Chapter 7

An introduction to unification grammars

In part II we have developed a formal theory of parsing schemata for context-free grammars. In part III we will apply this theory in several different directions.
In Chapters 7–9, we discuss parsing schemata for unification grammars.
In Chapters 10 and 11 we use parsing schemata to define Left-Corner and Head-Corner chart parsers. We will prove these to be correct as well.
In Chapters 12 and 13, subsequently, we derive a parsing schema for Tomita's algorithm as an example of an algorithm that is not item-based. As a result, we can cross-fertilize the Tomita parser with a *parallel* bottom-up Earley parser, yielding a parallel bottom-up Tomita parser.
In Chapter 14, finally, we discuss hard-wired implementations of parsing schemata, in the form of boolean circuits.

We will extend parsing schemata with feature structures, so that schemata for parsing unification grammars can be defined. In addition to items that describe how a parser deals with the context-free backbone of a grammar, we will extend the schema with a notation in which one can specify how features are transferred from one item to the other. Thus a formalism is obtained in which *feature percolation* in unification grammar parsing can be controlled explicitly. Chapter 7 is a brief, informal introduction. In Chapter 8 we give a lengthy, formal treatment of the formalism; some more practical aspects of unification grammar parsing are discussed in chapter 9.

Unification grammars — also called unification-based grammars, constraint-based grammars, or feature structure grammars — are of central importance to current computational linguistics. As these formalisms are not widely known among computer scientists, it seems appropriate to give an introduction that should provide some intuition about what we are going to formalize.

In 7.1 a preview is given of what parsing schemata with feature structures look like. While keeping the notion of feature structures deliberately abstract and vague, the general idea of such a parsing schema stands out rather clear. In 7.2, subsequently, feature structures and unification grammars are informally introduced by means of an example. We use the PATR formalism of Shieber [1986], with a tiny change in the notation. Anyone who is familiar with PATR can skip 7.2.

7.1 Unification-based parsing schemata: a preview

A thorough, formal treatment of unification grammars and parsing schemata for these grammars will be given in Chapter 8. As we will see, it requires quite some space and effort to do things properly. Parsing algorithms for unification grammars constitute a complex problem domain. A wealth of concepts is to be introduced, properly defined and — not the least problem — provided with clear and precise notations. We will jump ahead now and look at a glimpse of what we are heading for. An intuitive understanding of what we are trying to formalize may help the reader to get through the formal parts.

We address the following question: "*How can parsing schemata be enhanced with any kind of information that is added to the context-free backbone of a grammar?*" One may think of attribute grammars, unification grammars, affix grammars or any other formalism in which such information can be specified. We will be unspecific, for good reason. By refusing (for the moment) to use a particular formalism we cannot get sidetracked by all its sophisticated details.

In this section we recapitulate a simple context-free parsing schema, give an example of the use of other grammatical information, introduce (fragments of) a notation for it, and add this to the parsing schema.

As an example of a context-free parsing schema we recall the **Earley** schema of Example 4.32. For an arbitrary grammar $G \in \mathcal{CFG}$ we define a parsing system $\mathbb{P}_{Earley} = \langle \mathcal{I}, H, D \rangle$, where $\mathcal{I}$ denotes the domain of Earley items; H (the hypotheses) encodes the string to be parsed; D comprises the deduction steps that can be used to recognize items. Most deduction steps are of the form $\eta, \zeta \vdash \xi$. When the *antecedents* η and ζ have been recognized, then the *consequent* ξ can also be recognized. Some deduction steps have only a single antecedent. Moreover, in order to start parsing, an initial deduction step with no antecedents is included. $\mathbb{P}_{Earley}$ is defined by

$$\mathcal{I}_{Earley} = \{[A \rightarrow \alpha \bullet \beta, i, j] \mid A \rightarrow \alpha\beta \in P, 0 \leq i \leq j\};$$

