Subsumption

Let A, B be feature structures over the same signature. We say that A *subsumes* B ($A \sqsubseteq B$; also, A is *more general than* B, and B is *subsumed by*, or is *more specific than*, A) if the following conditions hold:

1. if A is an atomic AVM then B is an atomic AVM with the same atom;
2. for every $\textsc{f} \in \textsc{Feats}$, if $\textsc{f} \in dom(A)$ then $\textsc{f} \in dom(B)$, and $val(A, \textsc{f})$ subsumes $val(B, \textsc{f})$; and
3. if two paths are reentrant in A, they are also reentrant in B: if $\pi_1 \overset{A}{\leftrightsquigarrow} \pi_2$ then $\pi_1 \overset{B}{\leftrightsquigarrow} \pi_2$.

Subsumption

Example: Subsumption

$$[\,] \sqsubseteq \begin{bmatrix}\text{NUM}: & \mathit{sg}\end{bmatrix}$$

$$\begin{bmatrix}\text{NUM}: & X\end{bmatrix} \sqsubseteq \begin{bmatrix}\text{NUM}: & \mathit{sg}\end{bmatrix}$$

$$\begin{bmatrix}\text{NUM}: & \mathit{sg}\end{bmatrix} \sqsubseteq \begin{bmatrix}\text{NUM}: & \mathit{sg} \\ \text{PERS}: & \mathit{third}\end{bmatrix}$$

$$\begin{bmatrix}\text{NUM1}: & \mathit{sg} \\ \text{NUM2}: & \mathit{sg}\end{bmatrix} \sqsubseteq \begin{bmatrix}\text{NUM1}: & \boxed{1}\,\mathit{sg} \\ \text{NUM2}: & \boxed{1}\end{bmatrix}$$

Subsumption

Subsumption is a *partial* relation: not every pair of feature structures is comparable:

$$\left[\textsc{num}: \quad \mathit{sg}\right] \quad \begin{matrix} \not\sqsubseteq \\ \not\sqsupseteq \end{matrix} \quad \left[\textsc{num}: \quad \mathit{pl}\right]$$

A different case of incomparability is caused by the existence of different features in the two structures:

$$\left[\textsc{num}: \quad \mathit{sg}\right] \quad \begin{matrix} \not\sqsubseteq \\ \not\sqsupseteq \end{matrix} \quad \left[\textsc{pers}: \quad \mathit{third}\right]$$

Subsumption

Example: Subsumption

While subsumption informally encodes an order of information content among AVMs, sometimes the informal notion can be misleading:

$$\begin{bmatrix} \text{NUM}: & \textit{sg} \\ \text{PERS}: & \textit{third} \end{bmatrix} \begin{matrix} \not\sqsubseteq \\ \not\sqsupseteq \end{matrix} \begin{bmatrix} \text{AGR}: & \begin{bmatrix} \text{NUM}: & \textit{sg} \\ \text{PERS}: & \textit{third} \end{bmatrix} \end{bmatrix}$$

Subsumption

Some properties of subsumption:

Least element: the empty feature structure subsumes every feature structure: for every feature structure A, $[\,] \sqsubseteq A$

Refelxivity: for every feature structure A, $A \sqsubseteq A$

Transitivity: If $A \sqsubseteq B$ and $B \sqsubseteq C$ than $A \sqsubseteq C$.

Antisymmetry: Subsumption is antisymmetric: if $A \sqsubseteq B$ and $B \sqsubseteq A$ then $A = B$.

To summarize, subsumption is a partial, reflexive, transitive and antisymmetric relation; it is therefore a *partial order*.

Unification

- The *unification* operation, denoted '$\sqcup$', is defined over pairs of feature structures, and yields the most general feature structure that is more specific than both operands, if one exists:
  $A = B \sqcup C$ if and only if A is the most general feature structure such that $B \sqsubseteq A$ and $C \sqsubseteq A$.
- If such a structure exists, the unification *succeeds*, and the two arguments are said to be *unifiable* (or *consistent*). If none exists, the unification *fails*, and the operands are said to be *inconsistent*.

Unification

Example: Unification

Unification combines consistent information:

$$\begin{bmatrix}\text{NUM}: & \textit{sg}\end{bmatrix} \sqcup \begin{bmatrix}\text{PERS}: & \textit{third}\end{bmatrix} = \begin{bmatrix}\text{NUM}: & \textit{sg} \\ \text{PERS}: & \textit{third}\end{bmatrix}$$

Different atoms are inconsistent:

$$\begin{bmatrix}\text{NUM}: & \textit{sg}\end{bmatrix} \sqcup \begin{bmatrix}\text{NUM}: & \textit{pl}\end{bmatrix} = \top$$

Example: (continued)

Atoms and non-atoms are inconsistent:

$$\left[\textsc{num}: \quad \textit{sg}\right] \sqcup \textit{sg} = \top$$

Example: (continued)

Unification is absorbing:

$$\begin{bmatrix}\text{NUM}: & \textit{sg}\end{bmatrix} \sqcup \begin{bmatrix}\text{NUM}: & \textit{sg} \\ \text{PERS}: & \textit{third}\end{bmatrix} = \begin{bmatrix}\text{NUM}: & \textit{sg} \\ \text{PERS}: & \textit{third}\end{bmatrix}$$

Example: (continued)

Empty feature structures are identity elements:

$$[\,] \sqcup \left[\textsc{agr}: \left[\textsc{num}: \; sg\right]\right] = \left[\textsc{agr}: \left[\textsc{num}: \; sg\right]\right]$$

Example: (continued)

Reentrancy causes two consistent values to coincide:

$$\begin{bmatrix} \mathrm{F}: & \begin{bmatrix} \mathrm{NUM}: & \mathit{sg} \end{bmatrix} \\ \mathrm{G}: & \begin{bmatrix} \mathrm{PERS}: & \mathit{third} \end{bmatrix} \end{bmatrix} \sqcup \begin{bmatrix} \mathrm{F}: & \boxed{1} \\ \mathrm{G}: & \boxed{1} \end{bmatrix} = \begin{bmatrix} \mathrm{F}: & \boxed{1} \begin{bmatrix} \mathrm{NUM}: & \mathit{sg} \\ \mathrm{PERS}: & \mathit{third} \end{bmatrix} \\ \mathrm{G}: & \boxed{1} \end{bmatrix}$$

Example: (continued)

Variables can be (partially) instantiated:

$$\left[\mathrm{F}: \quad X\right] \sqcup \left[\mathrm{F}: \quad \left[\mathrm{H}: \quad b\right]\right] = \left[\mathrm{F}: \quad X(\left[\mathrm{H}: \quad b\right])\right]$$

Example: (continued)

Unification acts differently depending on whether the values are equal:

$$\begin{bmatrix} \text{F}: & \begin{bmatrix}\text{NUM}: & \textit{sg}\end{bmatrix} \\ \text{G}: & \begin{bmatrix}\text{NUM}: & \textit{sg}\end{bmatrix} \end{bmatrix} \sqcup \begin{bmatrix}\text{F}: & \begin{bmatrix}\text{PERS}: & \textit{3rd}\end{bmatrix}\end{bmatrix} = \begin{bmatrix} \text{F}: & \begin{bmatrix}\text{NUM}: & \textit{sg} \\ \text{PERS}: & \textit{3rd}\end{bmatrix} \\ \text{G}: & \begin{bmatrix}\text{NUM}: & \textit{sg}\end{bmatrix} \end{bmatrix}$$

