatterns are **declarative statements** which express correlations between grammatical functions like *S*, *V*, *O*, and their positions relative to a pattern like *SV(X)O*. For example, in (4):

(5) i. 1 states that ***Ann*** is ***S*** if it is followed by a unit that serves as *V* on certain scale, which itself precedes an *O*, if any;
ii. 2 states that ***asked*** is ***V*** if it follows a unit that serves as *S* on certain scale, that precedes it and serves as *S*, to precede *O*, on the other;
iii. 3 states that ***(him)*** is an optional **O** if it is preceded by a unit of *S V* (on certain scale) and postcedes *V*, which postcedes *S*.
iv. 4 states that ***the way*** is **O** if it is preceded by *S V (O)*.

Thus, if we see subpatterns 1, 2, 3 and 4 as statements declaring "constraints" to be "relaxed", this system could be adequately characterized as a **multiple constraint satisfaction system.**

Turn to other relevant ponits. In (1), bracketing is used to denote disjunction. This means that (1) describes both of the following:

(6) a. *Ann asked the way.*
b. *Ann asked him the way.*

Thus, (1) matches syntactic pattern SV(X)O if *the way* matches O. Furthermore, (1) matches another pattern SV(X)O′O if *the* and *way* match O′ and O, on the other, provided that O′ encodes determiner. Details of patterns will be discussed later.

Formation in (1) may serve as a **parse model** for formations such as in (7).

(7) a. *Ann asked (Bill) how to go there.*
b. *Ann asked (Bill) what to pick up.*

(1) serves as one of the parse models for (7), because of parallelisms such as:

(8) i. *the* (as *D*) ≈ *that* (as *C*)
ii. *way* (as *N*) ≈ *Ø to (go there)* (as *M* (for matrix) = *S V*)

I will discuss the notion of parse model in subsequent chapters.

2.1.3 Optimal redundancy in specifications

The entire program of PMA relies on a crucial assumption:

(9) There are **optimally redundant (perceptual) schemas** for syntax.

Optimal redundancy is a notion hard to define, but let me note a few points. Note crucially that an alternative analysis exists that the following table encodes instead of (4).

(10)					
	0.	***Ann***	***asked***	***(him)***	***the way***
	1.	***Ann***	*V*	*(X)*	O
	2.	*S*	***asked***	*(X)*	O
	3.	*S*	*V*	***(him)***	O
	4.	*S*	*V*	*(X)*	***the way***

Thus, the question is, What principles make us to prefer the specification in (4) over the "richer" specification in (10), despite the fact that both systems are redundant?

My answer is that those statements in (10) are **too specific** in that they are too general. Indeed, they are specialized for a particular syntax (and abstract semantics) of *S V (X) O*. To show this, it would be useful to note that the specification in (10) is incompatible with the syntax and semantics of *kiss*, for example. If we are forced to make a pattern matching analysis of *kiss* on the basis of the statements in (10), then we will have the following:

(11)					
	0.	***Ann***	***kissed***	***Ø***	***him***
	1.	***Ann***	*V*	*(X)*	O
	2.	*S*	***kissed***	*(X)*	O
	3.	*S*	*V*	***Ø***	O
	4.	*S*	*V*	*(X)*	***him***

This encoding wrongly states that *kiss* is a verb that instantiates *S V (X) O*, thereby admitting **Ann kissed Mother him* (presumably, in the sense of *Ann kissed him for (the sake of) Mother*).

This suggests that the argument structure of *kiss*, if any, is not *S V (X) O* but *S* ***V*** *O*. This confirms that schemas that are too general do not work properly for our purposes. Useful schemas should be optimally redundant.

What subpatterns are optimal, howevr? PMA claims that optimally redundant schemas take the form of **basic subpatterns**, such as:

(12) *S* ***V*** *(O)* for *V* in general, with elaborations such as:

i.	*S* ***V***	(e.g., *John* ***laugh***)
ii.	*S* ***V*** *O*	(e.g., *John* ***killed*** *his baby*)
iii.	*S* ***V*** *(X) O*	(e.g., *John* ***asked*** *(him) the way*)
iv.	*S* ***V*** *X O*	(e.g., *John* ***gave*** *him the paper*)

(13) ***S*** *V (O)* for *S* in general, with elaborations such as:

i. ***S*** *V* (e.g., ***John** smiled*)
ii. ***S*** *V* O (e.g., ***there** is a mistake in your paper*)

(14) *S V* **O** for *O* in general, with elaborations such as:
i. *S V* **O** (e.g., *You saw* ***him***)
ii. *S V (X)* **O** (e.g., *Ann asked (him)* ***the way***)
iii. *S V* ***(X)*** O (e.g., *Ann asked* ***(him)*** *the way*)
iv. *S V* ***X*** O (e.g., *Ann gave* ***him*** *a bicycle*)
v. *S V X* **O** (e.g., *Ann gave him* ***a bicycle***)

The list here is shown for purposes of illustration, and is not intended to exhaust all basic patterns of English syntax.

It is important to note that some of those basic subpatterns, e.g., those in (12), are analogous to **argument structures** (Grimshaw 1990). The similarity for its great theoretical importance, and some of the interesting issues raised by the similarity are discussed in Chapter 4. I note however that they are not the same. First, there are other basic subpatterns, such as those in (13) and in (14), that have nothing to do with verbs. I argue that subpatterns in (13) and in (14) are irreducible to argument structure.

Second, in PMA, prepositions are encoded as subpatterns.

(15) *S* ***P*** *(O)* for *P* in general, with elaborations such as:
i. *S* ***P*** (e.g., *take it* ***up***)
ii. *S* ***P*** O (e.g., *put it* ***on*** *the desk*)

(15)i describes the class of "intransitive" prepositions, to which I assume the class of "particles" corresponds. Some prepositions such as *off* belong to both (15)i and ii.