$$
\begin{array}{lcl}
H & = & \{[a_1, 0, 1], \ldots, [a_n, n-1, n]\};\\
D^{Init} & = & \{\ \vdash [S{\rightarrow}{\bullet}\gamma, 0, 0]\},\\
D^{Scan} & = & \{[A{\rightarrow}\alpha{\bullet}a\beta, i, j], [a, j, j+1]\ \vdash\ [A{\rightarrow}\alpha a{\bullet}\beta, i, j+1]\},\\
D^{Compl} & = & \{[A{\rightarrow}\alpha{\bullet}B\beta, i, j], [B{\rightarrow}\gamma{\bullet}, j, k]\ \vdash\ [A{\rightarrow}\alpha B{\bullet}\beta, i, k]\},\\
D^{Pred} & = & \{[A{\rightarrow}\alpha{\bullet}B\beta, i, j]\ \vdash\ [B{\rightarrow}{\bullet}\gamma, j, j]\},\\
D_{Earley} & = & D^{Init} \cup D^{Scan} \cup D^{Compl} \cup D^{Pred};
\end{array}
$$

where H varies according to the string $a_1 \ldots a_n$ that should be parsed. The second part of the usual set notation $\{\ldots \mid \ldots\}$ has been deleted in most cases; by definition, deduction steps may only use items from $\mathcal{I}$ and H.

We assume that the context-free backbone of a grammar is enhanced with additional syntactic, semantic or other linguistic information. Constituents, productions, and items can have certain *features*[1] that express information not present in the context-free part of the grammar. This information can be of different kinds. A typical use of features is the transfer of information through a parse tree. As an example, consider

> *In the production $S{\rightarrow}NP\ \ VP$, the semantics of S can be derived from the semantics of NP and VP by …*

If each word in the lexicon has some semantics associated with it, and for each production it is known how the semantics of the left-hand side is to be derived from the right-hand side, the semantics of the sentence can be obtained compositionally from its constituents.

Another typical, more syntactic way in which features are used is to constrain the set of sentences that is acceptable to the parser. A canonical example is

> *In the production $S{\rightarrow}NP\ \ VP$, there must be (some form of) agreement between NP and VP.*

The precise nature of the agreement is irrelevant here. Either constituent will have some features that could play a role in agreement, e.g.

> *the noun phrase "*the boy*" is masculine, third person singular,*

but the fact that agreement is required between NP and VP is a feature of the production, not a feature of each of the constituents individually.

Let us now enhance the Earley parser with such features. If we parse a sentence "The boy …", at some point we will recognize an item $[S{\rightarrow}NP{\bullet}\,VP, 0, 2]$. We could attach the previously stated information to the item, as follows

[1] At this level of abstraction, the word "feature" can be replaced by "attribute", "affix", etc. All of these stand for roughly the same concept, but refer to different kinds of formalisms.

The NP in $[S{\to}NP{\bullet}VP, 0, 2]$ *is masculine, third person singular.*
Hence the VP that is to follow must be masculine, third person singular.

Next we apply the *predict* step

$$[S{\to}NP{\bullet}VP, 0, 2] \ \vdash \ [VP{\to}{\bullet}{*v}\ NP, 2, 2],$$

in combination with a feature of the production $VP{\to}{*v}\ NP$:

In the production $VP{\to}{*v}\ NP$*, the agreement of VP is fully determined by the agreement of* $*v$*.*

Combining all this information, we obtain the following item annotated with features:

$[VP{\to}{\bullet}{*v}\ NP, 2, 2]$
VP must be masculine, third person singular;
hence $*v$ *must be masculine, third person singular.*

Gender plays no role in verb forms in English. Demanding that the verb form be masculine is irrelevant, but harmless. If the grammar doesn't specify gender for verb forms, it follows that every form of every verb can be used in combination with a masculine subject.

