# PM, Propositional Model, a Computational Psycholinguistic Model of Language Comprehension Based on a Relational Analysis of Written English

(Summary Paper)

Jerry T. Ball, PhD

www.DoubleRTheory.com Jerry@DoubleRTheory.com

© 2003

# Introduction

PM (Propositional Model) is a computational psycholinguistic model of written language (English) comprehension. It consists of a propositional system of representation and a processing mechanism for constructing propositional descriptions directly from input text. There is no separate process for the construction of syntactic representations, and no distinctly syntactic representations exist. In PM there is no distinction between syntactic and semantic processing, or between syntactic and semantic representations. Nor is there a clear distinction between grammar and lexicon.

PM's system of representation is propositionally and linguistically based. Propositional representations consist of linguistic descriptions of predicates (including predicate modifiers) along with linguistic descriptions of their associated arguments. The main predicate functions as the head of a propositional description. There are two basic types of argument descriptions: (a) object argument descriptions, and (b) propositional argument descriptions that are embedded in higher level propositional descriptions. They give propositional descriptions a recursive potential. Object argument descriptions are descriptions of the objects that participate in propositional descriptions. Object descriptions consist of terms which are base level non-relational elements and functions which are relational elements that modify terms. Terms function as the heads of object descriptions.

Propositional descriptions are perceptually based abstractions of linguistic input and they represent linguistic aspects of structure and meaning. Propositional descriptions contain no nonlinguistic entities. Propositional descriptions are associated with nonlinguistic representations which are constructed in parallel with propositional descriptions during processing. Nonlinguistic representations are perceptually based abstractions of nonlinguistic input and they represent nonlinguistic aspects of structure and meaning.

The basic relationship between linguistic and nonlinguistic representations is one of grounding. Nonlinguistic representations ground the sense and reference of related linguistic representations. Nonlinguistic representations of prototypes and exemplars (i.e., previously encountered instances) ground the sense of corresponding linguistic representations, and nonlinguistic representations. Linguistic representations may be directly related to other linguistic representations, and may gain much of their meaning from such associations, but linguistic representations are ultimately ground in nonlinguistic representations:



Humans have a very general ability to recognize the similarities and to abstract away from the differences between particular experiences. Humans make use of this ability in the creation of mental representations corresponding to their experience. Based on this ability and experience of language, humans construct representations which capture knowledge of language. Such representations vary in their level of abstraction, some capturing very general knowledge of language, and other capturing knowledge of specific linguistic constructions. Once constructed, these mental representations or schemas are available for use in subsequent language processing, with more specific and concrete schemas providing more predictive power than less specific and more abstract schemas.

PM makes use of schemas for the representation of knowledge of language. In PM there is an abstract schema of the form **lobj pred objl** (**pred** is short for predicate and **obj** is short for object description) which represents knowledge about the linear encoding, number and type of arguments which are associated with bivalent or transitive predicates. There is also likely to be a more concrete schema of the form **lobj hit objl**, reflecting specific knowledge about the transitive verb **hit**. And even more concrete schemas like **lobj** *hit the nail on the head*I are possible. Thus, PM assumes the existence of schemas at multiple levels of abstraction. The figure below is a tangled hierarchical diagram of some possible schemas for the verb **hit**. Of interest to note is that a schema which contains specific lexical items might be said to be part of the lexicon (assuming the schema is addressable via the lexical item it contains), whereas a schema which does not contain any specific lexical item might be said to be part of the grammar. But if abstract

schemas like **lobj pred obj** are directly associated with specific lexical items, this distinction loses its force.



PM's processing mechanism operates on the input text from left to right, activating learned schemas which correspond to individual lexical items or larger chunks of text, as it goes along. These schemas in turn establish expectations which both determine the possible structures and drive the processing mechanism. In PM there is effectively no overall grammar and no top down control mechanism—just the local preferences of individual lexical items and larger linguistic units which must be integrated together in the construction of a coherent representation for a piece of text.

### The Theoretical and Historical Basis of PM

From a linguistic perspective, PM's representational and processing commitments are most closely allied with the following linguistic approaches:

- Cognitive Linguistics (Johnson, 1987; Lakoff, 1988, 1987; Langacker, 1987, 1986)
- Case Grammar (Fillmore, 1977, 1971, 1968; Nilsen, 1973; Somers, 1987)
- Valency Grammar (Heringer, 1985; Somers, 1987)
- Functional Grammar (Dik, 1987b; Givon, 1989, 1984; Halliday, 1984)
- Traditional Grammar (Jackson, 1990; Jespersen, 1984, 1965; Quirk, Greenbaum, Leech & Svartvik, 1985, 1982).

Within the framework of Transformational Grammar (Chomsky, 1965, 1957; Radford, 1981), Jackendoff (1983, 1978) has been influential, although in general the basic assumptions of PM are not compatible with those of Transformational Grammar. The basic assumptions of PM are more compatible with those of Government and Binding Theory (Chomsky, 1995, 1988, 1982a, 1982b, 1981; Sells, 1985) than Transformational Grammar, and the advent of Government and Binding Theory is seen as an improvement over its predecessor. Government and Binding Theory is also in some ways more compatible with PM than are Lexical Functional Grammar (Bresnan, 1982, 1978; Sells, 1985) and Generalized Phrase Structure Grammar (Gazdar, Klein, Pullum & Sag, 1985; Sells, 1985). Lexical Functional Grammar and Generalized Phrase Structure Grammar retain some of the undesirable features of Transformational Grammar (e.g., phrase structure rules) that have been eliminated in Government and Binding Theory.

From a psychological perspective, PM is most compatible with psychological approaches which focus on propositional systems of representation and process and which espouse unified theories of cognition:

- ACT-R (Anderson & LeBiere, 1998; Anderson, 1993, 1983, 1976)
- Construction-Integration Model (Kintsch, 1998, 1988, 1977, 1974; Kintsch & van Dijk, 1978)
- Miller, 1978; Miller & Johnson-Laird, 1976
- Mental Models (Johnson-Laird, 1983)
- Clark & Clark 1977 ; Clark & Haviland, 1977 ; Haviland & Clark, 1974
- CAPS (Just & Carpenter, 1987)

The research of Miller and Johnson-Laird (1976) and Miller (1978) has been especially influential on the development of PM's system of representation. The discourse processing models put forward by Kintsch and van Dijk (1978) and Clark and Haviland (1977) have influenced the development of PM's processing mechanism. The psychologically based models of Anderson (1993, 1983, 1976; Anderson & LeBiere 1998), Just and Carpenter (1987), and Kintsch (1998, 1988, 1977, 1974; Kintsch & van Dijk 1978) are the most comprehension treatments of both representation and process

available. The most glaring omission in the research of Kintsch is the lack of a processing mechanism for constructing propositional representations from input texts. PM provides just such a mechanism.

