

# Foundations of Computational Linguistics

*man-machine communication in natural language*

ROLAND HAUSSER  
Computational Linguistics  
Universität Erlangen Nürnberg  
Germany



# Table of Contents

---

**Part I. Theory of Language**

---

<b>1. Computational language analysis</b> .....	15
1.1 Man-machine communication .....	15
1.2 Language science and its components .....	17
1.3 Methods and applications of computational linguistics .....	18
1.4 Electronic medium in recognition and synthesis .....	19
1.5 Second Gutenberg Revolution .....	22
<b>2. Technology and grammar</b> .....	26
2.1 Indexing and retrieval in textual databases .....	26
2.2 Using grammatical knowledge .....	28
2.3 Smart versus solid solutions .....	31
2.4 Beginnings of machine translation .....	33
2.5 Machine translation today .....	38
<b>3. Cognitive foundation of semantics</b> .....	42
3.1 Prototype of communication .....	42
3.2 From perception to recognition .....	44
3.3 Iconicity of formal concepts .....	46
3.4 Contextual I-propositions .....	50

---

3.5 Recognition and action .....	52
<b>4. Language communication .....</b>	<b>55</b>
4.1 Adding language .....	55
4.2 Modeling reference .....	57
4.3 Using literal meaning .....	59
4.4 Frege's principle .....	60
4.5 Surface compositionality .....	63
<b>5. Using language signs on suitable contexts .....</b>	<b>69</b>
5.1 Bühler's organon model .....	69
5.2 Pragmatics of tools and pragmatics of words .....	74
5.3 Finding the correct subcontext .....	75
5.4 Language production and interpretation .....	78
5.5 Thought as the motor of spontaneous production .....	82
<b>6. Structure and functioning of signs .....</b>	<b>84</b>
6.1 Reference mechanisms of different sign types .....	84
6.2 Internal structure of symbols and indices .....	88
6.3 Indices for repeating reference .....	90
6.4 Exceptional properties of icon and name .....	96
6.5 Pictures, pictograms, and letters .....	98

---

**Part II. Theory of Grammar**

---

<b>7. Generative grammar</b> .....	101
7.1 Language as a subset of the free monoid .....	101
7.2 Methodological reasons for generative grammar .....	104
7.3 Adequacy of generative grammars .....	106
7.4 Formalism of C-grammar .....	107
7.5 C-grammar for natural language .....	112
<b>8. Language hierarchies and complexity</b> .....	115
8.1 Formalism of PS-grammar .....	115
8.2 Language classes and computational complexity .....	117
8.3 Generative capacity and formal language classes .....	120
8.4 PS-Grammar for natural language .....	127
8.5 Constituent structure paradox .....	133
<b>9. Basic notions of parsing</b> .....	137
9.1 Declarative and procedural aspects of parsing .....	137
9.2 Fitting grammar onto language .....	138
9.3 Type transparency between grammar and parser .....	141
9.4 Input-output equivalence with the speaker-hearer .....	145

---

9.5 Desiderata of grammar for achieving convergence .....	147
<b>10. Left-associative grammar (LAG) .....</b>	<b>150</b>
10.1 Rule types and derivation order .....	150
10.2 Formalism of LA-grammar .....	154
10.3 Time-linear analysis .....	158
10.4 Absolute type transparency of LA-grammar .....	161
10.5 LA-grammar for natural language .....	164
<b>11. Hierarchy of LA-grammar .....</b>	<b>170</b>
11.1 Generative capacity of unrestricted LAG .....	170
11.2 LA-hierarchy of A-, B-, and C-LAGs .....	174
11.3 Ambiguity in LA-grammar .....	177
11.4 Complexity of grammars and automata .....	182
11.5 Subhierarchy of C1-, C2-, and C3-LAGs .....	183
<b>12. LA- and PS-hierarchies in comparison .....</b>	<b>191</b>
12.1 Language classes of LA- and PS-grammar .....	191
12.2 Subset relations in the two hierarchies .....	192
12.3 Non-equivalence of the LA- and PS-hierarchy .....	193
12.4 Comparing the lower LA- and PS-classes .....	196
12.5 Linear complexity of natural language .....	199

---

**Part III. Morphology and Syntax**

---

<b>13. Words and morphemes</b> .....	205
13.1 Words and word forms .....	205
13.2 Segmentation and concatenation .....	211
13.3 Morphemes and allomorphs .....	215
13.4 Categorization and lemmatization .....	217
13.5 Methods of automatic word form recognition .....	221
<b>14. Word form recognition in LA-Morph</b> .....	225
14.1 Allo-rules .....	225
14.2 Phenomena of allomorphy .....	232
14.3 Left-associative segmentation into allomorphs .....	238
14.4 Combi-rules .....	241
14.5 Concatenation patterns .....	245
<b>15. Corpus analysis</b> .....	251
15.1 Implementation and application of grammar systems .....	251
15.2 Subtheoretical variants .....	254
15.3 Building corpora .....	258
15.4 Distribution of word forms .....	260

---

15.5 Statistical tagging .....	264
<b>16. Basic concepts of syntax</b> .....	268
16.1 Delimitation of morphology and syntax .....	268
16.2 Valency .....	270
16.3 Agreement .....	274
16.4 Free word order in German ( <i>LA-D1</i> ) .....	276
16.5 Fixed word order in English ( <i>LA-E1</i> ) .....	283
<b>17. LA-syntax for English</b> .....	286
17.1 Complex fillers in pre- and postverbal position .....	286
17.2 English field of referents .....	294
17.3 Complex verb forms .....	296
17.4 Finite state backbone of LA-syntax ( <i>LA-E2</i> ) .....	299
17.5 Yes/no-interrogatives ( <i>LA-E3</i> ) and grammatical perplexity .....	304
<b>18. LA-syntax for German</b> .....	309
18.1 Standard procedure of syntactic analysis .....	309
18.2 German field of referents ( <i>LA-D2</i> ) .....	313
18.3 Verbal positions in English and German .....	320
18.4 Complex verbs and elementary adverbs ( <i>LA-D3</i> ) .....	324
18.5 Interrogatives and subordinate clauses ( <i>LA-D4</i> ) .....	333



---

**Part IV. Semantics and Pragmatics**

---

<b>19. Three system types of semantics</b> .....	343
19.1 Basic structure of semantic interpretation .....	343
19.2 Logical, programming, and natural languages .....	344
19.3 Functioning of logical semantics .....	348
19.4 Metalanguage-based versus procedural semantics .....	353
19.5 Tarski's problem for natural language semantics .....	355
<b>20. Truth, meaning, and ontology</b> .....	359
20.1 Analysis of meaning in logical semantics .....	359
20.2 Intension and extension .....	362
20.3 Propositional attitudes .....	364
20.4 Four basic ontologies .....	367
20.5 Sorites paradox and the treatment of vagueness .....	369
<b>21. Absolute and contingent propositions</b> .....	373
21.1 Absolute and contingent truth .....	373
21.2 Epimenides in a [+sense,+constructive] system .....	376
21.3 Frege's principle as homomorphism .....	379
21.4 Time-linear syntax with homomorphic semantics .....	384

---

21.5 Complexity of natural language semantics .....	387
<b>22. Database semantics .....</b>	<b>389</b>
22.1 Database metaphor of natural communication .....	389
22.2 Descriptive aporia and embarrassment of riches .....	392
22.3 Propositions as sets of coindexed proplets .....	395
22.4 Proplets in a classic database .....	396
22.5 Example of a word bank .....	399
<b>23. SLIM machine in the hearer mode .....</b>	<b>405</b>
23.1 External connections and motor algorithms .....	405
23.2 Ten SLIM states of cognition .....	407
23.3 Semantic interpretation of LA-SU syntax .....	413
23.4 Example of syntactic-semantic derivation ( <i>LA-E4</i> ) .....	419
23.5 From SLIM semantics to SLIM pragmatics .....	432
<b>24. SLIM machine in the speaker mode .....</b>	<b>436</b>
24.1 Subcontext as concatenated propositions .....	436
24.2 Tracking principles of LA-navigation .....	442
24.3 Interpreting autonomous LA-navigation with language .....	451
24.4 Subordinating navigation .....	456
24.5 LA-search and LA-inference .....	464

---

# Introduction

## 1. Requirements for modeling natural communication

- a theory of language which explains the natural transfer of information in a way that is functionally coherent, mathematically explicit, and computationally efficient,
- a description of language data which is empirically complete for all components of this theory of language, i.e. the lexicon, the morphology, the syntax, and the semantics, as well as the pragmatics and the representation of the internal context,
- a degree of precision in the description of these components which is sufficient for computation,

## 2. Consequences of using parsers

- *Competition*

Competing theories of grammar are measured with respect to the new standard of how well they are suited for efficient parsing and how well they fit into a theory of language designed to model the mechanism of natural communication.

- *Funding*

Computationally efficient and empirically adequate parsers for different languages are needed for an unlimited range of practical applications, which has a major impact on the inflow of funds for research, development, and teaching in this particular area of the humanities.

- *Verification*

Programming grammars as parsers allows testing their empirical adequacy automatically on arbitrarily large amounts of real data in the areas of word form recognition/synthesis, syntactic analysis/generation and semantic-pragmatic interpretation in both, the speaker and the hearer mode.

### 3. Principles of the SLIM theory of language

1. *Surface compositional* (methodological principle)

Syntactic-semantic composition assembles only concrete word forms, excluding the use of zero-elements, identity mappings, or transformations.

2. *Linear* (empirical principle)

Interpretation and production of utterances is based on a strictly time-linear derivation order.

3. *Internal* (ontological principle)

Interpretation and production of utterances is analyzed as cognitive procedures located inside the speaker-hearer.

4. *Matching* (functional principle)

Referring with language to past, current, or future objects and events is modeled in terms of pattern matching between language meaning and context.

**Part I.**  
**Theory of Language**

# 1. Computational language analysis

## 1.1 Man-machine communication

### 1.1.1 Restricted vs. nonrestricted communication

### 1.1.2 Example of restricted communication: a record-based database

	last name	first name	place	...
A1	Schmidt	Peter	Bamberg	...
A2	Meyer	Susanne	Nürnberg	...
A3	Sanders	Reinhard	Schwabach	...
	:	:	:	

### 1.1.3 Database query

Query:

```
select A#
where city = 'Schwabach'
```

Result:

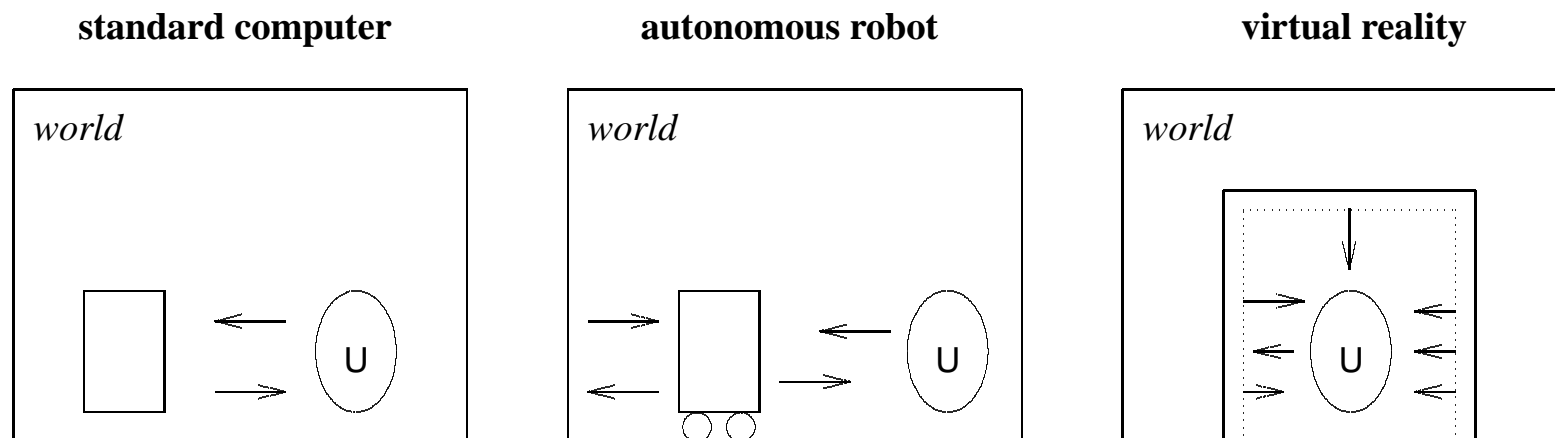
```
A3 Sanders Reinhard
```

### 1.1.4 Classic AI vs Nouvelle AI

Classic AI analyzes intelligent behavior as manipulating symbols. For example, a chess playing program operates in isolation from the rest of the world, using a fixed set of predefined pieces and a predefined board. The search space for a dynamic strategy of winning is astronomical. Yet a standard computer is sufficient because the world of chess is closed.

Nouvelle AI aims at the development of autonomous agents. In order to interact with their real world environment, they must continually keep track of changes by means of sensors. For this, nouvelle AI uses robots. The strategy of task level decomposition defines inferencing to operate directly on the local perception data.

### 1.1.5 Three types of man-machine communication





## 1.2 Language science and its components

### 1.2.1 Variants of language science

- *Traditional Grammar*
- *Theoretical Linguistics*
- *Computational Linguistics*

### 1.2.2 The components of grammar

- *Phonology*: Science of language sounds
- *Morphology*: Science of word form structure
- *Lexicon*: Listing analyzed words
- *Syntax*: Science of composing word forms
- *Semantics*: Science of literal meanings
- *Pragmatics*: Science of using language expressions

## 1.3 Methods and applications of computational linguistics

### 1.3.1 Methodology of parsing

1. *Decomposition* of a complex sign into its elementary components,
2. *Classification* of the components via lexical lookup, and
3. *Composition* of the classified components via syntactic rules in order to arrive at an overall grammatical analysis of the complex sign.

### 1.3.2 Practical tasks of computational linguistics

- Indexing and retrieval in textual databases
- Machine translation
- Automatic text production
- Automatic text checking
- Automatic content analysis
- Automatic tutoring
- Automatic dialog and information systems

## 1.4 Electronic medium in recognition and synthesis

### 1.4.1 Media of language

Nonelectronic media:

- *sounds* of spoken language
- *letters* of handwritten or printed language
- *gestures* of sign language

Electronic medium:

- Realization-dependent representations:
  - tape recording of spoken language
  - bitmap of written language
  - video recording of signed language
- Realization-independent representations:
  - digitally coded electronic sign sequences, e.g. ASCII

## 1.4.2 Transfer between realization-dependent and -independent representations

*recognition*:  $i \Rightarrow d$  transfer

Realization-dependent representations must be mapped into realization-independent ones.

*synthesis*:  $d \Rightarrow i$  transfer

Realization-independent representations must be mapped into realization-dependent ones.

## 1.4.3 Methods of $d \Rightarrow i$ transfer

Nonautomatic: typing spoken or written language into the computer

Automatic: Acoustic or optical pattern recognition

## 1.4.4 Desiderata of automatic speech recognition

The quality of automatic speech recognition should be at least equal to that of an average human hearer.

- *Speaker independence*

The system should understand speech of an open range of speakers with varying dialects, pitch, etc. – without the need for an initial learning phase to adapt the system to one particular user.

- *Continuous speech*  
The system should handle continuous speech at different speeds – without the need for unnatural pauses between individual word forms.
- *Domain independence*  
The system should understand spoken language independently of the subject matter – without the need of telling the system in advance which vocabulary is to be expected and which is not.
- *Realistic vocabulary*  
The system should recognize at least as many word forms as an average human.
- *Robustness*  
The system should recover gracefully from interruptions, contractions, and slurring of spoken language, and be able to infer the word forms intended.

### **1.4.5 The crucial question for designing truly adequate speech recognition**

*How should the domain and language knowledge best be organized?*

The answer is obvious:

*The domain and language knowledge should be organized within a functional theory of language which is mathematically and computationally efficient.*

## 1.5 Second Gutenberg Revolution

### 1.5.1 The First Gutenberg Revolution

Based on the technological innovation of printing with movable letters, it made a wealth of knowledge available to a broader public.

### 1.5.2 The Second Gutenberg Revolution

Based on the automatic processing of natural language in the electronic medium, it aims at facilitating access to specific pieces of information.

### 1.5.3 SGML: *standard generalized markup language*.

A family of ISO standards for labeling electronic versions of text, enabling both sender and receiver of the text to identify its structure (e.g. title, author, header, paragraph, etc.)

Dictionary of Computing, p. 416 (ed. Illingworth et al. 1990)

## 1.5.4 Newspaper text with SGML control symbols (excerpt)

```
<HTML>
<HEAD>
<TITLE>9/4/95 COVER: Siberia, the Tortured Land</TITLE>
</HEAD>
<BODY>
<!-- #include "header.html" -->
<P>TIME Magazine</P>
<P>September 4, 1995 Volume 146, No. 10</P>
<HR>
Return to <A href="../../../time/magazine/domestic/toc/
950904.toc.html">Contents page</A>
<HR>
<BR>
<!-- end include -->
<H3>COVER STORY</H3>
<H2>THE TORTURED LAND</H2>
<H3>An epic landscape steeped in tragedy, Siberia suffered
grievously under communism. Now the world's capitalists covet
its vast riches </H3>
<P><EM>BY <A href="../../../time/bios/eugenelinden.html">
EUGENE LINDEN</A>/YAKUTSK</EM>
<P>Siberia has come to mean a land of exile, and the place
easily fulfills its reputation as a metaphor for death and
```

## 1.5.5 Different types of text

- article
- book
- theater play
- movie script
- dictionary

## 1.5.6 TEI

Text encoding initiative: defines a DTD (*document type definition*) for the markup of different types of text in SGML.

## 1.5.7 Different goals of markup

- Function-oriented: SGML and TEI
- Print-oriented: T<sub>E</sub>X and L<sup>A</sup>T<sub>E</sub>X
- User-oriented: Winword, WordPerfect, etc.



## 1.5.8 Alphabetical list of word forms (excerpt)

10	in	STORY
146	in	suffered
1995	in	sun
20	its	than
4	its	that
a	LAND	The
a	land	the
a	landscape	the
a	like	the
a	LINDEN	the
above	Magazine	the
across	markers	the
and	mean	the
and	metaphor	the
and	midnight	the
and	midsummer	the
Arctic	million	through
as	mist	Throughout
as	more	to
barracks	mossy	to
bits	muting	TORTURED

## 2. Technology and grammar

### 2.1 Indexing and retrieval in textual databases

#### 2.1.1 Indexing

The indexing of a textual database is based on a table which specifies for each letter all the positions (addresses) where it occurs in the storage medium of the database.

#### 2.1.2 Advantages of an electronic index

- Power of search
- Flexibility
  - General specification of patterns
  - Combination of patterns
- Automatic creation of the index structure
- Ease, speed, and reliability
  - Query
  - Retrieval

### 2.1.3 Definition of recall and precision

*Recall* measures the percentage of relevant texts retrieved as compared to the total of relevant texts contained in the database.

For example: a database of several million pieces of text happens to contain 100 texts which are relevant to a given question. If the query returns 75 texts, 50 of which are relevant to the user and 25 are irrelevant, then the recall is  $50 : 100 = 50\%$ .

*Precision* measures the percentage of relevant texts contained in the result of a query.

For example: a query has resulted in 75 texts of which 50 turn out to be relevant to the user. Then the precision is  $50 : 75 = 66.6\%$ .

## 2.2 Using grammatical knowledge

### 2.2.1 Linguistic methods of optimization

#### A. Preprocessing the query

- Automatic query expansion

(i) The search words in the query are automatically ‘exploded’ into their full inflectional paradigm and the inflectional forms are added to the query.

(ii) Via a thesaurus the search words are related to all synonyms, hypernyms, and hyponyms. These are included in the query – possibly with all their inflectional variants.

(iii) The syntactic structure of the query, e.g. A sold x to B, is transformed automatically into equivalent versions, e.g. B was sold x by A, x was sold to B by A, etc., to be used in the query.

- Interactive query improvement

Prior to the search, the result of a query expansion is presented to the user to allow elimination of useless aspects of the automatic expansion and allow for an improved formulation of the query.

## B. Improving the indexing

- Letter-based indexing

This is the basic technology of search, allowing to retrieve the positions of each letter and each letter sequence in the database.

- Morphologically-based indexing

A morphological analyzer is applied during the reading-in of texts, relating each word form to its base form. This information is coded into an index which for any given word (base form) allows to find all corresponding (inflected) forms.

- Syntactically-based indexing

A syntactic parser is applied during the reading-in of texts, eliminating morphological ambiguities and categorizing phrases. The grammatical information is coded into an index which allows to find all occurrences of a given syntactic construction.

- Concept-based indexing

The texts are analyzed semantically and pragmatically, eliminating syntactic and semantic ambiguities as well as inferring special uses characteristic of the domain. This information is coded into an index which allows to find all occurrences of a given concept.

## C. Postquery processing

- The low precision resulting from a nonspecific formulation of the query may be countered by an automatic processing of the data retrieved. Because the raw data retrieved are small as compared to the database as a whole they may be parsed after the query and checked for their content. Then only those texts are given out which are relevant according to this post query analysis.

## 2.3 Smart versus solid solutions

### 2.3.1 Smart solutions

avoid difficult, costly, or theoretically unsolved aspects of the task at hand, such as

- Weizenbaum's Eliza program, which appears to understand natural language, but doesn't.
- Statistically-based tagging, which can guess the part of speech.
- Direct and transfer approaches in machine translation, which avoid understanding the source text.

### 2.3.2 Solid solutions

aim at a complete theoretical and practical understanding of the phenomena involved. Applications are based on ready-made off-the-shelf components such as

- Automatic word form analysis based on an online lexicon of the language and a rule-based morphological parser handling inflection, derivation and composition
- Syntactic parsing of free text based on the automatic word form analysis of the language
- Semantic interpretation of the syntactic analysis deriving the literal meaning
- Pragmatic interpretation relative to a context of use deriving the speaker meaning.

### 2.3.3 Comparison

- As long as the off-the-shelf components are not available, a smart solution seems initially cheaper and quicker. But smart solutions are costly to maintain and their accuracy cannot be substantially improved.
- The components of grammar are a long term investment that can be used again and again in a wide variety of different solid solutions. Improvements in the components of grammar lead directly to better applications.

### 2.3.4 Choice between smart or solid solution depends on application

- A smart solution providing a 70% recall in a large database is more than the user could hope to find by hand. Also, the user is not aware of what is missing in the query result.
- In contrast, the deficiencies of a smart solution providing 70% accuracy in machine translation are painfully obvious to the user. Furthermore, there are human translators available which are able to do a much better job.



## 2.4 Beginnings of machine translation

### 2.4.1 Language pairs

*French*  $\rightarrow$  *English* and *French*  $\leftarrow$  *English* are two different language pairs.

### 2.4.2 Formula to compute the number of language pairs

$n \cdot (n - 1)$ , where  $n =$  number of different languages

For example, an EU with 11 different languages has to deal with a total of  $11 \cdot 10 = 110$  language pairs.

