To appear in WCCFL-94

Type Underspecification and On-line Type Construction in the Lexicon

Jean-Pierre Koenig* and Daniel Jurafsky*†

University of California at Berkeley* and
International Computer Science Institute†

1 Introduction

Many recent lexical and syntactic theories have used TYPE HIERARCHIES to model linguistic generalizations, including valence alternations, morphological generalizations, subregularities, and positive exceptions (Bobrow & Webber 1980; Flickinger *et al.* 1985; Hudson 1984; Lakoff 1987; Pollard & Sag 1987; Jurafsky 1992; Briscoe *et al.* 1994; Pollard & Sag 1994).

Despite these successes, current type hierarchies are unable to model LEXICAL PRODUCTIVITY: the kind of productive patterns which occur in morphologically recursive languages, lexical borrowing and learning, and productive valence alternations (Hankamer 1989; Pinker 1989). This failure is due to the traditional conception of lexical types as generalizations over actual fully specified entries, functioning like redundancy rules. A lexical type hierarchy gives us an inventory of the word classes in a language, but does not tell us how to use productive processes like inflection to create new forms.

To model lexical productivity, we propose to underspecify the type hierarchy. For example, rather than store a type for each surface form of each word of the language, we store a single type for each root and each productive morphological template. Then these types are combined on-line to build types for surface forms in processing or producing an utterance, by an algorithm we call ON-LINE TYPE CREATION. Thus some of the burden typically borne by the lexical type hierarchy is shifted to the processing component, which combines these types on-line.

Although our system is embedded in the specific framework of CONSTRUCTION GRAMMAR (Fillmore *et al.* 1988; Kay 1990; Lakoff 1987; Goldberg 1991; Goldberg 1992; Koenig 1993), it is directly applicable to any typed theory such as HPSG. In fact, we show that on-line type construction can advantageously replace mechanisms like lexical rules which are used in HPSG to model lexical productivity.

2 Typed Lexicons

Theories of typed lexicons (Flickinger *et al.* 1985; Pollard & Sag 1987) rely on the intuition that lexical entries can be grouped into classes with common properties. These common properties can then be stated of the class as a whole, each member of the class inheriting the properties. This process of abstraction can proceed recursively; classes are grouped into more abstract classes, and so on. This set of words and classes and their relations is called a TYPE HIERARCHY. Consider the sample HPSG-style partial type hierarchy with English lexemes in Figure 1. (Our diagrams omit branches of the hierarchy for expository purposes).

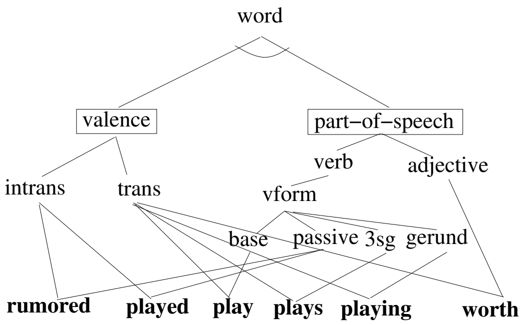

Figure 1: Part of a type hierarchy for HPSG

The generalization that **rumored** and **played** are both intransitive is captured by positing a class **intrans** with inheritance links to both **rumored** and **played** and which includes all properties common to intransitive verbs. This type **intrans** together with a similar general class **trans** PARTITION the type **valence** in our simple hierarchy. This means that any verb must belong to exactly one of the types **intrans** or **trans**. We call this a DISJUNCTIVE interpretation of type inheritance, since set-theoretically, objects which are of type **word** will belong either to the class **intrans** or the class **trans**.

The fact that **rumored** has both an intransitive valence and is a verb while **worth** is both transitive and an adjective is modeled by classifying words along multiple DIMENSIONS. These dimensions, (such as **valence** or **part-of-speech**) are represented in the diagram inside boxes. We mark a curved line on the link between **word** and the two dimensions **valence** and **part-of-speech**, indicating that every word must be specified for both valence and part-of-speech. The link from **valence** to **trans** indicates that the type **trans** is relevant for the **valence** dimension.

We call the relation between a type like **word** and its dimensions like **part-of-speech** a CONJUNCTIVE interpretation of type relations, since words must be specified for both part-of-speech and valence.

As mentioned above, although current type hierarchies have a number of advantages as a device for capturing generalizations, they are incapable of modeling productive processes. Notice in Figure 1 that the lexicon lists even the result of productive inflectional processes, such as *playing* and *played*.

3 On-Line Type Construction

We argue that to model all forms of lexical productivity, we need both a context-free approach to morphology (Selkirk 1982 and others) and a radical underspecification of the type hierarchy. This paper focuses on this second proposal, called ON-LINE TYPE CONSTRUCTION.[1] The gist of the proposal is that lexical entries are stored radically underspecified in the grammar. For example, Figure 2 shows part of the type hierarchy for Construction Grammar. The details of the hierarchy will be discussed later; the important intuition for now is that there is only one type for the lexeme *play*. In contrast to the compiled-out lexicon in Figure 1, we don't need to list a type for the inflected forms *playing*, *played*, etc. These forms of the word *play* are derived while interpreting or producing an utterance by combining *play* with inflectional templates. Since each of these forms is a new type, we call this TYPE CONSTRUCTION. We refer to our algorithm as ON-LINE to emphasize that types are not combined in advance.