From the perspective of Artificial Intelligence, PM is most indebted to the following:

- Preference Semantics (Wilks 1979, 1975a, 1975b, 1972)
- Conceptual Dependency Theory (Lytinen, 1986; Schank, 1975, 1972; Schank & Abelson, 1977; Wilensky, 1986)
- Conceptual Structures (Sowa, 1984).
- SHRDLU (Winograd 1983, 1972).

Wilks' Preference Semantics set the stage for the development of PM. Schank's Conceptual Dependency theory has provided numerous useful insights, although, the exclusive use of extremely abstract schemas that it espouses is not considered a reasonable model of language comprehension. Winograd's thesis and program (1972) is an impressive achievement and his shift towards a more cognitive orientation (1983) precipitated a similar shift in the development of PM.

## The Representation of Propositional and Object Descriptions

From a relational perspective, it is argued that the clause structure of written English has two basic elements:

- Propositional Descriptions
- Object Descriptions

Propositional descriptions represent the predicate-argument structure of clauses. They consist of a predicate, the head of the propositional description, and zero (in reduced argument constructions) to three arguments. The predicate consists of a main predicate which is typically a verb, adjective or preposition, associated predicate modifiers and perhaps a distinguished predicate specifier (i.e. the first element of the predicate). The arguments to a predicate are of two types: object descriptions and embedded propositional descriptions. Embedded propositional descriptions give propositional descriptions a recursive potential.

In total, nine basic propositional forms have been identified:

He went	pred(obj)	[went(he)]
He kissed me	pred(obj,obj)	[kissed(he,me)]
He gave me it	pred(obj,obj,obj)	[gave(he,me,it)]
Unfortunately, he went	pred(prop <sub>head</sub> )	[unfortunately([went(he)] <sub>head</sub> )]
He believes you like me	pred(obj,prop)	[believes(he,[like(you,me)])]
He kissed me by it	pred(prop <sub>head</sub> ,obj)	[by(kissed(he,me)] <sub>head</sub> ,it)]
He told me you like him	pred(obj,obj,prop)	[told(he,me,[like(you,him)])]
I like you and you like me	pred(prop <sub>head</sub> ,prop <sub>he</sub>	ad)

### [and([like(I,you)]<sub>head</sub>,[like(you,me)]<sub>head</sub>)] He ate, I sang and she sat pred(prop<sub>head</sub>,prop<sub>head</sub>,prop<sub>head</sub>) [and([ate(he)]<sub>head</sub>,[sang(I)]<sub>head</sub>,[sat(she)]<sub>head</sub>)]

In addition, there are four forms of predicate modification:

I am sad	pred{pred <sub>head</sub> }(obj)	[am{sad}(I)]
He went over it	pred{pred <sub>head</sub> }(obj,+obj)	[over{went}(he,it)]
He hit and kicked it	pred{pred <sub>head</sub> ,pred <sub>head</sub> }	[and{hit,kicked}(he,it)]
He hit, kicked and bit it	pred{pred <sub>head</sub> ,pred <sub>head</sub> ,pred	d <sub>head</sub> }
		[and{hit,kicked,bit}(he,it)]

In terms of notation, **pred** is a predicate description, **obj** is an object description, and **prop** is a propositional description. ()'s are used to circumscribe the arguments of the predicate. In these abstract schemas, the surface order of the predicate relative to the arguments is left unspecified, however, the order of the arguments is significant. The two forms **pred(obj,prop)** and **pred(prop<sub>head</sub>,obj)** differ in this latter respect. All of the basic propositional forms result in propositional descriptions when the predicate and arguments are instantiated. []'s are used to circumscribe a complete propositional description. ()'s are used to circumscribe a complete propositional description. ()'s are used to circumscribe a complete object description functions as an argument in one of the basic forms. Thus, the propositional description [went(he)] would be represented as [went((he))] if the inner ()'s had not been dropped. For embedded propositional descriptions the []'s will not be dropped since the ()'s surrounding the arguments of a predicate suggest that the arguments are object descriptions and not propositional descriptions. For predicate modification, {}'s are used to circumscribe the predicate being modified.

The basic propositional forms are annotated to reflect the **head** of the resulting form whenever the main predicate of the form is not the head. There are four instances of propositional modification and four instances of predicate modification where this is the case.

Object descriptions represent the function-term structure of noun phrases. They consist of a term which is the head of the object description, optional functional modifiers and an optional functional specifier. This term is typically a noun, although lexical items which are typically other parts of speech (e.g. adjective, present participle) may also be used as terms. This term may be modified by one or more functions which correspond to presupposed relations that are associated with the term. Functions may themselves be modified by function modifiers (e.g. adverbs). Finally, an optional function specifier which explicitly establishes the referential nature of the object description may occur.

Functions are the linguistically relational components of object descriptions and as such are largely responsible for determining the relational structure of such descriptions. In PM, functions (like predicates) can be classified in terms of the number and type of arguments they take. There are three basic types of arguments to functions: **terms**, other

**functions**, and **object descriptions**. Further, it is assumed that functions take at most two arguments—with the exception of conjunctions which can take (at least) three arguments—and that the arguments to a function must be of the same type. Additionally, it is assumed that the type of the function argument combination depends on the type of the arguments such that an object description combines with a function and forms an object description, a term combines with a function and forms either a complex term or an object description, and a function combines with a function to form a function. Finally, it is assumed that one or more of the arguments acts as the head of the resulting function argument structure. Given these assumptions, the following functional types have been identified:

Func <term<sub>head&gt;</term<sub>	=> Term
Func <term<sub>head,Term&gt;</term<sub>	=> Term
Func <term<sub>head,Term<sub>head</sub>&gt;</term<sub>	=> Term
Func <term<sub>head, Term<sub>head</sub>, Term<sub>head</sub>&gt;</term<sub>	=> Term
Func-Mod{Func <sub>head</sub> }	=> Func
Func-Mod{Func <sub>head</sub> ,Func <sub>head</sub> }	=> Func
Func-Mod{Func <sub>head</sub> ,Func <sub>head</sub> ,Func <sub>head</sub> }	=> Func
Func-Spec <term<sub>head&gt;</term<sub>	=> Obj
Func(Obj <sub>head</sub> )	=> Obj
Func(Obj <sub>head</sub> ,Obj)	=> Obj
Func(Obj <sub>head</sub> ,Obj <sub>head</sub> )	=> Obj
Func(Objhead,Objhead,Objhead)	=> Obj

In this notation, **func** is a function, **func-mod** and **func-spec** are subtypes of function corresponding to function modifiers and term specifiers, **obj** is an object description, **term** is a term, and the subscript  $_{head}$  marks the head of the resulting description. Terms are further identified by bracketing them with <>'s. The arguments of a function which are themselves functions are bracketed with { }'s. Arguments which are full object descriptions are bracketed with ()'s.