...or identical:

$$\begin{bmatrix} \text{F}: & \boxed{1}\begin{bmatrix}\text{NUM}: & \textit{sg}\end{bmatrix} \\ \text{G}: & \boxed{1} \end{bmatrix} \sqcup \begin{bmatrix}\text{F}: & \begin{bmatrix}\text{PERS}: & \textit{3rd}\end{bmatrix}\end{bmatrix} = \begin{bmatrix} \text{F}: & \boxed{1}\begin{bmatrix}\text{NUM}: & \textit{sg} \\ \text{PERS}: & \textit{3rd}\end{bmatrix} \\ \text{G}: & \boxed{1} \end{bmatrix}$$

Variable binding

Unification *binds* variables together. Let:

$$A = \left[\text{F}: \ \boxed{1}\left[\text{NUM}: \ \textit{sg}\right]\right] \qquad B = \left[\text{F}: \ \boxed{2}\left[\text{PERS}: \ \textit{third}\right]\right]$$

Then:

$$A \sqcup B = \left[\text{F}: \ \boxed{1}\boxed{2}\begin{bmatrix}\text{NUM}: & \textit{sg} \\ \text{PERS}: & \textit{third}\end{bmatrix}\right]$$

Of course, since the variables $\boxed{1}$ and $\boxed{2}$ occur nowhere else, they can be simply omitted and the result is equal to:

$$A \sqcup B = \left[\text{F}: \ \begin{bmatrix}\text{NUM}: & \textit{sg} \\ \text{PERS}: & \textit{third}\end{bmatrix}\right]$$

Variable binding

However, had either $\boxed{1}$ or $\boxed{2}$ occurred elsewhere (for example, as the value of some feature G in A), their values would have been modified as a result of the unification:

$$\begin{bmatrix} \text{F}: & \boxed{1}\begin{bmatrix}\text{NUM}: & \textit{sg}\end{bmatrix} \\ \text{G}: & \boxed{1} \end{bmatrix} \sqcup \begin{bmatrix} \text{F}: & \boxed{2}\begin{bmatrix}\text{PERS}: & \textit{third}\end{bmatrix}\end{bmatrix} =$$

$$\begin{bmatrix} \text{F}: & \boxed{3}\begin{bmatrix}\text{NUM}: & \textit{sg} \\ \text{PERS}: & \textit{third}\end{bmatrix} \\ \text{G}: & \boxed{3} \end{bmatrix}$$

Unification

Some properties of unification:

Idempotency: $A \sqcup A = A$

Commutativity: $A \sqcup B = B \sqcup A$

Associativity: $A \sqcup (B \sqcup C) = (A \sqcup B) \sqcup C$

Absorption: If $A \sqsubseteq B$ then $A \sqcup B = B$

Monotonicity: If $A \sqsubseteq B$ then for every C, $A \sqcup C \sqsubseteq B \sqcup C$ (if both exist).

Generalization

- Generalization (denoted $\sqcap$) is the operation that returns the most specific (or least general) feature structure that is still more general than both arguments.
- Unlike unification, generalization can never fail. For every pair of feature structures there exists a feature structure that is more general than both: in the most extreme case, pick the empty feature structure, which is more general than every other structure.

Generalization

Example: Generalization

Generalization reduces information:

$$\left[\text{NUM}: \quad \textit{sg}\right] \sqcap \left[\text{PERS}: \quad \textit{third}\right] = [\,]$$

Different atoms are inconsistent:

$$\left[\text{NUM}: \quad \textit{sg}\right] \sqcap \left[\text{NUM}: \quad \textit{pl}\right] = \left[\text{NUM}: \quad [\,]\right]$$

Example: (continued)

Generalization is restricting:

$$\begin{bmatrix}\text{NUM}: & \textit{sg}\end{bmatrix} \sqcap \begin{bmatrix}\text{NUM}: & \textit{sg} \\ \text{PERS}: & \textit{third}\end{bmatrix} = \begin{bmatrix}\text{NUM}: & \textit{sg}\end{bmatrix}$$

Example: (continued)

Empty feature structures are zero elements:

$$[\,] \sqcap \begin{bmatrix}\text{AGR}: & \begin{bmatrix}\text{NUM}: & \textit{sg}\end{bmatrix}\end{bmatrix} = [\,]$$

Reentrancies can be lost:

$$\begin{bmatrix}\text{F}: & \boxed{1}\begin{bmatrix}\text{NUM}: & \textit{sg}\end{bmatrix} \\ \text{G}: & \boxed{1}\end{bmatrix} \sqcap \begin{bmatrix}\text{F}: & \begin{bmatrix}\text{NUM}: & \textit{sg}\end{bmatrix} \\ \text{G}: & \begin{bmatrix}\text{NUM}: & \textit{sg}\end{bmatrix}\end{bmatrix} = \begin{bmatrix}\text{F}: & \begin{bmatrix}\text{NUM}: & \textit{sg}\end{bmatrix} \\ \text{G}: & \begin{bmatrix}\text{NUM}: & \textit{sg}\end{bmatrix}\end{bmatrix}$$

Generalization

Some properties of generalization:

Idempotency: $A \sqcap A = A$

Commutativity: $A \sqcap B = B \sqcap A$

Absorption: If $A \sqsubseteq B$ then $A \sqcap B = A$

Using feature structures for representing lists

Feature structures can be easily used to encode (finite) lists. As an example, consider the following representation of the list $\langle 1, 2, 3 \rangle$ (assuming a signature whose atoms include the numbers $1, 2, 3$):

Example: Feature structure encoding of a list

$$\begin{bmatrix} \text{FIRST}: & 1 \\ \text{REST}: & \begin{bmatrix} \text{FIRST}: & 2 \\ \text{REST}: & \begin{bmatrix} \text{FIRST}: & 3 \\ \text{REST}: & \textit{elist} \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

Using feature structures for representing lists

Example: A nested list

The list $\langle\langle 1, 2, 6, 7\rangle, \langle 3, 4\rangle, \langle 5\rangle\rangle$:

$$
\begin{bmatrix}
\text{FIRST}: & \begin{bmatrix} \text{FIRST}: & 1 \\ \text{REST}: & \begin{bmatrix} \text{FIRST}: & 2 \\ \text{REST}: & \begin{bmatrix} \text{FIRST}: & 6 \\ \text{REST}: & \begin{bmatrix} \text{FIRST}: & 7 \\ \text{REST}: & \textit{elist} \end{bmatrix} \end{bmatrix} \end{bmatrix} \end{bmatrix} \\
\text{REST}: & \begin{bmatrix} \text{FIRST}: & \begin{bmatrix} \text{FIRST}: & 3 \\ \text{REST}: & \begin{bmatrix} \text{FIRST}: & 4 \\ \text{REST}: & \textit{elist} \end{bmatrix} \end{bmatrix} \\ \text{REST}: & \begin{bmatrix} \text{FIRST}: & \begin{bmatrix} \text{FIRST}: & 5 \\ \text{REST}: & \textit{elist} \end{bmatrix} \\ \text{REST}: & \textit{elist} \end{bmatrix} \end{bmatrix}
\end{bmatrix}
$$

Adding features to rules

- Phrases, like words, have valued features and consequently, grammar non-terminals, too, are decorated with features.
- When a feature is assigned to a non-terminal symbol C, it means that this feature is appropriate for *all* the phrases of category C: it makes sense for it to appear in all the instances of C.
- Such categories interact, in the grammar, with other AVMs, through application of rules, and the specified values might thus undergo changes. In general, AVMs are changed as a result of rule application.
- We refer to such enriched categories as *generalized categories* (or *extended* ones), which have a *base category* and an associated feature structure.