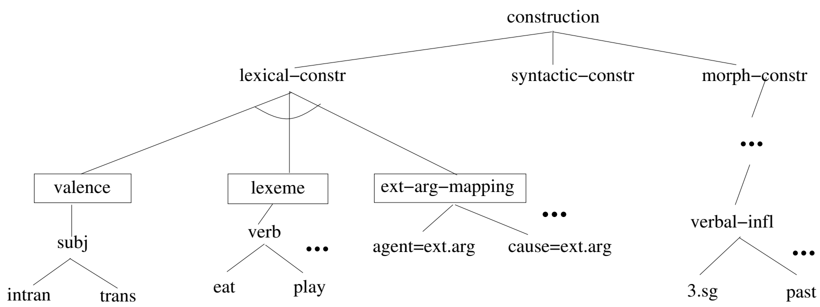

Figure 2: Part of the Construction Grammar Type Hierarchy

Our on-line type construction algorithm is a modification and extension of one of Carpenter's (1992) algorithms for off-line type construction. In this paper we cannot give the formal details of the algorithm.[2] Instead, we will work through an extended example, showing how a type inference system

[1] Neither type underspecification nor context-free morphology is sufficient by itself. Context-free morphology is ill-suited to model valence alternations which are not mediated by morphophonological change, such as extraposition. Type underspecification alone is not capable of handling the recursive aspects of morphology.

[2] Happily, constructing types on-line requires only a minor change in the mathematics of

presented with the sentences in (1) builds a fully specified entry for the various forms and valences of the word *play*. Figure 3 shows a simplification of the Construction Grammar type hierarchy with the newly-constructed types indicated in bold italic and connected by dotted lines.[3] As we mentioned

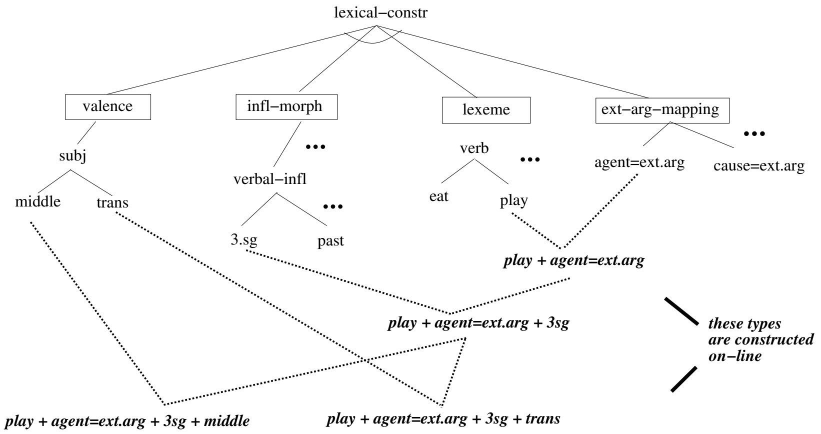

Figure 3: Example of On-Line Type Construction

before, the lexical dimension is underspecified because lexemes like *play* only specify stem phonology; predictable forms like *plays* or *played* are not present in the hierarchy. Consider (1a), which has the form *plays*.

(1) a. Claudio plays the sonata.
b. This sonata plays well on a piano-forte.
c. Carla played the sonata you love so much.

In order to determine the features of the fully-specified entry *plays*, we must construct a new type from the hierarchy. Since *plays* is a sub-type of **lexical-construction**, and **lexical-construction** has four dimensions we know that *plays* will need to be a subtype of exactly one type in each of the four dimensions **lexeme**, **external-argument mapping**, **inflection**, and **valence**.

Carpenter's algorithm for off-line type construction from a choice-network, allowing us to rely on his axiomatization. This is true because we can view the compiled-out type lattice as virtually present, although the entire lattice is not actually constructed. Thus any time his axioms refer to a join on the lattice, where a compiled-out type lattice merely looks up the join (via the greatest lower bound operation), the underspecified network must actually perform a complete unification at parse-time. If we interpret join in this manner, the only parts of Carpenter's axiomatization that need to be changed are those referring to the fail-type. Informally, the join of two types will produce the fail type if the feature structures associated with the two types fail to unify.

[3]For expository purposes, we (wrongly) assume in this figure that inflectional morphology is not modeled via context-free templates, but is a dimension of **lexical-constr.**

Suppose that the system has determined that the form *play* is of lexeme type **play**. Only some of the types in the other dimensions are compatible with lexeme **play**. For example, since play has an agent argument (and not an experiencer) only the **agent=ext.arg** type in the **external-argument mapping** dimension is compatible with it. The two types can be combined into a more specified entry in which the external-argument is specified as the player of the playing event, as shown in Figure 4[4]. We can read the new feature structure as specifying a STEM "play", a semantic form with two arguments, of which the PLAYER is the external-argument, and a valence (subcategorization) set with two elements.