It is important to note that most syntactic descriptions can be greatly simplified if the classes of verbs and prepositions are generalized. PMA posits a general class *R* (for relational terms) to subsume *V* and *P*, such that:

(16) *S* ***R*** *(O)* for *R*, = {*V*, *P*}[1]
i. *S* ***R*** for intransitives
ii. *S* ***R*** O for transitives

Admittedly, the generalzation is partial, because no ditransitive prepositions are ever known.

One of the most important consequences of this generalization would be that:

(17) Prepositions have "subjects" of their own.

This has a number of interesting effects. As I will show in Chapters 5 and 6, syntactic descriptions can benefit from this simple generalization.

To those "lexically based" subpatterns discussed above, one should add "functionally based" subpatterns, called **shifters**, such as:

(18) i. ***what*** *S (U) V* δ(*what*)
i′. ***what*** *(U) V*
ii. ***who*** *S (U) V* δ(*who*)
ii′. ***who*** *(U) V*
iii. ***how*** *S (U) V O* δ(*how*)
iv. δ(*that SV*) *V* ***that*** *S V*, where δ(*that SV*) realizes as preparatory

Shifters will be discussed in Section 2.2.2 in some detail, and in Chapter 3 and Appendix A in greater detail.

In addition to those in (18), more lexically specified shifters should exist, such as:

(19) i. *S V* ***pick*** δ_X ***up*** *X*
ii. *S V* ***take*** δ_X ***off*** *X*

Licensers of these (cataphoric) shifters are *up* and *off*, respectively.

These shifters should be distinguished from those in (20), where the leftward displacement of *X* is licensed by *C* (rather than *up* and *off*).

(20) i. *X (C) S V* ***pick*** δ(*X*) ***up***
ii. *X (C) S V* ***take*** δ(*X*) ***off***

These shifters are responsible for so-called topicalization, exemplified by like the following:

(21) a. *The ticket for the concert, John failed to pick up.*
b. *That strange hat, Thelonious did never took off.*

Under some examples given above, I should be noted explicitly that it is still unclear and under scrutiny what subpatterns are possible. No general restrictions on subpatterns are not found yet, though some relevant discussions will be presented in Appendix B relating to the "learnability" of language.

2.2 How Pattern Matching Analysis Works — A More Advanced Analysis

Turn now to a more complicated example for a better understanding of how PMA works.

From a very practical point of view, PMA consists in a method that assigns an encoding scheme in (23) to (22), which PMA assumes is the exact form of (7)b, repeated here for convenience.

(7) b. *Ann asked (Bill) what to pick up.*

(22) *Ann asked (Bill) what Ø$_1$ to pick Ø$_2$ up*

(23)	0.	***Ann***	***asked***	***(Bill)***	***what***	***Ø$_1$***	***to***	***pick***	***Ø$_2$***	***up***
	1.	***Ann***	*V*		*(O)*					
	2.	*S*	***asked***	*(O)*	*O*					
	3.	*S*	*V*	***(Bill)***	*O*					
	4.	*S*	*V*	*(O)*	***what***	*S*	*V*		δ(*what*)	
	5.				*C*	***Ø$_1$***	*V*			
	6.				*C*	*S*	***to***	*V*	*(O)*	
	7.				*C*	*S*	*(U)*	***pick***	*O*	*P*
	8.				*C*	*S*	*(U)*	*V*	***Ø$_2$***	
	9.				*C*	*S*	*(U)*	*V*	*P*	***up***

For the ease of comprehension, I will review below some basic properties of C/D table in (23).

2.2.1 Relativized categorization

Very roughly, *S* corresponds to what is usually called "subject", *V* to (main) "verb", *O* to "object", and *U* to "auxiliary", distinguished from main verb. *C* encodes the class of complementizer (e.g., *that*).

The question arises of how those "labels" are assigned. Some basic points will be briefly discussed in Section 2.3, and more details will be discussed in Chapter 3 and Appendix A.

2.2.2 Shifters

Specification in subpattern 4, *S V **what** S V* δ(*what*), deserves a special mention, since it has to do with a well-known case of **syntactic movement**, which is one of the greatest concerns of syntactic theory.

*S V **what** S V* δ*(what)*, which I will call an (**anaphoric**) **shifter**, gives a "schematic encoding" of *what* available for contexts like:

(24) i. *Ann asked (Bill) ___ to pick Ø up*
ii. *Chris said ___ you can't believe Ø*
iii. *___ you found on my desk is a novel of Borges*

Shifters take the form of either (i) *X Y* δ(*X*), where δ(*X*) is anaphorically bound by *X*, or (ii) δ(*X*) *Y X*, where δ(*X*) is cataphorically bound by *X*. In both cases, δ(*X*) realizes either as a gap, ∅, or as a personal (resumptive) pronoun (e.g., *he*) or an impersonal pronoun (e.g., dummy *it*).

In the case under question, *SV* ***what*** *SV* δ*(what)* is an instance of anaphoric shifter *X Y* δ*(X)* in that *X* = ***what***, *Y* = *S V* and δ(*X*) = ∅, putting aside effects of the matrix *SV* that subordinates it.

Shifters are a special case of **overriders** in that they override certain subpatterns. Specifically, ***what*** *John read* ∅ "overrides" ***John read it***, or ****John read what***. More generally, ***X*** *S V* δ*(X)* overrides ***S V X***, irrespective of the acceptability of ***S V X*** as such. In this sense, overriders are "parasitic" subpatterns that can be effective only when they override other subpatterns.