An important concept that must be introduced here is *consistency*. The features of an object are called *inconsistent* if they contain conflicting information. As an example, consider the sentence "The boy scout ...", where "scout" is known to be both a noun and a verb form. If we continue from the previous item and scan a $*v$, we would obtain

$[VP{\to}{*v}{\bullet}NP, 2, 3]$
VP must be masculine, third person singular;
hence $*v$ *must be masculine, third person singular.*
$*v$ *is either plural or first or second person singular.*

This is inconsistent and therefore not acceptable as a valid item.

We need to introduce a tiny bit of notation in order to enhance the **Earley** parsing schema with features. The notation will be explained, but not defined in a mathematical sense. We write

- $\varphi_0(A{\to}\alpha)$ for the features of a production $A{\to}\alpha$;
- $\varphi(X)$ for the features of a constituent X;
- $\varphi([A{\to}\alpha{\bullet}\beta, i, j])$ for the features of an item $[A{\to}\alpha{\bullet}\beta, i, j]$.

The index 0 for features of productions is to indicate that these are taken straight from the grammar. In both other cases, features may have accumulated by transfer from previously recognized constituents and/or items.
The features of an item comprise the features of the production and those of its constituents (as far as these are known yet). From an item, the features of each constituent mentioned in that item can be retrieved.
We will not (yet) define a domain of expressions in which features can be formulated. This is left to the imagination of the reader. We need some notation, however, to relate sets of features to one another. *Combining* the features of objects ξ and η is denoted by $\varphi(\xi) \sqcup \varphi(\eta)$. The square union ($\sqcup$) may be interpreted as conventional set union ($\cup$) if it is understood that we accumulate sets of features. Similarly, we write $\varphi(\xi) \sqsubseteq \varphi(\eta)$ (which may be interpreted as $\varphi(\xi) \subseteq \varphi(\eta)$) to denote that an object η has at least all features of an object ξ but may have other features as well.

We will now extend the **Earley** parsing schema with the possibility to include features of constituents, productions and items. The parsing schema is defined by a parsing system $\mathbb{P}_{Earley} = \langle \mathcal{I}_{Earley}, H, D_{Earley} \rangle$ for an arbitrary context-free grammar G, where the set H is determined by the string to be parsed. The domain is defined by

$$\begin{aligned}\mathcal{I}_{Earley} = \{[A{\rightarrow}\alpha{\bullet}\beta, i, j]_\xi \;\mid\; & A{\rightarrow}\alpha\beta \in P \;\wedge\; 0 \leq i \leq j \;\wedge\\ & \varphi_0(A{\rightarrow}\alpha\beta) \sqsubseteq \varphi(\xi) \;\wedge\; \mathit{consistent}(\varphi(\xi))\};\end{aligned}$$

The ξ symbol is used only for easy reference. Subscripting $[A{\rightarrow}\alpha{\bullet}\beta, i, j]$ with ξ means that we may refer to the item as ξ in the remainder of the formula. The unabbreviated, somewhat more cumbersome notation for the same definition is

$$\begin{aligned}\mathcal{I}_{Earley} = \{[A{\rightarrow}\alpha{\bullet}\beta, i, j] \;\mid\; & A{\rightarrow}\alpha\beta \in P \;\wedge\; 0 \leq i \leq j \;\wedge\\ & \varphi_0(A{\rightarrow}\alpha\beta) \sqsubseteq \varphi([A{\rightarrow}\alpha{\bullet}\beta, i, j]) \;\wedge\\ & \mathit{consistent}(\varphi([A{\rightarrow}\alpha{\bullet}\beta, i, j])) \qquad \}.\end{aligned}$$