### 2.4.3 Translating a French document in the EU

French  $\rightarrow$  English

French  $\rightarrow$  German

French  $\rightarrow$  Italian

French  $\rightarrow$  Dutch

French  $\rightarrow$  Swedish

French  $\rightarrow$  Spanish

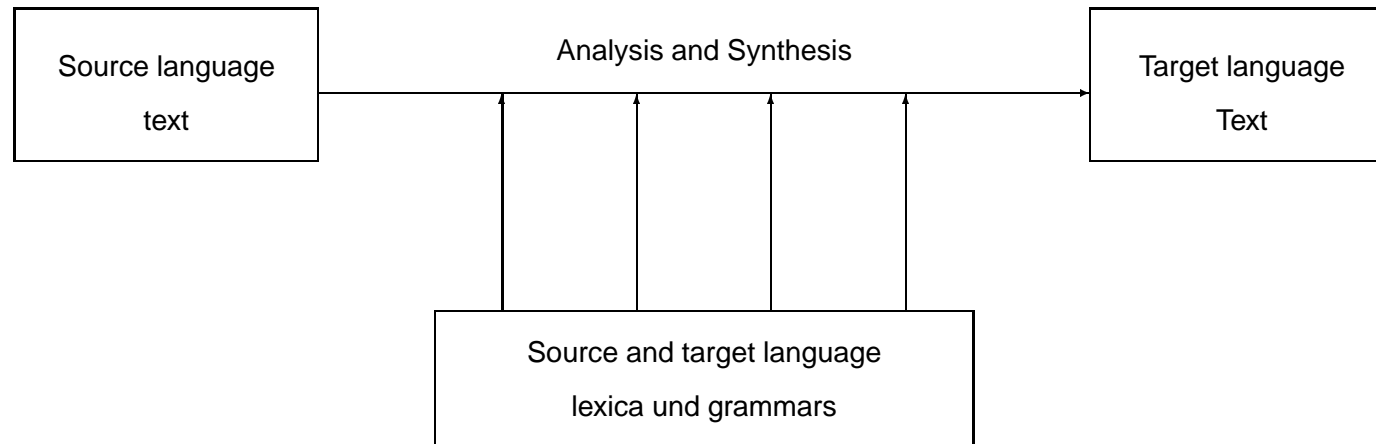
French  $\rightarrow$  Portugese

French  $\rightarrow$  Greek

French  $\rightarrow$  Danish

French  $\rightarrow$  Finnish

## 2.4.4 Schema of direct translation



## 2.4.5 What is FAHQT?

FULLY AUTOMATIC HIGH QUALITY TRANSLATION

## 2.4.6 Examples of automatic mis-translations

Out of sight, out of mind.  $\Rightarrow$  *Invisible idiot.*

The spirit is willing, but the flesh is weak.  $\Rightarrow$  *The whiskey is alright, but the meat is rotten.*

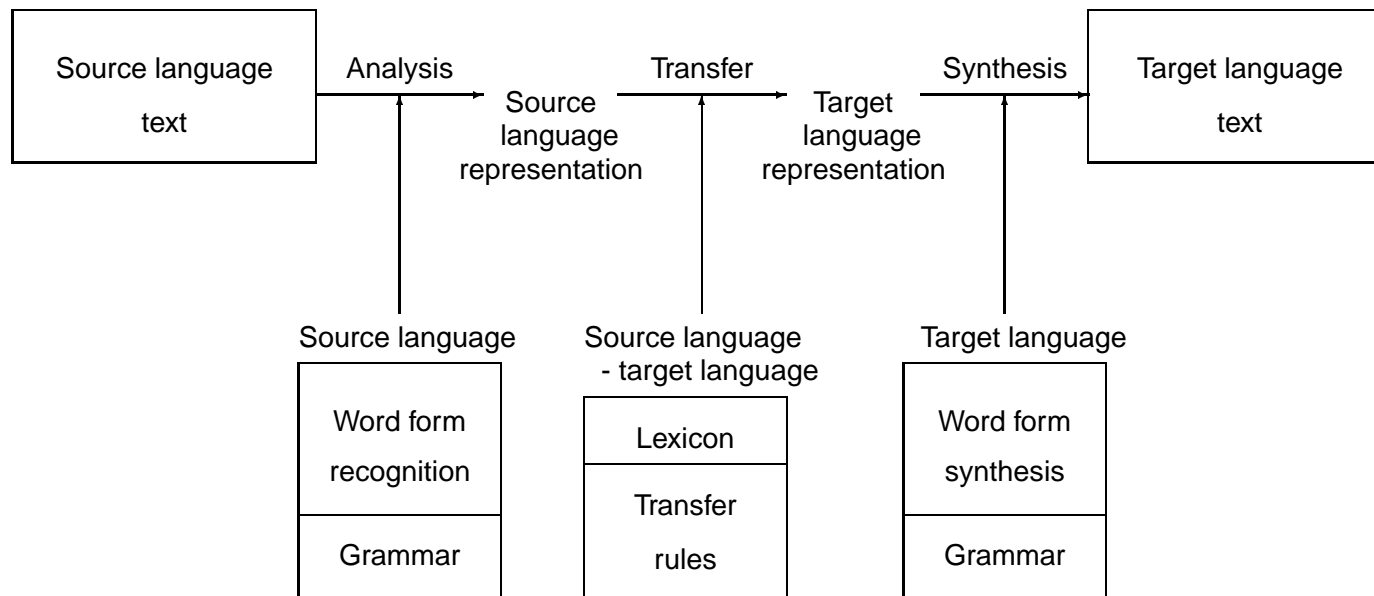
La Cour de Justice considère la création d'un sixième poste d'avocat général.  $\Rightarrow$  *The Court of Justice is considering the creation of a sixth avocado station.*

## 2.4.7 The transfer approach

aims for a modular separation of the

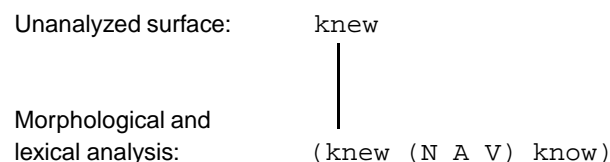
- source language analysis and target language synthesis, of
- linguistic data and processing procedures, and of the
- lexica for source language analysis, target language transfer, and target language synthesis.

## 2.4.8 Schema of the transfer approach



## 2.4.9 Three phrases of a word form transfer *English-German*

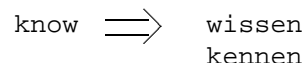
### 1. Source language analysis:



The source language analysis produces the syntactic category (N A V) of the inflectional form (categorization) and the base form know (lemmatization).

### 2. Source-target language transfer:

Using the base form resulting from the source language analysis, a source-target language dictionary provides the corresponding base forms in the target language.



### 3. Target language synthesis

Using the source language category (resulting from analysis) and the target language base forms (resulting from transfer), the desired target language word forms are generated based on target language morphology.

wußte	kannte
wußtest	kanntest
wußten	kannten
wußtet	kanntet

### 2.4.10 Shortcomings of the direct and the transfer approach

- Each language pair requires a special source-target component.
- Analysis and synthesis are limited to single sentences.
- Semantic and pragmatic analysis are avoided, attempting automatic translation without understanding the source language.

## 2.5 Machine translation today

### 2.5.1 Illustrating the importance of language understanding

- **Syntactic ambiguity in the source language**

1. Julia flew and crashed the air plane.

Julia (flew and crashed the air plane)

(Julia flew) and (crashed the air plane)

2. Susan observed the yacht with a telescope.

Susan observed the man with a beard.

3. The mixture gives off dangerous cyanide and chlorine fumes.

(dangerous cyanide) and (chlorine fumes)

dangerous (cyanide and chlorine) fumes

- **Lexical differences between source and target**

1. The men killed the women. Three days later they were caught.

The men killed the women. Three days later they were buried.

2. know: wissen savoir

kennen connaître

3. The watch included two new recruits that night.

- **Syntactic differences between source and target**

- German:

- Auf dem Hof sahen wir einen kleinen Jungen, der einem Ferkel nachlief.  
Dem Jungen folgte ein großer Hund.

- English:

- In the yard we saw a small boy running after a piglet.  
A large dog followed the boy.  
The boy was followed by a large dog.

- **Collocation and idiom**

strong current | high voltage (but: \*high current | \*strong voltage)

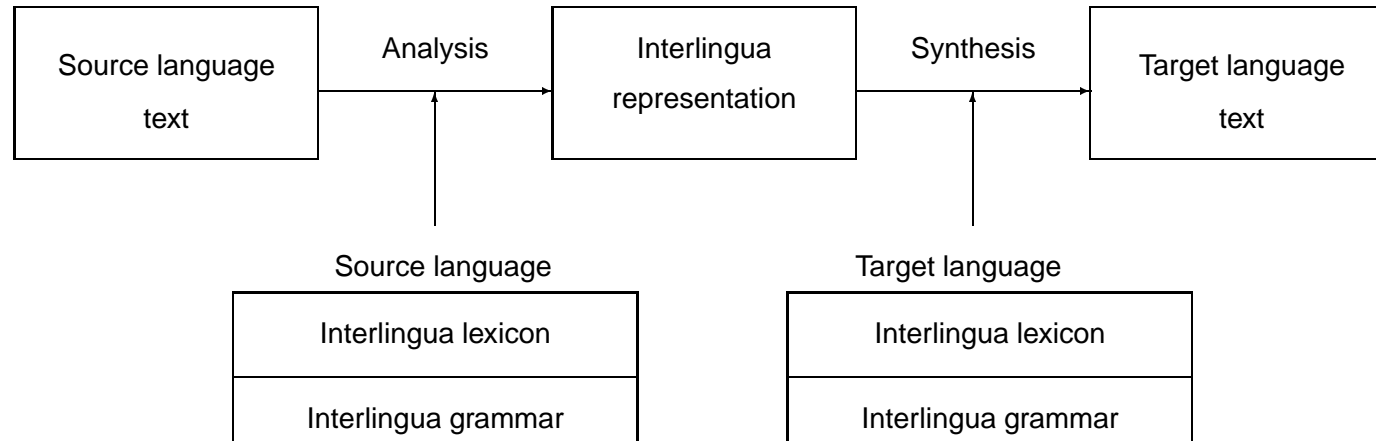
bite the dust | ins Gras beißen (but: \*bite the grass | \*in den Staub beißen)

## 2.5.2 Partial solutions for practical machine translation

1. *Machine aided translation* (MAT) supports human translators with comfortable tools such as on-line dictionaries, text processing, morphological analysis, etc.
2. *Rough translation* – as provided by an automatic transfer system – arguably reduces the translators' work to correcting the automatic output.
3. *Restricted language* provides a fully automatic translation, but only for texts which fulfill canonical restrictions on lexical items and syntactic structures.



### 2.5.3 Schema of the interlingua approach



### 2.5.4 Candidates proposed as interlingua

- an artificial logical language,
- a semi-natural language like Esperanto which is man-made, but functions like a natural language,
- a set of semantic primitives common to both, the source and the target language, serving as a kind of universal vocabulary.

## 3. Cognitive foundation of semantics

### 3.1 Prototype of communication

#### 3.1.1 Variants of language communication

- two speakers are located face to face and talk about concrete objects in their immediate environment
- two speakers talk on the telephone about events they experienced together in the past
- a merchant writes to a company to order merchandise in a certain number, size, color, etc., and the company responds by filling the order
- a newspaper informs about a planned extension of public transportation
- a translator reconstructs an English short story in German
- a teacher of physics explains the law of gravitation
- a registrar issues a marriage license
- a judge announces a sentence
- a conductor says: Terminal station, everybody please get off.
- a sign reads: Do not step on the grass!
- a professor of literature interprets an expressionistic poem
- an author writes a science fiction story
- an actor speaks a role

### 3.1.2 Prototype of communication

The basic prototype of natural communication is the direct face to face discourse of two partners talking about concrete objects in their immediate environment.

Possible alternatives: complete texts or signs of nature, such as smoke indicating fire.

### 3.1.3 Three components of the communication prototype

- Specification of the external *task environment*
- Structure of the *cognitive agent* including the internal *problem space*
- Specification of the *language*

### 3.1.4 Objects in the world of CURIOUS

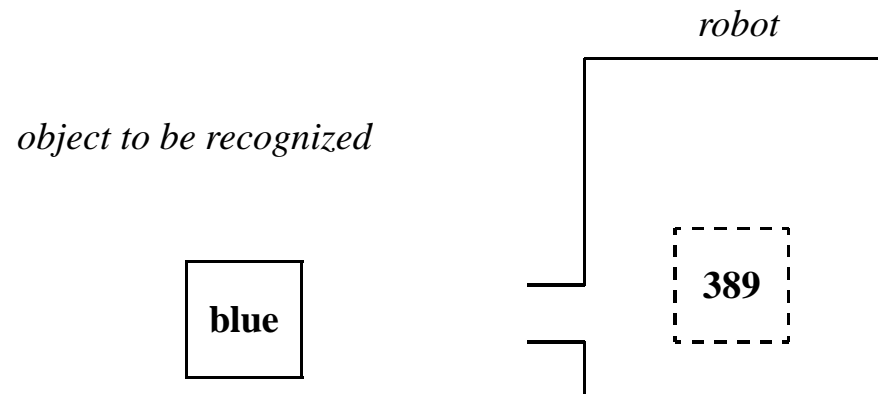
- triangles (scalene, isocetes, etc.)
- quadrangles (square, rectilinear, etc.)
- circles and ellipses

## 3.2 From perception to recognition

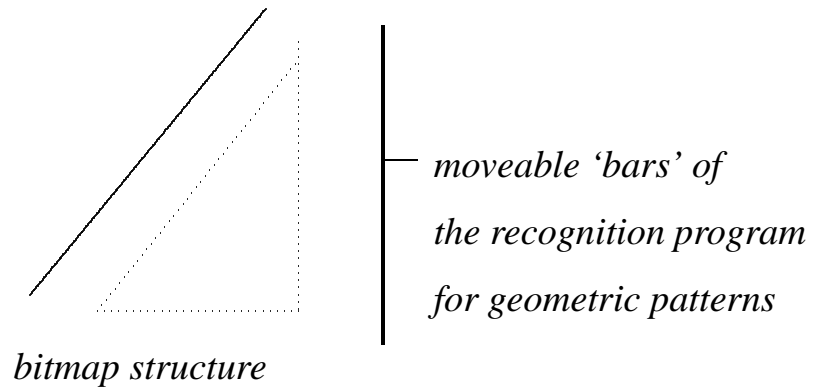
### 3.2.1 Two criteria to evaluate CURIOUS

- Measuring the active and reactive behavior (behavior test)
- Measuring the cognitive processing directly (cognition test)

### 3.2.2 Internal bitmap representation of external object



### 3.2.3 Analysis of an internal bitmap representation



### 3.2.4 Definition of the context

The context of a cognitive agent CA at a given point of time  $t$  includes

1. the total of all current cognitive parameter values  $CA_t$ ,
2. the logical analyses of the parameter values and their combinations (reconstructed patterns),
3. the conceptual structures used to classify the reconstructed patterns and their combinations.

## 3.3 Iconicity of formal concepts

### 3.3.1 I-concept<sub>loc</sub> of a square (token)

$$\left[ \begin{array}{l} \text{edge 1: 2cm} \\ \text{angle 1/2: } 90^0 \\ \text{edge 2: 2cm} \\ \text{angle 2/3: } 90^0 \\ \text{edge 3: 2cm} \\ \text{angle 3/4: } 90^0 \\ \text{edge 4: 2cm} \\ \text{angle 4/1: } 90^0 \end{array} \right]_{loc}$$

### 3.3.2 Definition: I-concept<sub>loc</sub>

An I-concept<sub>loc</sub> results from successfully matching an M-concept onto a corresponding parameter constellation at a certain space-time location.

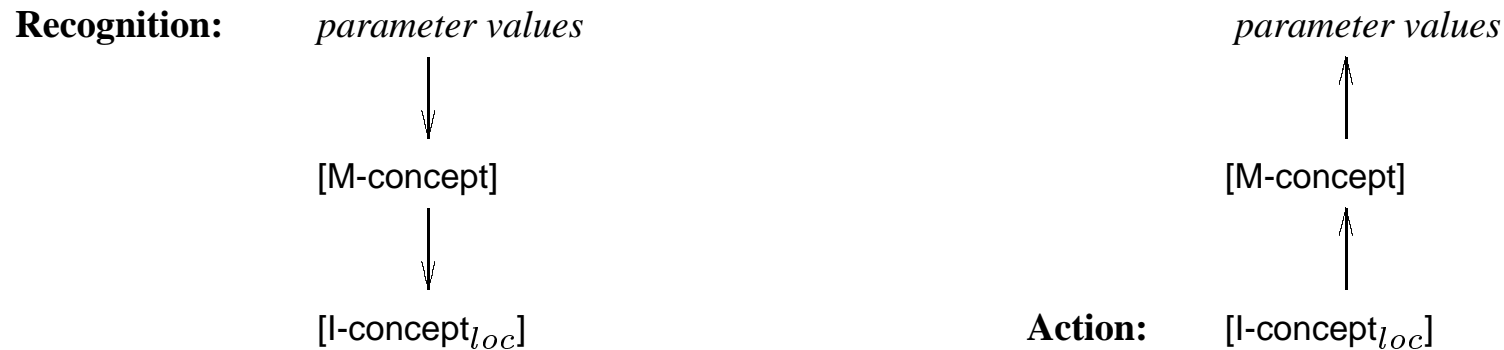
### 3.3.3 M-concept square (type)

$$\left[ \begin{array}{l} \text{edge 1: } \alpha \text{ cm} \\ \text{angle 1/2: } 90^0 \\ \text{edge 2: } \alpha \text{ cm} \\ \text{angle 2/3: } 90^0 \\ \text{edge 3: } \alpha \text{ cm} \\ \text{angle 3/4: } 90^0 \\ \text{edge 4: } \alpha \text{ cm} \\ \text{angle 4/1: } 90^0 \end{array} \right]$$

### 3.3.4 Definition: M-concept

An M-concept is the structural representation of a characteristic parameter constellation whereby certain parameter values are defined as variables.

### 3.3.5 Contextual recognition and action



### 3.3.6 Aspects of iconicity

- The parameter values of the internal context are images insofar as they reflect the corresponding structures of the real world.
- The reconstructed patterns (I-concepts<sub>loc</sub>) are images of parameter values because they are logical analyses of parameter values.
- The M-concepts of the internal context are images insofar as they (i) originate as abstractions over similar parameter constellations and (ii) characterize associated classes of reconstructed patterns.



### 3.3.7 Fallacious arguments against iconicity

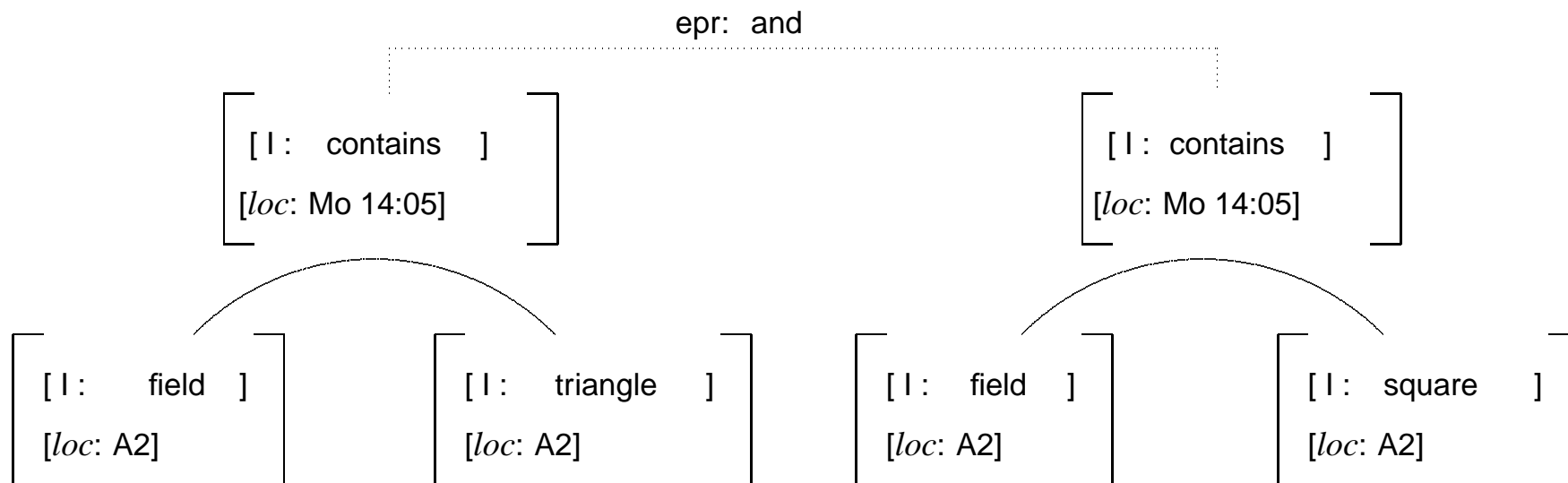
- If one were to surgically search the brain of a person who has seen a tree, one would not find such an image.
- In the case of, e.g., a triangle, what *kind* should the concept should be *exactly* : isoceles, scalene, right-angled? (BERKELEY 1685–1753).
- If there are pictures in the mind, then there must be someone to see them. Yet postulating a little man (homunculus) in the head to see the images would not do because the little man would have images in his little head in turn, requiring another homunculus, and so on. Since postulating a homunculus is of no help to understand the interpretation of images, the images themselves are concluded to be superfluous (HUME 1711–1776).

## 3.4 Contextual I-propositions

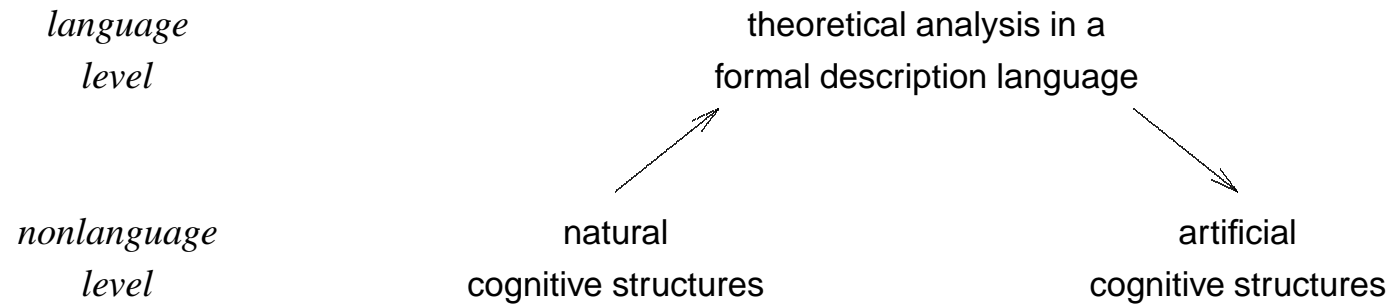
### 3.4.1 The three elements of basic propositions

<i>logic</i>	<i>world</i>	<i>language</i>
1. functor	relation	verb
2. argument	object	noun
3. modifier	property	adjective-adverbial

### 3.4.2 An example of two contextual propositions

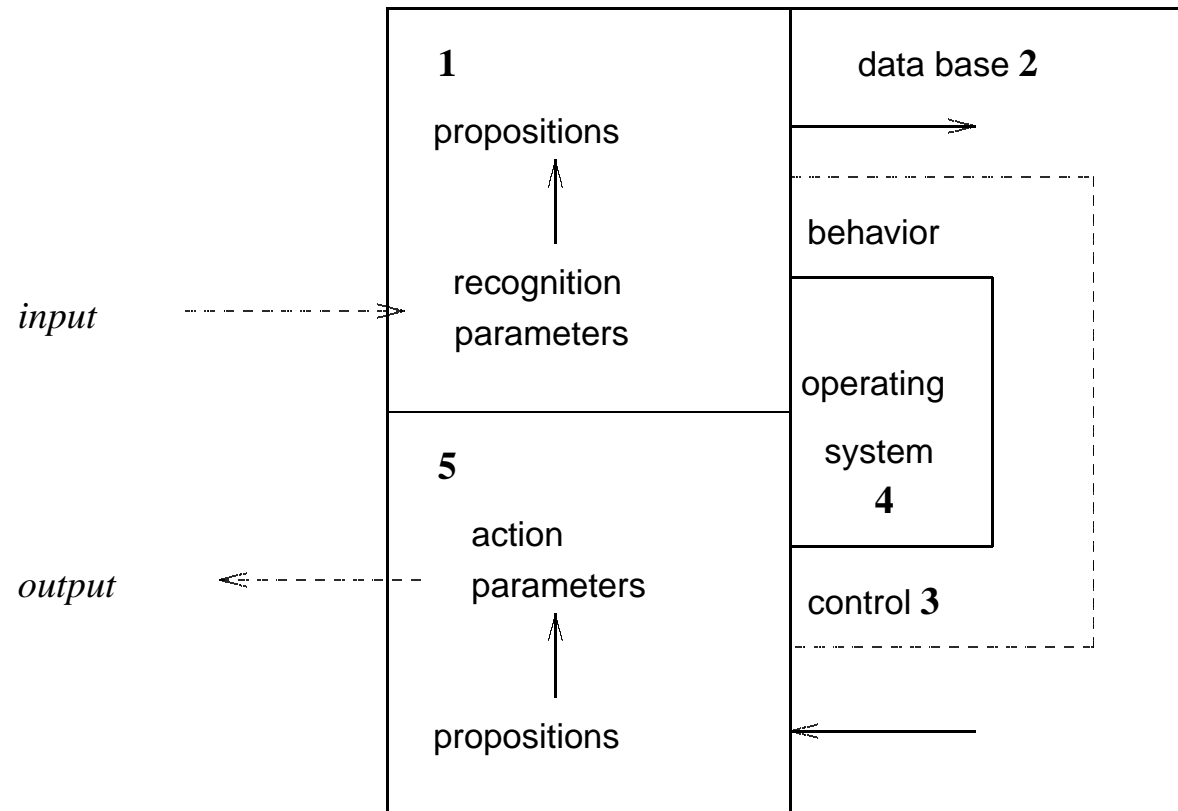


### 3.4.3 Artificial modeling of natural cognition



## 3.5 Recognition and action

### 3.5.1 Schematic structure of context-based cognition



### 3.5.2 Example of a behavior control program

1. Primary analysis of the current task environment:
  - (a) Move into the start field A1.
  - (b) Analyze the current field:
    - i. Approximate bitmap outline with edge program.
    - ii. Measure color value inside the bitmap outline.
    - iii. Derive I-proposition.
  - (c) Write I-proposition at index P-0.1 (present) into the database.
  - (d) If current field is not D4, move into the next field and enter state b. Otherwise go to 2.
2. Secondary analysis of current task environment (inferences):
  - (a) Count all triangles, rectangles, squares, red triangles, etc., in the primary analysis P-0.1 and write the result at index P-0.2 into the database.
  - (b) Compare the current secondary analysis P-0.2 with the previous secondary analysis P-1.2 and write the result (e.g. 'number of red triangle increased by 2') at index P-10.3 into the database.
3. Wait for 10 minutes.
4. Return to state 1.