I will show in subsequent chapters that most real "displacement" phenomena like *wh*-movement are described in terms of such shifters. One may wonder why even PMA needs such "dirty tricks" as shifters. The reason is as follows: Generally, to say that there are movements is to say that there are structures that some principles move "material(s) of", whether they are moved "blindly" or, alternatively, to satisfy conditions on well-formedness, which are independently given. In developing PMA, I found that it is impossible to reduce all kinds of movements into one single sort of notion. Some cases of movements should be described as "effects" of the (logical) satisfaction of well-formedness conditions, which are best understood as "surface structure constraints" in the sense of Perlmutter (1971), or output conditions in the sense of Ross (1967). Arguments in Chapters 5 and 6 will show that effects of cases like Raising, LF movement, are well characterized in this way. Nothing is moved in such cases. Some other cases, subsuming so-called *wh*-movement, are different. To describe them, PMA needs a special kind of subpatterns called overriders.

2.2.3 Structure-building by making use of overlaps

The analysis in (23) claims that the syntax for (22) is determined by **overlaps** among subpatterns 1, 2, ..., 9. Because this is one of the most important assumptions in the proposed framework, let me give a few notes on it.

The C/D table in (23) can be subdivided into two groups of subpatterns, one is the set of {1, 2, 3, 4}, responsible for substring *Ann asked Bill what*, and another is the set of {4, 5, 6, 7, 8, 9}, responsible for substring *what to pick up*. The role of subpattern 4 = *S V (O)* ***what*** *S V* is important because it is shared by the two sets, thereby "bridging" the two groups.

To clarify essential points, it is helpful to notice to abstract properties of a C/D table. For example, consider an abstract formation *F* = ***abcde***. Its minimum syntax is encoded in the following table, where the relative order of units is determined by making use of overlaps among ***a****B*, *A****b****C*, *AB****c***, *A****d****E*, and *AD****e***.

(25)

0.	***a***	***b***	***c***	***d***	***e***
1.	***a***	*B*			
2.	*A*	***b***	*C*		
3.	*A*	*B*	***c***		
4.			*A*	***d***	*E*
5.			*A*	*D*	***e***

Figure 2.1 below visualizes two implicit groupings (two squares) with respect to overlapping at ***c***.

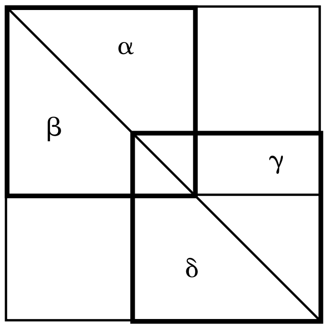

Figure 2.1

Interestingly, the groupings correspond to the domains of "dominance" determined by *b* (= *A**b**C*) and *d* (= *C**d**E*). So, specifications in a C/D table are so "grouped" that there can be as many "squares" as relational terms such as *V* and *P*.

One can find an asymmetry in grouping. In Figure 2.1, fewer specifications are found in area α than in area β, in that β is usually full of specifications. Likewise, fewer specifications are found in area γ than in are δ. This is admittedly a controversial point, but I suspect that it is because co-occurrence restrictions function differently when on the right and on the left. Leaving details to later discussions in Chapter 3, let me give brief notes on it.

PMA distinguishes between two sorts of "licensing". In the $n \times n$ matrix of subpatterns like (25), $r_{i,j}$ ($i < j$) states a "demand" of the ith unit on the jth unit, whereas $r_{i,j}$ ($i > j$) states an "admission" from the ith unit for the jth unit. For example, (25)1 = ***a**B* states that there must appear some unit to match *B* at the right of *a*. Psychologically, this is a "suspension" for a unit of category *B*. This suspension is created by the recognition of ***a**B*. In contrast, such suspension has nothing to do with specifications on the left. For example, (25)3 = *AB**c*** allows units of *A* and *B* sorts to appear in the specified order, without demanding them to be there. In other words, *AB**c*** only admits the realizations of *A* and *B* on the left.

This asymmetry affects the way that implications are utilized. In specifications on left-hand side, implications can be fully utilized. For example, if *B**c*** and ***AB*** are both true, then *AB**c*** must hold. In fact, the exact meaning of *AB**c*** is that *B**c*** holds as far as the condition *AB* is met.

This gives a hint for resolving the language learning paradox. As far as "discovery procedure" is concerned, *AB**c*** is preferable over *B**c***, despite the fact that *AB**c*** is apparently more complex than *Bc*. Note that simpler is not always better. Logically, *Bc* is implicationally weaker, and more importantly, harder to verify, than *ABc*. In other words, more simplicity in structural description should lead to more difficulty in learning what is so described.

This proposed use of (logical) implications in a statement is impossible for specifications on the right-hand side. The reason is that possible implications are not established yet because there are suspensions. This is why we find areas α and γ scarce.

2.2.4 Linguistic effects of overlapping

If there are overlaps among subpatterns, it is possible to make a "chain" of as many subpattern as possible. For illustration, let me give an abstract example of ***abcbcbc***.

(26)	0.	***a***	***b***	***c***	***b***	***c***	***b***	***c***
	1.	***a***	*B*	*(C)*				
	2.	*A*	***b***	*C*				
	3.	*A*	*B*	***c***				
	4.			*A*	***b***	*C*		
	5.			*A*	*B*	***c***		
	6.					*A*	***b***	*C*
	7.					*A*	*B*	***c***

It is easy to identify in this table local groupings, such as:

(27) i. $g_1 = \{2, 3\}$, $g_2 = \{4, 5\}$, and $g_3 = \{6, 7\}$
ii. $g_1' = \{1, g_1\}$, $g_2' = \{3, g_2\}$, and $g_3' = \{5, g_3\}$
iii. $h_1 = \{1, 2\}$, $h_2 = \{3, 4\}$, and $h_3 = \{5, 6\}$

It is clear that those groups have linguistic counterparts, such as follows:

(28) i. g_1 corresponds to a VP, and g_2 and g_3 to "adjuncts".
ii. g' corresponds to "clause", at least potentially.
iii. h corresponds to "generalized matrix".

