SESSION XI MODELING COMPLEX PROCESSES

N. John Castellan, Jr., *Presider*

A computer model of inflection learning

MICHAEL J. LIEBHABER
University of Kansas, Lawrence, Kansas

This paper presents a computer program that creates and modifies inflectional paradigms. A paradigm is a linguistic device used to index lexical entries that share a common inflection along a given dimension such as tense or case. Paradigms are useful because they offer a convenient method for predicting the form of a novel word and for explaining some aspects of language development. The model presented here is based on Pinker's (1984) inflection learning theory. The model covers a subset of the rules for learning inflectional paradigms that are outlined in the theory. The primary concern of the model is to understand how elements that are crucial to the theory, such as lexical entries and feature equations, may interact to form and expand word-specific paradigms. The model also represents an attempt to assess implementation details not specified by the theory. A brief overview of the model is given, followed by a sample run. Finally, some practical and theoretical considerations, such as the structure of paradigms, paradigm assignment to novel words, and suppletion, are discussed.

This project grew out of an attempt to model the inflection learning theory proposed by Pinker (1984). The model captures only a portion of the theory; therefore, this paper is not intended to be a critique of the proposed theory or of lexical-functional grammar (see Kaplan & Bresnan, 1982) upon which the theory rests. The purpose of the model is to provide a means for exploring practical and theoretical ideas related to inflection learning specifically and to model building in general. The program is written in Intellect-ul LISP on a SAGE II micro-computer.¹

The portion of the theory that the model covers is how children learn inflections for words. An inflection is a morphological alteration of a word that changes that word's grammatical category (e.g., the change from present to past tense). The theory modeled here proposes that children, as well as adults, use paradigms to extract, store, and produce inflections. A paradigm is a notational device for listing all of the inflections for a particular word.

Pinker uses two types of paradigms, word-specific and general, to represent inflectional information about words. Word-specific paradigms list the entire inflected form of a word (e.g., *walked*), whereas general paradigms list only the inflection (e.g., *-ed*). General paradigms, then, can be used during language processing to predict the correct inflection for words that share similar characteristics. Recent experimental evidence (Stemberger & Mac-

Whinney, 1986) supports the psychological validity of separate storage of whole inflected words and of inflectional affixes.

The program processes sentences word by word from left to right. There is no practical limit, other than computer memory restrictions, to the kind of words or the number of words per sentence that may be entered. Obviously, there are theoretical constraints on the input, such as segmentation, word order, and so on. This model does not "understand" what it is reading in that it does not assign a syntactic structure and semantic interpretation to the input.

ASSUMPTIONS OF THE THEORY

A major assumption underlying the inflection learning theory is that people use paradigms to encode some aspects of lexical knowledge. Pinker (1984) cited several reasons for assuming that language learners use paradigms to keep track of inflections. Primary among them is that paradigms can account for developmental phenomena, such as children's restricted initial usage of inflections on words and segmentation errors. In addition, paradigms account for learning of the zero, *-0*, morpheme; that is, they offer an explanation for the fact that some irregular verbs are learned as easily as regular verbs.

Grammatical Relationships

Another key element of the theory is grammatical relationships, including verb type, tense, aspect, and thematic roles. The relationships are assumed to be innate. Only

¹Address correspondence to Michael J. Liebhaver, Child Language Program, University of Kansas, 1043 Indiana, Lawrence, KS 66044.

Table 1
Grammatical Relationships Known by the Computer Model

Names	Values
Verb Type	Transitive, Intransitive
Tense	Past, Present
Aspect	Ongoing, Completed
Number	Singular, Plural
Thematic Role	Transitive-Agent, Transitive-Patient, Intransitive-Actor, Goal, Beneficiary
Object	Theme
Gender	Male, Female
Person	1st, 2nd, 3rd

a subset of the relationships that are proposed in the theory are used in the model. These are shown in Table 1.