### **3.5.3 CURIOUS as an autonomous agent (nouvelle AI)**

Without a carefully built physical grounding any symbolic representation will be mismatched to its sensors and actuators. These groundings provide the constraints on symbols necessary for them to be truly useful.'

R.A. Brooks, 1990, S. 6.

### **3.5.4 CURIOUS as a physical symbol system (classic AI)**

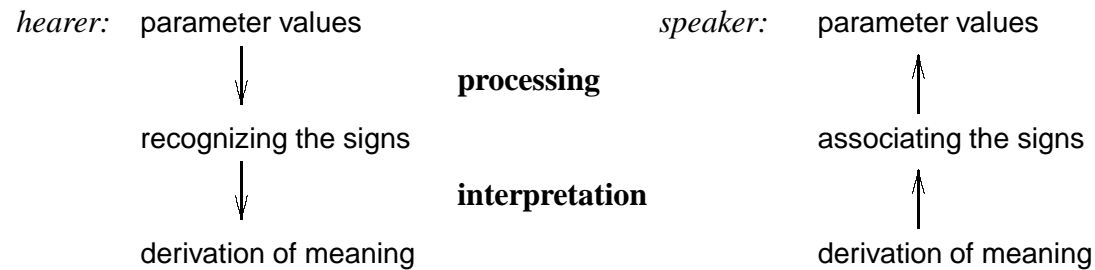
The total concept [of a physical symbol system] is the join of computability, physical realizability (and by multiple technologies), universality, the symbolic representation of processes (i.e. interpretability), and finally, symbolic structure and designation.

A. Newell & H. Simon 1975, p. 46

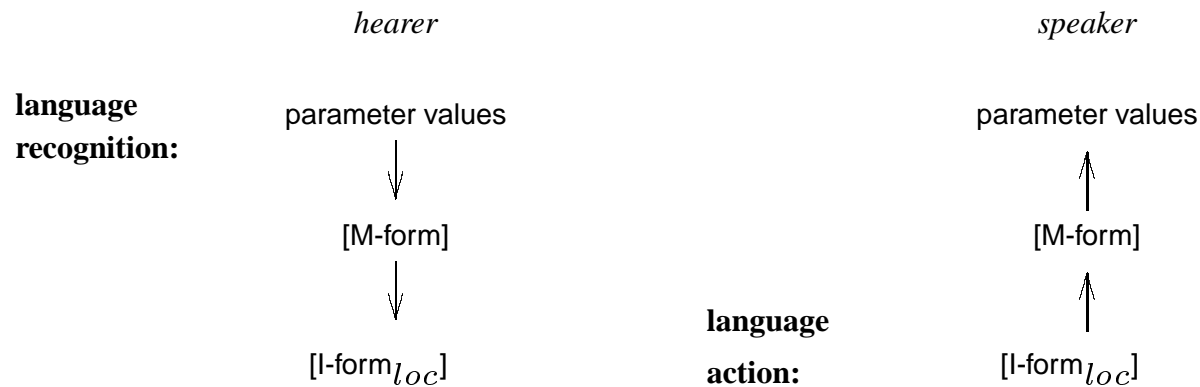
## 4. Language communication

### 4.1 Adding language

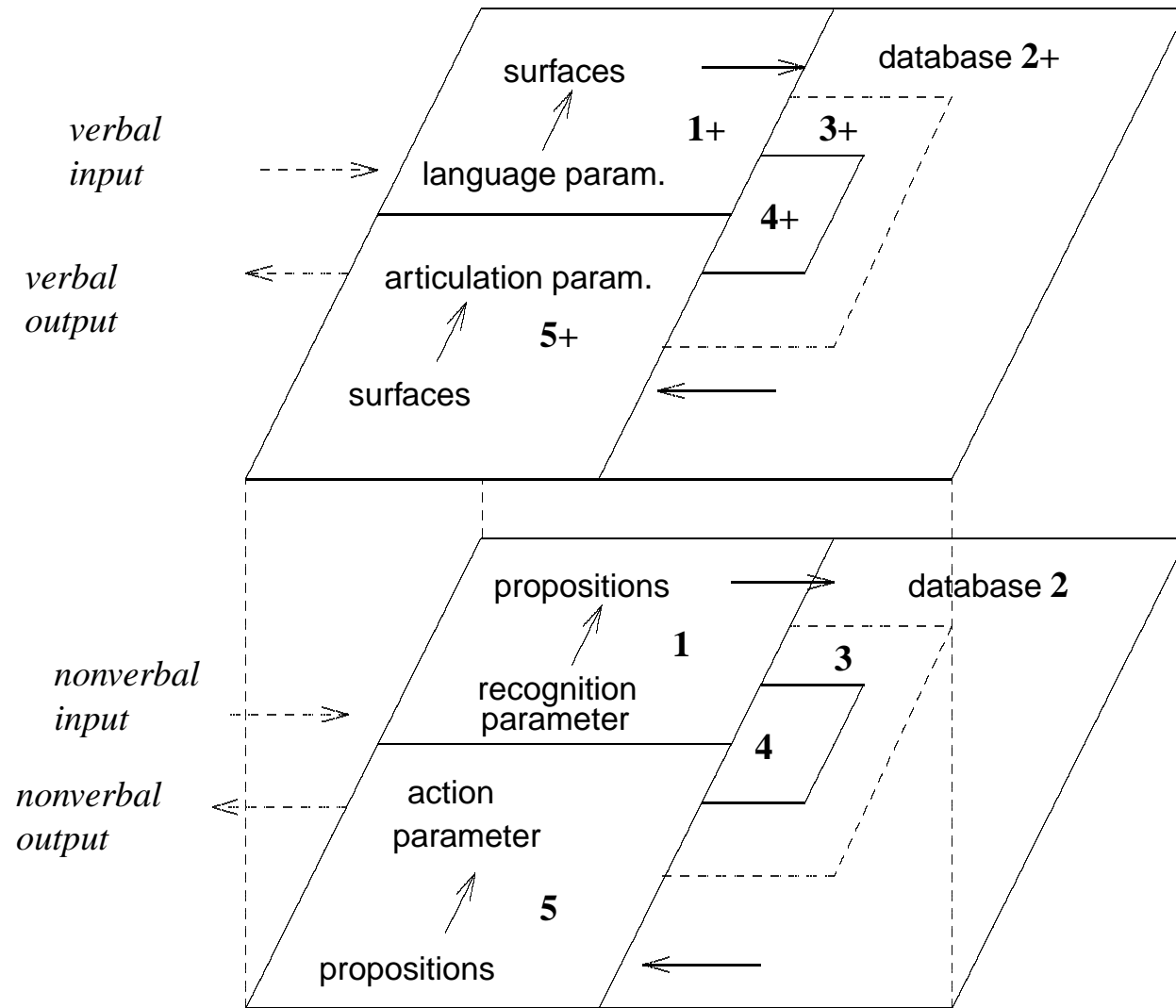
#### 4.1.1 Two subprocedures of language use



#### 4.1.2 Processing signs of language



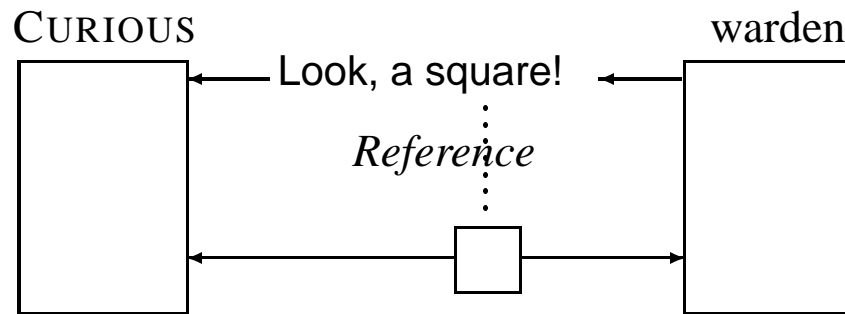
### 4.1.3 Expanded structure of CURIOUS



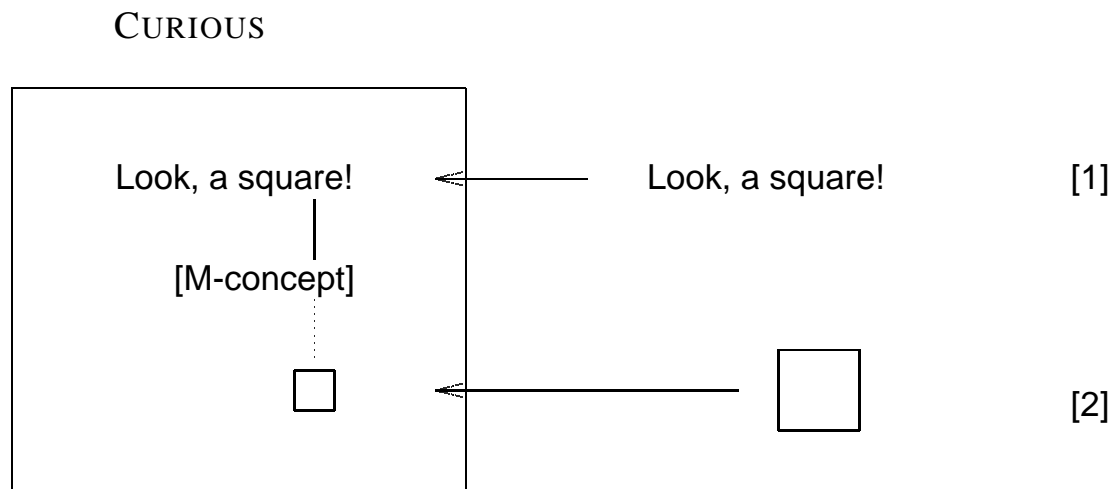


## 4.2 Modeling reference

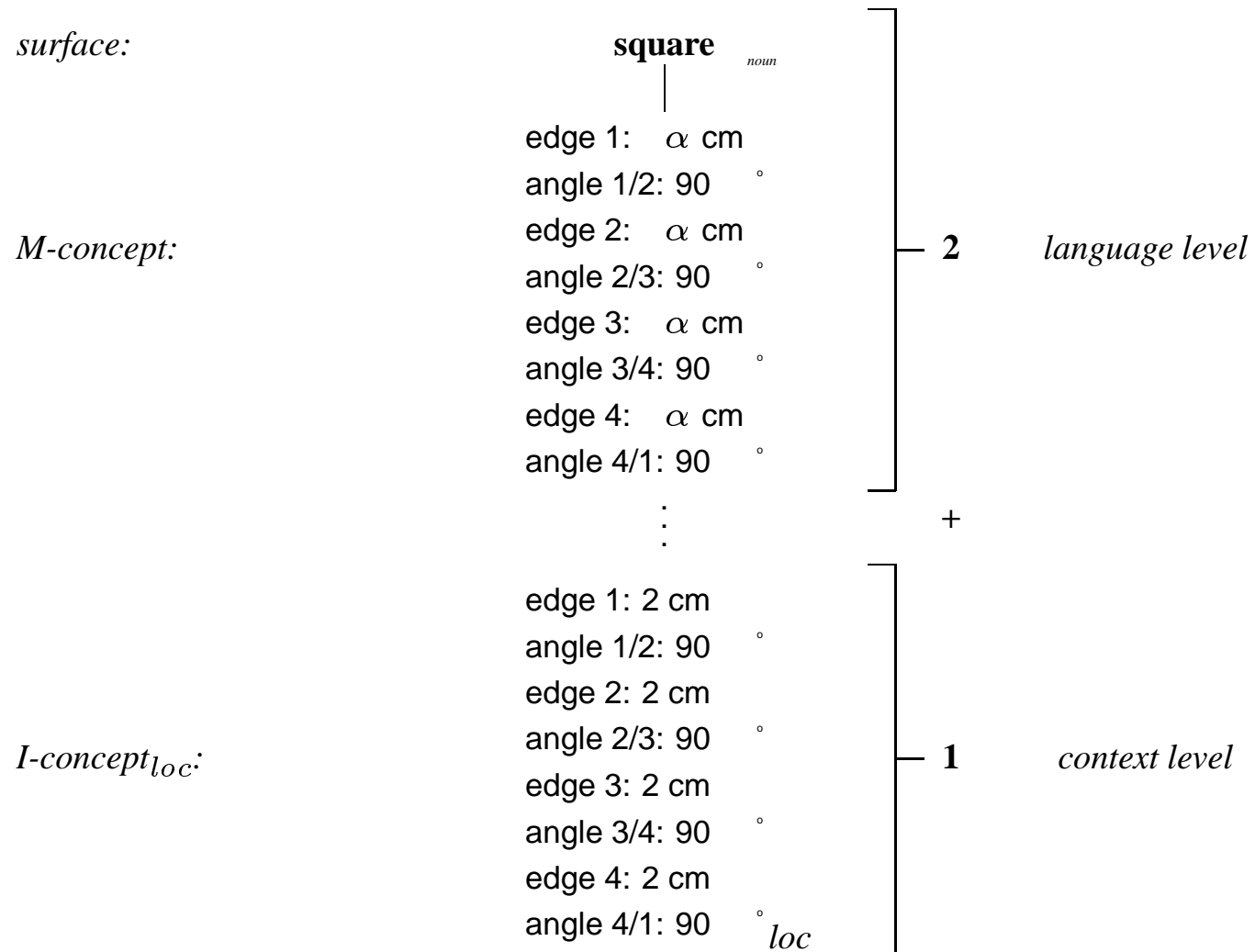
### 4.2.1 An external view of reference



### 4.2.2 Internal and external aspects of reference



### 4.2.3 Cognitive 2+1 level analysis of reference



## 4.3 Using literal meaning

### 4.3.1 Immediate and mediated reference

- *Immediate reference* is the speaker's or the hearer's reference to objects in the current task environment.
- *Mediated reference* is the speaker's or hearer's reference to objects which are not in the current task environment.

### 4.3.2 Two notions of meaning

- $\text{meaning}_1$  = property of signs, also called literal meaning
- $\text{meaning}_2$  = property of utterances, also called speaker meaning

### 4.3.3 First principle of pragmatics (PoP-1)

The speaker's utterance  $\text{meaning}_2$  is the use of the sign's literal  $\text{meaning}_1$  relative to an internal context.

## 4.4 Frege's principle

### 4.4.1 Frege's principle

The meaning of a complex expression is a function of the meaning of the parts and their mode of composition.

### 4.4.2 Different parts

- a.* The dog bites the *man*
- b.* The dog bites the *bone*

### 4.4.3 Different composition

- a.* The *dog* bites the *man*
- a'.* The *man* bites the *dog*

#### 4.4.4 Standard interpretation of Frege's principle

surface:	<b>a</b>	=	<b>a</b>		<b>a</b>	≠	<b>b</b>
	⋮		⋮		⋮		⋮
meaning <sub>1</sub> :	<b>A</b>	=	<b>A</b>		<b>A</b>	≠	<b>B</b>

#### 4.4.5 Syntactic ambiguity

They don't know how good meat tastes

#### 4.4.6 Paraphrase

The dog bit the man (active)

The man was bitten by the dog (passive)

### 4.4.7 Apparent exceptions (incorrect analysis)

	ambiguity	paraphrase
surface:	$a = a$	$a \neq b$
	⋮            ⋮	⋮            ⋮
meaning <sub>1</sub> :	$A \neq A'$	$A = B$

### 4.4.8 Syntactic ambiguity (correct analysis)

unanalyzed surface:	$a = a$	
analyzed surface:	$a \neq a'$	} scope of the Fregean principle
	⋮            ⋮	
meaning <sub>1</sub> :	$A \neq A'$	

### 4.4.9 Syntactic paraphrase

	<i>incorrect</i>	<i>correct</i>
surface:	$2 + 4 \neq 3 + 3$	$2 + 4 \neq 3 + 3$
	⋮            ⋮	⋮            ⋮
meaning <sub>1</sub> :	$6 = 6$	$2' + 4' \sim 3' + 3'$
	<b>identity</b>	<b>equivalence</b>

## 4.5 Surface compositionality

In its standard interpretation, Frege's principle corresponds to the principle of surface compositionality.

### 4.5.1 Surface compositionality I (SC-I principle)

An analysis of natural language is surface compositional if it uses only concrete word forms as the building blocks such that all syntactic and semantic properties of complex expression derive systematically from the syntactic category and the meaning<sub>1</sub> of their building blocks.

### 4.5.2 Consequences of surface compositionality

- Methodologically:  
Syntactic analyses are *concrete* because no kind of zero surface or underlying form may be used,
- Mathematically:  
Syntactic and semantic analyses may be of *low complexity*
- Functionally:  
The internal matching between meaning<sub>1</sub> and context may be extended from single words to the systematic syntactic-semantic *combination* of expressions.

## *Violating surface compositionality:* EXAMPLE I

### 4.5.3 Linguistic generalizations with transformational grammar

Transformations are supposed to be innate, yet have no function in communication.

### 4.5.4 Examples of ‘classical’ transformations

DEEP STRUCTURE:

SURFACE STRUCTURE:

*Passive:*

Peter closed the door

⇒ The door was closed by Peter

*Do-support:*

Peter not open the door

⇒ Peter didn't open the door

*Reflexivization*

Peter<sub>i</sub> shaves Peter<sub>i</sub>

⇒ Peter shaves himself

*There-insertion*

A hedgehog is in the garden

⇒ There is a hedgehog in the garden



*Pronominalization*

Peter<sub>i</sub> said that Peter<sub>i</sub> was tired      ⇒ Peter said that he was tired

*Relative clause formation*

Peter [Peter was tired]      ⇒ Peter, who was tired

*Main clause order in German*

Peter die Tür geschlossen hat      ⇒ Peter hat die Tür geschlossen

*Object raising*

Peter persuaded Jim [Jim sleeps]      ⇒ Peter persuaded Jim to sleep

*Subject-raising*

Peter promised Jim [Peter sleeps]      ⇒ Peter promised Jim to sleep

#### 4.5.5 Transformations and the standard interpretation of Frege's Principle

For a while, transformational grammar assumed the equivalence of

*active*    Everyone in this room speaks at least two languages

*passive*    At least two languages are spoken by everyone in this room

### **4.5.6 Transformations and Darwin's law: Form follows function**

The structure of, e.g., a duck foot is innate. Good science should explain its form in terms of its function. The same holds for innate cognitive structures, e.g., the language ability.

### **4.5.7 Cognitive variant of Occam's razor**

Entities or components of grammar should not be postulated as innate if they have no clearly defined function within natural communication.

### **4.5.8 Applications of the cognitive razor**

The cognitive razor applies to transformational grammar as well as all later variants of nativism including LFG, GPSG, and HPSG. Like transformational grammar, their linguistic generalizations are nonfunctional with respect to communication and inherently in violation of surface compositionality.

## *Violating surface compositionality:* EXAMPLE II

### **4.5.9 Definition of meaning by Grice**

Definiendum: U meant something by uttering x.

Definiens: For some audience A, U intends his utterance of x to produce in A some effect (response) E, by means of A's recognition of the intention.

### **4.5.10 Explaining the evolution of language**

Grice defines sentence meaning as an utterance type and utterance meaning as a token of this utterance type.

Problem:

How can a type evolve if it is already presupposed by the first utterance meaning?

### **4.5.11 Conflicting uses of convention**

Literal use: Conveying intentions by obeying conventions

Metaphoric use: Conveying intentions by violating conventions

### 4.5.12 Elementary notions suitable for computational implementation?

Recognition of an intention, producing some effect, intending for some audience...

### 4.5.13 Successful man-machine communication

L = a natural language, SH = a human speaker-hearer of L, CA = a cognitive agent.

- *Successful natural language interpretation*

CA communicates successfully in the hearer mode, if CA understands the L-utterance in the way intended by SH. In technical terms this means that CA correctly recreates the speaker meaning of the L-utterance in its database. The developers of CA can verify the procedure because (i) they themselves can understand the utterance in L and (ii) they can view the interpretation directly in the database of CA.

- *Successful natural language production*

CA communicates successfully in the speaker mode, if CA formulates its intentions in L in a way that SH can understand. This requires technically that CA maps a certain structure in its database into an L-utterance which SH can correctly reconstruct. The developers of CA can verify the procedure because (i) they have direct access to the database structure to be communicated and (ii) they themselves can understand the utterance in L.

## 5. Using language signs on suitable contexts

### 5.1 Bühler's organon model

#### 5.1.1 Theory of pragmatics

Analyzes the general principles of purposeful action.

Describes how a cognitive agent can achieve certain goals.

#### 5.1.2 Examples of pragmatic problems

- The use of a screw driver to fasten a screw
- The use of one's legs to go from  $a$  to  $b$
- The scavenging of the refrigerator in the middle of the night to fix a BLT sandwich and satisfy one's hunger
- The request that someone fix and serve the sandwich

### **5.1.3 Nonlinguistic and linguistic pragmatics**

Depending on whether or not the means employed are signs of language we speak of linguistic and nonlinguistic pragmatics.

### **5.1.4 Embedding linguistic in nonlinguistic pragmatics**

Just as language recognition and articulation may be analyzed as a phylo- and ontogenetic specialization of contextual (nonverbal) recognition and action (cf. 4.1.3), respectively, linguistic pragmatics may be analyzed as a phylo- and ontogenetic specialization of nonlinguistic pragmatics.

### **5.1.5 Language as an organon**

Embedding of linguistic pragmatics into nonlinguistic pragmatics:

PLATO (427(?)–347 BC)

KARL BÜHLER (1879–1963 AD)

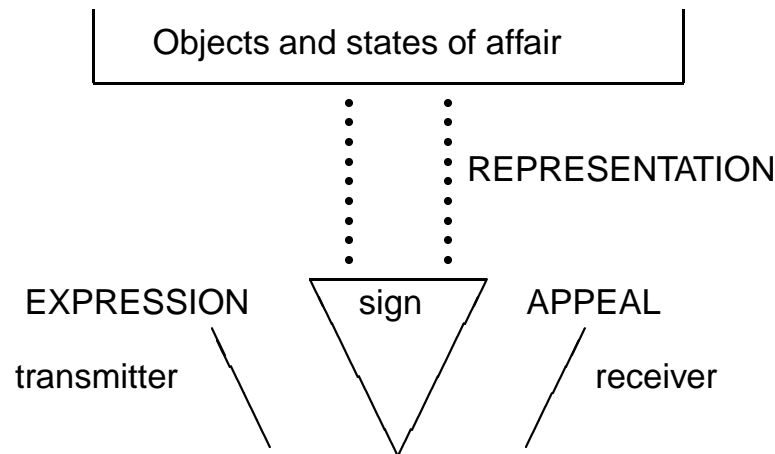
### 5.1.6 The tool character of language

Die Sprache ist dem Werkzeug verwandt; auch sie gehört zu den Geräten des Lebens, ist ein Organon wie das dingliche Gerät, das leibesfremde Zwischending; die Sprache ist wie das Werkzeug ein *geformter Mittler*. Nur sind es nicht die materiellen Dinge, die auf den sprachlichen Mittler reagieren, sondern es sind die lebenden Wesen, mit denen wir verkehren.

[Language is akin to the tool: language belongs to the instruments of life, it is an organon like the material instrument, a body-extraneous hybrid; language is – like the tool – a *purposefully designed mediator*. The only difference is that it is not material things which react to the linguistic mediator, but living beings with whom we communicate.]

K. Bühler 1934, p. XXI

### 5.1.7 Bühler's organon model



Representation refers to the language-based transfer of information. Expression refers to the way the transmitter produces the sign. Appeal refers to the way the sign affects the receiver beyond the bare content of the sign.

### 5.1.8 Shannon & Weaver's information theory 1949

Central notions besides transmitter and receiver are the band width of the channel, the redundancy and relative entropy of the codes, and the noise in the transmission. Its laws hold also in everyday conversation, but background noises, slurring of speech, hardness of hearing, etc., are not components of the natural communication mechanism.



### 5.1.9 Comparing organon model and CURIIOUS (4.1.3)

The organon model describes the relation between the ‘transmitter’ and the ‘receiver’ from an external viewpoint and is therefore limited to immediate reference.

The SLIM model of CURIIOUS describes the internal structure of the speaker-hearer and can therefore handle mediated reference in addition to immediate reference.