Determiners perform a function for object descriptions similar to the function that the auxiliary verbs perform for propositional descriptions. Both serve to complete the description of which they form a part by providing specifications which serve to fix the reference of the description. This correspondence provides a basis for generalizing about the structure of propositional and object descriptions in a way similar to the generalization put forward in **X-bar Theory** (see the description in Sells 1985). It suggests the existence of three levels of representation: (1) a base level, (2) an intermediate unspecified level above the base level, and (3) a fully specified level. For propositional description, and the fully specified level is the specified propositional description. For object descriptions, the base level is the term, the intermediate level is the unspecified object description, and the fully specified level is the fully specified level is the fully specified level is the term, the intermediate level is the unspecified object description.

The examples below explicate the possible forms:

Func <term<sub>head&gt;<sub>Term</sub></term<sub>	old man
	<old<man><sub>head</sub>&gt;</old<man>
Func <term<sub>head,Term &gt;<sub>Term</sub></term<sub>	can of beans
	<of<can <sub="">head,beans&gt;&gt;</of<can>
Func <term<sub>head,Term<sub>head</sub>&gt;<sub>Term</sub></term<sub>	prince and princess
	<and<prince head="" head,princess="">&gt;</and<prince>
Func <term<sub>head,Term<sub>head</sub>,Term<sub>head</sub>&gt;<sub>Term</sub></term<sub>	
	man, woman and child
	<and<man head="" head,child="" head,woman="">&gt;</and<man>
Func-Spec <term<sub>head&gt;<sub>Obj</sub></term<sub>	the man
·	(the <man><sub>head</sub>)</man>
Func(Obj <sub>head</sub> ) <sub>Obj</sub>	all the men
	(all(the <men><sub>head</sub>))</men>
Func(Obj <sub>head</sub> ,Obj) <sub>Obj</sub>	the man in the park
-	(in(the <man><sub>head</sub>,the<park>))</park></man>
Func(Obj <sub>head</sub> ,Obj <sub>head</sub> ) <sub>Obj</sub>	the man and the woman
	(and(the <man><sub>head</sub>,the<woman><sub>head</sub>))</woman></man>
Func(Obj <sub>head</sub> ,Obj <sub>head</sub> ,Obj <sub>head</sub> ) <sub>Obj</sub>	the man, the woman and the child
(and(the <ma< td=""><td>an&gt;<sub>head</sub>,the<woman><sub>head</sub>,the<child><sub>head</sub>))</child></woman></td></ma<>	an> <sub>head</sub> ,the <woman><sub>head</sub>,the<child><sub>head</sub>))</child></woman>
Func-Mod{Func <sub>head</sub> } <sub>Func</sub>	very old
	very{old <sub>head</sub> }
Func-Mod{Func <sub>head</sub> ,Func <sub>head</sub> } <sub>Func</sub>	black and white
	and{black <sub>head</sub> ,white <sub>head</sub> }
Func-Mod{Funchead,Funchead,Funchead}Funchead	<sub>nc</sub> red, white and blue
	and{red <sub>head</sub> ,white <sub>head</sub> ,blue <sub>head</sub> }

Despite the fact that PM's propositional and object representations are described in relational terms, PM representations are actually very consistent with the more detailed grammatical and functional treatments of Jespersen (1984, 1965) and Quirk, Greenbaum, Leech and Svartvik (1985, 1972). This consistency makes it possible to extend PM to handle grammatical details which are not currently modeled. For example, the more detailed representational system developed in Jespersen (1984) can be added to PM with only minor reworking. Further, the mapping from parts of speech and grammatical categories to PM's propositional categories is straightforward enough to consider the use of on-line dictionaries to bootstrap the development of functional language processing systems.

### The Processing of Propositional and Object Descriptions

The processing mechanism operates on the input text from left to right, identifying lexical units via the activation of associated schemas in long-term memory, and selecting from among the activated schemas. The activation and selection of schemas corresponding to the relational units in a piece of text is critical to the processing mechanism, since the relational units determine how selected schemas can be integrated together in the shortterm memory buffers. The processing mechanism also makes use of general rules of English word order as a guide to determining how to integrate various schemas. Schemas which are consistent with these general rules of word order need not explicitly specify that information. For example, English word order is such that the argument to a function which takes a single argument typically occurs to the right of that function in the input stream. Functions which follow this general rule can be represented by schemas which do not explicitly specify the location of the argument relative to the function. However, more specific schemas are likely to be associated with functions which do not follow the general rule. Thus, the function **ago** is likely to have a schema associated with it in which the argument is specified to occur before the function as in a week ago. The use of general rules of word order in combination with explicitly marked exceptions to the rules is consistent with the position of Pinker (2000).

In PM, function words (e.g. determiners) and prepositions are treated as relational lexical items and they are important to the processing mechanism. For example, the occurrence of a determiner establishes the context of an object description, whereas the occurrence of an auxiliary verb establishes the context of a propositional description. Likewise, the occurrence of a preposition marks the end of the previous propositional or object description and sets up an expectation for the occurrence of a subsequent object description. Thus, these often ignored sentence constituents are important markers for the processing mechanism. That they are short words in English is a reflection of the efficient encoding of these often occurring constituents, and not an indication of their minor importance for understanding. Their omission in contexts like newspaper headlines, often leads to difficulty in determining the meaning of those headlines.