$$
{}_{\textbf{play}}\begin{bmatrix} \text{STEM} & \text{"play"} \\ \text{SEM} & {}_{\textbf{play-reln}}\begin{bmatrix}\text{player \#1}\\ \text{music \#2}\end{bmatrix} \\ \text{VAL} & \{[\text{sem \#1}],[\text{sem\#2}]\} \end{bmatrix}
\sqcup
{}_{\textbf{agent=ext.arg}}\begin{bmatrix} \text{SEM} & {}_{\textbf{act-reln}}[\text{AGENT \#1}] \\ \text{EXT-ARG \#1} & \end{bmatrix}
=
{}_{\textbf{play + agent=ext.arg}}\begin{bmatrix} \text{STEM} & \text{"play"} \\ \text{SEM} & {}_{\textbf{play-reln}}\begin{bmatrix}\text{player \#1}\\ \text{music \#2}\end{bmatrix} \\ \text{EXT-ARG \#1} & \\ \text{VAL} & \{[\text{sem \#1}],[\text{sem\#2}]\} \end{bmatrix}
$$

Figure 4: The Constructed Type *play + agent=ext.arg*

Similarly, the only morphological type compatible with the form *plays* is the **3sg** type. We can therefore combine the type **3sg** with the type **play+agent=ext.arg** to derive the new type **play+agent=ext.arg+3sg**. Finally, we must choose between the middle and transitive valence templates (in this simplified graph). Combining each of them with the entry we've got so far creates two new types: **play+agent=ext.arg+3sg+middle** and **play+agent=ext.arg+3sg+trans**. These types are now fully-specified, but only the transitive type is compatible with the input (intuitively middles do not have objects). Sentences (1b) and (1c) would be analyzed similarly, although other choices would be made along the inflectional or valence dimensions.

4 Advantages of On-Line Type Construction

The last section illustrated how on-line type construction models lexical productivity, by combining underspecified abstract entries with valence and morphological templates. In this section, we show how type construction provides a motivated and direct solution to well-known morphological and valence problems. These include positive and negative exceptions, subregularities, the need for conjunctive versus disjunctive rule blocks, and suppletive stem selection. In each case we also compare our on-line typing approach to the only previous model of lexical productivity within a typing system, the

[4]Types are indicated in boldface to the lower left of the feature structure. Pound signs indicate identity of structure.

LEXICAL RULE approach assumed by HPSG. [5]

Our approach to lexical productivity — in addition to a context-free morphology — is to underspecify lexical entries and derive fully-specified entries on-line. The HPSG approach is to use a compiled-out type hierarchy to capture common properties of words, and lexical rules to handle productivity. Lexical rules in HPSG apply on-line to map fully-specified lexical entries into new fully-specified entries. On-line application means that the rules are used to build new lexical entries at the time of interpreting or producing an utterance. [6]

In HPSG, for example, the fact that the verb *eat* has both transitive and passive realizations as in (2)–(4) might be captured with a valence affecting lexical rule, while the morphological relation between *eats* and *ate* might be captured with an inflectional lexical rule.

(2) John ate a sandwich
(3) John eats a sandwich every day
(4) The sandwich was eaten

Figure 5 represents a simplified passive lexical rule, whose effect is basically to suppress the subject requirement and make an intransitive verb out of a transitive verb.

$$\begin{bmatrix} \text{CAT} & \text{V} \\ \text{FORM} & \text{active} \\ \text{VAL} & <\underset{\text{subj}}{\#1\ \text{NP}},\ \underset{\text{obj}}{\#2\ \text{NP}}> \end{bmatrix} \Longrightarrow \begin{bmatrix} \text{CAT} & \text{V} \\ \text{FORM} & \text{passive} \\ \text{VAL} & <\underset{\text{subj}}{\#2\ \text{NP}}> \end{bmatrix}$$

Figure 5: A simplified passive lexical rule

4.1 Exceptions and Subregularities

It is well-known that lexically-governed processes are subject to exceptions (Lakoff 1970). Consider the verb *rumored* in (5):

(5) a. It was rumored that the meeting would not be held.

[5] We dicuss here the theory of lexical rules sketched in Pollard & Sag (1987) and Pollard & Sag (1994). As we have recently become aware, some HPSG scholars have a view of the lexicon much closer to the one advocated in this paper (see for example Krieger & Nerbonne (1993) and Riehemann (1993)). We see this unexpected convergence of results as significant, although we cannot compare their approach to ours in this paper.

[6] Earlier theories of lexical rules assumed the REDUNDANCY interpretation of lexical rules, in which the lexicon contains all forms and the rule acts as a redundancy marker. Goldberg (1991) presents a number of linguistic arguments against this redundancy interpretation. Jurafsky (1992) argues that the redundancy interpretation is inconsistent with the Strong Competence Hypothesis of Bresnan & Kaplan (1982).

b. *John rumored that the meeting would not be held.

As (5) shows, there is no active form of the verb *rumored*. Verbs of this sort are traditionally refered to as POSITIVE EXCEPTIONS, since verbs like *rumored* can be thought of as obligatorily undergoing the passive rule or transformation.