Input

Much attention has been given to the type and amount of input that is necessary and sufficient for someone to learn a language. This is true in both data-driven (e.g., DeVilliers, 1985; Schwartz & Terrell, 1983; Snow, 1977) and model-driven (e.g., Selfridge, 1986; Wexler, 1982) child language research. Unfortunately, the input that is specified by most studies is either vague, from a computational point of view, theoretically naive, from an observational and empirical point of view, or both.

Pinker (1984) assumed that children can extract meanings of words and the relationships among them from a stream of input. Additionally, he stated that children can determine grammatical features, such as person and tense, from the context, both linguistic and nonlinguistic, in which the word appears. Therefore, the words that are input to the model are enriched with information about the meaning, including thematic roles (e.g., agent or instrument), of the word.

The model is essentially given a description of the context of the input. The argument for including contextual information in the input sentence is that children, with the possible exception of infants and those with severe handicaps, are aware of their surroundings. It is also assumed that children are able to infer meaning from discourse contexts (Pinker, 1981; Schlesinger, 1977; Wex-

ler & Culicover, 1980). Therefore, the contextual information allows the model to be aware of the utterance context.

Lexical Entries

The nature of lexical entries has also been the subject of research (Carey, 1978; Fowler, Napps, & Feldman, 1985; MacKay, 1976; Morton, 1969; Taft & Hambly, 1985). Lexical entries and the amount and kind of information contained in them are at the center of language processing for the model. The problem for this model, and for computer models in general, is to achieve a balance between theory and practicality when specifying the form and structure of lexical entries. That is, how much information is stored in an entry besides the phonological form of a word? Or at what point does a lexical entry become indistinguishable from other knowledge representations? The program assumes that lexical entries contain the word itself, pointers to the word's features and paradigm, and pointers to semantically related words.

REPRESENTATION OF LINGUISTIC DATA

The model was programmed in LISP. LISP was chosen because it allows flexible variable assignment and usage. Variable names, in addition to variable values, can be used as data in LISP. Furthermore, variables can be assigned properties that can be used to describe the attributes of a variable or class of variables. LISP seemed to offer a more psychologically plausible vehicle for representing data than other, generally available, programming languages.

Some of the major variables are shown in Table 2 and are discussed below. They are feature, lexical, and sentence variables. The use of these variables is described under the discussion of the model's processing. Linguistic data are represented in the program as individual values, lists of values, and properties of values and lists. Values are data that are assigned to a variable, and properties are data that describe features of a variable. For example, a sentence has a list of words as its value, whereas

Table 2
Names, Values, and Property Lists of Major Variables in the Model

Variable = Value	Property = Value
Fx = None	Feature Equation = (<innate feature name> = <input feature value>)
Lx = None	Name = <word> Part-of-Speech = <NOUN, VERB, ?> Strength = <1, 2, 3, ..., 100> Feature Equation = (<Fx> ...) Word-Specific Paradigm = <Sx> Context = (<input context word> ...)
LEXICON = (L1 L2 L3 ... LN)	None
SENTENCE = (<word-N>)	None
where <wordX> = (<word> <stress> (<feature value> ...) (<context word> ...))	

Note—x is a number generated by LISP when it creates a variable. It starts at 1 and increments sequentially for each new variable.

a lexical entry is defined only by its properties, or a set of features. The choice between value and property is relatively arbitrary to the program. However, the representations were chosen to correspond with plausible natural representations.

Feature Equations

A feature equation is formed by linking an innate grammatical feature and an input feature value, which are critical elements in the model since they connect innate and environmental features and are used to build the paradigms for the entry. Feature equations are properties of feature variables. The feature variable names that are used by a particular lexical entry are stored in the feature list of that lexical entry.

Lexical Entries and the Lexicon

A lexical entry is a combination of properties of a lexical entry variable. The properties describe various features of a word: the word itself; its part of speech, strength, and associated feature equations; the word-specific paradigm; and the context. The part of speech is inferred from natural categories (e.g., words that have thematic role markers are usually nouns, those representing actions are verbs, etc.). Strength is an indication of how often the word has been accessed. A lexical entry can have more than one feature equation. Only equation and paradigm variable names are kept in a lexical entry, not the values. Each lexical entry variable name is stored on a list called the lexicon.