In words: it is mandatory that all features of a production be contained in an item that is based on that production. The item may have other features as well, as long as this does not lead to an inconsistency.
The deduction steps are the usual context-free deduction steps, annotated with how the features of the consequent are determined by the features of the antecedents:

$$\begin{aligned}
D^{Init} &= \{\vdash [S{\rightarrow}{\bullet}\gamma, 0, 0]_\xi \;\mid\; \varphi(\xi) = \varphi_0(S{\rightarrow}\gamma)\},\\
D^{Scan} &= \{[A{\rightarrow}\alpha{\bullet}a\beta, i, j]_\eta, [a, j, j+1]_\zeta \;\vdash\; [A{\rightarrow}\alpha a{\bullet}\beta, i, j+1]_\xi\\
&\qquad\qquad \mid\; \varphi(\xi) = \varphi(\eta) \sqcup \varphi(a_\zeta)\},\\
D^{Compl} &= \{[A{\rightarrow}\alpha{\bullet}B\beta, i, j]_\eta, [B{\rightarrow}\gamma{\bullet}, j, k]_\zeta \;\vdash\; [A{\rightarrow}\alpha B{\bullet}\beta, i, k]_\xi\\
&\qquad\qquad \mid\; \varphi(\xi) = \varphi(\eta) \sqcup \varphi(B_\zeta)\},
\end{aligned}$$

$$
\begin{aligned}
D^{Pred} &= \{[A{\rightarrow}\alpha{\bullet}B\beta, i, j]_\eta \ \vdash\ [B{\rightarrow}{\bullet}\gamma, j, j]_\xi \\
&\qquad\qquad\qquad\qquad \mid\ \varphi(\xi) = \varphi(B_\eta) \sqcup \varphi_0(B{\rightarrow}\gamma)\}, \\
D_{Earley} &= D^{Init} \cup D^{Scan} \cup D^{Compl} \cup D^{Pred}.
\end{aligned}
$$

The items have been subscripted with identifiers ξ, η, ζ for easy reference. The notation $\varphi(X_\eta)$ is used for those features of the item η that relate to constituent X.

7.2 The example grammar UG_1

We will look at a very simple example of a unification grammar. Our example grammar does not pretend to have any linguistic relevance. Moreover, the example deviates slightly from the usual examples as given by, e.g., Shieber [1986]. It is not our purpose to advocate the felicity of unification grammars to encode linguistic phenomena, but to show how context-free backbones of natural language grammars can be enhanced with features. Hence, we take the context-free example grammar that has been used in chapter 2 and simply add features to that grammar.

The **Earley** schema of the previous section is too advanced, for the time being, and we will parse strictly bottom-up in CYK fashion. If constituents B and C are known for a production $A{\rightarrow}BC$, then A can be recognized and an appropriate feature structure for it will be constructed.

Different features of a constituent can be stored in a *feature structure*. For each word in the language, the lexicon contains a feature structure[2]. The lexicon entry for the word "catches", for example, might look as follows

$$
\text{catches} \longmapsto \begin{bmatrix}
cat & : & {*v} \\
head & : & \begin{bmatrix} tense: & present \\ agr: & \boxed{1} \begin{bmatrix} number: & singular \\ person: & third \end{bmatrix} \end{bmatrix} \\
subject & : & \begin{bmatrix} head: \begin{bmatrix} agr: \boxed{1} \end{bmatrix} \end{bmatrix} \\
object & : & [\ ]
\end{bmatrix}
$$

[2]If several different feature structures coexist for the same word, we will simply treat these as belonging to separate (homonym) words. Disjunction within feature structures is discussed in Section 9.4. While (a limited form of) disjunction is very useful for practical purposes, one can always interpret feature structures with disjunction as a compact representation of a set of non-disjunctive feature structures. Hence, from a theoretical point of view, disallowing disjunction is no limitation to the power of the formalism.

features are listed in an *attribute-value matrix* (AVM). Every word has a feature *cat* describing the syntactic category. "Catches" has a feature *head* that contains some relevant information about the verb form. Furthermore, there are features *subject* and *object*, describing properties of the subject and direct object of the verb. The value of a feature can be some atomic symbol (as for *cat*); an AVM (as for *head* and *subject*), or unspecified (as for *object*). Unspecified features are denoted by an empty AVM, also called a *variable*. The intended meaning, in this case, is that the verb *catches* does have a direct object, but its features do not matter.