Though (26) is abstract, it is admitted to find a few hints for linguistics by generalizing it:

(29) i. *A* corresponds to "subject" and *C* to "object".
ii. $D = \{A, C\}$ is the class of NP.
iii. *B* corresponds to class *R* of "relational" terms, i.e., a superclass of *V* and *P*.

The distinction (29)i implicitly encodes the **subject/nonsubject asymmetry**.

2.3 Decomposing Patterns by Diagonalization

In this section, I detail the steps required to arrive at the pattern matching analysis in (23), searching for the syntactic structure of (22). In addition to this, I discuss how **diagonalization** works as a method for decomposition of pattern into subpatterns, or simply **pattern decomposition**.

For example, consider (22), repeated here for convenience.

(22)[=(7)b] *Ann asked Bill what to pick up*.

PMA assumes that (22) corresponds to a formation:

(30) *Ann asked Bill what* $Ø_1$ *to pick* $Ø_2$ *up.*

PMA assumes that (30) results from the following set *W*:

(31) *W* = {*Ann*, *asked*, *Bill*, *what*, $Ø_1$, *to*, *pick*, $Ø_2$, *up*}

By **surface formations**, I mean structures like (22), which are given as phenomenological entities.

Note that what (22) represents, if anything, is phenomenological in its nature, and can never be mental. Thus, phonetics is relevant to (22), and irrelevant to (30). This implies that there must be a function ρ that converts (22) into (30) for comprehension, on the one hand, and another function π that converts (30) into (22) for production.

Conceptually, representations like (30) are analogous to the notion of "surface structures" (or S-structures) in the generative literature. This begs a question. What is the difference of (30) from (22), then? My position is that what is irrelevant for linguistic purposes is orthographic representations like (22) rather than those like (30).

Controversially, PMA counts "gaps", denoted by Ø, as words. Gaps are "less than lexical items", in that they have semantics and phonology of their own, thereby serving as "wild cards" that are able to match any unit.

PMA rejects to gratuitously appeal to syntactic movement, but this does not mean that PMA refuses to recognize gaps. Metatheoretically, there is a wealth of evidence to suggest that gaps are psychologically real. Indeed, there is a gap anywhere some material is felt to be "missing".[2] This makes clear that gaps are mental and could not be readily detected in terms of phonetics.

Theoretically, there is no necessity to assume that gaps are created by syntactic movements. Gaps should be independently motivated. My tentative position is that gaps are **underspecified lexical material, semantically and phonetically**, based on a (mis)interpretation of underspecification theory (Archangeli 1984, 1988).

2.3.1 Encoding contexts for words

Pattern matching analysis assumes that "words" in (31) are nothing but **subpatterns** 1 through 9 in the following, obtained by a method called **diagonalization**.

(32)	1.	***Ann***	*asked*	*(Bill)*	*what*	*Ø*	*to*	*pick*	*Ø*	*up*
	2.	*Ann*	***asked***	*(Bill)*	*what*	*Ø*	*to*	*pick*	*Ø*	*up*
	3.	*Ann*	*asked*	***(Bill)***	*what*	*Ø*	*to*	*pick*	*Ø*	*up*
	4.	*Ann*	*asked*	*(Bill)*	***what***	*Ø*	*to*	*pick*	*Ø*	*up*
	5.	*Ann*	*asked*	*(Bill)*	*what*	***Ø***	*to*	*pick*	*Ø*	*up*
	6.	*Ann*	*asked*	*(Bill)*	*what*	*Ø*	***to***	*pick*	*Ø*	*up*
	7.	*Ann*	*asked*	*(Bill)*	*what*	*Ø*	*to*	***pick***	*Ø*	*up*
	8.	*Ann*	*asked*	*(Bill)*	*what*	*Ø*	*to*	*pick*	***Ø***	*up*
	9.	*Ann*	*asked*	*(Bill)*	*what*	*Ø*	*to*	*pick*	*Ø*	***up***

For expository purposes, we say that this encoding scheme is the diagonalization of (30).

Each subpattern in (32) encodes a word in a **(syntactic) context** (of sentence), or an **environment** for its occurrence. For example, (32)4 encodes the occurrence of *what*, as a "context-free" unit, in the context of *Ann asked Bill* __ *Ø to pick Ø up*. For expository purposes, let $\xi(u)$ denote the context for unit u. With this, it is easy to see that nine subpatterns in (32) specify nine pairs of the form $(u, \xi(u))$, as follows:

(33)
1. (*Ann*, ξ(*Ann*))
2. (*asked*, ξ(*asked*))
3. (*Bill*, ξ(*Bill*))
4. (*what*, ξ(*what*))
5. $(\varnothing_1, \xi(\varnothing_1))$
6. (*to*, ξ(*to*))
7. (*pick*, ξ(*pick*))
8. $(\varnothing_2, \xi(\varnothing_2))$
9. (*up*, ξ(*up*))

One of the most crucial, and unusual, assumptions of PMA is that **(syntactic) contexts themselves can be represented and generalized.** A more explicit assumption is this:

(34) For a given unit u, pair $(\kappa(u), \xi(u))$ gives the proper representation of u, where $\kappa(u)$ specifies u's semantic and phonological **contents**, or more

adequately *u*'s **substance**, and $\xi(u)$ denotes a context for *u*'s occurrence.