The processing mechanism makes use of effective strategies during the processing of input texts. For lexical items which evoke strong preferences to be used in specific ways, those preferences can be immediately realized based solely on the prior context. For lexical items which evince multiple different uses, subsequent context may also be necessary to determine which use is relevant in the given context. The processing mechanism does not make extensive use of backtracking, but to the extent that it does, that backtracking essentially involves jumping back to the beginning of some chunk of text and is not like the formalized backtracking of a computational system like Prolog.

One automatic and two control processes have been introduced above. They include (a) an automatic spreading activation process, (b) a control process for selecting activated schemas from long-term memory and placing them in short-term memory buffers, and (c) a control process for integrating selected schemas in the short-term memory buffers. The automatic process of spreading activation and the control process by which activated

schemas are selected and placed in short-term memory will not be discussed further. Only the output of these two processes will be considered. Anderson (1983) and Anderson and LeBiere (1998) present a treatment of these two processes which is largely compatible with PM.

The remainder of this paper is concerned with a description of the process of integrating selected schemas in the short-term memory buffers. That process can be described algorithmically in terms of the individual processing steps required to integrate the schemas which have been selected for further processing. We begin the discussion of this process by walking through the steps involved in the processing of the following English sentence:

#### The boy likes the girl.

The processing of this sentence begins with the activation and selection of a schema corresponding to the first lexical unit. The word **the** is identified and a schema which reflects its status as a function which takes a term for and argument and forms an object description is selected. The word order of English is such that the term that goes with a function like **the** almost invariably occurs to the right of that function in the input text. Thus, the preference is for the function to await the appearance of this term before combining with it to form an object description. As a result of this preference, the function the is retained in a short-term memory buffer with its argument uninstantiated and the processing of this function is temporarily halted. The processing of the next lexical unit begins. The word boy is identified and determined to be a term. Since the function **the** is awaiting the occurrence of a term, it combines with the term **boy** to form the object description (the<boy>). This object description is retained in a short-term memory buffer for use in subsequent processing. The individual components of the object description (e.g., the and boy) are not separately maintained in the short-term memory buffer once they are combined, since it is assumed that set of short-term memory buffers has too limited a capacity to retain such individual components and since the separate maintenance of these components in short-term memory buffers would interfere with the processing mechanism. In general, it will be assumed that none of the components of propositional descriptions are separately maintained in memory once they are combined, however, there are reasons for suggesting that the subject may represent an exception to this assumption. For example, separately representing the subject in a shortterm memory buffer makes it more salient than the other arguments of a predicate and provides one way of explaining the asymmetry in the status of subjects and objects in English. The processing mechanism continues by identifying the next lexical unit. The word likes is identified and is determined to be a predicate which takes two object descriptions for arguments. The word order of English is such that the first object description of predicates like likes typically occurs before the predicate in the input text. This means that the first argument to the predicates likes should be available in a shortterm memory buffer for instantiation at the time the predicate is encountered. The processing mechanism searches the short-term memory buffers for an object description, identifies the object description the<boy> and instantiates it as the first argument of likes, forming the partially complete propositional description likes(the<boy>,Obj)prop.

The order of search is based on the recency of processing of the schemas in the shortterm memory buffers, such that the set of short-term memory buffers function very much like a stack. Thus, if there were two object descriptions separately available in short-term memory buffers at the time the predicates likes was processed, the most recently processed object description would be selected for instantiation as the first argument. Object descriptions which are not separately available in short-term memory buffers (i.e., arguments which have already been instantiated into a relational structure), are typically overlooked by the search mechanism. According to English word order, the second argument of a predicate which takes two object descriptions typically occurs after it in the input text. Therefore, the predicate must await the occurrence of the second object description. The partially completed propositional description is retained in a short-term memory buffer. The processing mechanism continues by identifying and processing the next lexical unit. The word **the** is identified and its functional schema is retrieved from memory. Since the predicate likes is expecting an object description and not a function, the function the cannot be instantiated as the second argument of likes. And since the term which the function the takes, is expected to occur to its right in the input text, this function must likewise wait to be completed. The processing mechanism continues by identifying the next lexical unit, girl, and determines it to be a term. Since the function the is expecting a term, it combines with the term girl to form the object description (the<girl>). This object description is in turn instantiated as the second argument of the predicate likes, forming the propositional description [likes(the<boy>,the<girl>)]. At the completion of processing of the input text, this propositional description is retained in a short-term memory buffer for subsequent processing.

An abbreviated notation for representing the steps in that processing is introduced. The first step involved in the processing of the preceding example is expressed in this notation as:

#### the => the<T1><sub>obj</sub>

On the left hand side of the arrow is the lexical item **the** and on the right hand side is the schema for the which is activated and selected for subsequent processing. In this schema, the is identified as a function which takes a term for an argument and forms an object description. At the time the schema is selected, the argument to the function is uninstantiated. Uninstantiated arguments are represented as variables beginning with a capital letter (e.g. T), which identifies the type of the variable, and ending in a single digit (e.g., 1) which functions to distinguish the variable from other variables of the same type. The possible types are T (term), F (function), PR (predicate), O (object description), and **P** (propositional description). The >'s around T1 also identify the type of T1 as a term. ()'s will be used to circumscribe object descriptions and the arguments of propostional descriptions, []'s will be used to circumscribe propositional descriptions and {}'s will be used to circumscribe the arguments of predicate and function modifiers. Since the schema for **the** is not yet a fully specified object description, it is not circumscribed by ( )'s, instead the subscript obj marks the schema as ultimately forming and object description. Functions which combine with terms to form complex terms will be circumscribed with <>'s to distinguish them from object descriptions when they are fully

specified and subscripted with  $_{term}$  when they are not. Partially completed propositional descriptions will be subscripted with  $_{prop}$  to reflect their fully specified type. Partially completed function modifiers will be subscripted with  $_{func}$  and partially completed predicate modifiers will be subscripted with  $_{pred}$  to reflect their fully specified type.

Continuing the processing leads to

boy => <boy> the<T1><sub>obj</sub> + <boy> => (the<boy>)

The left hand side need not always be a lexical item. It may also consist of a collection of schemas which are integrated as shown on the right hand side. The latter processing step above represents the instantiation of the term **boy** as the argument of the function **the**. Note that until the term **boy** is instantiated into the function **the**, there is no object description. Continuing with the processing gives:

likes => likes(O1,O2)prop

followed by

```
likes(O1,O2) prop + (the<boy>) => likes(the<boy>,O2) prop
```

Continuing to completion we have:

the => the<T2> obj
girl => <girl>
the<T2> obj + <girl> => (the<girl>)
likes(the<boy>,O2) prop + (the<girl>) => [likes(the<boy>,the<girl>)]

The final representation for the sentence is a propositional description consisting of the predicate **likes** along with the two object descriptions (**the**<**boy**>) and (**the**<**girl**>) which are the arguments of the predicate.