Input Sentence

Sentences are lists of lists; each sublist represents the information about a word in the sentence. Each word sublist contains the word, a stress indicator, a feature value, and some context words. Item 1 is an example of a two-word intransitive sentence, "Billy jumped":

((Billy 3 (intrans-actor) (friend))
(jumped 1 (past) (hop over))) (1)

Figure 1 shows the lexical representation for *jumped*.

Paradigms

Paradigms are represented in Pinker's (1984) model as multidimensional arrays of morphemes. The dimensions

Figure 1. Expanded lexical entry and word-specific paradigm for *jumped*.

of the array are the feature names. Feature values form levels within each dimension. Morphemes are said to be in a cell of the paradigm. The cells are referenced by dimensions, such as TENSE, and levels, such as PAST. These are illustrated in Figure 1 by the one-dimension paradigm, shown in LISP format, assigned to the word-specific paradigm variable S3. A paradigm must have at least one dimension, each dimension must have at least one level, and each level can have one or more lexical entries.

Paradigm variables are illustrated in Table 3. The program represents paradigms as lists of lists. Every dimension of a paradigm is a sublist of the paradigm, and every level of a given dimension is a sublist of that dimension list. Finally, the cell is a sublist of the level list and con-

Table 3
Names, Values, and Property Lists of Paradigm Variables

Variable = Value	Property = Value
Sx = (<word-specific paradigm>)	Stem = <word stem> Change = <0,1,2, ..., 100> GP = <Gx>
Gx = (<general paradigm>)	None
where <word-specific paradigm> = (<dimension1> (<levelA> (<Lx> ...)) ... (<levelZ> (<Lx> ...))) through (<dimensionN> (<levelA> (<Lx> ...)) ...)	
and <general paradigm> = (<dimension1> (<levelA> (<affix> ...)) ... (<levelZ> (<affix> ...))) through (<dimensionN> (<levelA> (<affix> ...)) ...)	

Note—x is a number generated by LISP (see Table 2). See the text for a discussion of other variables.

tains lexical entry variable names, Lx. Dimensions and levels appear only once in a paradigm; otherwise they would be redundant, as in item 2:

incorrect (TENSE (PRESENT (L2)) (PRESENT (L8)))
correct (TENSE (PRESENT (L2 L8))) (2)

Word-specific paradigm variables, Sx, have properties that identify the stem word, the number of times the stem has changed, and the associated general paradigm.

PROCESSING

Paradigms are formed in the inflection learning theory by (1) selecting a feature value (e.g., PLURAL) from the input utterance; (2) putting the value into a new feature equation, such as NUMBER = PLURAL, which gets added to the lexical entry (LE); (3) transforming the feature equation into a word-specific paradigm (WSP); and (4) removing word stems from the WSP to create a general paradigm (GP).

Pinker (1984) proposed 11 rules for accomplishing the above four steps. The rules are grouped according to their functions: setting up word-specific paradigms (I1-2),

removing incorrect dimensions (I3), setting up general paradigms (I4-I6), maintaining consistency among word-specific and general paradigms (I7), and adding new values to paradigms (I8-I11). The program includes rules I1, I2, I5, and parts of I8 through I11. The rules imply a flow of processing from I1 to I11, but the order is probably more an artifact of the development of the theory than it is an essential part of the process. Ordering is relevant only within a group (e.g., "setting up paradigms"), not between groups.

The essential steps performed by the model are in Figure 2 and are briefly reviewed here by way of an example. Figure 2 also shows input and output files and external routines. The innate lexical knowledge is entered first to initialize the model's memory. That knowledge represents the grammatical relations from Table 2 and some initial lexical entries and their word-specific paradigms. The initial values were necessary to test the modification of existing entries and paradigms. The model can function without the initial lexical entries. Only the grammatical relations are necessary to "prime" the model.