An important notion in AVMs is *coreference* (indicated by numbers contained in boxes). In the above example, the *head agr* feature is coreferenced with *subject head agr*, meaning that the agreement features of "catches" must be shared with the agreement features of its subject. Note, furthermore, that an entry within a nested structure of AVMs can be addressed by means of a *feature path*.

A first, very simple lexicon for the remainder of our canonical example sentence "the cat catches a mouse" is as follows:

$$\text{the, a} \longmapsto \left[\, cat: \ {*det} \right]$$

$$\text{cat, mouse} \longmapsto \begin{bmatrix} cat & : \ {*n} \\ head: & \begin{bmatrix} agr: \begin{bmatrix} number: & singular \\ person \ : & third \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

In order to parse the sentence, we need productions that tell us what to do with the features when we construct constituents. The syntactic categories of constituents are expressed by means of features, just like all other characteristic information. A formal, but somewhat austere way to express the construction of an NP from $*det$ and $*n$ is the following:

$$\begin{array}{l} X_0 \rightarrow X_1 \, X_2 \\ \quad \langle X_0 \ cat \rangle \doteq NP \\ \quad \langle X_1 \ cat \rangle \doteq {*det} \\ \quad \langle X_2 \ cat \rangle \doteq {*n} \\ \quad \langle X_0 \ head \rangle \doteq \langle X_2 \ head \rangle . \end{array} \tag{7.1}$$

That is, if we have constituents X_1, X_2 with *cat* features $*det$ and $*n$, respectively, we may create a new constituent with *cat* feature NP. Moreover, the *head* of X_0 is shared with the *head* of X_2.[3]

[3] In Chapter 8 we will make a distinction between *type identity* (denoted $=$) and *token identity* (denoted $\doteq$). As the distinction is not very relevant here, its introduction is postponed until Section 8.2, where we have developed the convenient terminology.

In most, if not all grammars it will be the case that all constituents have a *cat* feature. Hence we can simplify the notation of production (7.1) to

$$\begin{aligned} &NP \rightarrow {*det}\ {*n} \\ &\quad \langle NP\ head \rangle \doteq \langle {*n}\ head \rangle . \end{aligned} \tag{7.2}$$

The meaning of (7.1) and (7.2) is identical; the expression $\langle X_i\ \ cat \rangle \doteq A$ can be deleted when we substitute an A for X_i in the production. Thus we obtain context-free productions as usual, enhanced with so-called *constraints* that describe how the feature structures of the different constituents are related to one another. Hence, for the noun phrase "the cat" we may construct a feature structure with category NP and the *head* feature taken from the noun "cat:"

$$\text{the cat} \longmapsto \begin{bmatrix} cat & : NP \\ head: & \begin{bmatrix} agr: & \begin{bmatrix} number: & singular \\ person & : third \end{bmatrix} \end{bmatrix} \end{bmatrix} ;$$

similarly for "a mouse." For the construction of a VP, in the same vein, we employ the following production annotated with constraints:

$$\begin{aligned} &VP \rightarrow {*v}\ NP \\ &\quad \langle VP\ head \rangle \doteq \langle {*v}\ head \rangle \\ &\quad \langle VP\ subject \rangle \doteq \langle {*v}\ subject \rangle \\ &\quad \langle {*v}\ object \rangle \doteq \langle NP \rangle \end{aligned}$$

The verb phrase "catches a mouse" shares its *head* and *subject* features with the verb, while the entire (feature structure of the) NP is taken to be the direct object:

$$\text{catches a mouse} \longmapsto \begin{bmatrix} cat & : VP \\ head & : \begin{bmatrix} tense: & present \\ agr & : \boxed{1} \begin{bmatrix} number: & singular \\ person & : third \end{bmatrix} \end{bmatrix} \\ subject: & \begin{bmatrix} head: & \begin{bmatrix} agr: & \boxed{1} \end{bmatrix} \end{bmatrix} \\ object & : \begin{bmatrix} cat & : NP \\ head: & \begin{bmatrix} agr: & \begin{bmatrix} number: & singular \\ person & : third \end{bmatrix} \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