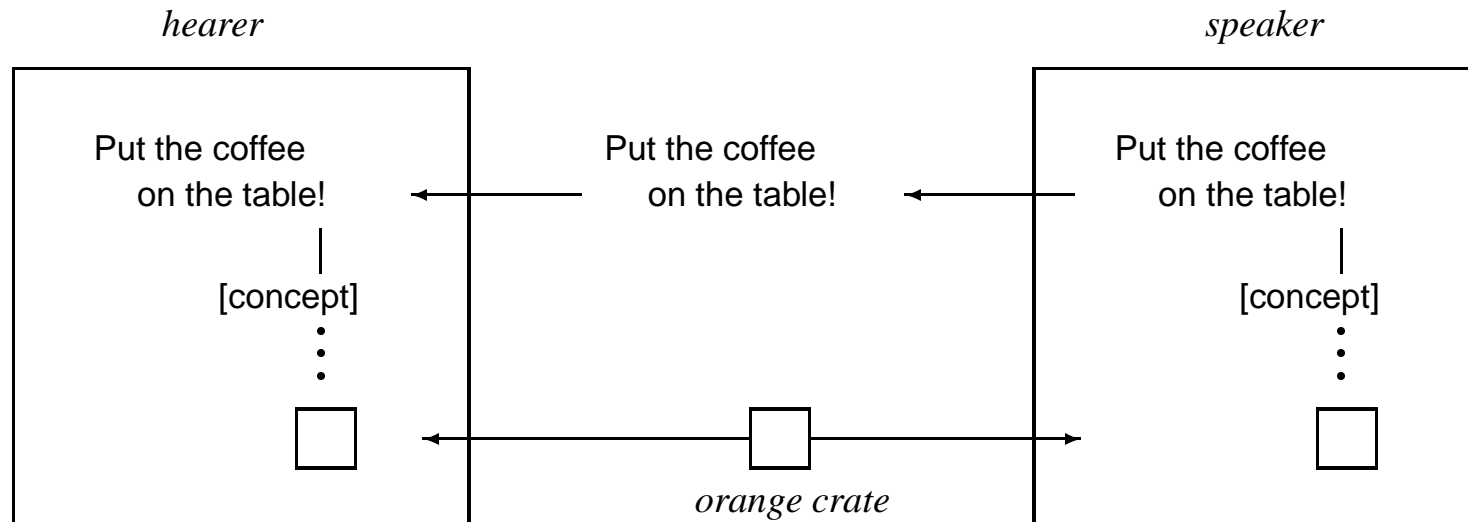
The organon function of ‘expression’ is to be located in component 5+ (language synthesis) of CURIIOUS.

The organon function of ‘appeal’ is to be located in component 1+ (language recognition) of CURIIOUS.

The organon function of ‘representation’ is performed by CURIIOUS in the lexical, syntactic, and semantic components of the language-based database structure 2+ and interpreted in relation to the contextual database structure 2.

## 5.2 Pragmatics of tools and pragmatics of words

### 5.2.1 Nonliteral use of the word table: Principle of best match



### 5.2.2 Central question of linguistic pragmatics

How does the speaker code the selection and delimitation of the used subcontext into the sign and how can these be correctly inferred by the hearer?

## 5.3 Finding the correct subcontext

### 5.3.1 Postcard example

New York, December 1, 1998

Dear Heather,

Your dog is doing fine. The weather is very cold. In the morning he played in the snow. Then he ate a bone. Right now I am sitting in the kitchen. Fido is here, too. The fuzzball hissed at him again. We miss you.

Love,  
Spencer

### 5.3.2 Parameters of origin of signs (STAR-point)

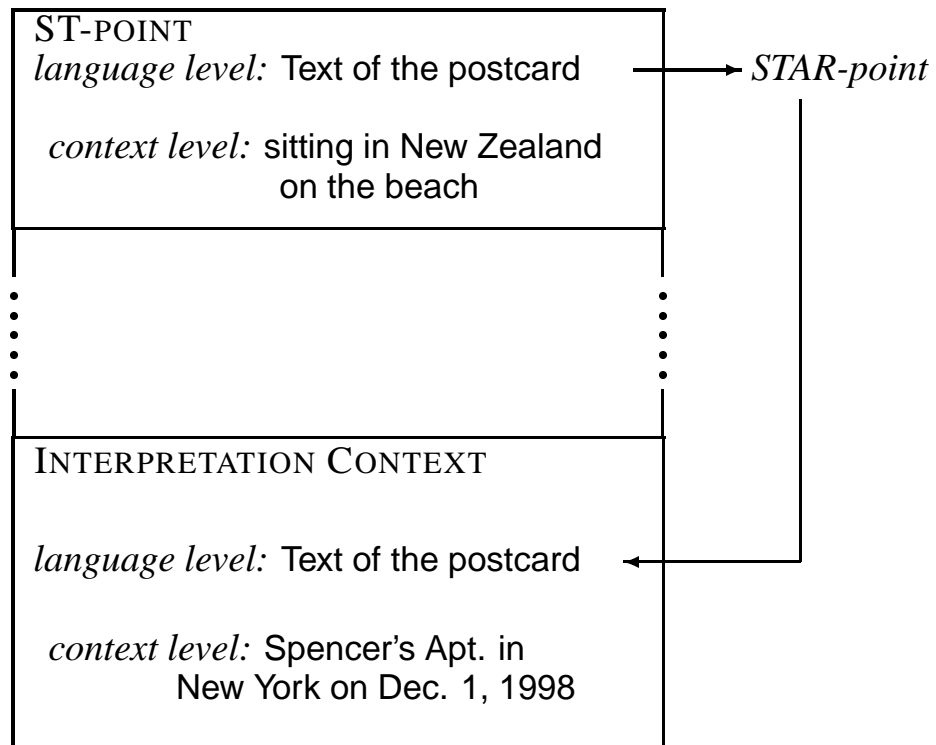
1. S = the **S**patial place of origin
2. T = the **T**emporal moment of origin
3. A = the **A**uthor
4. R = the intended **R**ecipient.

### 5.3.3 Second principle of pragmatics (PoP-2)

The STAR-point of the sign determines its primary positioning in the database by specifying the *entry context* of interpretation.

### 5.3.4 Primary positioning in terms of the STAR-point

Heather's cognitive representation:



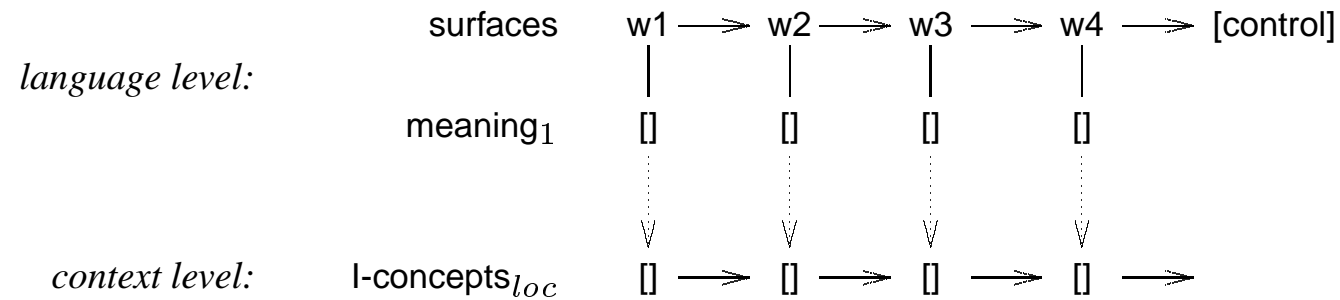
### 5.3.5 Fictitious STAR-point: Beginning of 'Felix Krull'

Indem ich die Feder ergreife, um in völliger Muße und Zurückgezogenheit – gesund übrigens, wenn auch müde, sehr müde ...

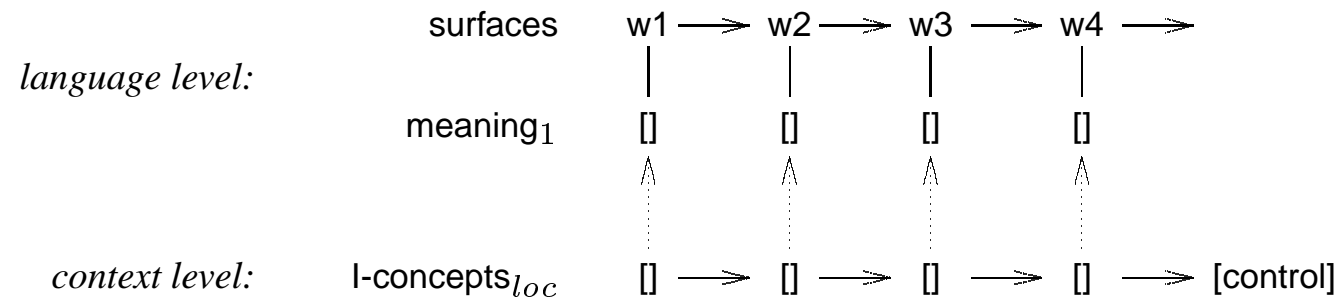
[While I seize the pen in complete leisure and seclusion – healthy, by the way – though tired, very tired ...]

## 5.4 Language production and interpretation

### 5.4.1 Schema of language interpretation (analysis)



### 5.4.2 Schema of language production (generation)



### 5.4.3 The time-linear structure of natural language signs

The basic structure of natural language signs is their *time-linear order*. This holds for the sentences in a text, the word forms in a sentence, and the allomorphs in a word form.

*Time-linear* means:

LINEAR LIKE TIME AND IN THE DIRECTION OF TIME.

#### 5.4.4 De Saussure's second law: *linear character of signs*

SECOND PRINCIPE; CARACTÈRE LINÉAIRE DU SIGNIFIANT.

Le signifiant, étant de nature auditive, se déroule dans le temps seul et a les caractères qu'il emprunte au temps: a) *représente une étendue*, et b) *cette étendue est mesurable dans une seule dimension*: c'est une ligne.

Ce principe est évident, mais il semble qu'on ait toujours négligé de l'énoncer, sans doute parce qu'on l'a trouvé trop simple; cependant il est fondamental et les conséquences en sont incalculables; son importance est égale à celle de la première loi. Tout le mécanisme de la langue en dépend.

[The designator, being auditory in nature, unfolds solely in time and is characterized by temporal properties: (a) *it occupies an expansion*, and (b) *this expansion is measured in just one dimension*: it is a line.

This principle is obvious, but it seems that stating it explicitly has always been neglected, doubtlessly because it is considered too elementary. It is, however, a fundamental principle and its consequences are incalculable. Its importance equals that of the first law. All the mechanisms of the language depend on it.]

F. de Saussure 1913/1972, p. 103



### **5.4.5 Third principle of pragmatics (PoP-3)**

The matching of word forms with their respective subcontexts is incremental whereby in production the elementary signs follow the time-linear order of the underlying thought path while in interpretation the thought path follows the time-linear order of the incoming elementary signs.

## **5.5 Thought as the motor of spontaneous production**

### **5.5.1 The once famous motto of behaviorism**

THOUGHT IS NONVERBAL SPEECH

### **5.5.2 The motto of the SLIM theory of language**

SPEECH IS VERBALIZED THOUGHT.

Thought is defined as the time-linear navigation of a focus point through the concatenated propositions of the internal database.

### **5.5.3 The role of time-linear order for the semantic interpretation**

Original order:

In the morning he played in the snow. Then he ate a bone.

Inverted order (incoherent):

Then he ate a bone. In the morning he played in the snow.

### 5.5.4 Alternative navigation through propositional content ( anti-temporal sequencing)

In the morning Fido ate a bone. Before that he played in the snow.

### 5.5.5 Modification of interpretation by changing sequencing

*a.* 1. In February, I visited the Equator. 2. There it was very hot. 3. In March, I was in Alaska. 4. There it was very cold.

*b.* 3. In March, I was in Alaska. 2. There it was very hot. 1. In February, I visited the Equator. 4. There it was very cold.

### 5.5.6 The time-linearity of speech

Speech is irreversible. That is its fatality. What has been said cannot be unsaid, except by adding to it: to correct here is, oddly enough, to continue.

R. Barthes, 1986, p. 76

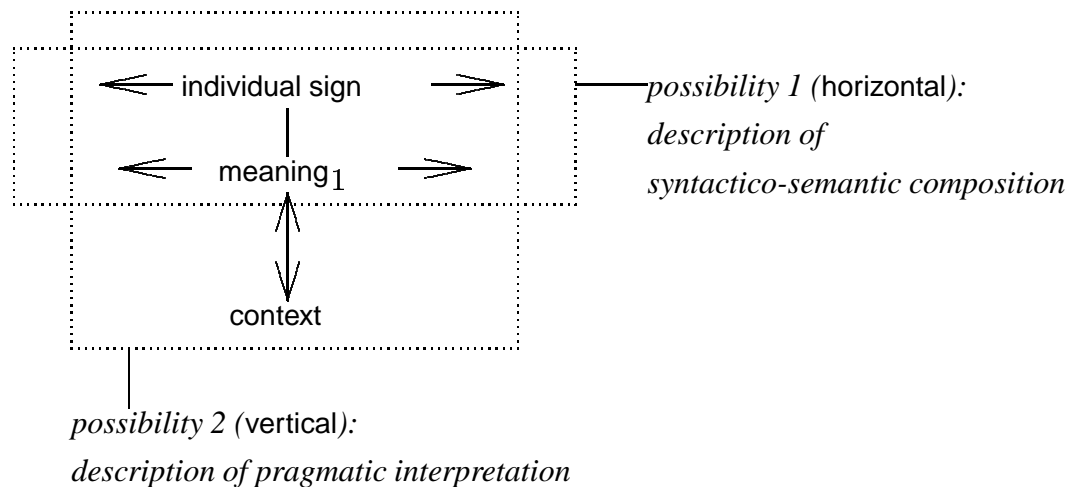
## 6. Structure and functioning of signs

### 6.1 Reference mechanisms of different sign types

#### 6.1.1 The mechanism of natural language communication as described so far

1. PoP-1: Language use as an internal matching between meaning<sub>1</sub> and a subcontext
2. PoP-2: Entrance subcontext determined by STAR-point of the sign
3. PoP-3: Derivation order is the time-linear sequence of words

#### 6.1.2 Alternatives of description



### 6.1.3 Example with minimal syntax

Me up. Weather nice. Go beach. Catch fish. Fish big. Me hungry. Fix breakfast. Eat fish. Feel good.

### 6.1.4 Fourth principle of pragmatics (PoP-4)

The meaning<sub>1</sub> of the sign type **symbol** is defined as an M-concept. Symbols refer from their place in a positioned sentence by matching their M-concept with suitable contextual referents (I-concepts<sub>loc</sub>).

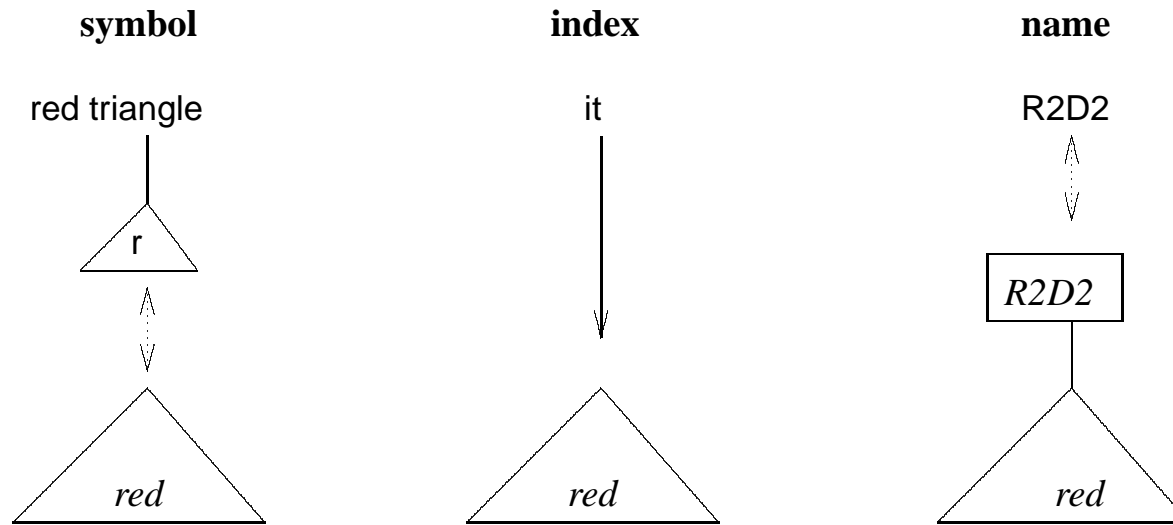
### 6.1.5 Fifth principle of pragmatics (PoP-5)

The meaning<sub>1</sub> of the sign type **index** is defined as a pointer. An index refers by pointing from its place in the positioned sentence into appropriate parameter values.

### 6.1.6 Sixth principle of pragmatics (PoP-6)

The reference mechanism of the sign type **name** is based on an act of naming which consists in adding a name-marker to the internal representation of the individual or object in question. A name refers by matching its surface with a corresponding marker.

### 6.1.7 Comparing iconic, indexical, and name-based reference



### 6.1.8 Reference in nonverbal and preverbal communication

1. nonverbal iconic reference consists in spontaneously imitating the referent by means of gestures or sounds,
2. nonverbal indexical reference consists in pointing at the referent, and
3. nonverbal name-based reference consists in pointing at the referent while simultaneously pronouncing a name.

### 6.1.9 Seventh principle of pragmatics (PoP-7)

The sign type *symbol* occurs as noun, verb, and adjective-adverbial. The sign type *index* occurs as noun and adjective-adverbial. The sign type *name* occurs only as noun.

### 6.1.10 Relation between sign types and parts of speech

name	Peter		
index	he	here	
symbol	man	old	see
	noun	adj-adv	verb

## 6.2 Internal structure of symbols and indices

### 6.2.1 Internal components of the sign types symbol and index

- The *surface* is constrained by the laws of acoustic articulation (in the original medium of spoken language).
- The *category* reflects the combinatorial properties of the part of speech and the inflectional class to which the sign belongs.
- The *meaning*<sub>1</sub> reflects the conceptual structures of the internal context and/or contains characteristic pointers to certain contextual parameters.
- The *glue* connecting surface, category, and *meaning*<sub>1</sub> consists in conventions which must be learned by each member of the language community.



### 6.2.2 De Saussure's first law

PREMIER PRINCIPE; L'ARBITRAIRE DU SIGNE.

Le lien unissant signifiant au signifié est arbitraire, ou encore, puisque nous entendons par signe le total résultant de l'association d'un signifiant à un signifié, nous pouvons dire plus simplement: *le signe linguistique est arbitraire.*

[THE FIRST LAW: ARBITRARINESS OF SIGNS

The link connecting the designator and the designated is arbitrary; and since we are treating a sign as the combination which results from connecting the designator with the designated, we can express this more simply as: *the linguistic sign is arbitrary.*]

F. de Saussure 1913/1972, p. 100

### 6.2.3 Possible functions of the sign type symbol

1. *Initial* reference to objects which have not been mentioned so far.
2. *Repeating* reference to referents which have already been introduced linguistically.
3. *Metaphorical* reference to partially compatible referents, both in initial and repeating reference.

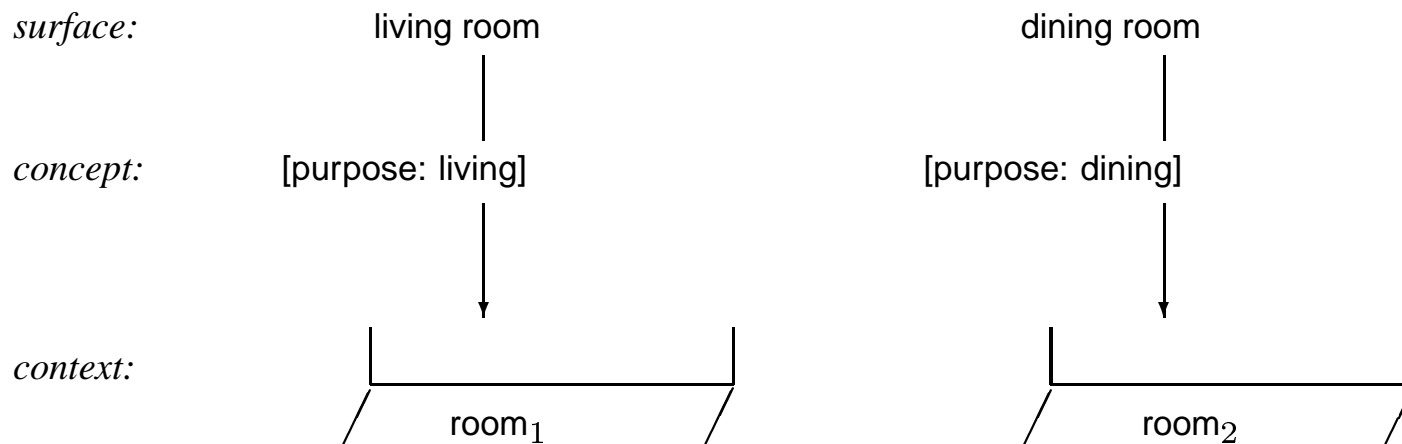
### 6.2.4 Structural basis of symbolic reference

- the minimal meaning<sub>1</sub> structure of symbols (M-concepts), and
- the limited selection of compatible referents available in the subcontext.

### 6.2.5 Integrating additional symbolic content to ensure reference

the table, the garden table, the green garden table, the small green garden table, the round small green garden table, the round small green garden table near the pool, etc.

### 6.2.6 Characterizing objects symbolically



## **6.3 Indices for repeating reference**

### **6.3.1 Pure indices**

Index words which contain no additional grammatical or symbolic meaning components, e.g. here, now, and you.

### **6.3.2 Nonpure indices**

Index words which incorporate symbolic-grammatical distinctions, e.g. between singular (I) and plural (we), between nominative (I, we) and oblique (me, us) case, etc.

### **6.3.3 Pointing area of third person pronouns**

It is outside of the STAR-point parameters and comprises all objects and persons that have been activated so far and are neither the speaker nor the hearer.

### **6.3.4 Repeating reference**

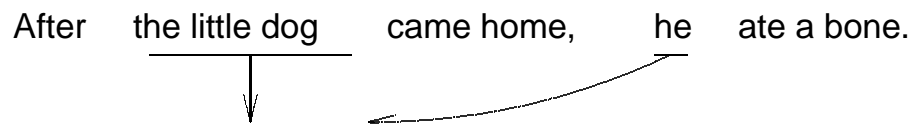
This special task reference arises in longer sentences and texts when a certain referent has already been introduced and needs to be referred to again.

### 6.3.5 Using third person pronouns for repeating reference

Because of their grammatical differentiation and the general nature of their pointing area, third person pronouns are ideally suited for a brief, precise, and versatile handling of repeating reference .

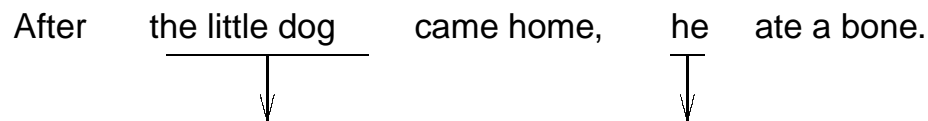
### 6.3.6 Indexically repeating reference

After the little dog came home, he ate a bone.



### 6.3.7 Indexical reference without coreference

After the little dog came home, he ate a bone.



### 6.3.8 Symbolically repeating reference I

After he came home, the little dog ate a bone.

The diagram illustrates coreference between the pronoun 'he' and the noun phrase 'the little dog'. A horizontal line is drawn under 'he', and another horizontal line is drawn under 'the little dog'. A curved arrow points from the line under 'the little dog' back to the line under 'he', indicating that both phrases refer to the same entity.

### 6.3.9 Symbolic reference without coreference

After he came home, the little dog ate a bone.

The diagram illustrates symbolic reference without coreference. A horizontal line is drawn under 'he', and another horizontal line is drawn under 'the little dog'. A vertical arrow points down from the line under 'he', and another vertical arrow points down from the line under 'the little dog'. This indicates that both phrases have symbolic reference to their respective referents, but they do not refer to the same entity.

### 6.3.10 Sentence structure blocking repeating reference

Anaphorical positioning:

After Peter<sub>*i*</sub> came home he<sub>*i*</sub> took a bath.

Peter<sub>*i*</sub> took a bath after he<sub>*i*</sub> came home.

%! Near Peter<sub>*i*</sub> he<sub>*i*</sub> sees a snake.

Cataphorical positioning:

After he<sub>*i*</sub> came home Peter<sub>*i*</sub> took a bath.

%! He<sub>*i*</sub> took a bath after Peter<sub>*i*</sub> came home.

Near him<sub>*i*</sub> Peter<sub>*i*</sub> saw a snake.