We handle verbs like *rumored* in our system by declaring them as subtypes of type **passive**. Thus *rumored* will correctly inherit all the morphological and syntactic properties common to all passives. In addition, this pretyping will prevent the active template from applying, since the active and passive types are incompatible, thus explaining the lack of an active form. By contrast, ordinary transitive lexemes like *love* can combine with either active or passive templates, because they are not pre-typed to either. Figure 6 shows these examples in our typed system. The absence of any inheritance link between **love** and either the **passive** or **transitive** constructions means it can combine with either provided they contain information compatible with its own. [7]

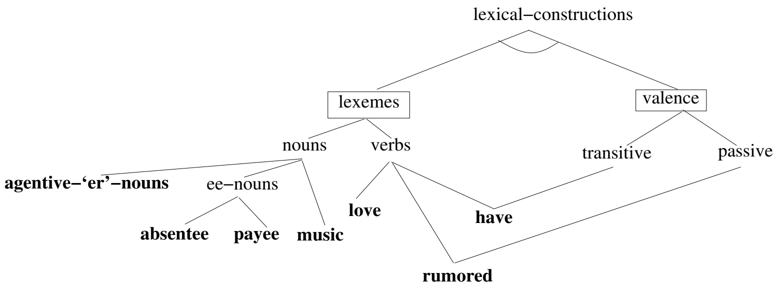

Figure 6: Positive and Negative Exceptions in a On-line Typing System

By contrast, a lexical rule system models positive exceptions by marking the offending lexical item with rule-specific *exception features*. A lexical rule will not apply to any lexical item which is marked with its exception feature. Since **rumored** is obligatorily passive we will need to specify in the entry for **rumored** the feature [PASSIVE-LEX-RULE –] to insure that the passive-lexical rule cannot apply. But in addition to marking the entry with this feature, a lexical rule system must still mark **rumored** as an instance of the passive type, because it acts syntactically like other passive forms. Thus the entry will be marked twice, once for being a passive template, and once

[7] Note also that so-called negative exceptions to the passive lexical-rule like *have* (although cf. Pinker (1989)) are also handled via *pre-typing* of the relevant entries.

(6) a. *The house was had by many people
b. This house was owned by many people

Abeillé (1990) gives other examples of negative exceptions from French.

for being an exception to the passive lexical rule. The on-line typing system accomplishes the same results with only type information.

More generally, on-line type construction allows us to model the difference between fully productive and semi-productive or subregular patterns via the structure of the type hierarchy. For example, Jurafsky (1992) proposes to capture the subregularity relating deverbal nouns in **-ee** (*addressee, employee*) by positing a super-type **-ee-nouns** representing semantically and morphophonologically common information. Since this type is not completely productive, it must be defined extensionally: each word it applies to must be listed in the stored hierarchy, as shown in Figure 6.[8] By contrast, the productive morphological construction **agentive-'er'-nouns** (*builder*,*mower*) is a leaf in the hierarchy; the lexemes it applies to are not stored in the grammar as a special class of stem. Thus 'builder' and 'mower' are constructed on-line, while 'absentee' is stored in the lexicon.

The positive and negative exceptions to regular processes discussed above are thus a special case of subregularities.

4.2 Stem selection algorithms for suppletive stems The combination of the AND/OR hierarchy and on-line type construction also allows for a direct account of suppletive stems. Consider the French lexeme *aller* in (7a)–(7d), whose various forms are based on four different suppletive stems, *all-*, *ir-*, *v-*, and *aill-*, respectively. Each stem occurs in an idiosyncratic morphological environment. The stem *ir-*, for example, is used in the future and conditional, the stem *v-* for the singular present indicative and imperative and third person plural present indicative, etc. (7) also exemplifies the many different valence and semantic entries for *aller*; it can mean 'go', 'leave', 'fit', etc. Significantly, these semantically different entries for *aller* all share the four stems; each entry can appear with any of the stems, and the endings which attach to these stems are regular. In Aronoff's (1976) terms, there are many French words which correspond to a single morpheme *aller*.

(7) a. Marc *allait* à Paris.
Marc go.impf to Paris
Marc went to Paris'

b. Marc s'en *ira*.
Marc refl of.it go.fut
Marc will leave.

c. Ce costume te *va* bien.
This suit you go.pr well
This suit fits you well (lit. goes well to you)

[8] The figure is somewhat simplified and does not show the internal structure of the **agentive-er-nouns** and **ee-nouns** constructions.

d. Il faut que tu y *ailles*
It must.pr that you there go.subj.pr
You must go there.