Example Run

The model was given the following sentence:

((Abby 1 (intrans-actor) (friend))
(tezam 1 (completed) (sleep))) (3)

The example follows the nonword *tezam* through the model. The result of the run will be a new lexical entry for *tezam* and revised paradigms.

The program processes item 3, one word at a time, using the following 7 steps. The first 4 steps process the word at the level of the lexical entry. Step 5 places the word in a WSP, and Step 6 forms a GP from the WSP.

1. Match the input feature value, COMPLETED, with its corresponding innate feature name, ASPECT. The match is constrained by the innate grammatical relationships defined in Table 2. If a match cannot be made, then enter another value and try matching again, or quit the program. Feature names cannot be entered since they are assumed to be innate.
2. Use the feature name and value to make a feature equation for the word. The resulting equation is: (ASPECT = COMPLETED).
3. Look for the word in the lexicon. If the word cannot be found, then create a new LE by generating a new LE variable, filling the new LE with the word, strength, and context from the input. Assign part of speech based on input feature value. Assign a WSP to the LE by doing the following:
 - (a) Use the WSP of a word that is already in the lexicon and that is related semantically, morphologically, or phonologically to the input word.
 - (b) If a related word is not found, then make a new WSP. The WSP is empty and will be filled in later.

Figure 2. Overview of model files and processing flow.

Otherwise, quit the program if a lexical entry was not found and a new one was not created for some reason.

The new lexical entry is: (TEZAM VERB 1 () S12 (SLEEP)).

4. Add the feature equation to the LE only if:

- (C1) it is not a duplicate of one that is already attached to the LE, or
- (C2) the feature name is in one of the LE's feature equations and it is a thematic role feature.

Constraint C1 prohibits redundant feature equations within a LE, and constraint C2 allows only thematic role features to have multiple equations. For example, a word such as *boy* can have agent and patient roles, but it cannot be singular and plural since plural requires a different form, *boys*.

The lexical entry is now: (TEZAM VERB 1 (F9) S12 (SLEEP)).

5. Add the lexical entry to its WSP by doing the following:

- (a) If the feature name *is not* a dimension of the WSP, then add it as a new dimension of the WSP. Add the feature value as a new level in the new dimension. Put the LE into its cell in the new level, replace the old WSP with the new one, and go on to 5d.
- (b) If the feature name *is* a dimension of the WSP, but the feature value *is not* a level in this dimension, then add the feature value as a new level of the dimension. Put the LE into its cell in the new level. Replace the old dimension with the revised dimension, and the old WSP with the new one. Go to 5d.
- (c) If the feature name *is* a dimension of the WSP and the feature value *is* a level in this dimension, then replace the LE in the cell with the new LE if:
 - (C3) the stress of the new LE is greater than the strength of the existing LE in the cell, or
 - (C4) the LE in the cell is preemptable.

Next, put the LE into its cell in the new level. Replace the old level, dimension, and WSP with the revised values. Go to 5d.

Constraint C3 is the program's version of the unique entry principle (UEP). The UEP limits the number of words that can occupy a cell in a paradigm (one word in most cases). The program is not flexible when it applies the UEP, and the word with the greater strength will always be placed in the cell. A cell entry is said to be preemptable, as in C4, when the entry is a prediction made from the affixes in the GP. Predicted entries can be preempted by words in the actual input.

(d) WSP processing is finished. The complete WSP for *tezam*, and related words, is in Table 4. Go on to Step 6.

6. If the WSP for the input word has more than one dimension, then try to form a GP with the following steps:

- (a) If the WSP already has a GP associated with it, then retrieve the stem of the word.
 - (C5) If the stem has not changed recently, then:
 - (1) Form the affix for the word by removing the stem from the word.
 - (2) Retrieve the LE from the cell, at the appropriate dimension and level, of the GP.
 - (3) If there is not a LE to retrieve, then add the affix to the GP and go to Step 7.
 - (4) If the new and old affixes are identical, then replacing the existing LE would be redundant, so go to Step 7.
 - (5) If this point is reached, then the old and new affixes conflict, so the entire WSP must be reanalyzed. Go to Step 6b.