### 6.3.11 Cross-sentential coreference

Peter<sub>*k*</sub> wanted to drive into the country. He<sub>*k*</sub> waited for Fido<sub>*i*</sub>. When the little dog<sub>*i*</sub> came home he<sub>*k*</sub> was glad.

### 6.3.12 Symbolically repeating reference II

After Fido came home, the little dog ate a bone.

### 6.3.13 Initial reference established by symbol, index, and name

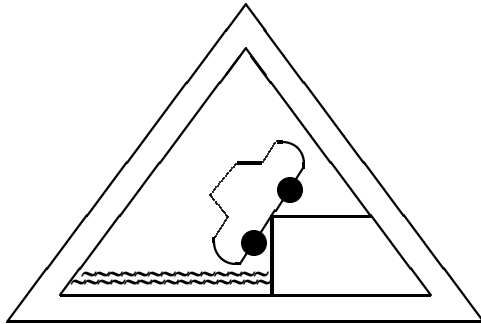
the little dog  
After he came home, he slept.  
Fido

### 6.3.14 Repeating reference using symbol, index, and name

After he came home, the little dog  
he slept.  
Fido

## 6.4 Exceptional properties of icon and name

### 6.4.1 Example of an icon (street sign)



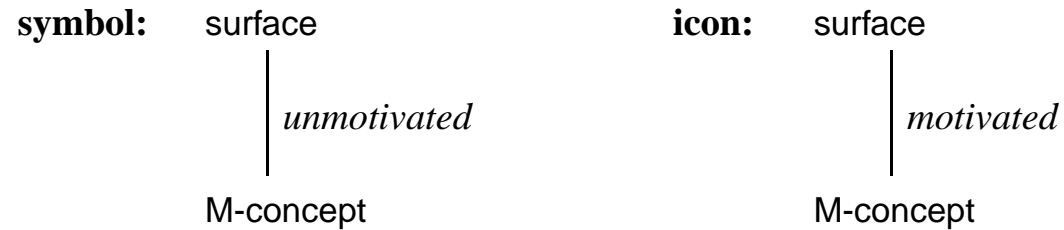
### 6.4.2 The controversy between *naturalists* and *conventionalists*

[The naturalists] maintained that all words were indeed ‘naturally’ appropriate to the things they signified. Although this might not always be evident to the layman, they would say, it could be demonstrated by the philosopher able to discern the ‘reality’ that lay behind the appearance of things. Thus was born the practice of conscious and deliberate etymology. The term itself (being formed from the Greek stem *etymo-* signifying ‘true’ or ‘real’) betrays its philosophical origin. To lay bare the origin of a word and thereby its ‘true’ meaning was to reveal one of the truths of ‘nature’.

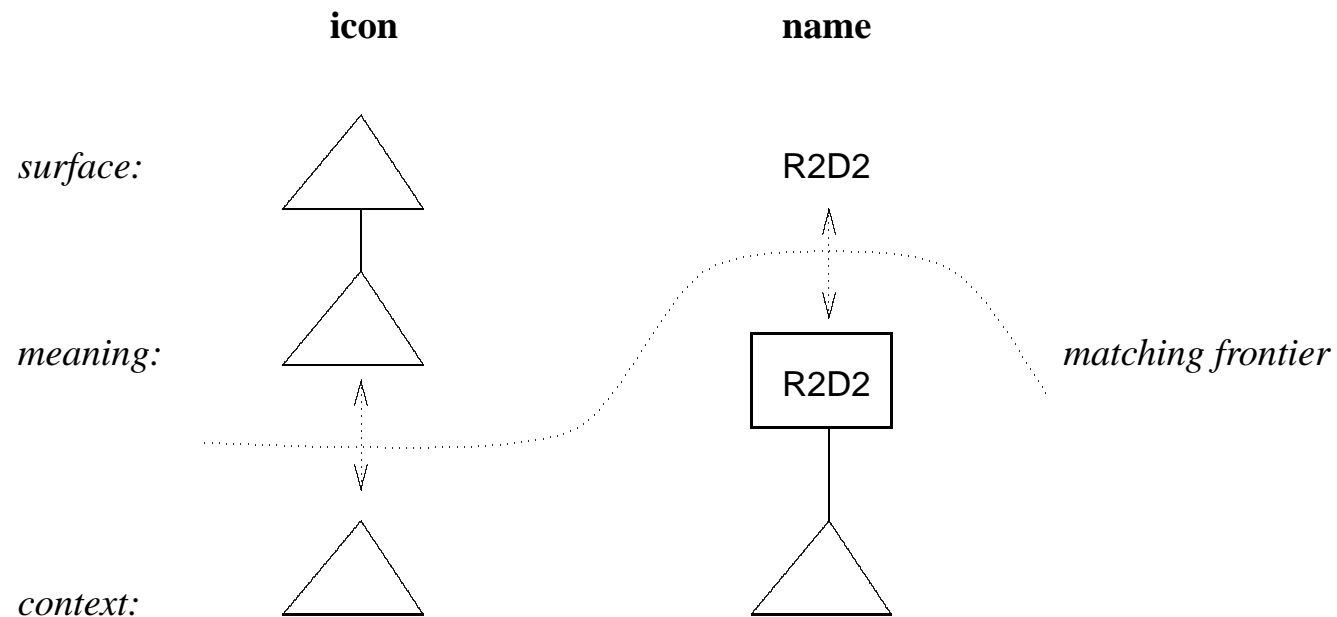
J. Lyons 1968, p. 4 f.



### 6.4.3 Comparing the structure of symbol and icon

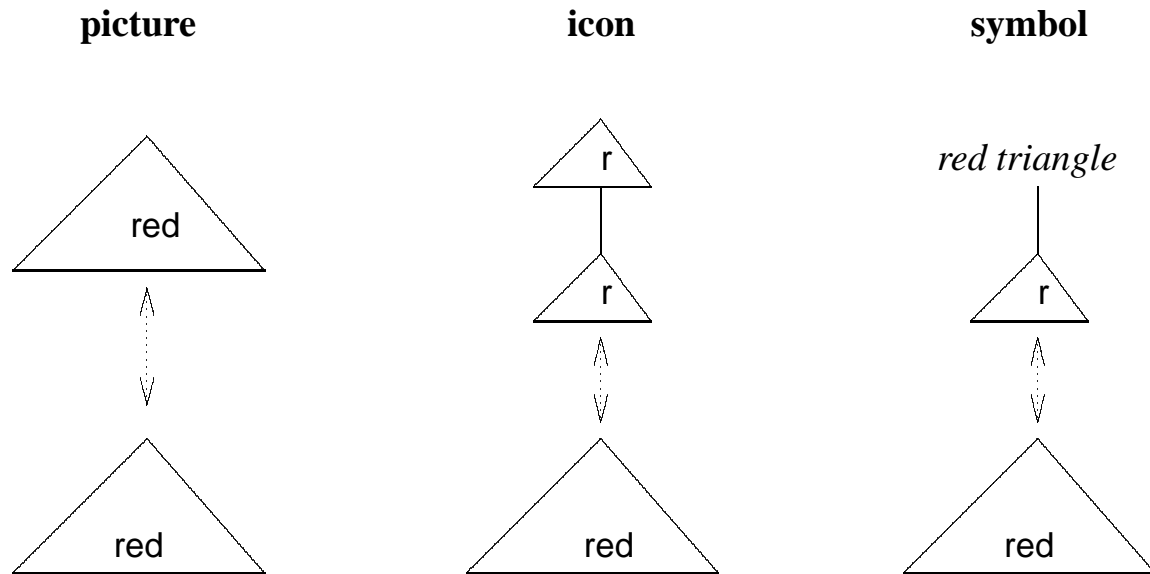


### 6.4.4 Comparing the structure of icon and name

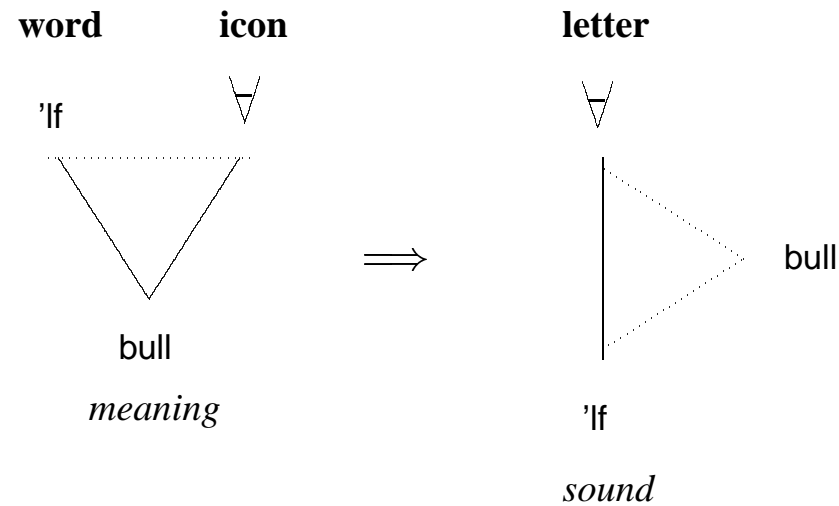


## 6.5 Pictures, pictograms, and letters

### 6.5.1 Transition from picture to icon to symbol



## 6.5.2 Rotation principle underlying transition from icon to letter



## 6.5.3 Aristotle on writing

Spoken words are the signs of mental experience and written words are the signs of spoken words.

DE INTERPRETATIONE, 1

**Part II.**  
**Theory of Grammar**

## 7. Generative grammar

### 7.1 Language as a subset of the free monoid

#### 7.1.1 Definition of language

A language is a set of word sequences.

#### 7.1.2 Illustration of the free monoids over $LX = \{a,b\}$

$\varepsilon$

a, b

aa, ab, ba, bb

aaa, aab, aba, abb, baa, bab, bba, bbb

aaaa, aaab, aaba, aabb, abaa, abab, abba, abbb, ...

...

#### 7.1.3 Informal description of the artificial language $a^k b^k$ (with $k \geq 1$ )

Its wellformed expressions consist of an arbitrary number of the word a followed by an equal number of the word b.

### 7.1.4 Wellformed expressions of $a^k b^k$

a b, a a b b, a a a b b b, a a a a b b b b, etc.,

### 7.1.5 Illformed expressions of $a^k b^k$

a, b, b a, b b a a, a b a b, etc.,

### 7.1.6 PS-grammar for $a^k b^k$

$$S \rightarrow a S b$$

$$S \rightarrow a b$$

A formal grammar may be viewed as a filter which selects the wellformed expressions of its language from the free monoid over the language's lexicon.

### 7.1.7 Elementary formalisms of generative grammar

1. Categorical or C-grammar
2. Phrase-structure or PS-grammar
3. Left-associative or LA-grammar

### **7.1.8 Algebraic definition**

The algebraic definition of a generative grammar explicitly enumerates the basic components of the system, defining them and the structural relations between them using only notions of set theory.

### **7.1.9 Derived formalisms of PS-grammar**

Syntactic Structures, Generative Semantics, Standard Theory (ST), Extended Standard Theory (EST), Revised Extended Standard Theory (REST), Government and Binding (GB), Barriers, Generalized Phrase Structure Grammar (GPSG), Lexical Functional Grammar (LFG), Head-driven Phrase Structure Grammar (HPSG)

### **7.1.10 Derived formalisms of C-grammar**

Montague grammar (MG), Functional Unification Grammar (FUG), Categorical Unification Grammar (CUG), Combinatory Categorical Grammar (CCG), Unification-based Categorical Grammar (UCG)

### **7.1.11 Examples of semi-formal grammars**

Dependency grammar (Tesnière 1959), systemic grammar (Halliday 1985), stratification grammar (Lamb ??)

## **7.2 Methodological reasons for generative grammar**

### **7.2.1 Grammatically well-formed expression**

the little dogs have slept earlier

### **7.2.2 Grammatically ill-formed expression**

\* earlier slept have dogs little the



### 7.2.3 Methodological consequences of generative grammar

- *Empirical*: formation of explicit hypotheses

A formal rule system constitutes an explicit hypothesis about which input expressions are well-formed and which are not. This is an essential precondition for incremental improvements of the empirical description.

- *Mathematical*: determining formal properties

A formal rule system is required for determining mathematical properties such as decidability, complexity, and generative capacity. These in turn determine whether the formalism is suitable for empirical description and computational realization.

- *Computational*: declarative specification for parsers

A formal rule system may be used as a declarative specification of the parser, characterizing its necessary properties in contrast to accidental properties stemming from the choice of the programming environment, etc. A parser in turn provides the automatic language analysis needed for the verification of the individual grammars.

## 7.3 Adequacy of generative grammars

### 7.3.1 Desiderata of generative grammar for natural language

The generative analysis of natural language should be simultaneously

- defined *mathematically* as a formal theory of low complexity,
- designed *functionally* as a component of natural communication, and
- realized *methodologically* as an efficiently implemented computer program in which the properties of formal language theory and of natural language analysis are represented in a modular and transparent manner.



### 7.4.3 Algebraic definition of C-grammar

A C-grammar is a quintuple  $\langle W, C, LX, R, CE \rangle$ .

1.  $W$  is a finite set of word form surfaces.
2.  $C$  is a set of categories such that
  - (a) *basis*  
 $u$  and  $v \in C$ ,
  - (b) *induction*  
 if  $X$  and  $Y \in C$ , then also  $(X/Y)$  and  $(X \setminus Y) \in C$ ,
  - (c) *closure*  
 Nothing is in  $C$  except as specified in (a) and (b).
3.  $LX$  is a finite set such that  $LX \subset (W \times C)$ .
4.  $R$  is a set comprising the following two rule schemata:
 
$$\alpha_{(Y/X)} \circ \beta_{(Y)} \Rightarrow \alpha\beta_{(X)}$$

$$\beta_{(Y)} \circ \alpha_{(Y \setminus X)} \Rightarrow \beta\alpha_{(X)}$$
5.  $CE$  is a set comprising the categories of *complete expressions*, with  $CE \subseteq C$ .

#### 7.4.4 Recursive definition of the infinite set $C$

Because the start elements  $u$  and  $v$  are in  $C$  so are  $(u/v)$ ,  $(v/u)$ ,  $(u \setminus v)$ , and  $(v \setminus u)$  according to the induction clause. This means in turn that also  $((u/v)/v)$ ,  $((u/v) \setminus u)$ ,  $(u/(u/v))$ ,  $(v/(u/v))$ , etc., belong to  $C$ .

#### 7.4.5 Definition of $LX$ as finite set of ordered pairs

Each ordered pair is built from (i) an element of  $W$  and (ii) an element of  $C$ . Which surfaces (i.e. elements of  $W$ ) take which elements of  $C$  as their categories is specified in  $LX$  by explicitly listing the ordered pairs.

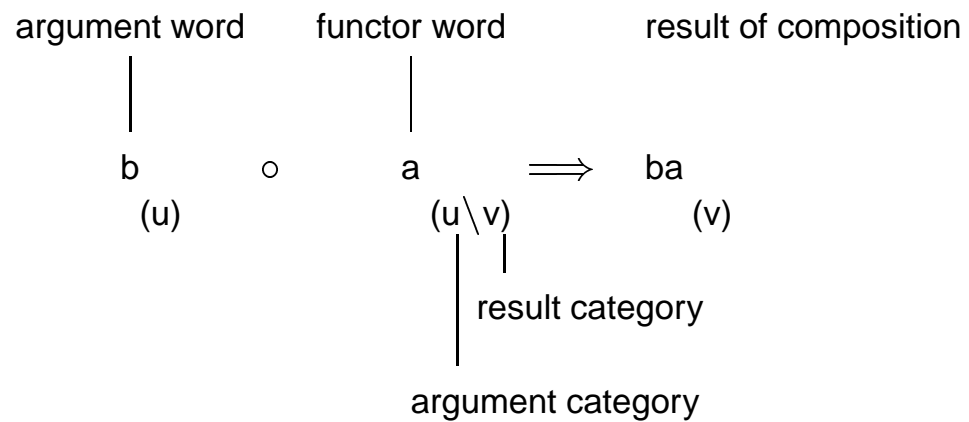
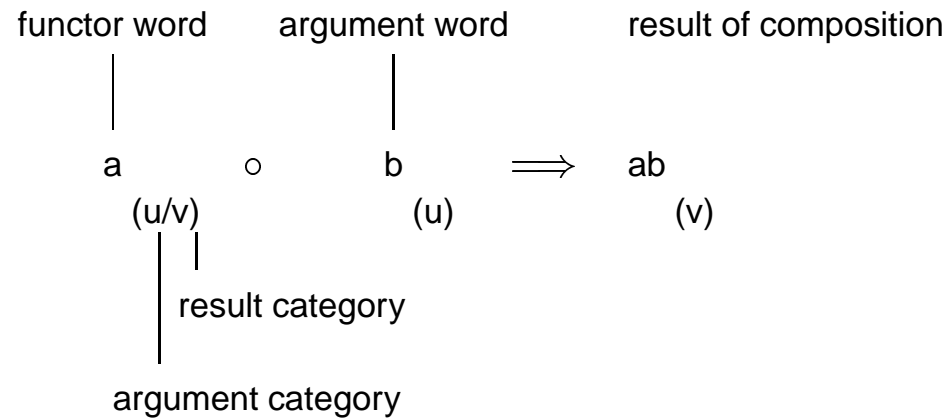
#### 7.4.6 Definition of the set of rule schemata $R$

The rule schemata use the variables  $\alpha$  and  $\beta$  to represent the surfaces of the functor and the argument, respectively, and the variables  $X$  and  $Y$  to represent their category patterns.

#### 7.4.7 Definition of the set of complete expressions $CE$

Depending on the specific  $C$ -grammar and the specific language, this set may be finite and specified in terms of an explicit listing, or it may be infinite and characterized by patterns containing variables.

### 7.4.8 Implicit pattern matching in combinations of bidirectional C-grammar



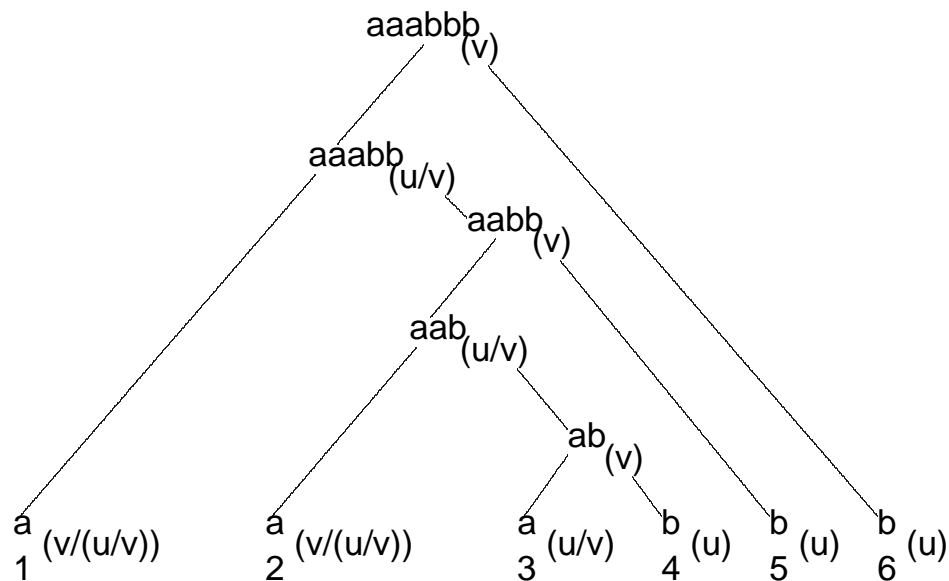
### 7.4.9 C-grammar for $a^k b^k$

$$LX =_{def} \{a_{(u/v)}, b_{(u)}, a_{(v/(u/v))}\}$$

$$CE =_{def} \{(v)\}$$

The word  $a$  has two lexical definitions with the categories  $(u/v)$  and  $(v/(u/v))$ , respectively, for reasons apparent in the following derivation tree.

#### 7.4.10 Example of $a^k b^k$ derivation, for $k = 3$



## 7.5 C-grammar for natural language

### 7.5.1 C-grammar for a tiny fragment of English

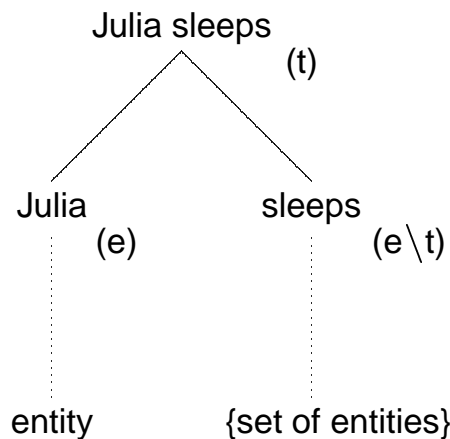
$LX =_{def} \{W_{(e)} \cup W_{(e \setminus t)}\}$ , where

$W_{(e)} = \{\text{Julia, Peter, Mary, Fritz, Suzy} \dots\}$

$W_{(e \setminus t)} = \{\text{sleeps, laughs, sings} \dots\}$

$CE =_{def} \{(t)\}$

### 7.5.2 Simultaneous syntactic and semantic analysis

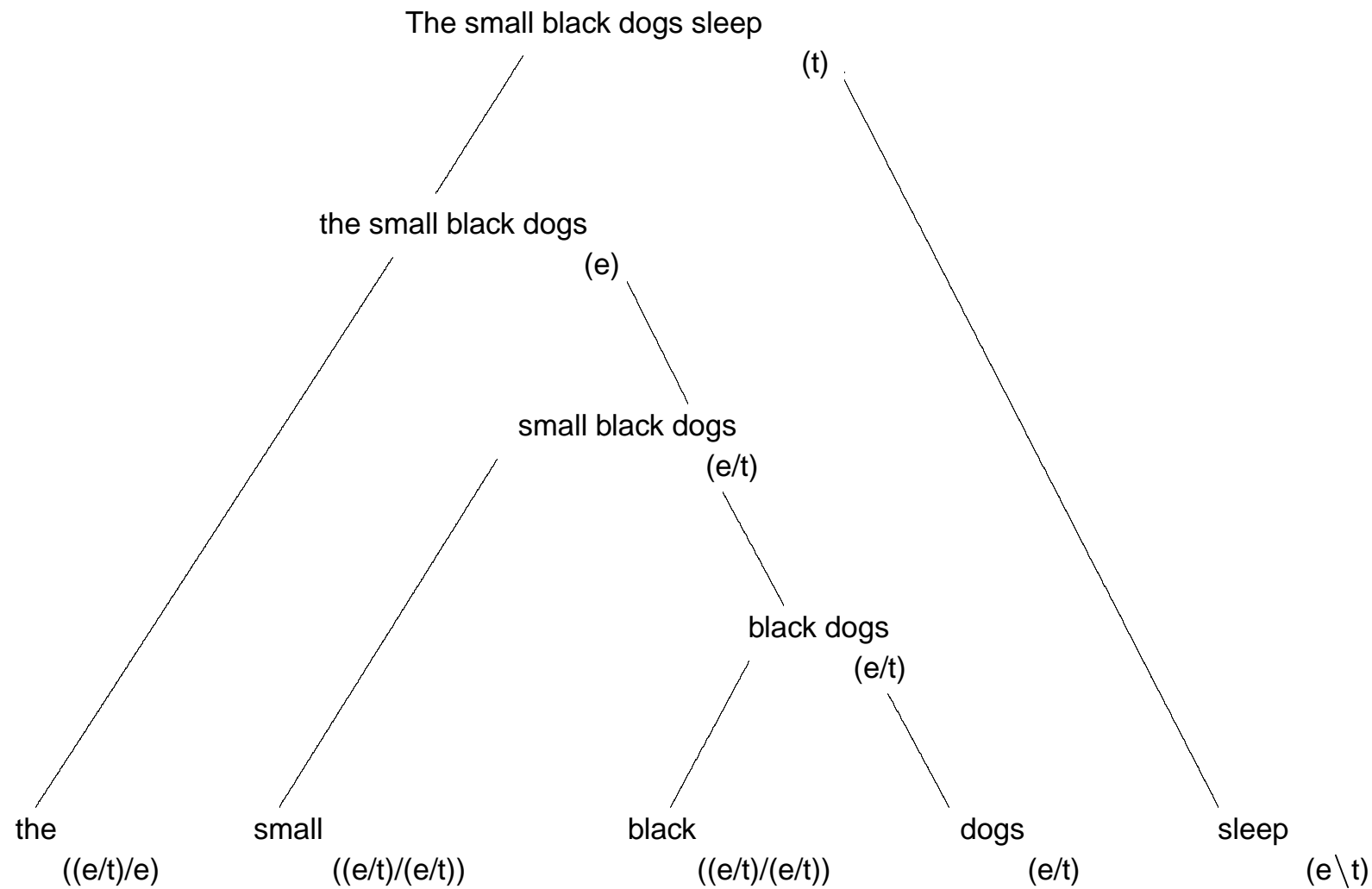


Denotations (in the model  $\mathcal{M}$ ):

entity              {set of entities}



### 7.5.3 C-analysis of a natural language sentence



### 7.5.4 C-grammar for example 7.5.3

$LX =_{def} \{ W_{(e)} \cup W_{(e \setminus t)} \cup W_{(e/t)} \cup W_{((e/t)/(e/t))} \cup W_{((e/t)/t)} \}$ , where

$W_{(e)} = \{ \text{Julia, Peter, Mary, Fritz, Suzy} \dots \}$

$W_{(e \setminus t)} = \{ \text{sleeps, laughs, sings} \dots \}$

$W_{(e/t)} = \{ \text{dog, dogs, cat, cats, table, tables} \dots \}$

$W_{((e/t)/(e/t))} = \{ \text{small, black} \dots \}$

$W_{((e/t)/t)} = \{ \text{a, the, every} \dots \}$

$CE =_{def} \{ (t) \}$

### 7.5.5 Empirical disadvantages of C-grammar for natural language

- Deriving expressions relative to a C-grammar has the character of problem solving.
- The handling of alternative word orders and agreement phenomena requires an extremely high degree of lexical ambiguities.