The system we propose can elegantly represent stem selection by associating with *aller* a bi-dimensional lexeme of entries and stems, with each stem leaving underspecified the endings to be filled in by the morphological-templates, as shown in Figure 7. Any actual form of *aller* combines one entry subtype and one stem subtype of *aller* together with the ordinary valence, inflectional and other templates. Note that the **/v/-** or **/ir/-** stems in the figure prespecify some morphosyntactic features. These features will constrain the contexts of occurences of words derived from these stems.[9]

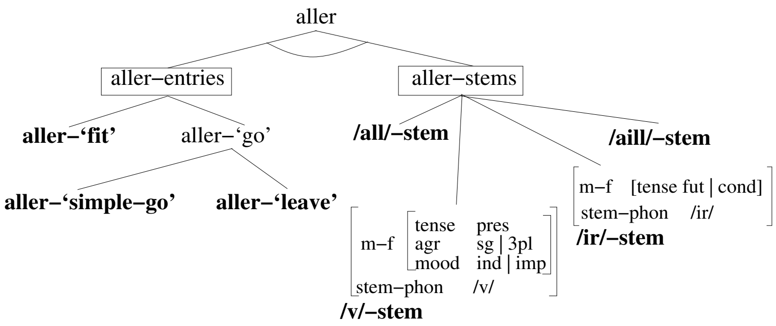

Figure 7: The type hierarchy for the French lexeme *aller*

By contrast, to insure the proper selection of stem in context, the lexical rules responsible for tense and person-endings in French must be sensitive to which of the four stems is appropriate in the sentence's morphosyntactic environment. A lexical rule (or hybrid) approach might propose an abstract *aller* entry shared by the more specific entries ('go', 'leave', 'fit'), as in our approach. However, in order to capture these generalizations about stems without type underspecification, this abstract entry would need to include ad-hoc features representing the phonology of all four possible 'stems' of *aller* and the morphological environments which condition them, duplicating independently-needed features. Moreover, when applying to *aller* inflectional lexical rules would need to implement a choice system, to insure that the correct stem is supplied for each morphological feature bundle. The AND/OR hierarchy assumed by such a hybrid system already implements such a choice system; thus implementing it via lexical rules is particularly inelegant.

[9]As suggested to us by Bill Ladusaw (p.c.) the same strategy can be profitably used for more mundane cases, like English *says* and *does*, which (although orthographically regular) have an irregular allomorph of the verbs with a lowered vowel when followed by the third-sigular affix. Each verb would have associated with it a set of forms with specific stem-phonology.

4.3 AND/OR Trees and Conjunctive vs. Disjunctive Rule Application

Another advantage of combining on-line type construction and AND/OR trees is that it can account for what has been called CONJUNCTIVE AND DISJUNCTIVE RULE APPLICATION (Anderson 1992). For example, Inkelas & Orgun (1993) have shown cases in Turkish in which more than one affix applies on a given cycle. We can model this obligatory concatenation of morphemes on the same cycle by using the dimension (AND) part of the type hierarchy to encode the fact that any form must express each of a certain set of affixes.

This section gives a simple example using the Latin verbal inflectional system. For the purposes of this example, we'll assume that the three affixes we consider belong to the same cycle. The structure of Latin verbs is represented informally in (8):

(8) *stem-(aspect)-(tense/mood)-(agreement+pass/act)*
ama-v-isse-m 'I wish I had loved'
ama-ba-r 'you were loved'

Figure 8 shows our proposal for a partial type hierarchy for Latin. It specifies that to construct a Latin verb one must choose a lexeme (here **amare**), and combine it with one template from each of three classes of choices: aspect-affix templates, tense/mood-affix templates, and person/voice ending templates.

In the case of **amavissem** in (8), the only choices compatible with the input form are the nodes at the end of branches a, b, c, and d, i.e. the **amare** lexeme, and the **perfect**, **past-subjunctive**, and **1st-singular-active** affix types. Thus the ordinary logic of AND/OR graphs, required independently for dimensional type systems, accounts for the Latin data with no additional morphology-specific mechanism.

By comparison, the lexical rule approach would need an ad-hoc device to account for the same data. Since these affixes are productive, each would be introduced by a productive lexical rule. Whereas ordinary lexical rules map words onto words, applying only one of these rules does not derive a well-formed word in this case. The three rules must apply together, and in sequence. We refer to this circumstance as a LEXICAL RULE CHAIN. Indeed, the situation is even somewhat more complex, since for affixes 2 and 3 (mood/tense and person endings), there are several possible values for the affixes. This means that capturing Latin verbal morphology with lexical rules requires chains of disjunctive sets of lexical rules. The on-line type creation algorithm avoids this unnecessary machinery.

5 An Ordering Paradox and Other Problems with Lexical Rules

Combining types via our on-line type construction algorithm is order-independent. It is therefore immune to ordering paradoxes, by contrast to rule-based ap-