## The Processing of Object Descriptions

The following object description includes a term, function, function modifier (i.e. adverb) and function specifier and introduces a range of considerations in the processing of object descriptions:

### The black robed judge.

In my preferred interpretation of this phrase, it is the **robe** and not the **judge** which is **black**. The processing of the preferred interpretation for this phrase proceeds as follows:

the => the<T1><sub>obj</sub>

 $black => black {F1}_{func}$ robed => robed<T2> term black {F1} func + robed<T2> term => black {robed}<T2> term judge => <judge> black {robed}<T2> + <judge> => <black {robed}<judge>> the<T1> obj + <black {robed}<judge>> => (the<black {robed}<judge>>)

In the preferred interpretation of this expression, **black** functions as an adverb (i.e. function modifier) rather than an adjective, since it modifies the adjective (or participial) **robed** (which is a function) rather than the noun **judge**. In PM, adverbs—when they occur as elements of object descriptions—are functions which take functions as their argument. Of course, **black** normally prefers to be an adjective rather than an adverb and it is unlikely that the processing of this expression is as straightforward as the example shows. Rather, the occurrence of the word **robed** after **black** triggers the preference for the adverbial use which must somehow replace the original preference. Thus, we might modify the processing as follows:

 $\label{eq:the_star} \begin{array}{l} the => the<T1>_{obj} \\ black => black<Ta>_{term} \\ robed => <robed<T2>_{term} \\ black<Ta>_{term} + robed<T2>_{term} => black{F1}_{func} + robed<T2>_{term} \\ black{F1}_{func} + robed<T2>_{term} => black{robed}<T2>_{term} \\ judge => <judge> \\ black{robed}<T2>_{term} + <judge> => <black{robed}<judge>> \\ the<T1>_{obj} + <black{robed}<judge> => (the<black{robed}<judge>>) \\ \end{array}$ 

The adverbial status of **black** is an interesting consequence of the participial use of **robed** which is derived from the noun **robe**. Thus, **black** would typically function as an adjective in modifying **robe** as in

The black robe. The robe is black.

However, since **robed** is itself functioning as a modifier in **the black robed judge**, **black** takes on the adverbial role of modifying a modifier. Note that there is an alternative (if dispreferred) reading of this expression in which **black** modifies **judge** and not **robed**. In this dispreferred reading **black** retains its more typical use and the expression can be processed as followed:

the => the<T1><sub>obj</sub> black => black<Ta><sub>term</sub> robed => <robed<T2><sub>term</sub> judge => <judge> robed<T2><sub>term</sub> + <judge> => <robed<judge>> black<Ta><sub>term</sub> + <robed<judge>> => <black<robed<judge>>>

### the<T1><sub>obj</sub> + <black<robed<judge>>> => (the<black<robed<judge>>>)

In the resulting representation, the "**robed judge**" is "**black**". Selection of this dispreferred reading can be facilitated by punctuation as in

### The black, robed judge

where the comma serves to emphasize the dual modification of **judge** by **black** and **robed**. And use of the conjunction **and** would make the dual modification even more explicit

### The black and robed judge

although the conjunction of such disparate properties is unusual.

### The Processing of Propositional Descriptions

As an example of the processing of propositional descriptions consider

### He walked slowly

in which the adverb **slowly** is assumed to be functioning as a predicate modifier (Thomason & Stalnaker, 1973). This sentence can be processed as followed:

 $\label{eq:he} \begin{array}{l} he => (he) \\ (he) + walked => walked(O1)_{prop} \\ slowly => slowly\{PR1\}_{pred} \\ walked(O1)_{prop} + slowly\{PR1\}_{pred} => slowly\{walked\}(O1)_{prop} \\ (he) + slowly\{walked\}(O1)_{prop} => [slowly\{walked\}(he)] \end{array}$ 

Note that the instantiation of **he** as the argument of **walked** is delayed until after the adverb **slowly** is processed. This avoids the need for the predicate modifier **slowly** to intrude on the propositional description [**walked**(**he**)] which would be created if **he** were immediately instantiated into **walked**. This is example of where delayed instantiation of the subject avoids the creation of a structure that would otherwise have to be dismantled.

If the adverb **slowly** is fronted as in

### Slowly, he walked

and **slowly** is treated as a predicate modifier, it is more difficult to avoid creation of an invalid structure. This may explain why fronted adverbs tend to be treated as sentential modifiers. The sentence can be processed as follows:

### slowly => slowly{PR1} pred

he => (he) (he) + walked => walked(O1) prop (he) + walked(O1) prop => [walked(he)] slowly{PR1}pred + [walked(he)] => [slowly{walked}(he)]

In the last step the predicate modifier **slowly** must be integrated into a complete propositional description which is atypical for the processing mechanism. An alternative last step is to convert **slowly** to a sentential adverb leading to:

slowly{PR1}pred + [walked(he)] => slowly(P1) + [walked(he)]
slowly(P1)prop + [walked(he)] => [slowly([walked(he)])]

Chafe (1970) argues that the fronting of adverbs like **slowly** has just this effect of converting them into sentential modifiers. One possible difference between **he walked slowly** and **slowly**, **he walked** is the suggestion that in the latter sentence that the act of walking was slow to start. This distinction is quite subtle. Indeed, I had overlooked it myself until it was pointed out to me (K. Paap, Oct. 1991, personal communication). While the position of the predicate **slowly** has only a subtle effect on meaning in the above sentences, the positional effect is more prominent in sentences containing multiple predicates. Consider

### He stopped walking slowly Slowly, he stopped walking.

The fronting of **slowly** results in a preferred interpretation of the second sentence which differs substantially from the first sentence. The processing of the first sentence can proceed as follows:

```
\label{eq:he} \begin{array}{l} he \Rightarrow (he) \\ stopped \Rightarrow stopped?? \\ stopped + walking \Rightarrow stopped\{PR1\} + walking(O1) \\ walking(O1) + slowly \Rightarrow walking(O1) + slowly\{PR2\}_{pred} \\ walking(O1)_{prop} + slowly\{PR2\}_{pred} \Rightarrow slowly\{walking\}(O1)_{prop} \\ stopped\{PR1\} + slowly\{walking\}(O1)_{prop} \Rightarrow \\ stopped\{slowly\{walking\}\}(O1)_{prop} = > \\ [stopped\{slowly\{walking\}\}(O1)_{prop} = > \\ [stopped\{slowly\{walking\}\}(he)] \end{array}
```