Such generalization of contexts has to do with a crucial characteristics of the brain. It is plausible to think that the associative nature of the memory makes words "remember" where they occur. In fact, there is a part/whole relation between u (or $\kappa(u)$) and $\xi(u)$.

What dispenses with phrase markers into which lexical items are inserted is exactly the incorporation of some effect of associative memory into the definition of words. I claim that this is exactly what syntactic structure "emerges" from.

2.3.2 Subindexing convention

The analysis in (32) does not reflect **scale effects**, by which one observes a variety of phrases. For example, note that there are as many segmentations of (22) as different scales on which analysis is made. Some of them are:

(35) i. [*Ann*][*asked*][*Bill*][*what* $Ø_1$ *to pick* $Ø_2$ *up*]
ii. [*Ann*][*asked*][*Bill*][[*what*][$Ø_1$ *to pick* $Ø_2$ *up*]]
iii. [*Ann*][*asked*][*Bill*][[[*what*][[$Ø_1$][*to pick* $Ø_2$ *up*]]]
iv. [*Ann*][*asked*][*Bill*][[[*what*][[$Ø_1$][[*to*][*pick* $Ø_2$ *up*]]]]

To encode the effects of the segmentation in (35)iv, for example, the C/D table in (32) is modified as follows:

(36)									
1.	***Ann***	*asked*	*Bill*	*what*	$Ø_1$	*to*	*pick*	$Ø_2$	*up*
2.	*Ann*	***asked***	*Bill*	*what*	$Ø_1$	*to*	*pick*	$Ø_2$	*up*
3.	*Ann*	*asked*	***Bill***	*what*	$Ø_1$	*to*	*pick*	$Ø_2$	*up*
4.1	*Ann*	*asked*	*Bill*	***what***	$Ø_1$	*to*	*pick*	$Ø_2$	*up*
4.2.1	*Ann*	*asked*	*Bill*	*what*	$\boldsymbol{Ø_1}$	*to*	*pick*	$Ø_2$	*up*
4.2.2.1	*Ann*	*asked*	*Bill*	*what*	$Ø_1$	***to***	*pick*	$Ø_2$	*up*
4.2.2.2	*Ann*	*asked*	*Bill*	*what*	$Ø_1$	*to*	***pick***	$\boldsymbol{Ø_2}$	***up***

Here, *pick Ø up* is analyzed as a simplex, VP-equivalent.

In (36), relevent scale effects are encoded by means of **subindexing** that gives indices such as 4.1, 4.2.1, 4.2.2.1. The following convention is assumed:

(37) Index *i.j.k* indicates:
i. the subpattern so indexed is the *i*th segment at the lowest resolution;
ii. the subpattern is the *j*th subsegment of the *i*th unit; and
iii. the subpattern is the *k*th sub(sub)segment of the *j*th subsegment of the *i*th segment.

With this convention, (36) is understood as an abbreviatory notation for the following:

(38)

1.0.0.0	***Ann***	*asked*	*Bill*	*what*	*Ø*$_1$	*to*	*pick*	*Ø*$_2$	*up*
2.0.0.0	*Ann*	***asked***	*Bill*	*what*	*Ø*$_1$	*to*	*pick*	*Ø*$_2$	*up*
3.0.0.0	*Ann*	*asked*	***Bill***	*what*	*Ø*$_1$	*to*	*pick*	*Ø*$_2$	*up*
4.1.0.0	*Ann*	*asked*	*Bill*	***what***	*Ø*$_1$	*to*	*pick*	*Ø*$_2$	*up*
4.2.1.0	*Ann*	*asked*	*Bill*	*what*	***Ø***$_1$	*to*	*pick*	*Ø*$_2$	*up*
4.2.2.1	*Ann*	*asked*	*Bill*	*what*	*Ø*$_1$	***to***	*pick*	*Ø*$_2$	*up*
4.2.2.2	*Ann*	*asked*	*Bill*	*what*	*Ø*$_1$	*to*	***pick***	***Ø***$_2$	***up***

This implies that the exact meaning of (35)iv, for example, is the following:

(39) [[[[*Ann*]]]][[[[*asked*]]]][[[[*Bill*]]]][[[[*what*]]]][[[[*Ø*$_1$]]]][[[[*to*]]]][[[[*pick Ø*$_2$ *up*]]]]

This may look strange, but the reason will be made clearer by discussion in the next section.

2.3.3 Multiple parallel parsing as self-organization

Pattern matching analysis countenances the idea of **parallel parsing on multiple scales**, in conformity with "connectionist" theories of the mind and the brain. To see this, it will be helpful to appeal to the diagram in Figure 2.2, which shows how *F* = *Ann asked Bill what to pick up* (= (22)) is parsed on nine different scales from the discourse-scale parse at the top to the word-scale parse at the bottom.