Adding features to rules

Example:

A feature structure that might be associated with phrases of category *NP* (noun phrases).

$$NP \begin{bmatrix} \text{NUM}: & [\,] \\ \text{PERS}: & [\,] \end{bmatrix}$$

A third person singular noun phrase such as lamb may be associated with:

$$NP \begin{bmatrix} \text{NUM}: & sg \\ \text{PERS}: & third \end{bmatrix}$$

Extended grammar rules

Example: Rules for imposing number agreement

$$(1) \quad \underset{[\text{NUM}: \ X]}{N} \rightarrow \underset{[\text{NUM}: \ X(sg)]}{lamb}$$

$$(2) \quad \underset{[\text{NUM}: \ X]}{N} \rightarrow \underset{[\text{NUM}: \ X(pl)]}{lambs}$$

$$(3) \quad S \rightarrow \underset{[\text{NUM}: \ X]}{NP} \quad \underset{[\text{NUM}: \ X]}{VP}$$

$$(4) \quad \underset{[\text{NUM}: \ X]}{NP} \rightarrow \underset{[\text{NUM}: \ X]}{D} \quad \underset{[\text{NUM}: \ X]}{N}$$

Scope of variables

The scope of a variable is the grammar rule in which it occurs. Reformulating rule (2) as

$$\begin{matrix} N \\ [\textsc{num}: \; Y] \end{matrix} \rightarrow \begin{matrix} \textit{lambs} \\ [\textsc{num}: \; Y(pl)] \end{matrix}$$

has no effect.
No sharing is implied by occurrences of the same variables in different rules, for example the occurrences of X in rules (1) and (2) above.

Declarativity

- Consider rule 4:

$$(4) \quad \begin{matrix} NP \\ \left[\textsc{num}: \; X\right] \end{matrix} \rightarrow \begin{matrix} D \\ \left[\textsc{num}: \; X\right] \end{matrix} \quad \begin{matrix} N \\ \left[\textsc{num}: \; X\right] \end{matrix}$$

- This rule stipulates that in order to form a noun phrase (NP) from the concatenation of a determiner (D) and a noun (N), the NUM features of the determiner and the noun must agree.
- The NUM feature of the noun phrase thus constructed is equal to that of either daughter.
- What the rule does *not* determine is an *order* of this value check.
- The undirectional view of agreement is a typical view of unification-based grammar formalisms.

Extending AVMs

- Multi-AVMs can be viewed as sequences of AVMs, with the important observation that some sub-structures can be shared among two or more AVMSs.
- In other words, the scope of variables is extended from a single AVM to a multi-AVM: the same variable can be associated with two sub-structures of different AVMs.
- The notion of well-formedness is extended to multi-AVMs.
- The function *val*, associating a value with features (and paths) has to be extended, too.
- If the value of the path π_1 leaving the i-th root of σ is reentrant with the value of the path π_2 leaving the j-th root, we write $(i, \pi_1) \overset{\sigma}{\leftrightsquigarrow} (j, \pi_2)$.

Extending AVMs

Example: Multi-AVM

Let σ be the multi-AVM:

$$\begin{bmatrix} \text{F}: & \begin{bmatrix} \text{G}: & a \\ \text{H}: & X \end{bmatrix} \end{bmatrix} \quad \begin{bmatrix} \text{G}: & Y \end{bmatrix} \quad \begin{bmatrix} \text{F}: & \begin{bmatrix} \text{H}: & b \\ \text{G}: & X \end{bmatrix} \\ \text{H}: & a \end{bmatrix}$$

Then $val(\sigma, 1, \langle \text{F} \rangle)$ is: $\begin{bmatrix} \text{G}: & a \\ \text{H}: & [\,] \end{bmatrix}$

whereas $val(\sigma, 3, \langle \text{F} \rangle)$ is: $\begin{bmatrix} \text{H}: & b \\ \text{G}: & [\,] \end{bmatrix}$

In this example, $(1, \langle \text{F H} \rangle) \overset{\sigma}{\leftrightsquigarrow} (3, \langle \text{F G} \rangle)$.

Example: (continued)

A multi-AVM can have an empty feature structure as an element:

$$\begin{bmatrix} \text{F} : & \begin{bmatrix} \text{G} : & a \\ \text{H} : & X \end{bmatrix} \end{bmatrix} \quad [\,] \quad \begin{bmatrix} \text{F} : & \begin{bmatrix} \text{H} : & b \\ \text{G} : & X \end{bmatrix} \\ \text{H} : & a \end{bmatrix}$$

Extending AVMs

The following is a valid multi-AVM:

$$\begin{bmatrix} \text{F} : & a \\ \text{G} : & \boxed{2} \end{bmatrix} \quad \boxed{2} \begin{bmatrix} \text{H} : & b \end{bmatrix}$$

The only restriction is that the same variable cannot be associated with two *different* elements in the sequence. Thus, the following is *not* a multi-AVM:

$$\boxed{2} \begin{bmatrix} \text{H} : & b \end{bmatrix} \quad \begin{bmatrix} \text{F} : & a \\ \text{G} : & \boxed{2} \end{bmatrix} \quad \boxed{2}$$

Subsumption

The notion of subsumption can be naturally extended from AVMs to multi-AVMs: if σ and ρ are two multi-AVMs *of the same length*, n, then $\sigma \sqsubseteq \rho$ if the following conditions hold:

1. every element of σ subsumes the corresponding element of ρ: for every i, $1 \leq i \leq n$, $val(\sigma, i, \epsilon) \sqsubseteq val(\rho, i, \epsilon)$; and
2. if two paths are reentrant in σ, they are also reentrant in ρ: if $(i, \pi_1) \overset{\sigma}{\leftrightsquigarrow} (j, \pi_2)$ then $(i, \pi_1) \overset{\rho}{\leftrightsquigarrow} (j, \pi_2)$.

Subsumption

Example: Multi-AVM subsumption

Let σ be: $\begin{bmatrix} \text{F} : & \begin{bmatrix} \text{G} : & a \\ \text{H} : & X \end{bmatrix} \end{bmatrix}$ $\begin{bmatrix} \text{G} : & c \end{bmatrix}$ $\begin{bmatrix} \text{F} : & \begin{bmatrix} \text{H} : & b \\ \text{G} : & X(d) \end{bmatrix} \\ \text{H} : & a \end{bmatrix}$

and ρ be: $\begin{bmatrix} \text{F} : & \begin{bmatrix} \text{G} : & a \\ \text{H} : & d \end{bmatrix} \end{bmatrix}$ $\begin{bmatrix} \text{G} : & c \end{bmatrix}$ $\begin{bmatrix} \text{F} : & \begin{bmatrix} \text{H} : & b \\ \text{G} : & d \end{bmatrix} \\ \text{H} : & a \end{bmatrix}$

Then σ does not subsume ρ, but $\rho \sqsubseteq \sigma$.