In the resulting representation **stopped** modifies **slowly{walking}** and not just **walking**. The second sentence can be processed as:

```
slowly => slowly{PR1}
he => (he)
stopped => stopped??
stopped + walking => stopped{PR1} + walking(O1)
stopped{PR1} + walking(O1) prop => stopped{walking}(O1) prop
```

```
(he) + stopped{walking}(O1) prop => [stopped{walking}(he)]
slowly{PR1} + [stopped{walking}(he)] =>
slowly(P1) prop + [stopped{walking}(he)]
slowly(P1) prop + [stopped{walking}(he)] => [slowly([stopped{walking}(he)])]
```

In this representation **slowly** modifies the propositional description **[stopped{walking}(he)]** giving a clearly different meaning.

Auxiliary verbs, modal auxiliaries and negatives are all typically predicate modifiers. Consider

### He could not have been walking slowly

which can be processed as follows:

```
he => (he)
could => could{PR1}<sub>pred</sub>
not => not{PR2}<sub>pred</sub>
have => have{PR3} pred
been => been{PR4}<sub>pred</sub>
been{PR4}pred + walking => been{PR4}pred + walking(O1) prop
slowly => slowly{PR5}pred
walking(O1) prop + slowly{PR5} => slowly{walking}(O1) prop
been{PR4}pred + slowly{walking}(O1) prop => been{slowly{walking}}(O1) prop
have{PR3 pred + been{slowly{walking}}(O1) prop =>
              have{been{slowly{walking}}}(O1) prop
not{PR2}<sub>pred</sub> + have{been{slowly{walking}}}(O1) prop =>
              not{have{been{slowly{walking}}}}(O1) prop
could{PR1}pred + not{have{been{slowly{walking}}}}(O1) prop =>
              could{not{have{been{slowly{walking}}}}}(O1) prop
(he) + could{not{have{been{slowly{walking}}}}}(O1) prop =>
               [could{not{have{been{slowly{walking}}}}}(he)]
```

The need to stack so many predicate modifiers could exceed the capacity of the shortterm memory buffers. It may be that there is a mechanism for combining predicate modifiers to avoid this stacking. Assuming such a mechanism (at least for predicate modifiers that occur before the main predicate), processing can proceed as follows:

```
\label{eq:pred} \begin{array}{l} he => (he) \\ could => could \{PR1\}_{pred} \\ not => not \{PR2\}_{pred} \\ could \{PR1\}_{pred} + not \{PR2\}_{pred} => could \{not \{PR2\}\}_{pred} \\ have => have \{PR3\}_{pred} \\ could \{not \{PR2\}\}_{pred} + have \{PR3\}_{pred} => could \{not \{have \{PR3\}\}\}_{pred} \\ been => been \{PR4\}_{pred} \end{array}
```

```
\label{eq:sould} \begin{aligned} & \operatorname{could} \{\operatorname{not} \{\operatorname{have} \{\operatorname{PR3}\}\} \}_{\operatorname{pred}} + \operatorname{been} \{\operatorname{PR3}\}\} \}_{\operatorname{pred}} \\ & \operatorname{could} \{\operatorname{not} \{\operatorname{have} \{\operatorname{PR3}\}\}\} \}_{\operatorname{pred}} + \operatorname{walking} => \\ & \operatorname{could} \{\operatorname{not} \{\operatorname{have} \{\operatorname{PR3}\}\}\} \}_{\operatorname{pred}} + \operatorname{walking}(\operatorname{O1})_{\operatorname{prop}} \\ & \operatorname{slowly} => \operatorname{slowly} \{\operatorname{PR5}\}_{\operatorname{pred}} \\ & \operatorname{walking}(\operatorname{O1})_{\operatorname{prop}} + \operatorname{slowly} \{\operatorname{PR5}\} => \operatorname{slowly} \{\operatorname{walking} \{\operatorname{O1})_{\operatorname{prop}} \\ & \operatorname{could} \{\operatorname{not} \{\operatorname{have} \{\operatorname{been} \{\operatorname{PR3}\}\}\} \}_{\operatorname{pred}} + \operatorname{slowly} \{\operatorname{walking} \{\operatorname{O1})_{\operatorname{prop}} => \\ & \operatorname{could} \{\operatorname{not} \{\operatorname{have} \{\operatorname{been} \{\operatorname{slowly} \{\operatorname{walking} \}\}\}\} (\operatorname{O1})_{\operatorname{prop}} \\ & \operatorname{could} \{\operatorname{not} \{\operatorname{have} \{\operatorname{been} \{\operatorname{slowly} \{\operatorname{walking} \}\}\}\} (\operatorname{O1})_{\operatorname{prop}} \\ & \operatorname{could} \{\operatorname{not} \{\operatorname{have} \{\operatorname{been} \{\operatorname{slowly} \{\operatorname{walking} \}\}\}\} (\operatorname{O1})_{\operatorname{prop}} \\ & \operatorname{could} \{\operatorname{not} \{\operatorname{have} \{\operatorname{been} \{\operatorname{slowly} \{\operatorname{walking} \}\}\}\} \} (\operatorname{he}) ] \end{aligned}
```

On my preferred reading of this sentence **slowly** is within the scope of the negative **not**. That is, what is negated is **walking slowly** and not just **walking**. Had the main predicate **walking** been integrated with the negative **not** before being integrated with **slowly**, the result would be a dispreferred reading:

### [slowly{could{not{have{been{walking}}}}}(he)]

Since negation is not factored out in PM as it is in the predicate calculus, negation will interact with sentential and predicate modification in interesting ways. Consider

### He did not stop walking slowly Slowly, he did not stop walking.

The latter sentence is somewhat difficult to interpret since it is not clear what it means to not stop walking in a slow manner. However, the first sentence has a clear interpretation (i.e. "it is not the case that he stopped walking slowly") which can be represented by

### [did{not{stop{slowly{walking}}}}(he)]

As a final example of the processing of sentences containing multiple predicate and propositional modifiers, consider