## 8. Language hierarchies and complexity

### 8.1 Formalism of PS-grammar

#### 8.1.1 Original definition

Published in 1936 by the American logician E. Post as *rewrite* or *Post production systems*, it originated in recursion theory and is closely related to automata theory.

#### 8.1.2 First application to natural language

Post's rewrite systems were first applied to natural language by N. Chomsky 1957 under the name of *phrase structure grammar*.

#### 8.1.3 Algebraic definition of PS-Grammar

A PS-grammar is a quadruple  $\langle V, V_T, S, P \rangle$  such that

1.  $V$  is a finite set of signs,
2.  $V_T$  is a proper subset of  $V$ , called *terminal symbols*,
3.  $S$  is a sign in  $V$  minus  $V_T$ , called *start symbol*, and
4.  $P$  is a set of rewrite rules of the form  $\alpha \rightarrow \beta$ , where  $\alpha$  is an element of  $V^+$  and  $\beta$  an element of  $V^*$ .

### 8.1.4 Restrictions of PS-rule schemata

#### 0. Unrestricted PS-rules:

The left hand side and the right hand side of a type 0 rule each consist of arbitrary sequences of terminal and nonterminal symbols.

#### 1. Context-sensitive PS-rules:

The left hand side and the right hand side of a type 1 rule each consist of arbitrary sequences of terminal and nonterminal symbols whereby the right hand side must be at least as long as the left hand side.

Example:  $A B C \rightarrow A D E C$

#### 2. Context-free PS-rules:

The left hand side of a type 2 rule consists of exactly one variable. The right hand side of the rule consists of a sequence from  $V^+$ .

Examples:  $A \rightarrow BC$ ,  $A \rightarrow bBCc$ , etc.

#### 3. Regular PS-rules:

The left hand side of a type 3 rule consists of exactly one variable. The right hand side consists of exactly one terminal symbol and at most one variable.

Examples:  $A \rightarrow b$ ,  $A \rightarrow bC$ .

## 8.2 Language classes and computational complexity

### 8.2.1 Different restrictions on a generative rule schema result in

- different *types of grammar* which have
- different *degrees of generative capacity* and generate
- different *language classes* which in turn exhibit
- different *degrees of computational complexity*.

### 8.2.2 Basic degrees of complexity

1. *Linear complexity*  
 $n, 2n, 3n$ , etc.
2. *Polynomial complexity*  
 $n^2, n^3, n^4$ , etc.
3. *Exponential complexity*  
 $2^n, 3^n, 4^n$ , etc.
4. *Undecidable*  
 $n \cdot \infty$

### 8.2.3 Polynomial vs. exponential complexity (M.R.Garey & D.S. Johnson 1979)

	problem size n		
time complexity	10	50	100
$n^3$	.001 seconds	.125 seconds	1.0 seconds
$2^n$	.001 seconds	35.7 years	$10^{15}$ centuries

### 8.2.4 Application to natural language

The Limas corpus comprises a total of 71 148 sentences. Of these, there are exactly 50 which consist of 100 word forms or more, whereby the longest sentence in the whole corpus consists of 165 words.

## 8.2.5 PS-grammar hierarchy of formal languages (Chomsky hierarchy)

rule restrictions	types of PS-grammar	language classes	degree of complexity
type 3	regular PSG	regular languages	linear
type 2	context-free PSG	context-free languages	polynomial
type 1	context-sensitive PSG	context-sensitive lang.	exponential
type 0	unrestricted PSG	rec. enum. languages	undecidable

## 8.3 Generative capacity and formal language classes

### 8.3.1 Essential linguistic question regarding PS-grammar

Is there is a type of PS-grammar which generates exactly those structures which are characteristic of natural language?

### 8.3.2 Structural properties of regular PS-grammars

The generative capacity of regular grammar permits the recursive repetition of single words, but without any recursive correspondences.

### 8.3.3 Regular PS-grammar for $ab^k$ ( $k \geq 1$ )

$$V =_{def} \{S, B, a, b\}$$

$$V_T =_{def} \{a, b\}$$

$$P =_{def} \{S \rightarrow a B, \\ B \rightarrow b B, \\ B \rightarrow b \}$$



### 8.3.4 Regular PS-grammar for $\{a, b\}^+$

$$V =_{def} \{S, a, b\}$$

$$V_T =_{def} \{a, b\}$$

$$P =_{def} \{S \rightarrow a S, \\ S \rightarrow b S, \\ S \rightarrow a, \\ S \rightarrow b\}$$

### 8.3.5 Regular PS-grammar for $a^m b^k$ ( $k, m \geq 1$ )

$$V =_{def} \{S, S_1, S_2, a, b\}$$

$$V_T =_{def} \{a, b\}$$

$$P =_{def} \{S \rightarrow a S_1, \\ S_1 \rightarrow a S_1, \\ S_1 \rightarrow b S_2, \\ S_2 \rightarrow b\}$$

### 8.3.6 Structural properties of context-free PS-grammars

The generative capacity of context-free grammar permits the recursive generation of pairwise inverse correspondences, e.g.  $a b c \dots c b a$ .

### 8.3.7 Context-free PS-grammar for $a^k b^{3k}$

$$V =_{def} \{S, a, b\}$$

$$V_T =_{def} \{a, b\}$$

$$P =_{def} \{ S \rightarrow a S b b b, \\ S \rightarrow a b b b \}$$

### 8.3.8 Context-free PS-grammar for $WW^R$

$$V =_{def} \{S, a, b, c, d\}, V_T =_{def} \{a, b, c, d\}, P =_{def} \{ S \rightarrow a S a, \\ S \rightarrow b S b, \\ S \rightarrow c S c, \\ S \rightarrow d S d, \\ S \rightarrow a a, \\ S \rightarrow b b, \\ S \rightarrow c c, \\ S \rightarrow d d \}$$

### 8.3.9 Why $WW$ exceeds the generative capacity of context-free PS-grammar

aa  
abab  
abcabc  
abcdabcd  
...

do not have a reverse structure. Thus, despite the close resemblance between  $WW^R$  and  $WW$ , it is simply impossible to write a PS-grammar like 8.3.8 for  $WW$ .

### 8.3.10 Why $a^k b^k c^k$ exceeds the generative capacity of context-free PS-grammar

a b c  
a a b b c c  
a a a b b b c c c  
...

cannot be generated by a context-free PS-grammar because it requires a correspondence between three different parts – which exceeds the *pairwise* reverse structure of the context-free languages such as the familiar  $a^k b^k$  and  $WW^R$ .

### 8.3.11 Structural properties of context-sensitive PS-grammars

Almost any language one can think of is context-sensitive; the only known proofs that certain languages are not CSL's are ultimately based on diagonalization.

J.E. Hopcroft and J.D. Ullman 1979, p. 224

### 8.3.12 PS-grammar for context-sensitive $a^k b^k c^k$

$$V =_{def} \{S, B, C, D_1, D_2, a, b, c\}$$

$$V_T =_{def} \{a, b, c\}$$

$$P =_{def} \left\{ \begin{array}{ll} S \rightarrow a S B C, & \text{rule 1} \\ S \rightarrow a b C, & \text{rule 2} \\ C B \rightarrow D_1 B, & \text{rule 3a} \\ D_1 B \rightarrow D_1 D_2, & \text{rule 3b} \\ D_1 D_2 \rightarrow B D_2, & \text{rule 3c} \\ B D_2 \rightarrow B C, & \text{rule 3d} \\ b B \rightarrow b b, & \text{rule 4} \\ b C \rightarrow b c, & \text{rule 5} \\ c C \rightarrow c c \} & \text{rule 6} \end{array} \right.$$

The rules 3a–3d jointly have the same effect as the (monotonic)

*rule 3*      $C B \rightarrow B C$ .

### 8.3.13 Derivation of a a a b b b c c c

	intermediate chains	rules
1.	S	
2.	a S B C	(1)
3.	a a S B C B C	(1)
4.	a a a b C B C B C	(2)
5.	a a a b B C C B C	(3)
6.	a a a b B C B C C	(3)
7.	a a a b B B C C C	(3)
8.	a a a b b B C C C	(4)
9.	a a a b b b C C C	(4)
10.	a a a b b b c C C	(5)
11.	a a a b b b c c C	(6)
12.	a a a b b b c c c	(6)

### 8.3.14 Structural properties of recursive languages

The context-sensitive languages are a proper subset of the recursive languages. The class of recursive languages is not reflected in the PS-grammar hierarchy because the PS-rule schema provides no suitable restriction (cf. 8.1.4) such that the associated PS-grammar class would generate exactly the recursive languages.

A language is recursive if and only if it is decidable, i.e., if there exists an algorithm which can determine in finitely many steps for arbitrary input whether or not the input belongs to the language. An example of a recursive language which is not context-sensitive is the Ackermann function.

### 8.3.15 Structural properties of unrestricted PS-grammars

Because the right hand side of a rule may be shorter than the left hand side, a type 0 rules provides for the possibility of *deleting* parts of sequences already generated. For this reason, the class of recursively enumerable languages is undecidable.

## 8.4 PS-Grammar for natural language

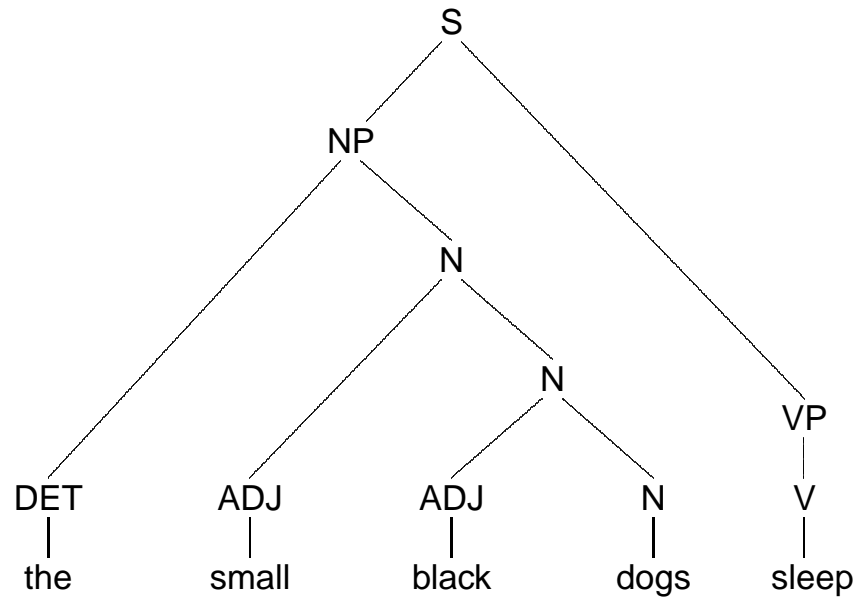
### 8.4.1 PS-grammar for example 7.5.4

$V =_{def} \{S, NP, VP, V, N, DET, ADJ, \text{black}, \text{dogs}, \text{little}, \text{sleep}, \text{the}\}$

$V_T =_{def} \{\text{black}, \text{dogs}, \text{little}, \text{sleep}, \text{the}\}$

$P =_{def} \{$   
     $S \rightarrow NP VP,$   
     $VP \rightarrow V,$   
     $NP \rightarrow DET N,$   
     $N \rightarrow ADJ N,$   
     $N \rightarrow \text{dogs},$   
     $ADJ \rightarrow \text{little},$   
     $ADJ \rightarrow \text{black},$   
     $DET \rightarrow \text{the},$   
     $V \rightarrow \text{sleep}\}$

## 8.4.2 PS-grammar analysis of example 7.5.4

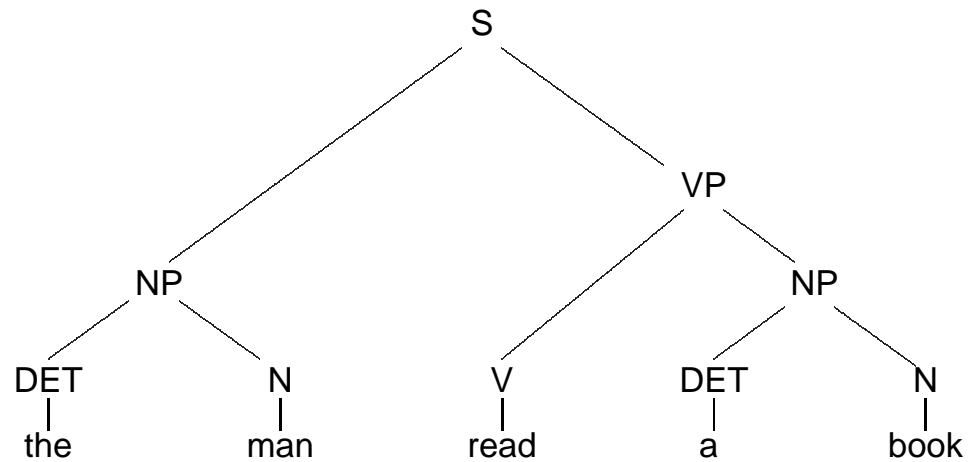


## 8.4.3 Definition of constituent structure

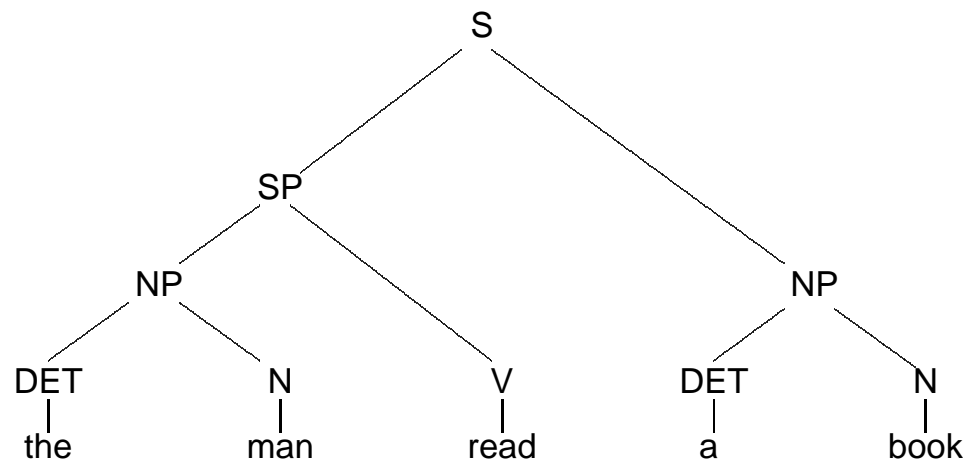
1. Words or constituents which belong together semantically must be dominated directly and exhaustively by a node.
2. The lines of a constituent structure may not cross (*nontangling condition*).



### 8.4.4 Correct constituent structure analysis



### 8.4.5 Incorrect constituent structure analysis

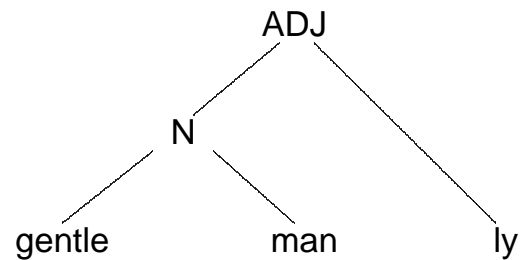


## 8.4.6 Origin of constituent structure

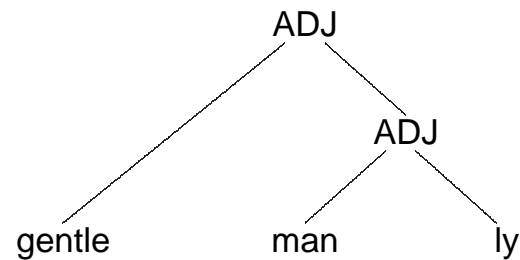
Historically, the notion of constituent structure evolved from the *immediate constituent analysis* of the American structuralist L. BLOOMFIELD (1887–1949) and the distribution tests of his student Z. Harris.

## 8.4.7 Immediate constituents in PS-grammar:

*correct:*



*incorrect:*



### 8.4.8 Substitution test

*correct substitution:*

Suzanne has [eaten] an apple



Suzanne has [cooked] an apple

*incorrect substitution:*

Suzanne has [eaten] an apple



\* Suzanne has [desk] an apple

### 8.4.9 Movement test

*correct movement:*

Suzanne [has] eaten an apple



[has] Suzanne eaten an apple (?)

*incorrect movement:*

Suzanne has eaten [an] apple



\* [an] Suzanne has eaten apple

### 8.4.10 Purpose of constituent structure

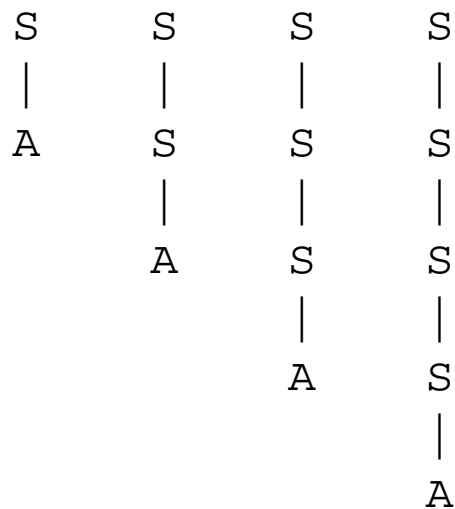
The distribution tests seemed important methodologically in order to support intuitions about the *correct segmentation* of sentences. The distinction between linguistically correct and incorrect phrase structures trees seemed necessary because for any finite string the number of possible phrase structures is infinite.

### 8.4.11 Infinite number of trees over a single word

Context-free rules:  $S \rightarrow S, S \rightarrow A$

Indexed bracketing:  $(A)_S, ((A)_S)_S, (((A)_S)_S)_S, (((((A)_S)_S)_S)_S)_S, \text{ etc.}$

Corresponding trees:

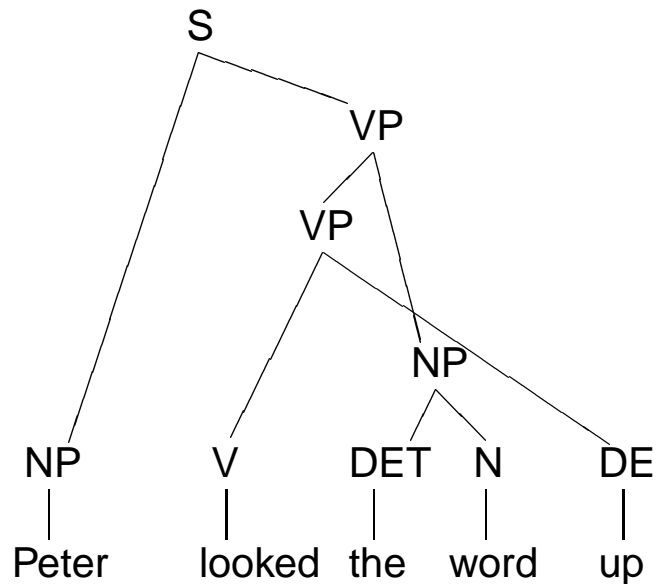


## 8.5 Constituent structure paradox

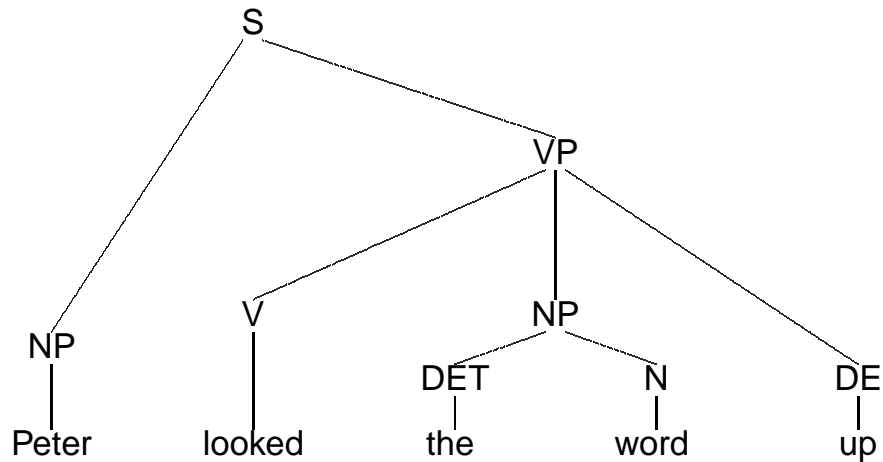
### 8.5.1 Constituent structure from the viewpoint of the SLIM theory of language

- Constituent structure and the distribution tests claimed to support it run counter to the time-linear structure of natural language.
- The resulting phrase structure trees have no communicative purpose.
- The principles of constituent structure cannot always be fulfilled.

### 8.5.2 Violating the second condition of 8.4.3



### 8.5.3 Violating the first condition of 8.4.3



### 8.5.4 Assumptions of transformational grammar

In order to maintain constituent structure as innate, transformational grammar distinguishes between a hypothetical deep structures claimed to be universal and the concrete language dependent surface structure.

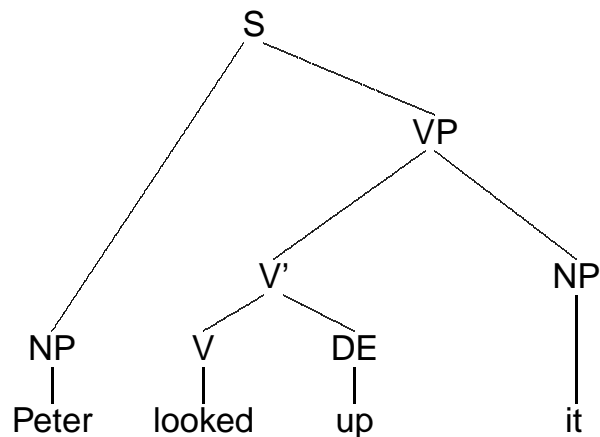
- Thereby the two levels are assumed to be semantically equivalent,
- deep structures need not be grammatical, but must obey constituent structure, and
- surface structures must be grammatical, but need not obey constituent structure.