(b) If this WSP does not have a GP associated with it, it must be reanalyzed. Make a new GP variable if necessary. Attach its name to the property list of the WSP (see Table 3). This will associate the WSP with the GP.

(c) Only abstract a WSP into a GP if:

- (C6) the WSP has more than one dimension, or
- (C7) the WSP has one dimension, but has more than one level in that dimension.

Otherwise, go to Step 7.

Constraints C6 and C7 exclude one-word paradigms from GP analysis. They correspond to rule I9(c) in the theory (Pinker, 1984, p. 198). The next step, 6d, forms the GP from the WSP and is illustrated in Figure 3 for the TENSE dimension.

(d) Perform the following for each dimension in the WSP:

- (1) For each level in the current dimension, extract the string of letters that are common

Table 4
Word-Specific Paradigm for the Example Run of the Model

Number	Present Tense		Past Tense	
	Ongoing	Completed	Ongoing	Completed
Singular	extezamin	extezam	tezamin	tezam
Plural	exteamin	exteam	teamin	team

Note—"Ongoing" and "Completed" are aspect features.

Figure 3. Word stem and affix extraction process for the TENSE dimension from the word-specific paradigm in the example run of the model.

to all of the words in this level. This will give one "word" for each level in this dimension.

- (2) Look through each of the extracted letter strings for the STEM and AFFIXes. There will be one stem and one or more affixes (one affix for each level). The STEM of this word is the word formed from the letters that are common to all of the strings. The AFFIXes are the portions of the strings that are unique to that string.

General paradigm processing is now complete. The entire GP for the WSP in Table 4 is:

((TENSE (Present (ex-)) (Past (0-)))
 (ASPECT (Ongoing (-in)) (Completed (-0)))
 (NUMBER (Singular (-z-)) (Plural (-0-))))

7. The processing for the current word is completed. If there are more words in the sentence, then get the next word in the sentence. Return to Step 3 and process the word. Otherwise, the INFLECT program is finished.

DISCUSSION

The model deviates from the rules proposed in the theory in three ways. First, formation of feature equations is driven by features in the input rather than by sampling from innate features (see Step 1). This was done to eliminate the need for resampling the innate feature list until a correct match was found. Second, the model assumes that stems of words are not known. This poses some

difficulties for GP abstraction, in Step 6, which are discussed below. Third, conflicting entries in a GP cell cause the entire GP to be reanalyzed to ensure that all of the affixes are correct (Step 6a5). The unique entry principle and preemptability are employed in the theory to remove conflicts. These methods assume that other entries in the GP are unaffected by the new word causing the conflict.

Some practical and theoretical considerations became apparent during the course of the development of the model. Practical considerations center on methods for representing paradigms and on procedural questions. Word-specific paradigm selection and general paradigm formation are two major theoretical issues.

Practical Considerations

Programming the model was not difficult overall. However, the most complex process, general paradigm abstraction, was not coded in the model. Decisions concerning how different kinds of data should be represented were made with theoretical and procedural constraints in mind. Procedurally, directly assigned values are more easily accessed than are property lists. However, theoretical conceptualizations of human cognition, and the model's representation of it, outweigh the inconvenient access.

The representation of word-specific paradigms underwent a few revisions. The initial attempt at a model used trees to represent paradigms, feature names and values occupied alternate levels within the tree, and lexical entries were the terminal elements. This method of representation is acceptable when feature name, value, and word correspondences are low (i.e., paradigms with few words and feature names). However, a paradigm tree must be expanded breadth-wise (i.e., broadened) to maintain intra-level consistency (e.g., for all elements on a level to have the same feature name or value) when new elements are added. This representation strategy quickly depletes available memory. An alternative strategy, breaking paradigms into lists of similar elements, is used in the current version of the model.