A sentence, finally, can be constructed from an NP and VP as follows:

$$\begin{aligned} &S \rightarrow NP\ VP \\ &\quad \langle S\ head \rangle \doteq \langle VP\ head \rangle \\ &\quad \langle VP\ subject \rangle \doteq \langle NP \rangle \end{aligned}$$

The sentence shares its head with the VP. The *subject* feature of the VP is shared with all features of the NP. Note that (by coreference) the subject of the verb phrase has (*head*) *agreement* third person singular. An NP can be substituted for the subject only if it has the same agreement. If the NP were to have a feature $\langle head\ agr\ number\rangle$ with value *plural*, then the S would obtain both *singular* and *plural* as values for its $\langle head\ agr\ number\rangle$ feature (because it is shared with the $\langle subject\ head\ agr\ number\rangle$ feature of the VP, which is shared with the $\langle head\ agr\ number\rangle$ feature of the VP). Such a clash of values would constitute an inconsistency, as discussed in Section 7.1. As a feature structure for S we obtain

$$\textsf{the cat catches a mouse} \longmapsto \begin{bmatrix} cat & : S \\ head : & \begin{bmatrix} tense: & present \\ agr & : \begin{bmatrix} number: & singular \\ person & : third \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

The entire sentence appears to have less features than its constituing parts NP and VP. That is because some features were present only to guarantee agreement between subject and verb. As the sentence has been produced, the agreement must have been okay, hence there is no need to retain this information explicitly in the feature structure for an S.

Above we have shown how syntactic constraints can be incorporated into the features of a grammar. We will also give an example of how semantic information can be collected from the lexicon and transferred upwards to contribute to the semantics of the sentence. We will use a very simple unification grammar UG_1. A relevant part of the lexicon for UG_1 is shown in Figure 7.1, the productions annotated with constraints are shown in Figure 7.2 The head of each feature structure is extended with a feature *trans*(*lation*), which is only a first, easy step towards translation of the constituent to its corresponding semantics. The translation of a verb is a predicate with the (translation of the) subject as first argument and the (translation of the) object as second argument.

The production $NP \rightarrow *det\ *n$ has been extended with another clause, stating that the *head trans* features of $*det$ and $*n$ are to be shared. Thus we obtain, for example

$$\textsf{a mouse} \longmapsto \begin{bmatrix} cat & : NP \\ head : & \begin{bmatrix} agr & : \begin{bmatrix} number: & singular \\ person & : third \end{bmatrix} \\ trans: & \begin{bmatrix} pred: & \textsf{mouse} \\ det & : - \end{bmatrix} \end{bmatrix} \end{bmatrix}.$$

Because the translation of the subject and object are used as arguments for the translation of the verb, the relevant properties of subject and object are moved

$$\textsf{the} \longmapsto \begin{bmatrix} cat & : & {*}det \\ head & : & \begin{bmatrix} trans : \begin{bmatrix} det : + \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

$$\textsf{a} \longmapsto \begin{bmatrix} cat & : & {*}det \\ head & : & \begin{bmatrix} trans : \begin{bmatrix} det : - \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

$$\textsf{cat} \longmapsto \begin{bmatrix} cat & : & {*}n \\ head & : & \begin{bmatrix} agr & : & \begin{bmatrix} number : singular \\ person : third \end{bmatrix} \\ trans & : & \begin{bmatrix} pred : \textsf{cat} \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

$$\textsf{mouse} \longmapsto \begin{bmatrix} cat & : & {*}n \\ head & : & \begin{bmatrix} agr & : & \begin{bmatrix} number : singular \\ person : third \end{bmatrix} \\ trans & : & \begin{bmatrix} pred : \textsf{mouse} \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