### Unfortunately, he did not stop walking slowly.

This sentence contains a sentential modifier **unfortunately** which must finally be integrated with the rest of the sentence. A shortened version of the processing is shown below:

```
unfortunately => unfortunately(P1) prop
he did not stop walking slowly => [did{not{stop{slowly{walking}}}}(he)]
unfortunately(P1) prop + [did{not{stop{slowly{walking}}}}(he)] =>
[unfortunately([did{not{stop{slowly{walking}}}}(he)])]
```

In the processing of predicate modifiers, PM adheres to the **Main Predicate Proximity Principle**. This principle says that the order of occurrence of predicate and propositional modifiers determines their scope relative to each other and to the main predicate-for predicates occurring on the same side of the main predicate. According to this principle, it is not possible (or is at least highly dispreferred in the unmarked case) for the predicate not to take on a wider scope than the predicate did or the predicate unfortunately in this This principle is discussed in detail in Ball (forthcoming). example. Besides representational considerations, the processing mechanism provides additional support for this principle, since the Main Predicate Proximity Principle follows from the assumption that the elements in short-term memory buffers are ordered in terms of the recency of processing. That is, since predicates closer to the main predicate and to its left (i.e. prior to it) will have been processed more recently than predicates that are further from the main predicate and to its left, their arguments will be instantiated first when those arguments become available and predicates closer to the main predicate will have smaller scope as a result. The relative scoping of predicates on differing sides of the main predicate is not determined by this principle, although it appears that predicate adverbs to the right of the main predicate tend to have smaller scope than auxiliaries, modals and negatives to the left, and sentential adverbs tend to have larger scope than these same elements regardless of the side.

We can consider the effect on meaning of the position of the word slowly. Consider

Slowly, he stopped walking He slowly stopped walking He stopped slowly walking He stopped walking slowly.

In the first sentence, **slowly** is likely functioning as a propositional modifier since it occurs outside the propositional description **he stopped walking** leading to:

```
[slowly([stopped{walking}(he)])]
```

In the second sentence, **slowly** occurs within the subject argument and is likely functioning as a predicate modifier leading to:

```
[slowly{stopped{walking}}(he)]
```

In the third sentence, the location of **slowly** suggest two possible structures depending on which verb it modifies:

[slowly{stopped}{walking}(he)] (he stopped slowly...walking)
[stopped{slowly{walking}(he)] (he stopped...slowly walking)

In the fourth sentence, **slowly** prefers to modify **walking** leading to:

```
[stopped{slowly{walking}}(he)]
```

However, with emphasis this can be changed to

```
[slowly([stopped{walking}(he)])] (he stopped walking...slowly)
```

### Low Level Semantic Influence

Semantic information that is not currently modeled in PM is assumed to play an important role in resolving certain types of ambiguities (e.g. prepositional phrase attachment, resolution of verb object argument preferences). This is an open area of research, but Latent Semantic Analysis (LSA) techniques (Landauer & Dumais, 1997; Kintsch, 1998) offer prospects for providing the kind of low level associations between lexical items needed to resolve such amibiguities. On the other hand, adequate mechanisms for the processing of corresponding nonlinguistic representations are not yet available and this remains a gap in PM. Nonetheless, it is assumed that a reasonable level of performance in language understanding can be achieved with available techniques.

# Bibliography

Anderson, J. R. (1976). Language, Memory and Thought. Hillsdale, NJ: LEA.

- Anderson, J. R. (1983). *The Architecture of Cognition*. Cambridge, MA: Harvard University Press.
- Anderson, J. R. (1993). Rules of the Mind. Hillsdale, NJ: LEA.
- Anderson, J. R. & C. LeBiere (1998). *The Atomic Components of Thought*. Hillsdale, NJ: LEA.
- Ball, J. (forthcoming). PM, Propositional Model, a Computational Psycholinguitic Model of Language Comprehension Based on a Relational Analysis of Written English.
- Bresnan, J. (ed.) (1982). *The Mental Representation of Grammatical Relations*. Cambridge, MA: The MIT Press.
- Bresnan, J. (1978). "A Realistic Transformational Grammar." In *Linguistic Theory and Psychological Reality*. Edited by M. Halle, J. Bresnan & G. A. Miller. Cambridge, MA: The MIT Press.
- Chafe, W. (1970). *Meaning and the Structure of Language*. Chicago: University of Chicago Press.
- Chomsky, N. (1995). The Minimalist Program. Cambridge, MA: The MIT Press.
- Chomsky, N. (1988). Language and problems of knowledge: the Managua lectures. Cambridge, MA: The MIT Press.
- Chomsky, N. (1982a). "On the Representation of Form and Function." In *Perspectives* on Mental Representation. Edited by J. Mehler, E. Walker & M. Garrett. Hillsdale, NJ: LEA.
- Chomsky, N. (1982b). Some Concepts and Consequences of the Theory of Government and Binding. Cambridge, MA: The MIT Press.
- Chomsky, N. (1981). Lectures on Government and Binding. Dordrecht-Holland: Foris.
- Chomsky, N. (1970). "Remarks on nominalization." In *Readings in Transformational Grammar*. Edited by R. Jacobs & P. Rosenbaum. Boston: Ginn.

Chomsky, N. (1965). Aspects of the Theory of Syntax. Cambridge, MA: The MIT Press.

Chomsky, N. (1957). Syntactic Structures. The Hague: Mouton.