The change in paradigm representation led to a change in the algorithms that added new elements to the paradigms. The resulting methods were simpler since they had to manipulate only one portion of a list (i.e., the dimension being updated). The old method had to perform the same operations, but a given dimension could be repeated a number of times across a level in the tree.

Other programming issues that need to be addressed are associating a word with a WSP, extracting stems and affixes from WSPs, and deciding what constitutes a match for the association and abstraction process. Procedurally, associating a word with a WSP may be better handled by parallel semantic, morphologic, and phonologic searches than by the current sequential method. Extraction can be accomplished by a pattern matching process that remembers its relative position in each word and also monitors

success or failure of preceding matches. These considerations have theoretical counterparts, and the programming solutions should be driven by the theory.

Theoretical Considerations

Theoretical problems that need to be resolved include selecting word-specific paradigms, explicating suppletive handling procedures, and circumventing problems caused by lack of data.

Selection of word-specific paradigms. The selection of the correct WSP for new words is not addressed by the theory. The program, for a given input word, searches context lists of all lexical entries for words that match the input context. If one is found, the program assigns the WSP to the new word. In other words, the program performs a semantic search. Searches along the morphological and phonological dimensions of a word can also be made. These are acknowledged, but not implemented, in the program. Pinker (1984, p. 194) also acknowledged the importance of morphological and phonological information in WSP assignment. The sequential logic and interpretation of these matches will be difficult to implement. For example, what match should be done first, and what does it mean when one match (e.g., phonological) is made, but not any others?

An additional consideration is the degree of similarity required before a match can be made. The exactness of the match is not controlled in the program, so the selected lexical entry may or may not be appropriate. This may be satisfactory initially, and it does account for semantic overextension errors, such as the one made by the beginning reader in the following example:

- Mother: What are you reading?
 Child: A story about keeping water.
 M: Keeping water?
 C: Yeah, around the house.
 M: What's the title of the story?
 C: Ways to Save Water at Home. (4)

However, some constraints must eventually be placed on the search and match procedure.

Suppletion. Suppletion became a problem for the model when GPs were being abstracted from WSPs. The program can handle suppletive forms such as in item 5, up to a point.

- go* and *went* → STEM = 0 AFFIXES = go went
run and *ran* → STEM = rn AFFIXES = -u- -a- (5)

Since the words within each pair are semantically similar, they will be placed in the same WSP. However, the model relies on morphology for abstraction and will give the results shown in item 5. The "0" indicates no stem, which is correct since the program does not know that it can arbitrarily consider the present-tense morpheme as the stem.

Suppletive forms cause problems only when they reach the level of GP abstraction. One method for avoiding the

problems illustrated in item 5 is to limit the application of the abstraction process to WSPs that have a sufficient number of lexical entries. This is what Pinker (1984, p. 198) does in rule I9(c). However, this raises the question of what is a sufficient number. The program requires at least two words in a dimension of a WSP before it will attempt to form a GP from that WSP. Another method for avoiding problems, which is also used by Pinker, is to assume that word stems are known before GP abstraction is attempted. However, stems must be abstracted at some point in language development, and assuming their existence avoids learnability issues. For example, are stems actually learned through abstraction, or is one cell in a paradigm innately recognized as the stem?

Insufficient data. Suppletion touched on another issue in paradigm usage, namely, how is GP abstraction handled when there are only a few words in the WSP undergoing abstraction? The correctness of stems and affixes of verbs depends on the completeness of the WSP. The program reanalyzes a WSP every time one is accessed. Reanalysis can be limited by tracking the number of times a stem has changed relative to the number of times the WSP in which it appears is accessed (constraint C5) and by avoiding WSPs with too few entries (as in the theory and in constraints C6 and C7).