$$\textsf{catches} \longmapsto \begin{bmatrix} cat & : & {*}v \\ head & : & \begin{bmatrix} tense & : & present \\ agr & : & \boxed{1} \begin{bmatrix} number : singular \\ person : third \end{bmatrix} \\ trans & : & \begin{bmatrix} pred & : & \textsf{catch} \\ arg1 & : & \boxed{2} \begin{bmatrix} \, \end{bmatrix} \\ arg2 & : & \boxed{3} \begin{bmatrix} \, \end{bmatrix} \end{bmatrix} \end{bmatrix} \\ subject & : & \begin{bmatrix} head : \begin{bmatrix} agr & : & \boxed{1} \\ trans & : & \boxed{2} \end{bmatrix} \end{bmatrix} \\ object & : & \begin{bmatrix} head : \begin{bmatrix} trans : \boxed{3} \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

Figure 7.1: Part of the lexicon for UG_1

$$
\begin{array}{l}
S \rightarrow NP\ VP \\
\quad \langle S\ head \rangle \doteq \langle VP\ head \rangle \\
\quad \langle VP\ subject \rangle \doteq \langle NP \rangle \\[1ex]
VP \rightarrow {*v}\ NP \\
\quad \langle VP\ head \rangle \doteq \langle {*v}\ head \rangle \\
\quad \langle VP\ subject \rangle \doteq \langle {*v}\ subject \rangle \\
\quad \langle {*v}\ object \rangle \doteq \langle NP \rangle \\[1ex]
NP \rightarrow {*det}\ {*n} \\
\quad \langle NP\ head \rangle \doteq \langle {*n}\ head \rangle \\
\quad \langle {*n}\ head\ trans \rangle \doteq \langle {*det}\ head\ trans \rangle
\end{array}
$$

Figure 7.2: Some productions of UG_1

upward to a feature structure for the entire sentence. The reader may verify that, following the same steps as before, we obtain

$$
\textsf{the cat catches a mouse} \longmapsto \begin{bmatrix} cat : S \\ head : \begin{bmatrix} tense : present \\ agr : \begin{bmatrix} number : singular \\ person : third \end{bmatrix} \\ trans : \begin{bmatrix} pred : \textsf{catch} \\ arg1 : \begin{bmatrix} pred : \textsf{cat} \\ det : + \end{bmatrix} \\ arg2 : \begin{bmatrix} pred : \textsf{mouse} \\ det : - \end{bmatrix} \end{bmatrix} \end{bmatrix} \end{bmatrix} .
$$

Other features can be added likewise. We can add a *modifier* feature to the translation, in which modifiers like adjectives, adverbs and prepositional phrases can be stored. For a noun phrase "the very big, blue cat" we could envisage a feature structure as in Figure 7.3.

A noun phrase can include any number of modifiers, hence these are stored by means of a *list*. More sophisticated feature structure formalisms as, e.g., HPSG [Pollard and Sag, 1988], have special constructs for lists. Such constructs are convenient for notation, but not necessary. As shown in Figure 7.3, lists can be expressed in the basic formalism as well. In Section 9.5 a more complicated example is shown where lists are used for *subcategorization* of verbs.

$$
\begin{bmatrix}
cat & : & NP \\
head & : & \begin{bmatrix}
agr & : & \begin{bmatrix} number & : & singular \\ person & : & third \end{bmatrix} \\
trans & : & \begin{bmatrix}
pred & : & \mathsf{cat} \\
det & : & + \\
mod & : & \begin{bmatrix}
first & : & \begin{bmatrix}
trans & : & \mathsf{big} \\
mod & : & \begin{bmatrix}
first & : & \begin{bmatrix} trans & : & \mathsf{very} \\ mod & : & no \end{bmatrix} \\
rest & : & no
\end{bmatrix}
\end{bmatrix} \\
rest & : & \begin{bmatrix}
first & : & \begin{bmatrix} trans & : & \mathsf{blue} \\ mod & : & no \end{bmatrix} \\
rest & : & no
\end{bmatrix}
\end{bmatrix}
\end{bmatrix}
\end{bmatrix}
\end{bmatrix}
$$

Figure 7.3: feature structure of "the very big, blue cat"