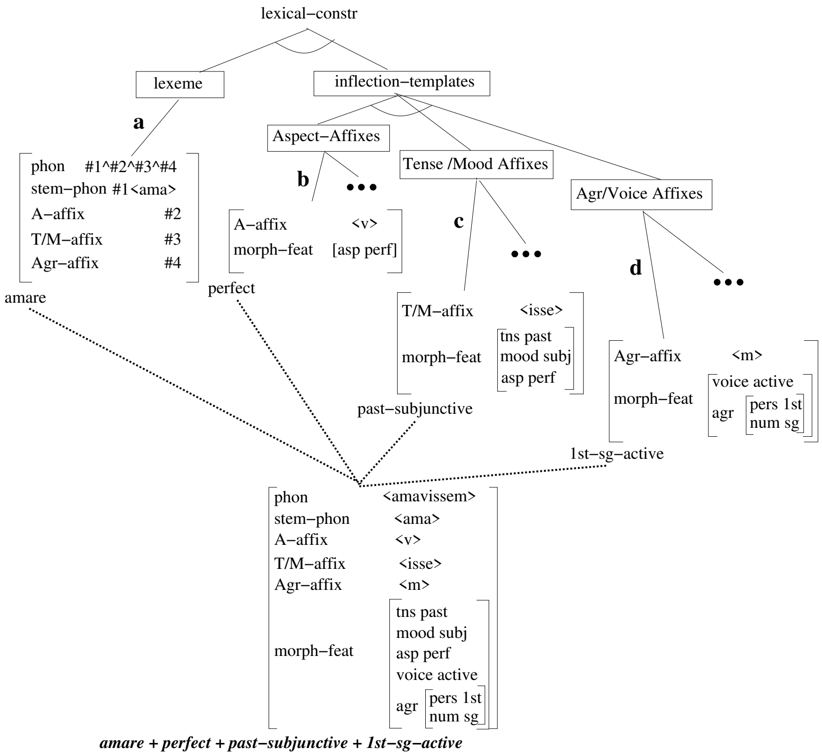

Figure 8: Latin Morphology: A Simplified Example

proaches, like the lexical rule approach to valence alternations taken in classical HPSG.

Data from the French passive-reflexive and impersonal-inversion valence constructions provide such a paradox. Consider (9), which illustrates the basic transitive use of the verb *vendre* 'to sell'.

(9) Jacques vend des livres
Jacques sell.pr indef books
Jacques sells books

(10) illustrate the middle or passive-reflexive use of the same verb (*se vendre*).[10] The lexical rule approach would derive *se vendre* (as in (10)) from *vendre* (as in (9)) via a passive-reflexive lexical rule which moves the

[10]In all examples involving first or second person pronouns, the pronoun refers metonymically to the author of the book being sold.

object into subject position and adds a reflexive marker which agrees with the subject. (Notice that the reflexive clitic (*t'* or *nous*) agrees with the subject.)

(10) a. A la dernière foire du livre tu t'es bien vendu.
At the last book fair you sold well

b. A la dernière foire du livre nous nous sommes bien vendus.
At the last book fair we sold well

Figure 9 sketches the lexical rule in an HPSG-style notation.

$$\left[\begin{array}{ll} \text{HEAD} & [\text{CLI } \{...\}] \\ \text{SUBJ} & \langle\#1 \text{ NP}\rangle \\ \text{OBJ} & \langle\#2 \text{ NP}\rangle \\ \text{ARG–S} & \langle\#1, \#2, ...\rangle \end{array}\right]_{\textbf{trans}} \Longrightarrow \left[\begin{array}{ll} \text{HEAD} & [\text{CLI } \{[[\text{agr } \#3\text{lex}]\ [\text{refl } +]]\}] \\ \text{SUBJ} & \langle\#2 \text{ NP[agr } \#3]\rangle \\ \text{OBJ} & \langle\rangle \\ \text{ARG–S} & \langle\#2,...\rangle \end{array}\right]_{\textbf{intrans}}$$

Figure 9: The Passive Reflexive Lexical Rule

(11) illustrates the impersonal-inversion valence of *se vendre*. The lexical rule approach would derive this valence of *se vendre* via an impersonal-inversion lexical rule which moves the subject into object position and adds an expletive subject *il*.

(11) Il se vend deux cent livres par an à Paris
It 3rd.refl sell two hundred books by year to Paris
'Two hundred books are sold each year in Paris'

$$\left[\begin{array}{ll} \text{HEAD} & [\text{CLI } \{...\}] \\ \text{SUBJ} & \langle\#1 \text{ NP}\rangle \\ \text{OBJ} & \langle\rangle \\ \text{ARG–S} & \langle\#1, ...\rangle \end{array}\right]_{\textbf{intrans}} \Longrightarrow \left[\begin{array}{ll} \text{HEAD} & [\text{CLI } \{...\}] \\ \text{SUBJ} & \langle\#2 \text{ NP[expl]}\rangle \\ \text{OBJ} & \langle\#1 \text{ NP}\rangle \\ \text{ARG–S} & \langle\#2,\#1, ...\rangle \end{array}\right]_{\textbf{trans}}$$

Figure 10: The Impersonal Inversion Lexical Rule

Figure 10 sketches the impersonal inversion rule. Note that this rule must apply after the passive-reflexive rule. The contrast between (12) and (13) shows that the impersonal-inversion construction only applies to intransitive verbs. It cannot therefore apply directly to the transitive form of *vendre*, as shown in (14).

(12) Il est arrivé deux personnes
it be.pr arrive.ppt two people
Two people arrived.

(13) *Il mange des champignons Jacques
It eats.pr indef mushrooms Jacques
Jacques eats mushrooms.

(14) *Il vend des livres Jacques
It sell.pr indef books Jacques
Jacques sells books.

(14) is bad because we must first derive an intransitive valence by applying the passive-reflexive lexical rule. In other words, the passive-reflexive feeds the impersonal-inversion rule. We sketch this as follows:

(15) Jean a vendu un livre $\stackrel{pass-refl}{\Rightarrow}$ Un livre s'est vendu $\stackrel{imp-inv}{\Rightarrow}$ Il s'est vendu un livre.