Unification

In the same way, the notion of unification can be extended to multi-AVMs (of the same length): we say that ρ is the unification of σ_1 and σ_2 (and write $\rho = \sigma_1 \sqcup \sigma_2$) if σ_1, σ_2 and ρ are of the same length, and ρ is the most general multi-AVM that is more specific than both σ_1 and σ_2.

Rules and grammars

- An extended context-free rule consists of two components: a context-free rule, and a multi-AVM of the same length.
- A unification grammar consists of a set of extended context-free rules and an extended category that serves as the *start symbol*.

Unification grammars

Example: G_1, a unification grammar for E_0

$$
\begin{array}{llcll}
(1) & S & \rightarrow & \underset{\left[\text{NUM}: \; X\right]}{NP} & \underset{\left[\text{NUM}: \; X\right]}{VP} \\
(2) & \underset{\left[\text{NUM}: \; X\right]}{NP} & \rightarrow & \underset{\left[\text{NUM}: \; X\right]}{D} & \underset{\left[\text{NUM}: \; X\right]}{N} \\
(3) & \underset{\left[\text{NUM}: \; X\right]}{VP} & \rightarrow & \underset{\left[\text{NUM}: \; X\right]}{V} &
\end{array}
$$

Example: (continued)

$$(4) \quad \begin{matrix} VP \\ [\text{NUM}: \; X] \end{matrix} \rightarrow \begin{matrix} V \\ [\text{NUM}: \; X] \end{matrix} \quad \begin{matrix} NP \\ [\text{NUM}: \; Y] \end{matrix}$$

$$(5,6) \quad \begin{matrix} N \\ [\text{NUM}: \; X] \end{matrix} \rightarrow \begin{matrix} lamb \\ [\text{NUM}: \; X(sg)] \end{matrix} \; | \; \begin{matrix} sheep \\ [\text{NUM}: \; X] \end{matrix} \; | \; \ldots$$

$$(7,8) \quad \begin{matrix} V \\ [\text{NUM}: \; X] \end{matrix} \rightarrow \begin{matrix} sleeps \\ [\text{NUM}: \; X(sg)] \end{matrix} \; | \; \begin{matrix} sleep \\ [\text{NUM}: \; X(pl)] \end{matrix} \; | \; \ldots$$

$$(9,10) \quad \begin{matrix} D \\ [\text{NUM}: \; X] \end{matrix} \rightarrow \begin{matrix} a \\ [\text{NUM}: \; X(sg)] \end{matrix} \; | \; \begin{matrix} two \\ [\text{NUM}: \; X(pl)] \end{matrix} \; | \; \ldots$$

Rule application

- Forms and sentential forms
- Derivations
- Derivation trees
- Language

Forms

- *Forms* are generalized and are composed of “sequences” of generalized categories, that is, of a sequence of base categories or words, augmented by a multi-AVM of the same length.
- We use Greek letters such as α, β as meta-variables over forms. For example, following is a form of length two:

$$\begin{array}{cc} \mathit{NP} & \mathit{VP} \\ \left[\textsc{num}: \quad Y\right] & \left[\textsc{num}: \quad Y\right] \end{array}$$

Derivations

Derivation is a binary relation over generalized forms. Let α be a generalized form, and $B_0 \rightarrow B_1 B_2 \ldots B_k$ a grammar rule, where the B_i are all generalized categories, and where reentrancies might occur among elements of the form or the rule. Application of the rule to α consists of the following steps:

- Matching the rule's head with some element of the form that has the same base category;
- Replacing the selected element in the form with the body of the rule, producing a new form.

Derivations

Example: Matching

Suppose that

$$\alpha = \underset{[\text{NUM}: \quad Y]}{NP} \quad \underset{[\text{NUM}: \quad Y]}{VP}$$

is a (sentential) form and that

$$\rho = \underset{[\text{NUM}: \quad X]}{VP} \rightarrow \underset{[\text{NUM}: \quad X]}{V} \quad \underset{[\text{NUM}: \quad W]}{NP}$$

is a rule. Let the selected element of α be its second element, namely the extended category

$$\underset{[\text{NUM}: \quad Y]}{VP}$$

Example: (continued)

This extended category matches the head of the rule ρ, as the base categories are identical (VP) and the AVMs associated with them are unifiable (consistent). The result of the unification is the extended category

$$\begin{matrix} VP \\ \left[\textsc{num}: \quad Z\right] \end{matrix}$$

which is equivalent to

$$\begin{matrix} VP \\ \left[\textsc{num}: \quad [\,]\right] \end{matrix}$$

An additional effect of the unification is that the variables Y of the form and X of the rule are unified, too.

Replacement

- The two feature structures (associated with the head of the rule and with the selected element) are unified in their respective *contexts*: the body of the rule and the form.
- When some variable X in the form is unified with some variable Y in the rule, all occurrences of X in the form and of Y in the rule are modified: they are all set to the unified value.
- The replacement operation inserts the modified rule body into the modified form, replacing the selected element of the form.
- The variables of the resulting form are then systematically renamed.

Derivation

Example: Derivation step

Let

$$\alpha = \begin{matrix} NP \\ [\text{NUM}: \; Y] \end{matrix} \quad \begin{matrix} VP \\ [\text{NUM}: \; Y] \end{matrix}$$

$$\rho = \begin{matrix} VP \\ [\text{NUM}: \; X] \end{matrix} \rightarrow \begin{matrix} V \\ [\text{NUM}: \; X] \end{matrix} \quad \begin{matrix} NP \\ [\text{NUM}: \; W] \end{matrix}$$

be a form and a rule, respectively. The unification of the rule's head with the second element of α succeeds, and identifies the values of X and Y . After replacement and variable renaming we obtain:

$$\beta = \begin{matrix} NP \\ [\text{NUM}: \; X_1] \end{matrix} \quad \begin{matrix} V \\ [\text{NUM}: \; X_1] \end{matrix} \quad \begin{matrix} NP \\ [\text{NUM}: \; W_1] \end{matrix}$$

Example: (continued)

Now assume that the (terminal) rule

$$\begin{matrix} V \\ \left[\text{NUM}: \quad Y\right] \end{matrix} \rightarrow \begin{matrix} herds \\ \left[\text{NUM}: \quad Y(sg)\right] \end{matrix}$$

is to be applied to β. The value of the variable X_1 in the form is set, through unification, to *sg*, and the resulting form is:

$$\gamma = \begin{matrix} NP \\ \left[\text{NUM}: \quad X_2\right] \end{matrix} \quad \begin{matrix} herds \\ \left[\text{NUM}: \quad X_2(sg)\right] \end{matrix} \quad \begin{matrix} NP \\ \left[\text{NUM}: \quad W_2\right] \end{matrix}$$

Note that the first *NP* had its feature structure modified, even though it did not participate directly in the rule application.