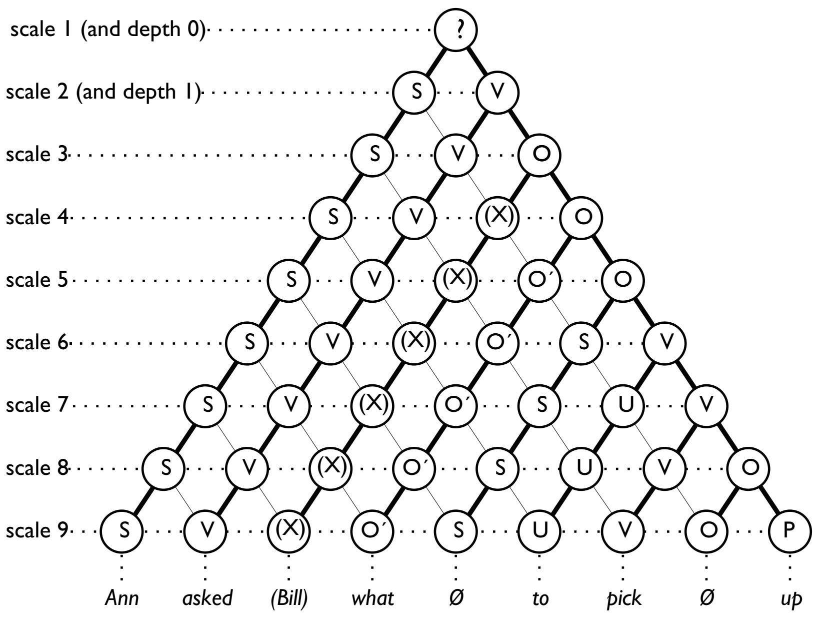

Figure 2.2

The topmost node need not be identified even if it can be the same as O at the rightmost node on scale 5. Thick links indicate units' identities across scales, to which I call **scale-invariance**. For this reason, what this diagram describes is different from what rewrite rules generate. This property should account for the "strange" bracketing in (39). It shows nesting of degree 4 (e.g., [[[[*Ann*]]]]) because it is an analysis on the scale 7 **relative to** the scale 4. Generally, an analysis on the scale *m* relative the scale *n* shows nesting of degree ($m - n + 1$). Thus, the analysis on scale 9 relative to the scale itself if given as (40), without contradiction to (39).

(40) [*Ann*][*asked*][*Bill*][*what*][$Ø_1$][*to*][*pick*][$Ø_2$][*up*]

Note that *F* = *Ann asked Bill what* $Ø_1$ *to pick* $Ø_2$ *up* matches eight patterns on scales 2-9, though some of them are controversial.

(41) i. SV matches *F* on scale 2.
ii. SVO matches *F* on scale 3.
iii. SV(X)O matches *F* on scale 4.
iv. SV(X)O′O matches *F* on scale 5, where O′ is a "dummy" of O, and a sort of "determiner" of SV.[3]
v. SV(X)O′SV, matches *F* on scale 6.
vi. SV(X)O′SUV, matches *F* on scale 7.
vii. SV(X)O′SUVO, matches *F* on scale 8.
viii. SV(X)O′SUVOP, matches *F* on scale 9.

It should be emphasized here that what diagrams like those above specify is conceptually different from what most linguists believe "trees" specify. First of all, in the proposed conception, patterns on different scales **parse in parallel**, though in interaction with each other.

This diagram illustrates how units on different scales correspond to each other. Parsing in terms of multi-scale pattern matching is a sort of **self-organization** that starts from an "disorderly" state, where all possible relations exist, to an "orderly" state, as diagrammed in Figure 2.2.

Thus, **hierarchical organization**, which is usually assumed, is not fully responsible for different parses on different scales. Between two adjacent scales, only "simplification rules" or possibly interaction schemas are posited. Under the assumption that categorization assumes adjacency, rules of simplification are as follows:

(42) Given two adjacent units, *a* and *b*, on scale *i*, either rule (i) of "vacuous" simplification, (ii) of simplification, or (iii) mutation (or replacement) apply:

i. *a* and *b*, without simplification, correspond to a' and b', respectively, on scale $i + 1$; or
ii. *a* and *b*, with simplification, correspond to either a' or b' on scale $i + 1$.
iii. *a* and *b* correspond on scale $i + 1$ to *c* by replacement of *ab* by *c* ($c \neq a, b$).

For an example of mutation, consider the difference in categorization of *to*. On scale 9, where units are words, *to* is categorized as *U*. On other larger scales, it is "merged" into *V*.

Without detailed discussion, I note that the system described in (42), which results in the structure in Figure 2.2, could be characterized as a Lindenmeyer system.[4] I will give a brief discussion on this issue in Chapter 3.

2.4 Composing (Sub)patterns by Superposition

Putting aside scale effects, **columnwise unification**, or equivalently **superposition**, of the subpatterns in (36) results in composite pattern 0 in (43), called **base pattern**, which is equated with the representation of (30).

(43)

0.	***Ann***	***asked***	***Bill***	***what***	***Ø***	***to***	***pick***	***Ø***	***up***
	⥣	⥣	⥣	⥣	⥣	⥣	⥣	⥣	⥣
1.	***Ann***	*asked*	*Bill*	*what*	*Ø*	*to*	*pick*	*Ø*	*up*
2.	*Ann*	***asked***	*Bill*	*what*	*Ø*	*to*	*pick*	*Ø*	*up*
3.	*Ann*	*asked*	***Bill***	*what*	*Ø*	*to*	*pick*	*Ø*	*up*
4.	*Ann*	*asked*	*Bill*	***what***	*Ø*	*to*	*pick*	*Ø*	*up*
5.	*Ann*	*asked*	*Bill*	*what*	***Ø***	*to*	*pick*	*Ø*	*up*
6.	*Ann*	*asked*	*Bill*	*what*	*Ø*	***to***	*pick*	*Ø*	*up*
7.	*Ann*	*asked*	*Bill*	*what*	*Ø*	*to*	***pick***	*Ø*	*up*
8.	*Ann*	*asked*	*Bill*	*what*	*Ø*	*to*	*pick*	***Ø***	*up*
9.	*Ann*	*asked*	*Bill*	*what*	*Ø*	*to*	*pick*	*Ø*	***up***

Unification operator is denoted by ⥣.

In a sense, the most practically relevant question in pattern matching analysis is: What is the procedure to determine details of a **schema-based encoding** like (23), starting from an **item-based encoding** like (43) here?