Lack of information causes malformed GPs, which are evident in the paradigms for the Quiché words in item 6:

- xatoq'ik* → you cried
koq'ik → he cries (6)

The paradigms and extracted affixes are in Table 5. Only the person and aspect dimensions are illustrated. *Oq'ik* is the correct stem, but the affixes are incorrect. If a child has only the limited WSP shown in Table 5, and uses the GP to produce a novel word (e.g., second person, ongoing action), a conflict would arise because either prefix, *xat-* or *k-*, is possible (*kat-* is the correct prefix). Fortunately, constraints C6 and C7 prevent the GP abstraction process from starting. But, adding one more cell to the WSP causes abstraction to occur, and the affixes still are wrong.

The general paradigm in Table 5 leads to some testable hypotheses. Children could be expected to overgeneralize affixes randomly in unsubstantiated cells in the WSP, as in *xat-* and *k-*. Abstraction also predicts that overgeneralization errors should persist for particular words and for stages of WSP completeness. Words with large complex paradigms should show a wide variation in production for a longer period of time relative to words with smaller and simpler paradigms. The stage of WSP completeness could also be inferred from the types of affixes that are added to words produced by children.

Meta-paradigms. Many of the problems discussed above can be avoided by allowing general paradigms to communicate with each other. This issue was not addressed in the theory or in the model. Interparadigm communication would let paradigms compare information,

Table 5
Partial Paradigms for the Quiché Word OQ'IK, "To Cry"

WSP =		ASPECT	
		Ongoing	Completed
	2nd		xatoq'ik
PERSON			
	3rd	koq'ik	
Abstraction			
	ASPECT	Ongoing	Completed
	Common	koq'ik	xatoq'ik
	Affixes	k-	xat-
	Stem	oq'ik	
	PERSON	2nd	3rd
	Common	xatoq'ik	koq'ik
	Affixes	xat-	k-
	Stem	oq'ik	
GP =		ASPECT	
		Ongoing	Completed
	2nd	xat-/k-	xat-/k-
PERSON			
	3rd	xat-/k-	xat-/k-

Note—WSP = word-specific paradigm. GP = general paradigm.

such as affixes, and would presumably circumvent problems caused by lack of data within one paradigm.

Lexical access. Even if children know the meanings of words they attempt to analyze, as Pinker suggests, the question of how a particular lexical entry is accessed is unanswered. Entries could be accessed directly by words, indirectly by context, or both. Multiple entries would be accessed, and hence activated by the direct method. The indirect method would result in one entry, or very few entries, becoming active. Experimental data are mixed as to the role of context in lexical access. Most research (Duffy, 1986; Schubert & Eimas, 1977; Schwanenflugel & Shoben, 1985; Seidenberg, Waters, Sanders, & Langer, 1984; Sharkey & Mitchell, 1985) supports the view that context defines a set of words that can be accessed but does not actively "select" a word. Since the inflection learning theory presented here relies primarily on words to access lexical entries directly, then the model should have difficulty learning words, such as homonyms and ambiguous words, that activate many lexical entries. The theory would predict no difference in the rate of learning homonyms or other ambiguous words, whereas experimental evidence may be expected to show that typical acquisition errors, such as overgeneralization (e.g., item 7), should persist longer during language acquisition than for words that do not have homonyms (e.g., item 8).

* She readed the book. (7)

* She goed to the store. (8)

CONCLUSION

The program presented in this paper is an initial attempt to model the acquisition and use of inflectional paradigms. The model successfully builds feature equations and limited word-specific paradigms. General paradigms were discussed, but were not implemented in the model. Additional consideration must be given to the form of paradigm representation and to interparadigm relationships before the model can be extended to include general paradigms. Specifically, three questions must be addressed. First, how are word-specific paradigms found for novel words? This becomes a question of what makes words similar. Second, what constrains rule application, as in the case of regular and suppletive morphology? And finally, how do paradigms interact? The answers to these questions are not settled and depend on one's theoretical orientation. However, they must be addressed in order for a complete, testable theory to be developed.