Derivation

Example: Derivation step (continued)

Assume now that γ is expanded by applying to its first element the rule

$$\begin{matrix} NP \\ [\text{NUM}: \; X] \end{matrix} \rightarrow \begin{matrix} D \\ [\text{NUM}: \; X] \end{matrix} \; \begin{matrix} N \\ [\text{NUM}: \; X] \end{matrix}$$

In this case, unification of the first element of γ with the head of the rule binds the value of X in the rule to *sg*:

$$\delta = \begin{matrix} D \\ [\text{NUM}: \; X_3] \end{matrix} \; \begin{matrix} N \\ [\text{NUM}: \; X_3] \end{matrix} \; \begin{matrix} \textit{herds} \\ [\text{NUM}: \; X_3(sg)] \end{matrix} \; \begin{matrix} NP \\ [\text{NUM}: \; W_3] \end{matrix}$$

Example: (continued)

If we now tried to apply the (terminal) rule

$$\begin{matrix} D \\ [\text{NUM}: \quad Y] \end{matrix} \longrightarrow \begin{matrix} \textit{two} \\ [\text{NUM}: \quad Y(pl)] \end{matrix}$$

to the first element of δ, this attempt would have caused unification failure.

Derivation

Example: Derivation with ϵ-rules

Let

$$\alpha = \overset{A}{\begin{bmatrix} \mathrm{F}: & X \end{bmatrix}} \quad \overset{B}{\begin{bmatrix} \mathrm{F}: & X \\ \mathrm{G}: & Y \end{bmatrix}} \quad \overset{C}{\begin{bmatrix} \mathrm{G}: & Y \end{bmatrix}}, \qquad \rho = \overset{B}{\begin{bmatrix} \mathrm{F}: & Z \\ \mathrm{G}: & Z \end{bmatrix}} \rightarrow \epsilon$$

be a form and a rule, respectively. Applying the rule to the second element of the form yields:

$$\overset{A}{\begin{bmatrix} \mathrm{F}: & W \end{bmatrix}} \quad \overset{C}{\begin{bmatrix} \mathrm{G}: & W \end{bmatrix}}$$

Derivation

Example: Derivation can modify information

Let

$$\alpha = \overset{A}{\left[\mathrm{F}: \quad a\right]} \quad \overset{B}{\left[\mathrm{G}: \quad b\right]}, \qquad \rho = \overset{A}{\left[\mathrm{F}: \quad a\right]} \rightarrow \overset{A}{\left[\mathrm{F}: \quad c\right]}$$

be a form and a rule, respectively. Applying the rule to the first element of the form yields:

$$\overset{A}{\left[\mathrm{F}: \quad c\right]} \quad \overset{B}{\left[\mathrm{G}: \quad b\right]}$$

Notice that in the result, the value of F in A was modified from a to c.

Derivation

- The full derivation relation is, as usual, the reflexive-transitive closure of rule application.
- A form is *sentential* if it is derivable from the start symbol.

Derivation

Consider a derivation of the sentence two sheep sleep with the grammar G_1. After each rule is applied, the variables in the obtained form are renamed.

Example: Derivation

The derivation starts with the start symbol, which is the extended category S. Applying rule (1), one gets:

$$\begin{array}{cc} NP & VP \\ \left[\text{NUM}: \quad X_1\right] & \left[\text{NUM}: \quad X_1\right] \end{array}$$

Example: (continued)

It is now possible to select the leftmost element in the above sentential form and to apply rule (2). Renaming all occurrences of X in rule (2) to X_2, one gets the following sentential form:

$$\begin{array}{ccc} D & N & VP \\ [\text{NUM}: \; X_2] & [\text{NUM}: \; X_2] & [\text{NUM}: \; X_2] \end{array}$$

Now select the rightmost element in the above form and apply rule (3), renaming all occurrences of X_2 to X_3:

$$\begin{array}{ccc} D & N & V \\ [\text{NUM}: \; X_3] & [\text{NUM}: \; X_3] & [\text{NUM}: \; X_3] \end{array}$$

Example: (continued)

The leftmost element is selected, and (the terminal) rule (10) is applied, binding X_3 to *pl*:

$$\overset{\textit{two}}{\left[\textit{num}: \quad X_3(\textit{pl})\right]} \quad \overset{N}{\left[\text{NUM}: \quad X_3\right]} \quad \overset{V}{\left[\text{NUM}: \quad X_3\right]}$$

In the same way, rule (6) can be applied to the middle element in this form, and rule (8) to the rightmost, resulting in:

$$\overset{\textit{two}}{\left[\text{NUM}: \quad X_3(\textit{pl})\right]} \quad \overset{\textit{sheep}}{\left[\text{NUM}: \quad X_3(\textit{pl})\right]} \quad \overset{\textit{sleep}}{\left[\text{NUM}: \quad X_3(\textit{pl})\right]}$$

Thus the string two sheep sleep is indeed a sentence.

Derivation trees

Example: Snapshots of a derivation sequence

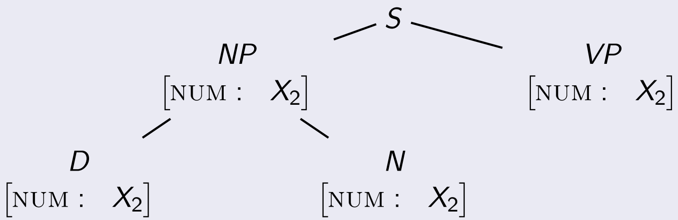
We begin with the start symbol, the extended category S, which is expanded by applying rule (1), yielding (after renaming):

$$S$$

$$NP \begin{bmatrix} \text{NUM} : & X_1 \end{bmatrix} \qquad VP \begin{bmatrix} \text{NUM} : & X_1 \end{bmatrix}$$

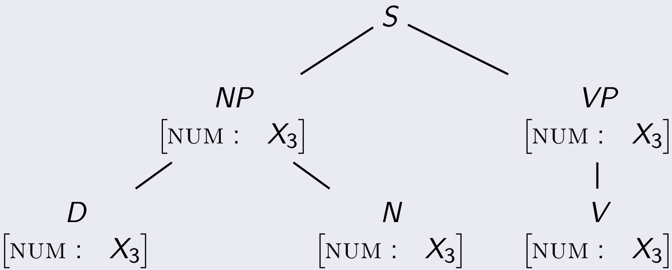
Example: (continued)

The next step is the application of rule (2) to the leftmost element in the frontier of the tree. Since this application results in binding X_1 with X_2, we rename all occurrences of X_1 in the tree to X_2, obtaining the following tree:



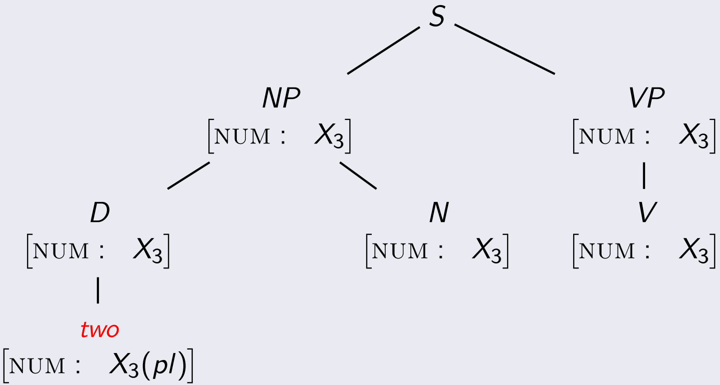
Example: (continued)

Now select the rightmost element in the frontier of the above tree and apply rule (3), renaming all occurrences of X_2 in the tree to X_3; the following tree is obtained:



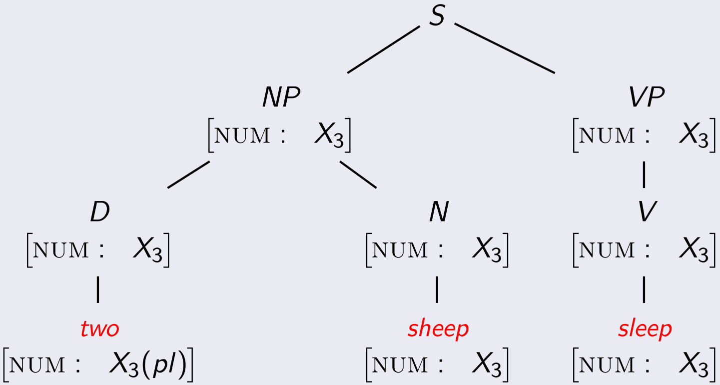
Example: (continued)

Next, the leftmost element is selected, and (the terminal) rule (10) is applied, binding X_3 to *pl*:



Example: (continued)

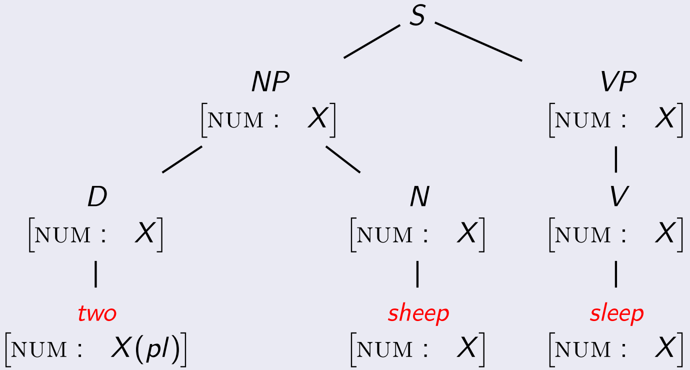
Similarly, rule (6) can be applied to the middle element in the frontier, and rule (8) to the rightmost, yielding:



Derivation trees

The final derivation tree for the same sentence:

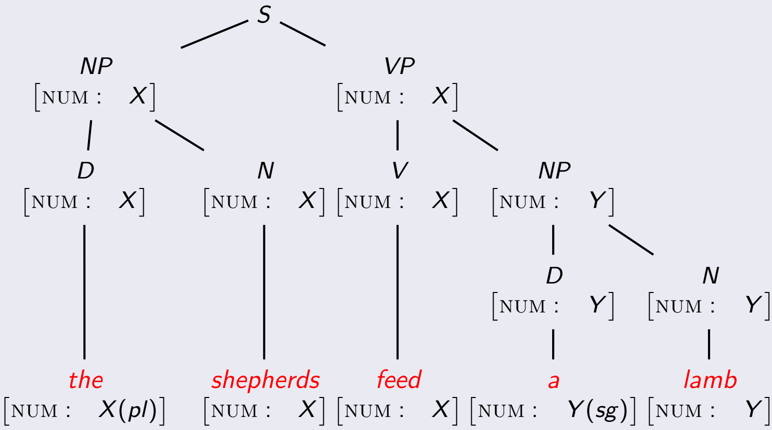
Example: Derivation tree



Derivation trees

The final derivation tree for the sentence the shepherds feed a lamb:

Example: Derivation tree



Languages

- To determine whether a sequence of words, $w = a_1 \cdots a_n$, is in $L(G)$, consider a derivation in G whose first form consists of the start symbol (an extended category, viewed as an extended form of length 1), and whose last form is $\langle w, \sigma' \rangle$.
- Let $\langle w, \sigma \rangle$ be an extended form obtained by concatenating $A_1, \ldots, A_n$, where each A_i is a lexical entry of the word a_1.
- We say that $w \in L(G)$ if and only if σ' be a multi-AVM that is unifiable with σ: $\sigma \sqcup \sigma'$ does not fail.

Languages

Example: Language

Given this definition, observe, for example, that the string two sheep sleep is in the language generated by the example grammar G_1; we have seen a derivation sequence for this string. The first and the last elements of this sequence, namely the feature structures associated with the words two and sleep, are identical to lexical entries of G_1. However, the middle element, namely the feature structure associated with sheep, is more specific than (subsumed by) the lexical entry of sheep.

Languages

The language generated by the grammar G_1 *is* context free:

Example: A context-free grammar G_1'

$S \longrightarrow S_{sg} \mid S_{pl}$

$S_{sg} \longrightarrow NP_{sg}\ VP_{sg}$

$NP_{sg} \longrightarrow D_{sg}\ N_{sg}$

$VP_{sg} \longrightarrow V_{sg}$

$VP_{sg} \longrightarrow V_{sg}\ NP_{sg} \mid V_{sg}\ NP_{pl}$

$D_{sg} \longrightarrow a$

$N_{sg} \longrightarrow lamb \mid sheep \mid \cdots$

$V_{sg} \longrightarrow sleeps \mid \cdots$

$S_{pl} \longrightarrow NP_{pl}\ VP_{pl}$

$NP_{pl} \longrightarrow D_{pl}\ N_{pl}$

$VP_{pl} \longrightarrow V_{pl}$

$VP_{pl} \longrightarrow V_{pl}\ NP_{sg} \mid V_{pl}\ NP_{pl}$

$D_{pl} \longrightarrow two$

$N_{pl} \longrightarrow lambs \mid sheep \mid \cdots$

$V_{pl} \longrightarrow sleep \mid \cdots$

Imposing case control

- The extensions of the CFG formalism can be used for imposing various constraints on generated languages. Here we suggest a solution for the problem of controlling the case of a noun phrase.
- First, add pronouns to the grammar:

$$(2.1)\quad \underset{[\text{NUM}:\ X]}{NP} \rightarrow \underset{[\text{NUM}:\ X]}{D}\ \underset{[\text{NUM}:\ X]}{N}$$

$$(2.2)\quad \underset{[\text{NUM}:\ X]}{NP} \rightarrow \underset{[\text{NUM}:\ X]}{PropN}$$

$$(2.3)\quad \underset{[\text{NUM}:\ X]}{NP} \rightarrow \underset{[\text{NUM}:\ X]}{Pron}$$

Imposing case control

- Additionally, the following terminal rules are needed:

$$\begin{array}{c} PropN \\ \left[\mathrm{NUM}: \quad sg\right] \end{array} \quad \rightarrow \quad \text{Jacob} \mid \text{Rachel} \mid \ldots$$

$$\begin{array}{c} Pron \\ \left[\mathrm{NUM}: \quad sg\right] \end{array} \quad \rightarrow \quad \text{she} \mid \text{her} \mid \ldots$$

Imposing case control

- The additional rules allow sentences such as

  She herds the sheep
  Jacob loves her

- but also non-sentences such as

  $*$*Her herds the sheep*
  $*$*Jacob loves she*

Imposing case control

- We add a feature, CASE, to the feature structures associated with nominal categories: nouns, pronouns, proper names and noun phrases.
- What should the values of the CASE feature be?