- Clark, H. (1983). "Making sense of nonce sense." In *The Process of Language Understanding*. Edited by G. Flores d'Arcais & R. Jarvella. NY: John Wiley.
- Clark, H. & S. Haviland (1977). "Comprehension and the Given-New Contract." In *Discourse Production and Comprehension*, Volume 1, pp. 1-39. Edited by R. Freedle. Norwood, NJ: Ablex.
- Dik, S. (1987b). "Some Principles of Functional Grammar." In *Functionalism in Linguistics* pp. 81-100. Edited by R. Dirven & V. Fried. Philadelphia: John Benjamin.
- Fillmore, C. (1968). "The case for case." In *Universals in Linguistic Theory*. Edited by E. Bach & R. Harms. Chicago: Holt, Rinehart and Winston.
- Fillmore, C. (1971). "Some Problems for Case Grammar." In Monograph Series on Language and Linguistics – 22<sup>nd</sup> Annual Roundtable. Edited by R. O'Brien. Washington, DC: Georgetown University School of Language and Linguistics.
- Fillmore, C. (1977). "The Case for Case Reopened." In *Syntax and Semantics*, Volume 8. Edited by P. Cole. NY: Academic Press.
- Gazdar, G. E. Klein, G. Pullum & I. Sag (1985). *Generalized Phrase Structure Grammar*. Cambridge, MA: Harvard University Press.
- Givon, T. (1984). *Syntax: a Functional-Typological Introduction*. Amsterdam: John Benjamins Publishing Company.
- Givon, T. (1989). Mind, Code and Context. Hillsdale, NJ: LEA.
- Halliday, M. (1984). An Introduction to Functional Grammar. Cambridge, MA: Harvard University Press.
- Haviland, S. & H. Clark (1974). "What's new? Acquiring new information as a process in comprehension." *Journal of Verbal Learning and Verbal Behavior*, 13.
- Heringer, H, (1985). "The Verb and its Semantic Power: Association as a Basis for Valence Theory." *Journal of Semantics*, Volume 4, pp. 79-99.
- Jackendoff, R. (1978). "Grammar as Evidence for Conceptual Structure." In *Linguistic Theory and Psychological Reality*. Edited by M. Halle, J. Bresnan, & G. A. Miller. Cambridge, MA: The MIT Press.

Jackendoff, R. (1983). Semantics and Cognition. Cambridge, MA: The MIT Press.

- Jackson, H. (1990). *Grammar and Meaning: a Semantic Approach to English Grammar*. NY: Longman.
- Jespersen, O. (1965). *The Philosophy of Grammar*. NY: W. W. Norton & Company.
- Jespersen, O. (1984). Analytic Syntax. Chicago: The University of Chicago Press.
- Johnson, M. (1987). The Body in the Mind. Chicago: The University of Chicago Press.
- Johnson-Laird, P. (1983). Mental Models. Cambridge, MA: Harvard University Press.
- Just, M. & P. Carpenter (1987). *The Psychology of Reading and Language Comprehension*. Boston, MA: Allyn and Bacon.
- Kintsch, W. (1974). The Representation of Meaning in Memory. Hillsdale, NJ: LEA.
- Kintsch, W. (1977). Memory and Cognition. NY: John Wiley & Sons.
- Kintsch, W. (1988). "The Role of Knowledge in Discourse Comprehension: a Construction-Integration Model." *Psychological Review*, 95, pp. 163-182.
- Kintsch, W. (1998). *Comprehension, a Paradigm for C ognition*. NY: Cambridge University Press.
- Kintsch, W. & T. van Dijk (1978). "Toward a Model of Text Comprehension and Production." *Psychological Review*, 85, pp. 363-394.
- Lakoff, G. (1987). *Women, Fire and Dangerous Things*. Chicago: The University of Chicago Press.
- Lakoff, G. (1988). "Cognitive Semantics." In *Meaning and Mental Representation*. Edited by U. Eco, M. Santambrogio & P. Violi. Indianapolis: Indiana University Press.
- Langacker, R. (1986). "An Introduction to Cognitive Grammar." *Cognitive Science*, 10, pp. 1-40.
- Langacker, R. (1987). *Foundations of Cognitive Grammar*, Volume 1. Stanford, CA: Stanford University Press.
- Landauer, T. & S. Dumais (1996). "A solution to Plato's problem: The Latent Semantic Analysis theory of the acquisition, induction, and representation of knowledge. *Psychological Review*, 104, pp. 211-240.
- Lytinen, A. (1986). "Dynamically combining syntax and semantics in natural language processing." In *Proceedings AAAI 1986*.

- Miller, G. A. (1978). "Semantic Relations among Words." In *Linguistic Theory and Psychological Reality*. Edited by M. Halle, J. Bresnan & G. A. Miller. Cambridge, MA: The MIT Press.
- Miller, G. A., & P. Johnson-Laird (1976). *Language and Perception*. Cambridge: MA: Harvard University Press.
- Pinker, S. (2000). Words and Rules, the Ingredients of Language. NY: HarperCollins.
- Quirk, R., S. Greenbaum, G. Leech, & J. Svartvik (1985). A Comprehensive Grammar of the English Language. London: Longman.
- Quirk, R., S. Greenbaum, G. Leech, & J. Svartvik (1972). A Grammar of Contemporary English. London: Longman.
- Radford, A. (1981). Transformational Syntax. NY: Cambridge University Press.
- Schank, R. (1972). "Conceptual Dependency." Cognitive Psychology, pp. 82-123.
- Schank, R. (1975). Conceptual Information Processing. Amsterdam: North Holland.
- Schank, R., & R. Abelson (1977). Scripts, Plans, Goals and Understanding. Hillsdale, NJ: LEA.
- Sells, P. (1985). *Lectures on Contemporary Syntactic Theories*. Chicago: University of Chicago Press.
- Somers, H. (1987). Valency and Case in Computational Linguistics. Edinburgh: Edinburgh University Press.
- Sowa, J. (1984). Conceptual Structures: Information Processing in Mind and Machine. Reading, MA: Addison-Wesley Publishing Company.
- Thomason, R., & R. Stalnaker (1973). "A Semantic Theory of Adverbs." *Linguistic Inquiry*, 4, pp. 195-220.
- Wilensky, R. (1986). "Some Problems and Proposals for Knowledge Representation." Report No. UCB/CSD 86/294. Computer Science Division (EECS), University of California, Berkeley.
- Wilks, Y. (1972). *Grammar, Meaning and the Machine Analysis of Language*. London: Routledge & Kegan Paul.

- Wilks, Y. (1973). "An Artificial Intelligence Approach to Machine Translation." In *Computer Models of Thought and Language*. Edited by R. Schank and K. Colby. San Francisco: Freeman.
- Wilks, Y. (1975a). "Preference Semantics." In *Formal Semantics*. Edited by E. Keenan. NY: Cambridge University Press.
- Wilks, Y. (1975b). "An Intelligent Analyzer and Understander of English." *Communications of the ACM*, 18, pp. 264-274.
- Wilks, Y. (1979). "Frames, Semantics and Novelty." In *Frame Conceptions and Text Understanding*. Edited by D. Metzing. Berlin: de Gruyter.
- Winograd, T. (1972). Understanding Natural Language. NY: Academic Press.
- Winograd, T. (1983). Language as a Cognitive Process: Syntax. Reading, MA: Addison-Wesley.