REFERENCES

- CAREY, S. (1978). The child as word learner. In M. Halle, J. Bresnan, & G. A. Miller (Eds.), *Linguistic theory and psychological reality* (pp. 264-293). Cambridge, MA: MIT Press.
- DEVILLIERS, J. G. (1985). Learning how to use verbs: Lexical coding and the influence of the input. *Journal of Child Language*, 12, 587-595.
- DUFFY, S. A. (1986). Role of expectations in sentence integration. *Journal of Experimental Psychology: Learning, Memory, & Cognition*, 12, 208-219.
- FOWLER, C. A., NAPPS, S. E., & FELDMAN, L. (1985). Relations among regular and irregular morphologically related words in the lexicon as revealed by repetition priming. *Memory & Cognition*, 13, 241-255.
- KAPLAN, R. M., & BRESNAN, J. (1982). Lexical-functional grammar: A formal system for grammatical representation. In J. Bresnan (Ed.), *The mental representation of grammatical relations* (pp. 173-281). Cambridge, MA: MIT Press.
- MACKEY, D. G. (1976). On the retrieval and lexical structure of verbs. *Journal of Verbal Learning & Verbal Behavior*, 15, 169-182.
- MORTON, J. (1969). Interaction of information in word recognition. *Psychological Review*, 76, 165-178.
- PINKER, S. (1981). Comments on the paper by Wexler. In C. L. Baker & J. J. McCarthy (Eds.), *The logical problem of language acquisition*. Cambridge, MA: MIT Press.
- PINKER, S. (1984). *Language learnability and language development*. Cambridge, MA: Harvard University Press.
- SCHLESINGER, I. M. (1977). The role of cognitive development and linguistic input in language development. *Journal of Child Language*, 4, 153-169.
- SCHUBERT, R. E., & EIMAS, P. D. (1977). Effects of context on the classification of words and nonwords. *Journal of Experimental Psychology: Human Perception & Performance*, 3, 27-36.
- SCHWANENFLUGEL, P. J., & SHOBN, E. J. (1985). The influence of sentence constraint on the scope of facilitation for upcoming words. *Journal of Memory & Language*, 24, 232-252.
- SCHWARTZ, R. G., & TERRELL, B. Y. (1983). The role of input frequency in lexical acquisition. *Journal of Child Language*, 10, 57-64.
- SEIDENBERG, M. S., WATERS, G. S., SANDERS, M., & LANGER, P. (1984). Pre- and post-lexical loci of contextual effects on word recognition. *Memory & Cognition*, 12, 315-328.
- SELFIDGE, M. (1986). A computer model of child language learning. *Artificial Intelligence*, 29, 171-216.
- SHARKEY, N. E., & MITCHELL, D. C. (1985). Word recognition in a

- functional context: The use of scripts in reading. *Journal of Memory & Language*, **24**, 253-270.
- SNOW, C. E. (1977). Mothers' speech research: From input to interaction. In C. E. Snow & C. A. Ferguson (Eds.), *Talking to children: Language input and acquisition* (pp. 31-49). New York: Cambridge University Press.
- STEMBERGER, J. P., & MACWHINNEY, B. (1986). Frequency and the storage of regularly inflected forms. *Memory & Cognition*, **14**, 17-26.
- TAFT, M., & HAMBLY, G. (1985). The influence of orthography on phonological representations in the lexicon. *Journal of Memory & Language*, **24**, 320-335.
- WEXLER, K. (1982). A principle theory for language acquisition. In E. Wanner & L. R. Giletman (Eds.), *Language acquisition: The state of the art* (pp. 288-315). New York: Cambridge University Press.
- WEXLER, K., & CULICOVER, P. (1980). *Formal principles of language acquisition*. Cambridge, MA: MIT Press.

NOTE

1. Intellect-ul LISP is a trademark of PCD Systems. The source code for the model is available from the author. SAGE II is a trademark of SageMicro of Reno, Nevada. The SAGE II has a Motorola 68000 CPU, 256K of RAM, and runs the UCSD P-system operating system.