2.4.1 Reducing redundancy by schematization

The necessary and sufficient conditions of pattern matching of (22) is to obtain a **schematic encoding table** (23), where specifications are made in schematic terms of grammatical functions and/or categories like *S*, *V*, *O*.

(23)

0.	***Ann***	***asked***	*(Bill)*	***what***	*Ø*$_1$	***to***	***pick***	*Ø*$_2$	***up***
1.	***Ann***	*V*		*(O)*					
2.	*S*	***asked***	*(O)*	*O*					
3.	*S*	*V*	***(Bill)***	*O*					
4.	*S*	*V*	*(O)*	***what***	*S*	*V*		d(*what*)	
5.				*C*	***Ø***$_1$	*V*			
6.				*C*	*S*	***to***	*V*	*(O)*	
7.				*C*	*S*	*(U)*	***pick***	*O*	*P*
8.				*C*	*S*	*(U)*	*V*	***Ø***$_2$	
9.				*C*	*S*	*(U)*	*V*	*P*	***up***

Here, *S* encodes subject, *V* verb, *O* object, *U* auxiliary, *D* determiner, and *N* (head) noun.

(23) should be contrasted with the **itemic encoding table** in (32) (and (36)), where specifications are made in terms of lexical items, *Ann*, *asked*, *Bill*, *what*, *Ø*, *to*, *pick*, *up*, *Ø*.

2.4.2 Conversion of token-based to type-based encoding

Thus, the most practically relevant question is: What is the procedure for determination of variables, or categories, so that an analysis arrives at (23)?

My answer is, most abstractly, that determination into a schematic encoding table (23) from an itemic encoding table (36) is to assign appropriate labels to $r_{i,j}$ in (45), under the condition (44).

(44) $$\begin{aligned} \boldsymbol{u}_{1,1} &= \textbf{\textit{Ann}}, \\ \boldsymbol{u}_{2,2} &= \textbf{\textit{asked}}, \\ \boldsymbol{u}_{3,3} &= \textbf{\textit{Bill}}, \\ \boldsymbol{u}_{4,4} &= \textbf{\textit{what}}, \\ \boldsymbol{u}_{5,5} &= \boldsymbol{\emptyset}, \\ \boldsymbol{u}_{6,6} &= \textbf{\textit{to}}, \\ \boldsymbol{u}_{7,7} &= \textbf{\textit{pick}}, \\ \boldsymbol{u}_{8,8} &= \boldsymbol{\emptyset}, \\ \boldsymbol{u}_{9,9} &= \textbf{\textit{up}} \end{aligned}$$

(45) $$\begin{array}{lccccccccc}
0. & \boldsymbol{u}_1 & \boldsymbol{u}_2 & \boldsymbol{u}_3 & \boldsymbol{u}_4 & \boldsymbol{u}_5 & \boldsymbol{u}_6 & \boldsymbol{u}_7 & \boldsymbol{u}_8 & \boldsymbol{u}_9 \\
 & \Uparrow\Downarrow & \Uparrow\Downarrow & \Uparrow\Downarrow & \Uparrow\Downarrow & \Uparrow\Downarrow & \Uparrow\Downarrow & \Uparrow\Downarrow & \Uparrow\Downarrow & \Uparrow\Downarrow \\
1. & \boldsymbol{u}_{1,1} & u_{1,2} & u_{1,3} & u_{1,4} & u_{1,5} & u_{1,6} & u_{1,7} & u_{1,8} & u_{1,9} \\
2. & u_{2,1} & \boldsymbol{u}_{2,2} & u_{2,3} & u_{2,4} & u_{2,5} & u_{2,6} & u_{2,7} & u_{2,8} & u_{2,9} \\
3. & u_{3,1} & u_{3,2} & \boldsymbol{u}_{3,3} & u_{3,4} & u_{3,5} & u_{3,6} & u_{3,7} & u_{3,8} & u_{3,9} \\
4. & u_{4,1} & u_{4,2} & u_{4,3} & \boldsymbol{u}_{4,4} & u_{4,5} & u_{4,6} & u_{4,7} & u_{4,8} & u_{4,9} \\
5. & u_{5,1} & u_{5,2} & u_{5,3} & u_{5,4} & \boldsymbol{u}_{5,5} & u_{5,6} & u_{5,7} & u_{5,8} & u_{5,9} \\
6. & u_{6,1} & u_{6,2} & u_{6,3} & u_{6,4} & u_{6,5} & \boldsymbol{u}_{6,6} & u_{6,7} & u_{6,8} & u_{6,9} \\
7. & u_{7,1} & u_{7,2} & u_{7,3} & u_{7,4} & u_{7,5} & u_{7,6} & \boldsymbol{u}_{7,7} & u_{7,8} & u_{7,9} \\
8. & u_{8,1} & u_{8,2} & u_{8,3} & u_{8,4} & u_{8,5} & u_{8,6} & u_{8,7} & \boldsymbol{u}_{8,8} & u_{8,9} \\
9. & u_{9,1} & u_{9,2} & u_{9,3} & u_{9,4} & u_{9,5} & u_{9,6} & u_{9,7} & u_{9,8} & \boldsymbol{u}_{9,9}
\end{array}$$

But it should be emphasized that what pattern matching analysis attempts to give is not tables like the following, which are insufficiently schematic:

(46)