### 8.5.5 Example of a formal transformation

$$[[V \text{ DE}]_{V'} [\text{DET N}]_{NP}]_{VP} \Rightarrow [V [\text{DET N}]_{NP} \text{ DE}]_{VP}$$

### 8.5.6 Applying transformation 8.5.3

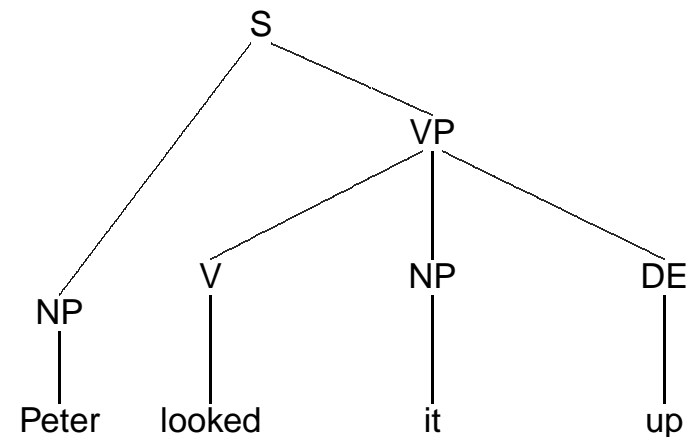
*deep structure:*



*transformation*

$\Rightarrow$

*surface structure:*



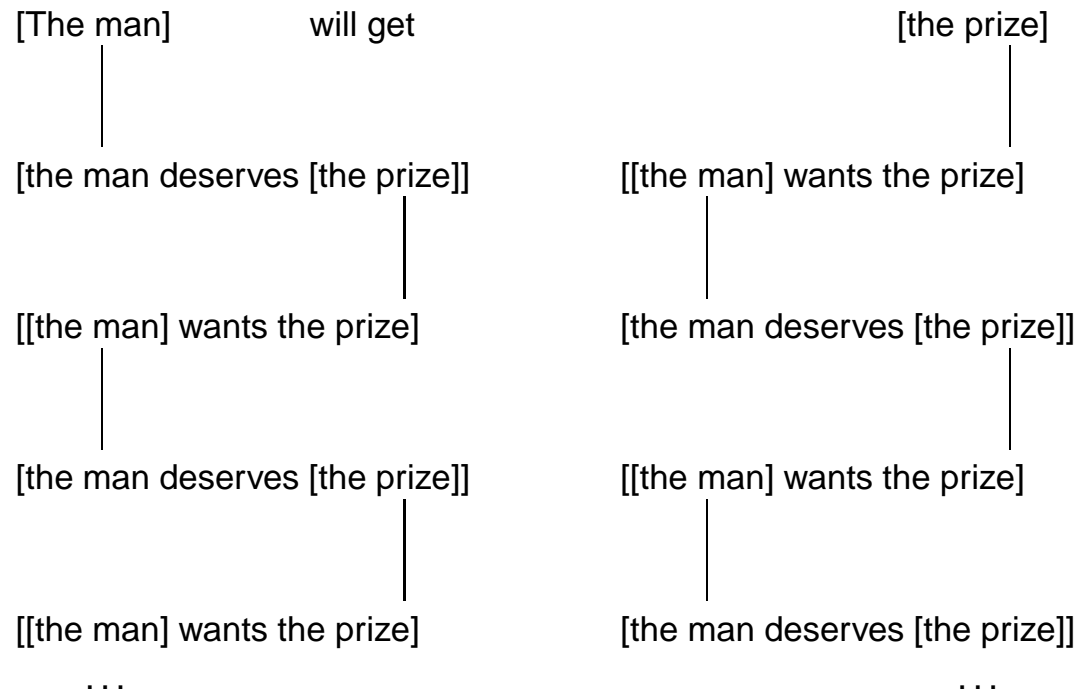
### 8.5.7 Mathematical consequences of adding transformations to PS-grammar

While the context-free deep structure is of low polynomial complexity ( $n^3$ ), adding transformations raises complexity to recursively enumerable. In other words, transformational grammar is undecidable.

## 8.5.8 Example of a Bach-Peters-sentence

The man who deserves it will get the prize he wants.

## 8.5.9 Deep structure of a Bach-Peters-sentence





## 9. Basic notions of parsing

### 9.1 Declarative and procedural aspects of parsing

#### 9.1.1 Declarative & procedural aspects in linguistics

- The *declarative* aspect of computational language analysis is represented by a generative grammar, written for the specific language to be analyzed within a general, mathematically well-defined formalism.
- The *procedural* aspect of computational language analysis comprises those parts of the computer program which interpret and apply the general formalism in the automatic analysis of language input.

#### 9.1.2 Example

rule 1:  $A \rightarrow B C$

rule 2:  $B \rightarrow c d$

rule 3:  $C \rightarrow e f$

## 9.2 Fitting grammar onto language

### 9.2.1 Context-free structure in German

Der Mann, ( <i>the man</i> )					schläft. ( <i>sleeps</i> ).
	der die Frau, ( <i>who the woman</i> )				liebt, ( <i>loves</i> )
		die das Kind, ( <i>who the child</i> )			sieht, ( <i>sees</i> )
			das die Katze	füttert, ( <i>who the cat</i> )	( <i>feeds</i> )

### 9.2.2 Alternative implications of natural language not being context-free

1. PS-grammar is the only elementary formalism of generative grammar, for which reason one must accept that the natural languages are of high complexity and thus computationally intractable.
2. PS-grammar is not the only elementary formalism of generative grammar. Instead, there are other elementary formalisms which define other language hierarchies whose language classes are orthogonal to those of PS-grammar.

### 9.2.3 Possible relations between two grammar formalisms

- *No equivalence*

Two grammar formalisms are not equivalent, if they generate/recognize different language classes; this means that the two formalisms are of different generative capacity.

- *Weak equivalence*

Two grammar formalisms are weakly equivalent, if they generate/recognize the same language classes; this means that the two formalisms have the same generative capacity.

- *Strong equivalence*

Two grammar formalisms are strongly equivalent, if they are (i) weakly equivalent, and moreover (ii) produce the same structural descriptions; this means that the two formalisms are no more than *notational variants*.

### 9.2.4 Weak equivalence between C-grammar and PS-grammar

The problem arose of determining the exact relationships between these types of [PS-]grammars and the categorial grammars. I surmised in 1958 that the BCGs [Bidirectional Categorial Grammar *à la* 7.4.1] were of approximately the same strength as [context-free phrase structure grammars]. A proof of their equivalence was found in June of 1959 by Gaifman. ... The equivalence of these different types of grammars should not be too surprising. Each of them was meant to be a precise explicatum of the notion *immediate constituent grammars* which has served for many years as the favorite type of American descriptive linguistics as exhibited, for instance, in the well-known books by Harris [1951] and Hockett [1958].

Y. Bar-Hillel 1960 [1964, p. 103]

## 9.2.5 General relations between notions of generative grammar

- *Languages* exist independently of generative grammars. A given language may be described by different formal grammars of different grammar formalisms.
- *Generative grammars* is (i) a general formal framework or (ii) a specific rule system defined for describing a specific language within the general framework.
- *Subtypes of generative grammars* result from different restrictions on the formal framework.

- *Language classes*

The subtypes of a generative grammar may be used to divide the set of possible languages into different language classes.

Nota bene: *languages* exist independently of the formal grammars which may generate them. The *language classes*, on the other hand, do not exist independently, but result from particular restrictions on particular grammar formalisms.

- *Parsers*

Parsers are programs of automatic language analysis which are defined for whole subtypes of generative grammars.

- *Complexity*

The complexity of a subtype of generative grammar is determined over the number of *primitive operations* needed by an equivalent abstract automaton or parsing program for analyzing expressions in the worst case.

## 9.3 Type transparency between grammar and parser

### 9.3.1 Natural view of a parser as the motor or driver of a grammar

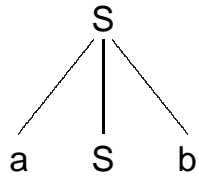
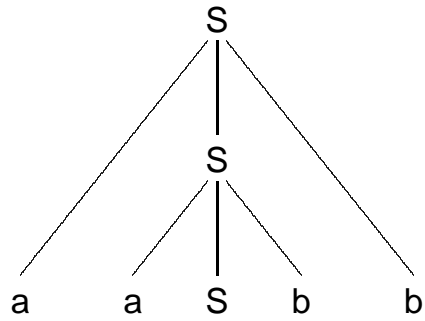
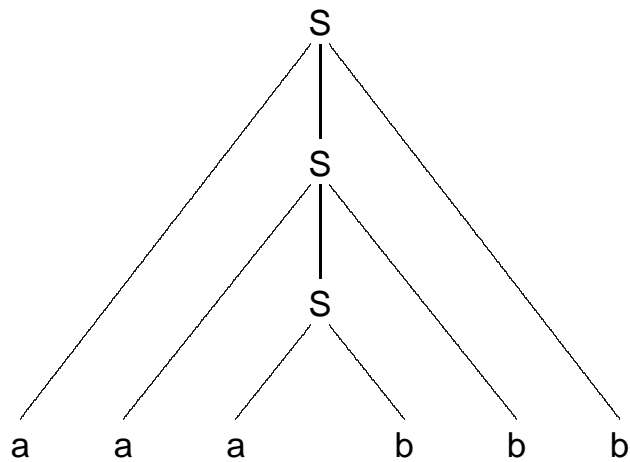
Miller and Chomsky's original (1963) suggestion is really that grammars be realized more or less directly as parsing algorithms. We might take this as a methodological principle. In this case we impose the condition that the logical organization of rules and structures incorporated in the grammar be mirrored rather exactly in the organization of the parsing mechanism. We will call this *type transparency*.

R.C. Berwick & A.S. Weinberg 1984, p. 39.

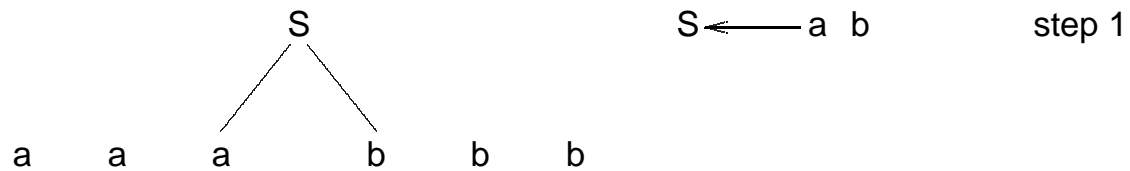
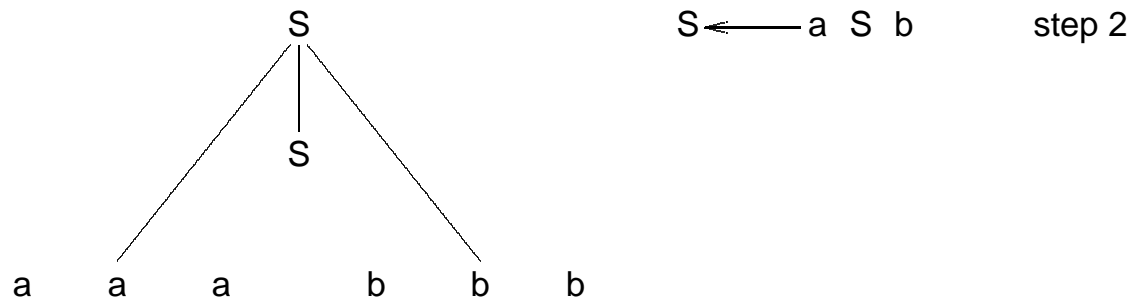
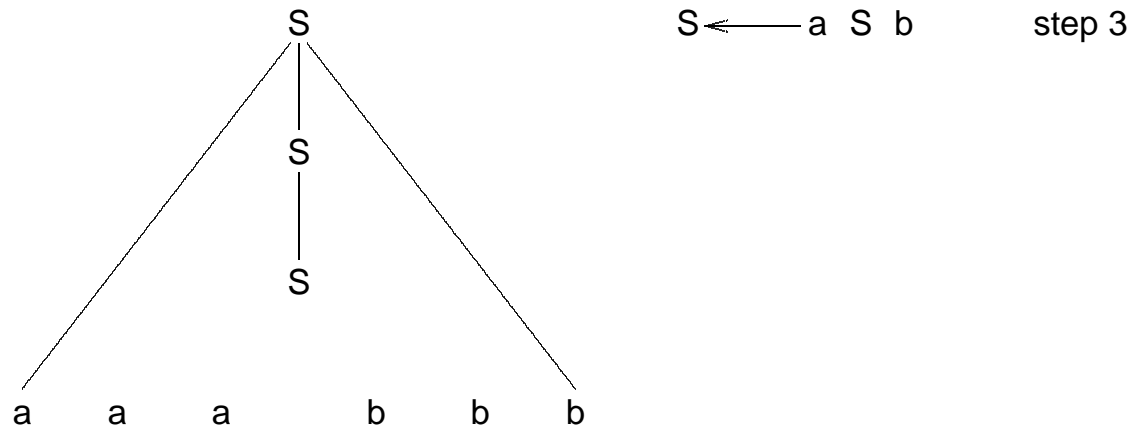
### 9.3.2 Definition of absolute type transparency

- For any given language, parser and generator use the *same* formal grammar,
- whereby the parser/generator applies the rules of the grammar *directly*.
- This means in particular that the parser/generator applies the rules in the *same order* as the grammatical derivation,
- that in each rule application the parser/generator takes the *same input* expressions as the grammar, and
- that in each rule application the parser/generator produces the *same output* expressions as the grammar.

### 9.3.3 Top-down derivation of a a a b b b


 $S \longrightarrow a S b$       step 1

 $S \longrightarrow a S b$       step 2

 $S \longrightarrow a b$       step 3

### 9.3.4 Bottom-up derivation of a a a b b b



### 9.3.5 The Earley algorithm analyzing $a^k b^k$

.aaabbb

.S

| a.aabbb

.ab -> a.b

.aSb -> a.Sb

| aa.aabbb

a.abbb -> aa.bb

a.aSbbb -> aa.Sbbb

| aaa.bbb      aaab.bb

aa.aabbb -> aaa.bbb -> aaab.bb -> ...

aa.aSbbb -> aaa.Sbbb

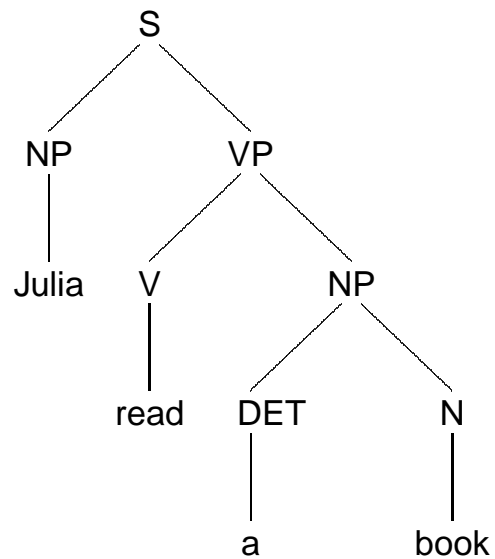


## 9.4 Input-output equivalence with the speaker-hearer

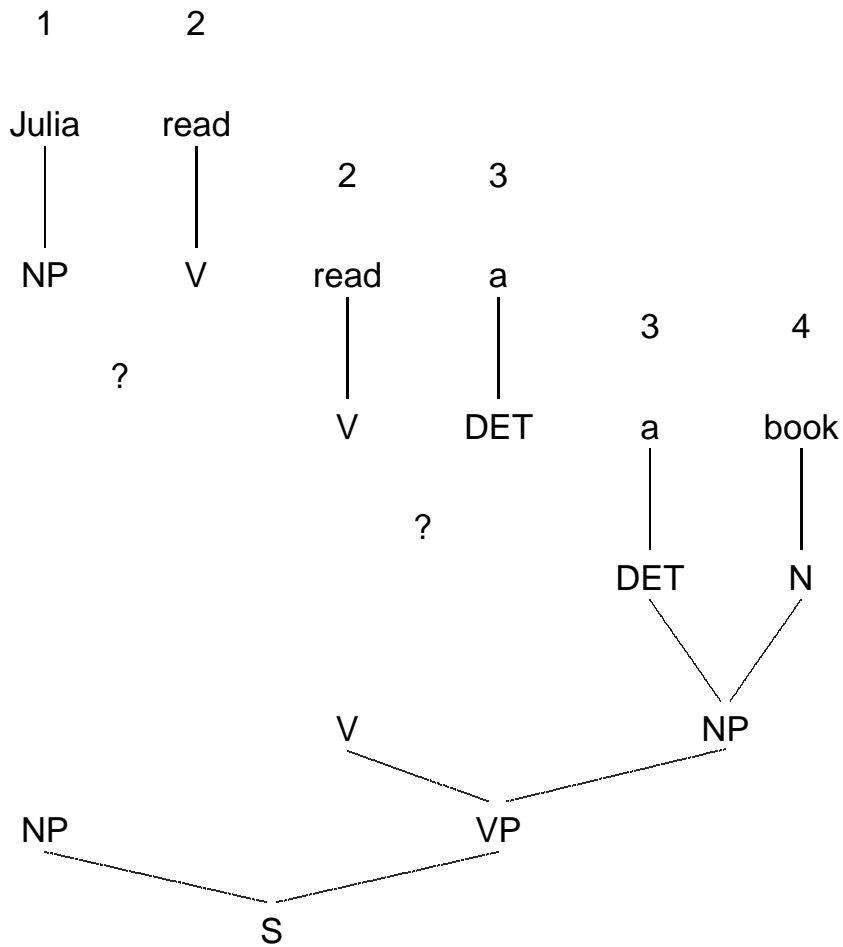
### 9.4.1 Context-free PS-grammar for a simple sentence of English

- |       |         |        |        |
|-------|---------|--------|--------|
| 1. S  | → NP VP | 5. V   | → read |
| 2. NP | → DET N | 6. DET | → a    |
| 3. VP | → V NP  | 7. N   | → book |
| 4. NP | → Julia |        |        |

### 9.4.2 PS-grammar analysis (*top-down derivation*)



### 9.4.3 Attempt of a time-linear analysis in PS-grammar



## 9.5 Desiderata of grammar for achieving convergence

### 9.5.1 Symptoms of lacking convergence in nativism

- Development of ever new derived systems instead of consolidation.
- Additional mechanisms regarded as descriptively necessary have consistently degraded mathematical and computational properties.
- Empirical work has lead continuously to problems of the type descriptive aporia and embarrassment of riches.
- Practical systems of natural language processing pay either only lip service to the theoretical constructs of nativism or ignore them altogether.

### 9.5.2 Reasons for lacking convergence of nativism

- Nativism is empirically underspecified because it does not include a functional theory of communication.
- The PS-grammar formalism adopted by nativism is incompatible with the input-output conditions of the speaker-hearer.

### 9.5.3 Properties of PS-grammar

- *Mathematical:*

Practical parsing algorithms exist only for context-free PS-grammar. It is of a sufficiently low complexity ( $n^3$ ), but not of sufficient generative capacity for natural language. Extensions of the generative capacity for the purpose of describing natural language turned out to be of such high complexity (undecidable or exponential) that no practical parse algorithm can exist for them.

- *Computational:*

PS-grammar is not type transparent. This prevents using the automatic traces of parsers for purposes of debugging and upscaling grammars. Furthermore, the indirect relation between the grammar and the parsing algorithm requires the use of costly routines and large intermediate structures.

- *Empirical:*

The substitution-based derivation order of PS-grammar is incompatible with the time-linear structure of natural language.

### 9.5.4 Desiderata of a generative grammar formalism

1. The grammar formalism should be mathematically well-defined and thus
2. permit an explicit, declarative description of artificial and natural languages.
3. The formalism should be recursive (and thus decidable) as well as
4. type transparent with respect to its parsers and generators.
5. The formalism should define a hierarchy of different language classes in terms of structurally obvious restrictions on its rule system (analogous – but orthogonal – to the PS-grammar hierarchy),
6. whereby the hierarchy contains a language class of low, preferably linear, complexity the generative capacity of which is sufficient for a complete description of natural language.
7. The formalism should be input-output equivalent with the speaker-hearer (and thus use a time-linear derivation order).
8. The formalism should be suited equally well for production (in the sense of mapping meanings into surfaces) and interpretation (in the sense of mapping surfaces into meanings).

## 10. Left-associative grammar (LAG)

### 10.1 Rule types and derivation order

#### 10.1.1 The notion *left-associative*

When we combine operators to form expressions, the order in which the operators are to be applied may not be obvious. For example,  $a + b + c$  can be interpreted as  $((a + b) + c)$  or as  $(a + (b + c))$ . We say that  $+$  is *left-associative* if operands are grouped left to right as in  $((a + b) + c)$ . We say it is *right-associative* if it groups operands in the opposite direction, as in  $(a + (b + c))$ .

A.V. Aho & J.D. Ullman 1977, p. 47

#### 10.1.2 Incremental left- and right-associative derivation

*left-associative:*

$$\begin{array}{l} a \\ (a + b) \\ ((a + b) + c) \\ (((a + b) + c) + d) \\ \dots \end{array} \quad \Longrightarrow$$

*right-associative:*

$$\begin{array}{l} a \\ (b + a) \\ (c + (b + a)) \\ (d + (c + (b + a))) \\ \dots \end{array} \quad \Longleftarrow$$

### 10.1.3 Left-associative derivation order

Derivation is based on the principle of possible *continuations*

Used to model the time-linear structure of language

### 10.1.4 Irregular bracketing structures corresponding to the trees of C- and PS-grammar

$(( (a + b) + (c + d) ) + e)$   
 $((a + b) + ((c + d) + e))$   
 $(a + ((b + c) + (d + e)))$   
 $((a + (b + c)) + (d + e))$   
 $(( (a + b) + c) + (d + e))$   
 ...

The number of these irregular bracketings grows exponentially with the length of the string and is infinite, if bracketings like (a), ((a)), (((a))), etc., are permitted.

### 10.1.5 Irregular bracketing structure

Derivation is based on the principle of possible *substitutions*

Used to model constituent structure

### 10.1.6 The principle of possible continuations

Beginning with the first word of the sentence, the grammar describes the possible continuations for each sentence start by specifying the rules which may perform the next grammatical composition (i.e., add the next word).

### 10.1.7 Schema of left-associative rule in LA-grammar

$$r_i: \text{cat}_1 \text{ cat}_2 \Rightarrow \text{cat}_3 \text{ rp}_i$$

### 10.1.8 Schema of a canceling rule in C-grammar

$$\alpha_{(Y|X)} \circ \beta_{(Y)} \Rightarrow \alpha\beta_{(X)}$$

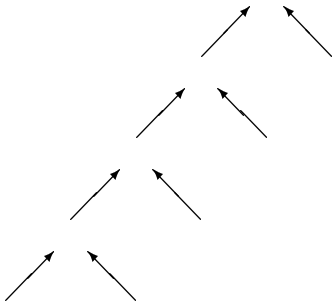
### 10.1.9 Schema of a rewrite rule in PS-grammar

$$A \rightarrow B C$$



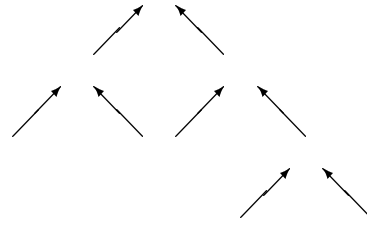
### 10.1.10 Three conceptual derivation orders

LA-grammar



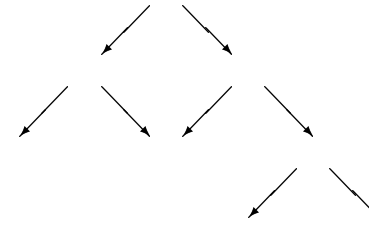
*bot.-up left-associative*

C-grammar



*bottom-up amalgamating*

PS-grammar



*top-down expanding*

## 10.2 Formalism of LA-grammar

### 10.2.1 Algebraic definition of LA-grammar

A left-associative grammar (or LA-grammar) is defined as a 7-tuple  $\langle W, C, LX, CO, RP, ST_S, ST_F \rangle$ , where

1.  $W$  is a finite set of *word surfaces*.
2.  $C$  is a finite set of *category segments*.
3.  $LX \subset (W \times C^+)$  is a finite set comprising the *lexicon*.
4.  $CO = (co_0 \dots co_{n-1})$  is a finite sequence of total recursive functions from  $(C^* \times C^+)$  into  $C^* \cup \{\perp\}$ , called *categorial operations*.
5.  $RP = (rp_0 \dots rp_{n-1})$  is an equally long sequence of subsets of  $n$ , called *rule packages*.
6.  $ST_S = \{(cat_s \ rp_s), \dots\}$  is a finite set of *initial states*, whereby each  $rp_s$  is a subset of  $n$  called start rule package and each  $cat_s \in C^+$ .
7.  $ST_F = \{(cat_f \ rp_f), \dots\}$  is a finite set of *final states*, whereby each  $cat_f \in C^*$  and each  $rp_f \in RP$ .

### 10.2.2 A concrete LA-grammar is specified by

1. a lexicon  $LX$  (cf. 3),
2. a set of initial states  $ST_S$  (cf. 6),
3. a sequence of rules  $r_i$ , each defined as an ordered pair  $(co_i, rp_i)$ , and
4. a set of final states  $ST_F$ .