Imposing case control

(11) $\overset{\textit{PropN}}{\begin{bmatrix} \text{NUM}: & X \\ \text{CASE}: & Y \end{bmatrix}} \rightarrow \overset{\textit{Rachel}}{\begin{bmatrix} \text{NUM}: & X(sg) \\ \text{CASE}: & Y \end{bmatrix}}$

(12) $\overset{\textit{PropN}}{\begin{bmatrix} \text{NUM}: & X \\ \text{CASE}: & Y \end{bmatrix}} \rightarrow \overset{\textit{Jacob}}{\begin{bmatrix} \text{NUM}: & X(sg) \\ \text{CASE}: & Y \end{bmatrix}}$

(13) $\overset{\textit{Pron}}{\begin{bmatrix} \text{NUM}: & X \\ \text{CASE}: & Y \end{bmatrix}} \rightarrow \overset{\textit{she}}{\begin{bmatrix} \text{NUM}: & X(sg) \\ \text{CASE}: & Y(nom) \end{bmatrix}}$

(14) $\overset{\textit{Pron}}{\begin{bmatrix} \text{NUM}: & X \\ \text{CASE}: & Y \end{bmatrix}} \rightarrow \overset{\textit{her}}{\begin{bmatrix} \text{NUM}: & X(sg) \\ \text{CASE}: & Y(acc) \end{bmatrix}}$

Imposing case control

Percolating the value of the CASE feature from the lexical entries to the category *NP*:

$$(2.1)\quad \underset{\begin{bmatrix}\text{NUM}: & X\\ \text{CASE}: & Y\end{bmatrix}}{NP} \rightarrow \underset{\begin{bmatrix}\text{NUM}: & X\end{bmatrix}}{D}\quad \underset{\begin{bmatrix}\text{NUM}: & X\\ \text{CASE}: & Y\end{bmatrix}}{N}$$

$$(2.2)\quad \underset{\begin{bmatrix}\text{NUM}: & X\\ \text{CASE}: & Y\end{bmatrix}}{NP} \rightarrow \underset{\begin{bmatrix}\text{NUM}: & X\\ \text{CASE}: & Y\end{bmatrix}}{PropN}$$

$$(2.3)\quad \underset{\begin{bmatrix}\text{NUM}: & X\\ \text{CASE}: & Y\end{bmatrix}}{NP} \rightarrow \underset{\begin{bmatrix}\text{NUM}: & X\\ \text{CASE}: & Y\end{bmatrix}}{Pron}$$

Imposing case control

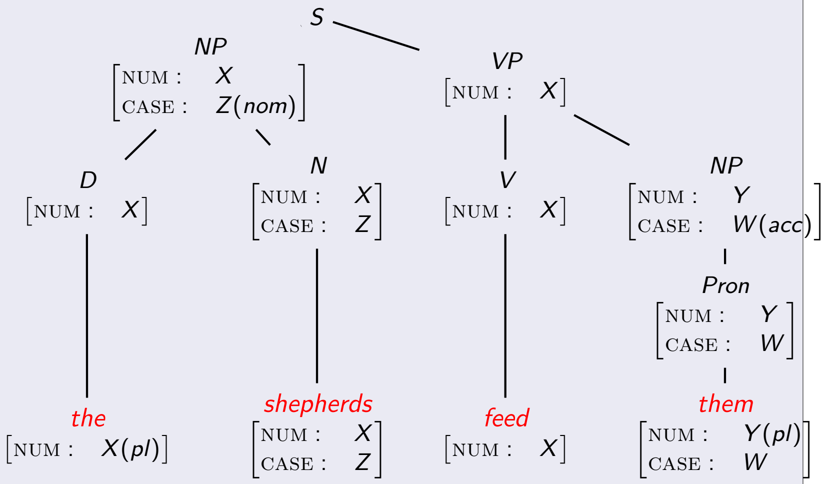
Imposing the constraint:

$$(1') \quad S \rightarrow \underset{\begin{bmatrix} \text{NUM}: & X \\ \text{CASE}: & nom \end{bmatrix}}{NP} \quad \underset{\begin{bmatrix} \text{NUM}: & X \end{bmatrix}}{VP}$$

$$(4') \quad \underset{\begin{bmatrix} \text{NUM}: & X \end{bmatrix}}{VP} \rightarrow \underset{\begin{bmatrix} \text{NUM}: & X \end{bmatrix}}{V} \quad \underset{\begin{bmatrix} \text{NUM}: & Y \\ \text{CASE}: & acc \end{bmatrix}}{NP}$$

Derivation tree with case control

Example: Derivation tree with case control



Example: (continued)

This tree represents a derivation which starts with the initial symbol, S, and ends with multi-AVM σ', where

$$\sigma' = \overset{\textit{the}}{\begin{bmatrix}\text{NUM}: & X(pl)\end{bmatrix}} \quad \overset{\textit{shepherds}}{\begin{bmatrix}\text{NUM}: & X \\ \text{CASE}: & Z\end{bmatrix}} \quad \overset{\textit{feed}}{\begin{bmatrix}\text{NUM}: & X\end{bmatrix}} \quad \overset{\textit{them}}{\begin{bmatrix}\text{NUM}: & Y(pl) \\ \text{CASE}: & W(acc)\end{bmatrix}}$$

This multi-AVM is unifiable with (but not identical to!) the sequence of lexical entries of the words in the sentence, which is:

$$\sigma = \overset{\textit{the}}{\begin{bmatrix}\text{NUM}: & [\,]\end{bmatrix}} \quad \overset{\textit{shepherds}}{\begin{bmatrix}\text{NUM}: & pl \\ \text{CASE}: & [\,]\end{bmatrix}} \quad \overset{\textit{feed}}{\begin{bmatrix}\text{NUM}: & pl\end{bmatrix}} \quad \overset{\textit{them}}{\begin{bmatrix}\text{NUM}: & pl \\ \text{CASE}: & acc\end{bmatrix}}$$

Imposing subcategorization constraints

We use the extended formalism for a naïve solution to the subcategorization problem; reminder:

intransitive verbs: sleep, walk, run, laugh, . . .

transitive verbs (with a nominal object): feed, love, eat, . . .

Imposing subcategorization constraints

First, the lexical entries of verbs are extended:

Example: Lexical entries for verbs

$$\underset{\begin{bmatrix}\text{NUM}: & X \\ \text{SUBCAT}: & \mathit{intrans}\end{bmatrix}}{V} \rightarrow \underset{\begin{bmatrix}\text{NUM}: & X(sg)\end{bmatrix}}{\textit{sleeps}} \mid \underset{\begin{bmatrix}\text{NUM}: & X(pl)\end{bmatrix}}{\textit{sleep}} \mid \ldots$$

$$\underset{\begin{bmatrix}\text{NUM}: & X \\ \text{SUBCAT}: & \mathit{trans}\end{bmatrix}}{V} \rightarrow \underset{\begin{bmatrix}\text{NUM}: & X(sg)\end{bmatrix}}{\textit{feeds}} \mid \underset{\begin{bmatrix}\text{NUM}: & X(pl)\end{bmatrix}}{\textit{feed}} \mid \ldots$$

Imposing subcategorization constraints

Second, the rules that involve verbs and verb phrases are extended:

Example: Modified rules for verb phrases

$$(4.1)\quad \underset{\begin{bmatrix}\text{NUM}: & X\end{bmatrix}}{VP} \rightarrow \underset{\begin{bmatrix}\text{NUM}: & X \\ \text{SUBCAT}: & \textit{intrans}\end{bmatrix}}{V}$$

$$(4.2)\quad \underset{\begin{bmatrix}\text{NUM}: & X\end{bmatrix}}{VP} \rightarrow \underset{\begin{bmatrix}\text{NUM}: & X \\ \text{SUBCAT}: & \textit{trans}\end{bmatrix}}{V} \quad \underset{\begin{bmatrix}\text{NUM}: & Y\end{bmatrix}}{NP}$$