(16) Jean a vendu un livre $\stackrel{imp-inv}{\Rightarrow}$ *Il a vendu un livre Jean

Unfortunately, there are other agreement facts which show that the impersonal inversion construction must apply before the passive-reflexive rule, causing an ordering paradox. We mentioned above that the reflexive clitic introduced by the passive-reflexive lexical rule must agree with the subject. Note that in sentences where both lexical rules have applied ((17a) below), the reflexive critic agrees with the expletive subject introduced by the impersonal inversion construction. Thus (17aa) is ungrammatical because the reflexive clitic *t'* agrees with the extraposed pronoun *toi*.

(17) a. *A la dernière foire du livre, il ne t'est bien vendu que toi.
At the last fair of.the book it not refl.2sg be.pr well sell.ppt that you
At the last book fair only you sold well.

b. (?)A la dernière foire du livre, il ne s'est bien vendu que toi.
At the last fair of.the book it not refl.3sg be.pr well sell.ppt that you
At the last book fair only you sold well.

Intuitively, the clitic must be introduced after the surface subject is in place. Since in (17a) the surface subject is introduced by the impersonal inversion rule, the passive reflexive rule which introduces the clitic must follow the inversion rule.

But we showed above that the passive-reflexive rule must apply before the impersonal-inversion construction, to create the appropriate intransitive environment for the application of the impersonal construction. Hence the ordering paradox.[11]

[11] This paradox does not depend on the details of our description of the passive reflexive rule. For example, one might abstract the reflexive clitic agreement facts away from the rule itself, making them constraints on all reflexive verbs which would be inherited by the output of the rule. Since the reflexive rule and the clitic agreement information are now separated, we might attempt to order the inheritance of the reflexive type differently from the passive-reflexive rule. However, this proposal solves the ordering paradox by introducing an inelegant and otherwise unmotivated non-monotonicity: requiring inherited information to apply after other lexical rules.

Our underspecified approach to lexical productivity does not run into any ordering paradox, since patterns do not apply in any order. The passive-reflexive construction (using an HPSG-like notation for comparison with the lexical rule approach) is sketched in Figure 11. The construction says that the valence requirement corresponding to the external-argument is lexically satisfied, and that the subject agrees with a clitic-affix. The impersonal-inversion construction, also in Figure 11 says that one valence requirement is assigned the object function and the subject is an expletive NP. Both patterns, together with other linking patterns, account for the data in (9)–(17a).

SEM[CONT[EXT−ARG #4]]
HEAD [CLI {[[agr #1] [refl +]]}]
SUBJ <#2 NP[agr #1]>
OBJ <...>
LEX−SAT {#3[SEM|CONT #4]}
ARG−S {#2, #3, ...}

passive−reflexive

SUBJ <#1 NP[expl]>
OBJ <#2 NP>
ARG−S {#1, #2, ...}

impersonal−inversion

Figure 11: The Passive Reflexive and Impersonal Inversion Valence Constructions

The crucial difference between the two treatments lies in the fact that the lexical rule approach operates on fully specified entries. It must therefore decide before the final valence-affecting rule applies which argument is the subject of the output of the passive-reflexive rule and will have to agree with the reflexive clitic.

Finally, we turn to computational considerations. Consider trying to parse the Latin form *amavissem* in (8). Since this word contains productive tense, aspect, and mood suffixes, it will not be listed in the lexicon. So in order to find the lexical entry for this word, we must recover its base form, say the infinitive *amare* 'to love'. How can we do this? If inflection is represented by lexical rules, we must run the lexical rules *backwards*, one at a time, stripping off phonological material at the right-edge of the word at each step, til we come to the base verb, and can check the lexicon.

But now we need to build the proper feature structure for the input form *amavissem*, and so we will have to run each lexical rules forwards again starting from the base infinitive verb to build the surface form!

The lexical rule approach requires this inefficient computation because lexeme-specific information is stored in a specific fully specified base form.[12]

[12] We wish to stress again that this argument rests on the assumption that lexical rules map fully specified and inflected words into fully specified and inflected words, without employing the kind of underspecified entries or context-free morphology we propose.

Our underspecified type system eschews this extra computational cost entirely, since lexical entries are stored radically underspecified and there is only one abstract entry for all inflectional and valence alternates of a word.

This argument shows that the lexical rule approach has a greater computational cost than on-line type construction.

6 Conclusion

We have introduced a new theory of on-line type creation for inheritance-based grammars like Construction Grammar or HPSG which, together with a context-free theory of morphology, accounts for productivity in the lexicon. Because it is based on type theory, our model retains the ability of types to model positive exceptions and subregularities. Because the lattice of types is constructed on-line, we can directly model true productivity in the lexicon.

We have also shown that this direct way of modeling lexical productivity has distinct advantages over other approaches to lexical productivity in lexically oriented theories like HPSG. This includes the ability to model exceptions to productive processes, complex stem suppletion like French *aller* and complex inflectional morphology all without proposing additional mechanisms like exception features or lexical rule chains. Our system also avoids ordering paradoxes which plague lexical rules, like their transformational ancestors.