0.	***Ann***	***asked***	***Bill***	***what***	***Ø***	***to***	***pick***	***Ø***	***up***
1.	***Ann***	*V*	*O*	*C*	*S*	*U*	*V*	*O*	*P*
2.	*S*	***asked***	*O*	*C*	*S*	*U*	*V*	*O*	*P*
3.	*S*	*V*	***Bill***	*C*	*S*	*U*	*V*	*O*	*P*
4.	*S*	*V*	*O*	***what***	*S*	*U*	*V*	*O*	*P*
5.	*S*	*V*	*O*	*C*	***Ø***	*U*	*V*	*O*	*P*
6.	*S*	*V*	*O*	*C*	*S*	***to***	*V*	*O*	*P*
7.	*S*	*V*	*O*	*C*	*S*	*U*	***pick***	*O*	*P*
8.	*S*	*V*	*O*	*C*	*S*	*U*	*V*	***Ø***	*P*
9.	*S*	*V*	*O*	*C*	*S*	*U*	*V*	*O*	***up***

While they are apparently valid in that they are not overspecifications, encodings like this are too specific to provide linguistically (and psychologically) significant generalizations. The reason is, roughly, that subpatterns in (46) are too much "accommodated" to contexts, and they will fail to reflect the "emergence" of syntax. As I will discuss in some detail in Section 2.1.3, and in Chapter 3, Appendices A and B in more detail, the point is that, basically, there is an **optimally redundant** encoding of (22) (and (30)). The encoding can be obtained by reducing redundancy in (46). This point is related to another, equally important point of **optimal schematicity**.

In this light, compare the C/D table in (46) with the one in (23). The difference between their specifications is that, in (23), (i) there are fewer specifications (with more blanks), and (ii) some specifications are different. I will defer detailed discussion of these issues until Chapter 3.

2.4.3 Bridging two groups of subpatterns

In (23), two groupings can be easily recognized. One is the domain of *ask*, from 1 to 4, and another is the domain of *pick (up)*, from 4 to 9. This point deserves a mention, because, phenomenologically, it is rare for a subpattern contains more than four units. This is because the *n*-arity of argument structure has this kind of properties.

It is critical to note that the two groups of *ask* and *(to) pick* are "bridged" by *S V **what** S V* δ(*what*), encoded by subpattern 4 in (23). In a sense, it is a enzyme of merger of *ask* and *pick (up)*.

Some readers may wonder if subpattern 4 in (23) can be replaced by a weaker one, such as:

(23)	0.	***Ann***	***asked***	***Bill***	***what***	*Ø*	***to***	***pick***	*Ø*	***up***
	4′.				***what***	*S*	*(U)*	*V*	δ(*what*)	

But this replacement nullifies our crucial assumption of a redundancy-based description, which enables overlapping. Note that the overlap of *what* with the first domain is necessary for the composition of (30) and therefore (22) to be unique. Furthermore, the overlap must contain *V (X)* because if it lacks *(X)*, it wrongly licenses expressions, such as:

(47) **Ann asked what*$_i$ *Ø to pick Ø*$_i$ *up Bill*

Admittedly, it is redundant to have both (23)4 and 4′. But PMA will not eliminate this kind of redundancy, because a crucial use of it is made in its description of natural language syntax.

2.5 Concluding Remarks

In this chapter, I showed how forms such as *Ann asked (him) the way*, *Ann asked Bill what to pick up*, are composed from respective sets of **subpatterns**, which one may reasonably think are proper representations of words (and idiomatic phrases, constructions). But this identification is possible only when words are thought to be so structured that they remember optimal information of contexts in which they occur. Thus, it is conceptually misguided to treat words as syntactically unstructured, isolated elements that must be inserted into, or associated to, "slots" of certain templates, e.g., phrase markers. If words are what we have called subpatterns, then syntax "emerges" when words combine with each other.

Notes

1. It seems that verbs and prepositions differ minimally as to whether they agree for tense, ignoring the fact that prepositions constitute an almost closed class.

2. It seems that this feeling corresponds to the distinction between syntax and morphology. As I will discuss in Chapter 6 in some detail, there is no gap felt between *was* and *killed* in *The duckling was killed by John*, though there are reasons to believe that the subject of *kill* is missing there.

3. This parse is possible only for cases that Ross (1969) calls **sluicing**, such as follows.

i. *Ann asked Bill how, but he didn't know how to order such wines.*

ii. *I don't know why, but Professor Y. is delighful today.*

iii. *I am sorry, but no one can smoke here.*

Incidentally, this phenomenon can be seen as a special form of right-node raising (Ross 1967; Postal 1974).

4. For details of Lindenmayer systems or *L* systems, see Rozenberg and Salomaa (1980), Rozenberg and Salomaa, *eds.* (1992), and Vitányi (1980). Let me remark only on some relevant properties of *L* systems here. A Lindenmeyer system *G* is a triple $<\Sigma, P, A>$, where:

i. Σ is a set of symbols, without the distinction between terminal and preterminal symbols. Σ can be seen as a union of terminal and preterminal symbols such that $\Sigma = V_N \cup V_T$.

ii. *P* is a set of "productions" of the form $\alpha_1 \cdots \alpha_m \rightarrow \beta_1 \cdots \beta_n$ $(\alpha_i, \beta_i \in \Sigma)$.

iii. *A* is a special symbol, called an "axiom", from which all derivations start. *A* may or may not be in Σ.

Rewrites in an *L* system are said to be "parallel" in that all symbols of the input string have to be rewritten at the same time. For example, $G = <\{a, b\}, \{a \rightarrow aa, b \rightarrow bb\}, ab>$ generates a language $\{ab, aabb, aaaabbbb, ...\}$, i.e., $a^{2^n}b^{2^n}$ $(n \geq 1)$. This means that it is a natural interpretation to see derivational steps in L systems as "generations" in (cellular) development.

Note that it has important effects if an L system has "vacuous" productions of the form $x \rightarrow x$. If $a \rightarrow a$, $b \rightarrow b$ are added to *P* in *G* above, the generated language no longer is $a^{2^n}b^{2^n}$; rather, it is a^+b^+.