Imposing subcategorization constraints

Example: Derivation of a shepherd feeds two sheep

$$S \overset{(1)}{\Rightarrow} \underset{[\text{NUM}: \; X]}{NP} \; \underset{[\text{NUM}: \; X]}{VP}$$

$$\overset{(2)}{\Rightarrow} \underset{[\text{NUM}: \; X]}{D} \; \underset{[\text{NUM}: \; X]}{N} \; \underset{[\text{NUM}: \; X]}{VP}$$

$$\overset{(4.2)}{\Rightarrow} \underset{[\text{NUM}: \; X]}{D} \; \underset{[\text{NUM}: \; X]}{N} \; \underset{\begin{bmatrix}\text{NUM}: & X \\ \text{SUBCAT}: & trans\end{bmatrix}}{V} \; \underset{[\text{NUM}: \; Y]}{NP}$$

Example: Derivation of a shepherd feeds two sheep

$$\stackrel{(4.2)}{\Rightarrow} \quad \underset{\begin{bmatrix}\text{NUM}: & X\end{bmatrix}}{D} \quad \underset{\begin{bmatrix}\text{NUM}: & X\end{bmatrix}}{N} \quad \underset{\begin{bmatrix}\text{NUM}: & X \\ \text{SUBCAT}: & trans\end{bmatrix}}{V} \quad \underset{\begin{bmatrix}\text{NUM}: & Y\end{bmatrix}}{NP}$$

$$\stackrel{(1)}{\Rightarrow} \quad \underset{\begin{bmatrix}\text{NUM}: & X\end{bmatrix}}{D} \quad \underset{\begin{bmatrix}\text{NUM}: & X\end{bmatrix}}{N} \quad \underset{\begin{bmatrix}\text{NUM}: & X \\ \text{SUBCAT}: & trans\end{bmatrix}}{V} \quad \underset{\begin{bmatrix}\text{NUM}: Y\end{bmatrix}}{D} \quad \underset{\begin{bmatrix}\text{NUM}: Y\end{bmatrix}}{N}$$

$$\stackrel{*}{\Rightarrow} \quad \underset{\begin{bmatrix}\text{NUM}: & sg\end{bmatrix}}{a} \quad \underset{\begin{bmatrix}\text{NUM}: & sg\end{bmatrix}}{shepherd} \quad \underset{\begin{bmatrix}\text{NUM}: & sg \\ \text{SUBCAT}: & trans\end{bmatrix}}{feeds} \quad \underset{\begin{bmatrix}\text{NUM}: pl\end{bmatrix}}{two} \quad \underset{\begin{bmatrix}\text{NUM}: pl\end{bmatrix}}{sheep}$$

G_2, a complete E_0-grammar

Example: G_2, a complete E_0-grammar

$$S \rightarrow \underset{\begin{bmatrix}\text{NUM}: & X \\ \text{CASE}: & nom\end{bmatrix}}{NP} \quad \underset{\begin{bmatrix}\text{NUM}: & X\end{bmatrix}}{VP}$$

$$\underset{\begin{bmatrix}\text{NUM}: & X \\ \text{CASE}: & Y\end{bmatrix}}{NP} \rightarrow \underset{\begin{bmatrix}\text{NUM}: & X\end{bmatrix}}{D} \quad \underset{\begin{bmatrix}\text{NUM}: & X \\ \text{CASE}: & Y\end{bmatrix}}{N}$$

$$\underset{\begin{bmatrix}\text{NUM}: & X \\ \text{CASE}: & Y\end{bmatrix}}{NP} \rightarrow \underset{\begin{bmatrix}\text{NUM}: & X \\ \text{CASE}: & Y\end{bmatrix}}{Pron} \mid \underset{\begin{bmatrix}\text{NUM}: & X \\ \text{CASE}: & Y\end{bmatrix}}{PropN}$$

Example: (continued)

$$\underset{\left[\text{NUM}: \; X\right]}{VP} \rightarrow \underset{\begin{bmatrix}\text{NUM}: & X \\ \text{SUBCAT}: & intrans\end{bmatrix}}{V}$$

$$\underset{\left[num: \; X\right]}{VP} \rightarrow \underset{\begin{bmatrix}\text{NUM}: & X \\ \text{SUBCAT}: & trans\end{bmatrix}}{V} \quad \underset{\begin{bmatrix}\text{NUM}: & Y \\ \text{CASE}: & acc\end{bmatrix}}{NP}$$

$$\underset{\begin{bmatrix}\text{NUM}: & X \\ \text{SUBCAT}: & intrans\end{bmatrix}}{V} \rightarrow \underset{\left[\text{NUM}: \; X(sg)\right]}{\color{red}{sleeps}} \;\mid\; \underset{\left[\text{NUM}: \; X(pl)\right]}{\color{red}{sleep}} \;\mid\; \ldots$$

$$\underset{\begin{bmatrix}\text{NUM}: & X \\ \text{SUBCAT}: & trans\end{bmatrix}}{V} \rightarrow \underset{\left[\text{NUM}: \; X(sg)\right]}{\color{red}{feeds}} \;\mid\; \underset{\left[\text{NUM}: \; X(pl)\right]}{\color{red}{feed}} \;\mid\; \ldots$$

Example: (continued)

$$\underset{\begin{bmatrix}\text{NUM}: & X\\ \text{CASE}: & Y\end{bmatrix}}{N} \rightarrow \underset{\begin{bmatrix}\text{NUM}: & X(sg)\\ \text{CASE}: & Y\end{bmatrix}}{lamb} \mid \underset{\begin{bmatrix}\text{NUM}: & X(pl)\\ \text{CASE}: & Y\end{bmatrix}}{lambs} \mid \ldots$$

$$\underset{\begin{bmatrix}\text{NUM}: & X\\ \text{CASE}: & Y\end{bmatrix}}{Pron} \rightarrow \underset{\begin{bmatrix}\text{NUM}: & X(sg)\\ \text{CASE}: & Y(nom)\end{bmatrix}}{she} \mid \underset{\begin{bmatrix}\text{NUM}: & X(sg)\\ \text{CASE}: & Y(acc)\end{bmatrix}}{her} \mid \ldots$$

$$\underset{\begin{bmatrix}\text{NUM}: & X\\ \text{CASE}: & Y\end{bmatrix}}{PropN} \rightarrow \underset{\begin{bmatrix}\text{NUM}: & X(sg)\\ \text{CASE}: & Y\end{bmatrix}}{Rachel} \mid \underset{\begin{bmatrix}\text{NUM}: & X(sg)\\ \text{CASE}: & Y\end{bmatrix}}{Jacob} \mid \ldots$$

$$\underset{\begin{bmatrix}\text{NUM}: & X\end{bmatrix}}{D} \rightarrow \underset{\begin{bmatrix}\text{NUM}: & X(sg)\end{bmatrix}}{a} \mid \underset{\begin{bmatrix}\text{NUM}: & X(pl)\end{bmatrix}}{two} \mid \ldots$$