To conclude, we would like to emphasize an implicit theme of our on-line model, when compared with compiled-out type hierarchies (Carpenter 1992). Where a compiled-out theory lists all possible linguistic types in the grammar, our model shifts some of the burden of representing linguistic knowledge out of the grammar and into the computational processes of generation and interpretation. To a certain extent, recent theories of on-line lexical rules (Pollard & Sag 1994) have made this same shift from earlier redundancy models by allowing new forms to be added to the lexicon. But our model shifts even more out of the grammar, as the underspecified lexicon includes no fully-specified lexemes, only blueprints for their eventual construction. In current work we are examining an extension of this use of the processing component, to model other lexical phenomena like blocking.

Acknowledgments

We thank Bob Carpenter, Tony Davis, Chuck Fillmore, Sharon Inkelas, Paul Kay, Bill Ladusaw, Orhan Orgun, Carl Pollard, Ivan Sag, participants in the Berkeley Typed Feature Structure seminar, and the WCCFL talk audience for valuable comments on the content of this paper. All remaining errors are ours.

References

ABEILLÉ, ANNE. 1990. Lexical and syntactic rules in a tree adjoining grammar. In *Proceedings of the 28th ACL*, 292–298, Pittsburgh, PA.

ANDERSON, STEPHEN. 1992. *Amorphous Morphology*. Cambridge: Cambridge University Press.

ARONOFF, MARK. 1976. *Word Formation in Generative Grammar*. Cambridge: MIT Press.

BOBROW, ROBERT J., & BONNIE WEBBER. 1980. Knowledge representation for syntactic/semantic processing. In *Proceedings of the First National Conference on Artificial Intelligence*, 316–323. Morgan Kaufmann.

BRESNAN, JOAN, & RONALD M. KAPLAN. 1982. Introduction: Grammars as mental representations of language. In *The Mental Representation of Grammatical Relations*, ed. by Joan Bresnan. Cambridge, MA: MIT Press.

BRISCOE, TED, ANN COPESTAKE, & ALEX LASCARIDES, 1994. Blocking. manuscript.

CARPENTER, BOB. 1992. *The logic of typed feature structures*. Cambridge: Cambridge University Press.

FILLMORE, CHARLES J., PAUL KAY, & M. C. O'CONNOR. 1988. Regularity and idiomaticity in grammatical constructions: The case of let alone. *Language* 64.501–538.

FLICKINGER, DANIEL, CARL POLLARD, & THOMAS WASOW. 1985. Structure-sharing in lexical representation. In *Proceedings of the 23rd ACL*, 262–267, Chicago.

GOLDBERG, ADELE E. 1991. On the problems with lexical rule accounts of argument structure. In *Proceedings of the 13th Annual Cognitive Science Conference*, 729–733, Chicago.

—— 1992. The inherent semantics of argument stucture: the case of the English ditransitive construction. *Cognitive Linguistics* 3.37–74.

HANKAMER, JORGE. 1989. Morphological parsing and the lexicon. In *Lexical Representation and Process*, ed. by William Marslen-Wilson, 392–408. Cambridge, MA: MIT Press.

HUDSON, RICHARD. 1984. *Word Grammar*. Oxford: Basil Blackwell.

INKELAS, SHARON, & ORHAN ORGUN. 1993. Level economy, derived environment effects, and the treatment of exceptions. In *Proceedings of the Workshop on Recent Developments in Lexical Phonology*, ed. by Richard Wiese. Düsseldorf: H. Heine Universität.

JURAFSKY, DANIEL, 1992. *An On-line Computational Model of Human Sentence Interpretation: A Theory of the Representation and Use of Linguistic Knowledge*. Berkeley, CA: University of California dissertation. Available as UC Berkeley Computer Science Division TR #92/676.

KAY, PAUL. 1990. Even. *Linguistics and Philosophy* 13.59–216.

KOENIG, JEAN-PIERRE. 1993. Linking constructions vs. linking rules: Evidence from French. In *Proceedings of the 19th BLS*, 217–231, Berkeley, CA.

KRIEGER, HANS-ULRICH, & JOHN NERBONNE. 1993. Feature-based inheritance networks for computational lexicons. In *Inheritance, Defaults, and the Lexicon*, ed. by Ted Briscoe, Valeria de Paiva, & Ann Copestake, 90–136. Cambridge: Cambridge University Press.

LAKOFF, GEORGE. 1970. *Irregularity in Syntax*. New York: Holt, Rinehart, and Winston.

——. 1987. *Women, Fire, and Dangerous Things*. Chicago: University of Chicago Press.

PINKER, STEVEN. 1989. *Learnability and Cognition: The acquisition of argument structure*. Cambridge: MIT Press.

POLLARD, CARL, & IVAN A. SAG. 1987. *Information-Based Syntax and Semantics: Volume 1: Fundamentals*. Chicago: University of Chicago Press.

——, & ——. 1994. *Head-Driven Phrase Structure Grammar*. Chicago: University of Chicago Press.

RIEHEMANN, SUSANNE, 1993. *Word Formation in Lexical Type Hierarchies*. University of Tuebingen thesis.

SELKIRK, ELISABETH. 1982. *The Syntax of Words*. Cambridge: MIT Press.