### 10.2.3 LA-grammar for $a^k b^k$

$$LX =_{def} \{[a(a)], [b(b)]\}$$

$$ST_S =_{def} \{[(a) \{r_1, r_2\}]\}$$

$$r_1: (X) (a) \Rightarrow (aX) \{r_1, r_2\}$$

$$r_2: (aX) (b) \Rightarrow (X) \{r_2\}$$

$$ST_F =_{def} \{[\varepsilon rp_2]\}.$$

### 10.2.4 LA-grammar for $a^k b^k c^k$

$LX =_{def} \{[a(a)], [b(b)], [c(c)]\}$

$ST_S =_{def} \{[(a) \{r_1, r_2\}]\}$

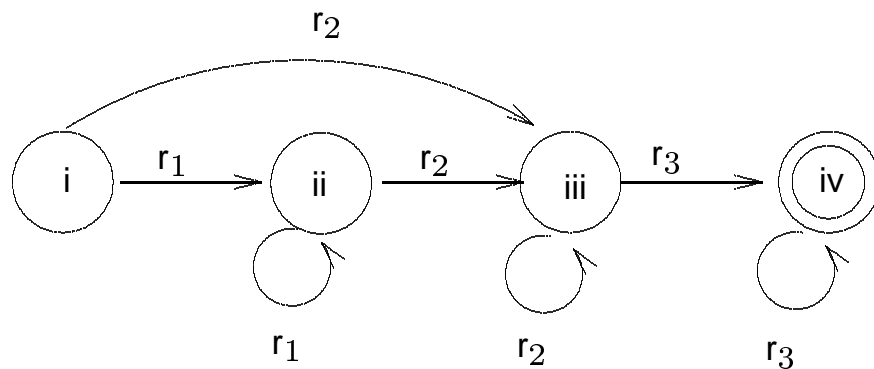
$r_1: (X) (a) \Rightarrow (aX) \{r_1, r_2\}$

$r_2: (aX) (b) \Rightarrow (Xb) \{r_2, r_3\}$

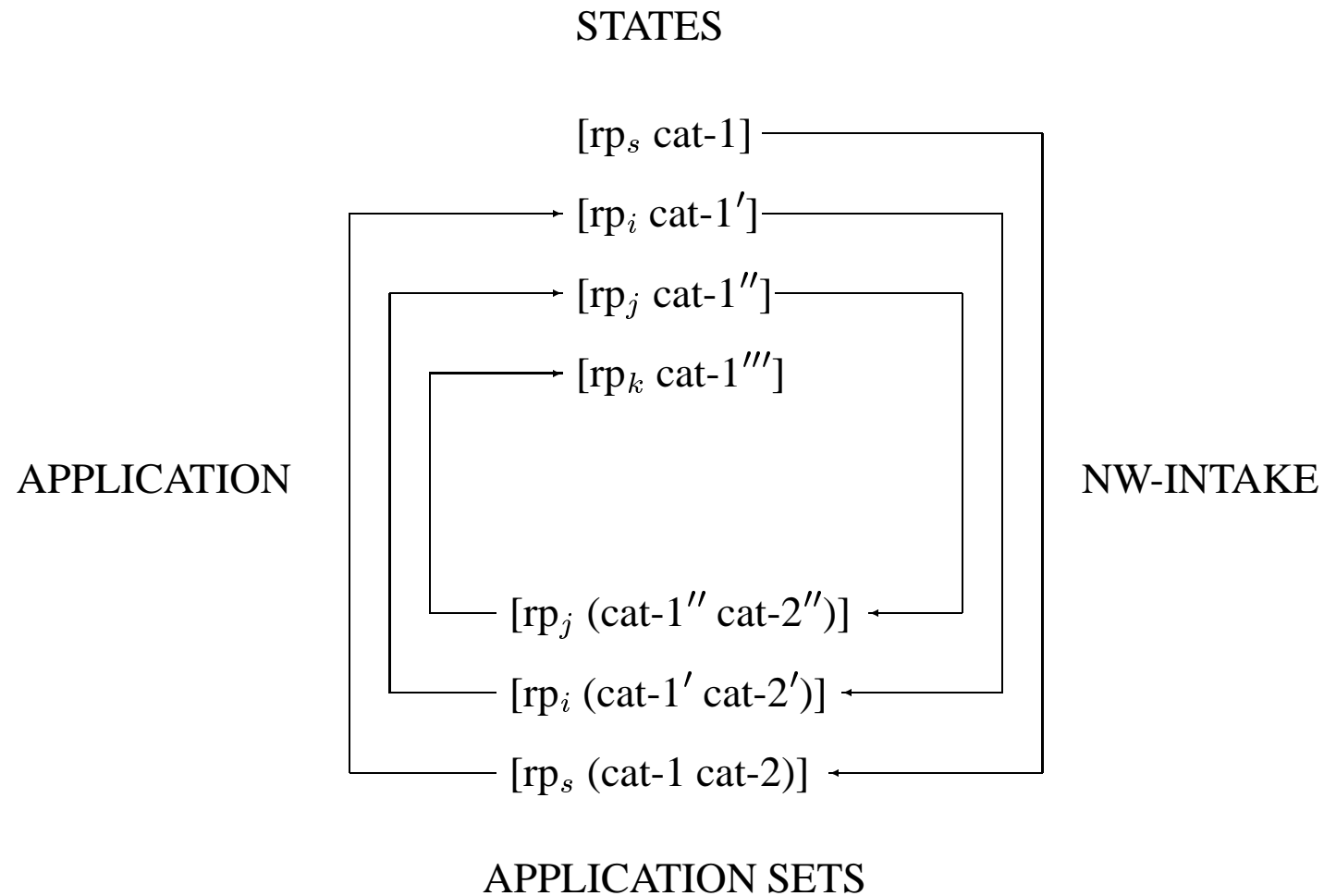
$r_3: (bX) (c) \Rightarrow (X) \{r_3\}$

$ST_F =_{def} \{[\varepsilon rp_3]\}$ .

### 10.2.5 The finite state backbone of the LA-grammar for $a^k b^k c^k$

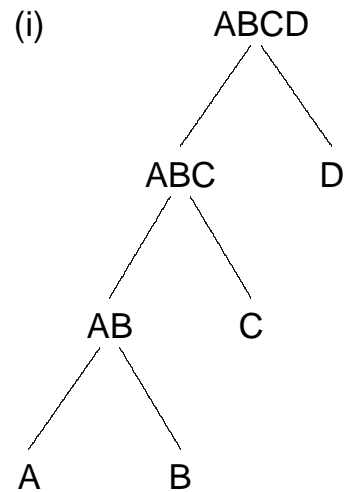


## 10.2.6 Recursion of left-associative algorithm



## 10.3 Time-linear analysis

### 10.3.1 LA-trees as structured lists



(ii) ABCD  
 (D  
 ABC)  
 (C  
 AB)  
 (B  
 A)

(iii) (A  
 B)  
 (AB  
 C)  
 (ABC  
 D)  
 ABCD

### 10.3.2 LA-grammar derivation of $a^k b^k$ for $k=3$

```

NEWCAT>  a a a b b b
          *START-0
          1
            (A) A
            (A) A
          *RULE-1
          2
            (A A) A A
            (A) A
          *RULE-1
          3
            (A A A) A A A
            (B) B
          *RULE-2
          4
            (A A) A A A B
            (B) B
          *RULE-2
          5
            (A) A A A B B
            (B) B
          *RULE-2
          6
            (NIL) A A A B B B

```

### 10.3.3 Interpretation of a history section

```
active rule package:      *START-0
composition number:      1
sentence start:          (A) A
next word:               (A) A
successful rule:         *RULE-1
next composition number: 2
result:                  (A A) A A
```

### 10.3.4 Overlap between history sections

```
active rule package:      *RULE-1
composition number:      2
sentence start :         (A A) A A
next word:               (A) A
successful rule :        *RULE-1
next composition number: 3
result:                  (A A A) A A A
```



## 10.4 Absolute type transparency of LA-grammar

### 10.4.1 Parsing aaabbbccc with active rule counter

```

NEWCAT> a a a b b b c c c
; 1: Applying rules (RULE-1 RULE-2)           (A A B) A A A B
; 2: Applying rules (RULE-1 RULE-2)           (B) B
; 3: Applying rules (RULE-1 RULE-2)           *RULE-2
; 4: Applying rules (RULE-2 RULE-3)           5
; 5: Applying rules (RULE-2 RULE-3)           (A B B) A A A B B
; 6: Applying rules (RULE-2 RULE-3)           (B) B
; 7: Applying rules (RULE-3)                 *RULE-2
; 8: Applying rules (RULE-3)                 6
; Number of rule applications: 14.           (B B B) A A A B B B
                                           (C) C
                                           *RULE-3
                                           7
                                           (C C) A A A B B B C
                                           (C) C
                                           *RULE-3
                                           8
                                           (C) A A A B B B C C
                                           (C) C
                                           *RULE-3
                                           9
                                           (NIL) A A A B B B C C C

```

## 10.4.2 Generating a representative sample in $a^k b^k c^k$

```
NEWCAT> (gram-gen 3 '(a b c))
```

Parses of length 2:

```
A B
  2 (B)
A A
  1 (A A)
```

Parses of length 3:

```
A B C
  2 3 (NIL)
A A B
  1 2 (A B)
A A A
  1 1 (A A A)
```

Parses of length 4:

```
A A B B
  1 2 2 (B B)
A A A B
  1 1 2 (A A B)
A A A A
  1 1 1 (A A A A)
```

Parses of length 5:

```
A A B B C
  1 2 2 3 (B)
A A A B B
```

```
  1 1 2 2 (A B B)
A A A A B
  1 1 1 2 (A A A B)
```

Parses of length 6:

```
A A B B C C
  1 2 2 3 3 (NIL)
A A A B B B
  1 1 2 2 2 (B B B)
A A A A B B
  1 1 1 2 2 (A A B B)
```

Parses of length 7:

```
A A A B B B C
  1 1 2 2 2 3 (B B)
A A A A B B B
  1 1 1 2 2 2 (A B B B)
```

Parses of length 8:

```
A A A B B B C C
  1 1 2 2 2 3 3 (C)
A A A A B B B B
  1 1 1 2 2 2 2 (B B B B)
```

Parses of length 9:

```
A A A B B B C C C
```

```

  1 1 2 2 2 3 3 3   (NIL)
A A A A B B B B C
  1 1 1 2 2 2 2 3   (B B B)

```

Parses of length 10:

```

A A A A B B B B C C
  1 1 1 2 2 2 2 3 3   (B B)

```

Parses of length 11:

```

A A A A B B B B C C C
  1 1 1 2 2 2 2 3 3 3   (B)

```

Parses of length 12:

```

A A A A B B B B C C C C
  1 1 1 2 2 2 2 3 3 3 3   (NIL)

```

### 10.4.3 Complete well-formed expression in $a^k b^k c^k$

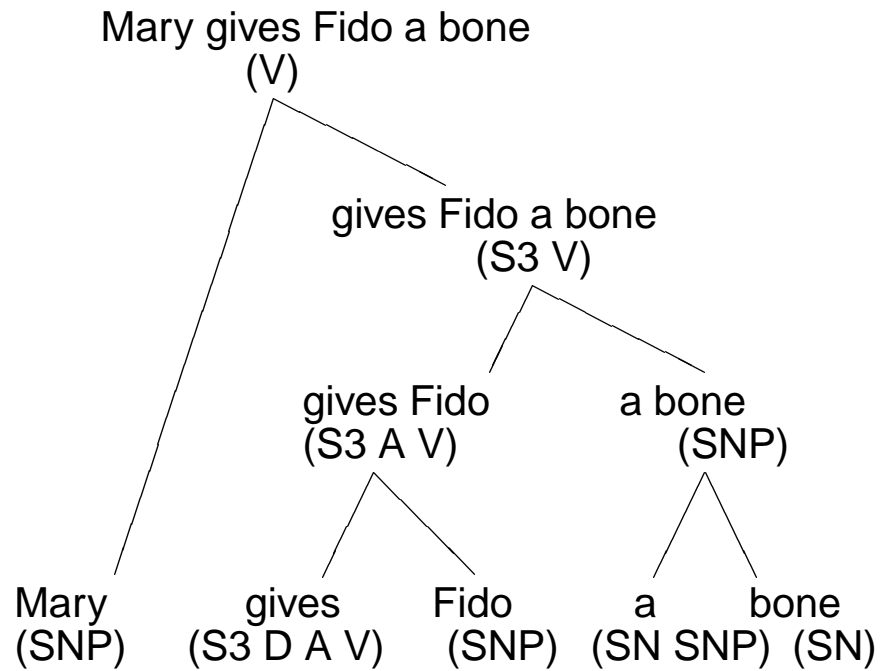
```

A A A B B B C C C
  1 1 2 2 2 3 3 3   (NIL)

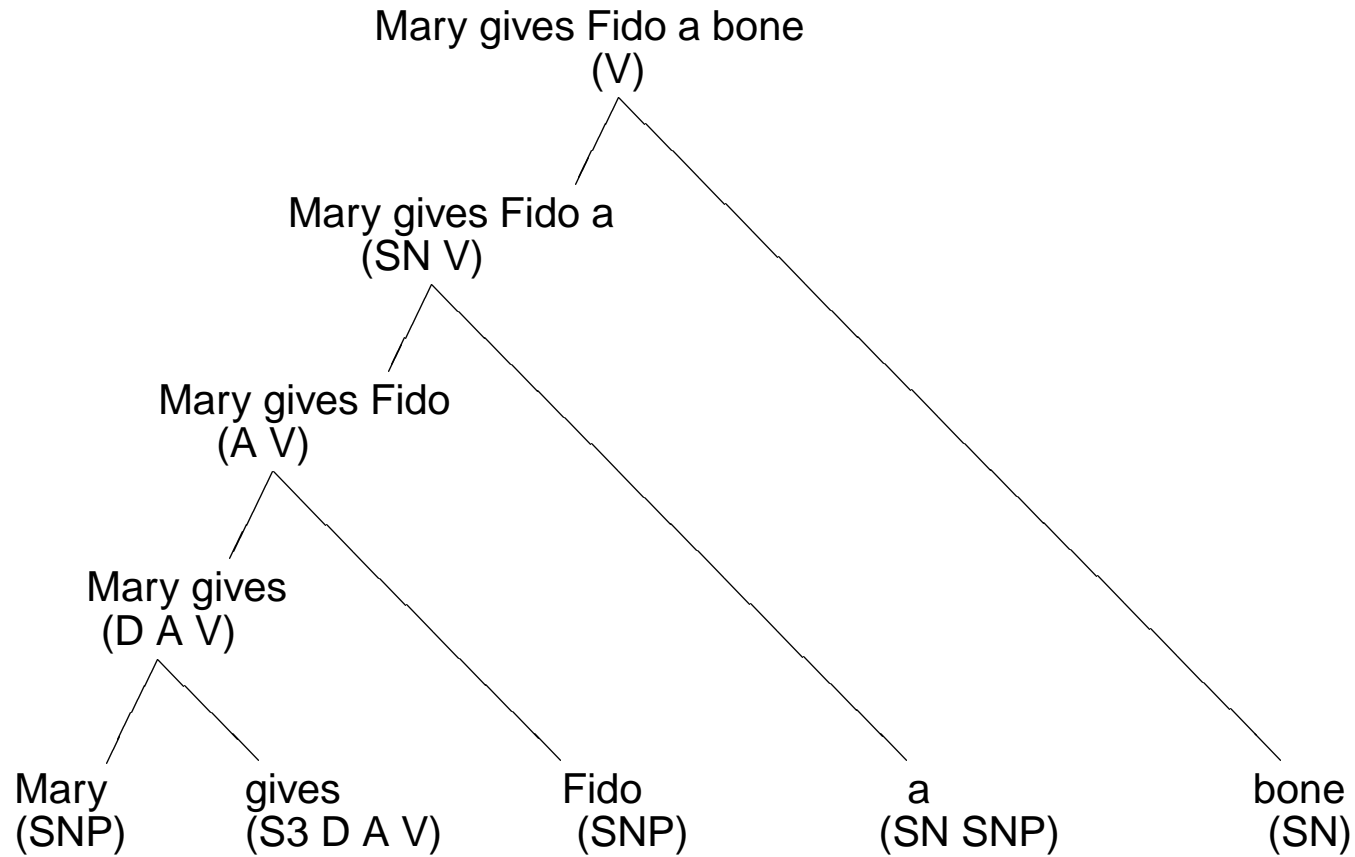
```

## 10.5 LA-grammar for natural language

### 10.5.1 Constituent structure analysis in C-grammar



## 10.5.2 Time-linear analysis in LA-grammar



**10.5.3 Categorical operation combining Mary and gives** $(\text{SNP}) (\text{N D A V}) \Rightarrow (\text{D A V})$ **10.5.4 Categorical operation combining Mary gives and Fido** $(\text{D A V}) (\text{SNP}) \Rightarrow (\text{A V})$ **10.5.5 Categorical operation combining Mary gives Fido and a** $(\text{A V}) (\text{SN SNP}) \Rightarrow (\text{SN V})$ **10.5.6 Categorical operation combining Mary gives Fido a and book** $(\text{SN V}) (\text{SN}) \Rightarrow (\text{V})$

## 10.5.7 Left-associative parsing of example 10.5.2

NEWCAT> Mary gives Fido a bone \.

\*START

1

(SNP) MARY  
(S3 D A V) GIVES

\*NOM+FVERB

2

(D A V) MARY GIVES  
(SNP) FIDO

\*FVERB+MAIN

3

(A V) MARY GIVES FIDO  
(SN SNP) A

\*FVERB+MAIN

4

(SN V) MARY GIVES FIDO A  
(SN) BONE

\*DET+NOUN

5

(V) MARY GIVES FIDO A BONE  
(V DECL) .

\*CMPLT

6

(DECL) MARY GIVES FIDO A BONE .

## 10.5.8 Analysis of a discontinuous element

NEWCAT> Fido dug the bone up \.

\*START

1

(SNP) FIDO  
(N A UP V) DUG

\*NOM+FVERB

2

(A UP V) FIDO DUG  
(SN SNP) THE

\*FVERB+MAIN

3

(SN UP V) FIDO DUG THE  
(SN) BONE

\*DET+NOUN

4

(UP V) FIDO DUG THE BONE  
(UP) UP

\*FVERB+MAIN

5

(V) FIDO DUG THE BONE UP  
(V DECL) .

\*CMPLT

6

(DECL) FIDO DUG THE BONE UP .



## 10.5.9 LA-analysis of ungrammatical input

```
NEWCAT> the young girl give Fido the bone \.
```

```
ERROR
```

```
Ungrammatical continuation at: "GIVE"
```

```
*START
```

```
1
```

```
(SN SNP) THE
```

```
(ADJ) YOUNG
```

```
*DET+ADJ
```

```
2
```

```
(SN SNP) THE YOUNG
```

```
(SN) GIRL
```

```
*DET+NOUN
```

```
3
```

```
(SNP) THE YOUNG GIRL
```

# 11. Hierarchy of LA-grammar

## 11.1 Generative capacity of unrestricted LAG

### 11.1.1 Generative capacity of unrestricted LA-grammar

Unrestricted LA-grammar accepts and generates all and only the recursive languages.

### 11.1.2 Theorem 1

Unrestricted LA-grammar accepts and generates *only* the recursive languages.

*Proof:* Assume an input string of finite length  $n$ . Each word in the input string has a finite number of readings ( $> 0$ ).

Combination step 1: The finite set of start states  $ST_S$  and all readings of the first word  $w_1$  result in a finite set of well-formed expressions  $WE_1 = \{(ss' rp_S) \mid ss' \in (W^+ \times C^+)\}$ .

Combination step  $n$ : Combination step  $k-1$ ,  $k > 1$ , has produced a finite set of well-formed expressions  $WE_k = \{(ss' rp_i) \mid i \in n, ss' \in (W^+ \times C^*) \text{ and the surface of each } ss' \text{ has length } k\}$ . The next word  $w_{k+1}$  has a finite number of readings.

Therefore, the Cartesian product of all elements of  $WE_k$  and all readings of the current next word will be a finite set of pairs. Each pair is associated with a rule package containing a finite set of rules. Therefore, combination step  $k$  will produce only finitely many new sentence starts. The derivation of this finite set of new sentence starts is decidable because the categorial operations are defined to be total recursive functions.

Q.E.D.

### 11.1.3 Theorem 2

Unrestricted LA-grammar accepts and generates *all* recursive languages.

*Proof:* Let  $L$  be a recursive language with the alphabet  $W$ . Because  $L$  is recursive, there is a total recursive function  $\varrho: W^* \rightarrow \{0,1\}$ , i.e., the characteristic function of  $L$ . Let  $LAG^L$  be an LA-grammar defined as follows:

The set of word surfaces of  $LAG^L$  is  $W$ .

The set of category segments  $C =_{def} W \cup \{0,1\}$ .

For arbitrary  $e, f \in W^+$ ,  $[e (f)] \in LX$  if and only if  $e = f$ .

$$LX =_{def} \{[a (a)], [b (b)], [c (c)], [d (d)], \dots\}$$

$$ST_S =_{def} \{[(seg_c) \{r_1, r_2\}]\}, \text{ where } seg_c \in \{a, b, c, d, \dots\}$$

$$r_1: (X) (seg_c) \Rightarrow (X seg_c) \quad \{r_1, r_2\}$$

$$r_2: (X) (seg_c) \Rightarrow \varrho(X seg_c) \quad \{ \}$$

$$ST_F =_{def} \{[(1) rp_2]\}$$

After any given combination step, the rule package  $rp_1$  offers two choices: application of  $r_1$  to continue reading the input string, or application of  $r_2$  to test whether the input read so far is a well-formed expression of  $L$ . In the

latter case, the function  $\varrho$  is applied to the concatenation of the input categories, which are identical to the input surfaces. If the result of applying  $r_2$  is  $[(1) rp_2]$ , the input surface is accepted; if it is  $[(0) rp_2]$ , it is rejected.

Since the categorial operations of  $LAG^L$  can be any total recursive function,  $LAG^L$  may be based on  $\varrho$ , the characteristic function of  $L$ . Therefore,  $LAG^L$  accepts and generates any recursive language.

Q.E.D.

#### 11.1.4 Definition of the class of A-LAGs.

The class of A-LAGs consists of unrestricted LA-grammars and generates *all* recursive languages.

## 11.2 LA-hierarchy of A-, B-, and C-LAGs

### 11.2.1 Parameters of complexity

- The *amount* of computation per rule application required in the worst case.
- The *number* of rule applications relative to the length of the input needed in the worst case.

### 11.2.2 Main approaches to restricting LA-grammar

*R1*: Restrictions on the form of categorial operations in order to limit the maximal amount of computation required by arbitrary rule applications.

*R2*: Restrictions on the degree of ambiguity in order to limit the maximal number of possible rule applications.

### 11.2.3 Possible restrictions on categorial operations

*R1.1*: Specifying upper bounds for the *length* of categories;

*R1.2*: Specifying restrictions on *patterns* used in the definition of categorial operations.

### 11.2.4 Definition of the class of B-LAGs.

The class of *bounded* LA-grammars, or B-LAGs, consists of grammars where for any complete well-formed expression E the length of intermediate sentence start categories is bounded by  $k \cdot n$ , where n is the length of E and k is a constant.

### 11.2.5 Rule schemata with constant categorial operations

$$r_i: (\text{seg}_1 \dots \text{seg}_k \text{ X}) \text{ cat}_2 \Rightarrow \text{cat}_3 \text{ rp}_i$$

$$r_i: (\text{X seg}_1 \dots \text{seg}_k) \text{ cat}_2 \Rightarrow \text{cat}_3 \text{ rp}_i$$

$$r_i: (\text{seg}_1 \dots \text{seg}_m \text{ X seg}_{m+1} \dots \text{seg}_k) \text{ cat}_2 \Rightarrow \text{cat}_3 \text{ rp}_i$$

### 11.2.6 Rule schema with nonconstant categorial operation

$$r_i: (\text{X seg}_1 \dots \text{seg}_k \text{ Y}) \text{ cat}_2 \Rightarrow \text{cat}_3 \text{ rp}_i$$

### 11.2.7 Definition of the class of C-LAGs.

The class of *constant* LA-grammars, or C-LAGs, consists of grammars in which no categorial operation  $\text{co}_i$  looks at more than k segments in the sentence start categories, for a finite constant k.

### 11.2.8 The hierarchy of A-LAGs, B-LAGs, and C-LAGs

The class of A-LAGs accepts and generates all recursive languages, the class of B-LAGs accepts and generates all context-sensitive languages, and the class of C-LAGs accepts and generates many context-sensitive, all context-free, and all regular languages.



## 11.3 Ambiguity in LA-grammar

### 11.3.1 Factors determining the number of rule applications

The number of rule application in an LA-derivation depends on

1. the length of the input;
2. the number of rules in the rule package to be applied in a certain combination to the analyzed input pair;
3. the number of readings existing at each combination step.

### 11.3.2 Impact on complexity

- Factor 1 is grammar-independent and used as the length  $n$  in formulas characterizing complexity .
- Factor 2 is a grammar-dependent constant.
- Only factor 3 may push the total number of rule applications beyond a linear increase. Whether for a given input more than one rule in a rule package may be successful depends on the input conditions of the rules.

### 11.3.3 Regarding factor 3: Possible relations between the input conditions of two rules

1. *Incompatible* input conditions: if there exist no input pairs which are accepted by both rules.

Examples:           (a X) (b)                               (a X) (b)  
                         (c X) (b)                               (a X) (c)

2. *Compatible* input conditions: if there exists at least one input pair accepted by both rules and there exists at least one input pair accepted by one rule, but not the other.

Examples:           (a X) (b)  
                         (X a) (b)

3. *Identical* input conditions: if all input pairs are either accepted by both rules or rejected by both rules.

### 11.3.4 Definition of unambiguous LA-grammars

An LA-grammar is unambiguous if and only if (i) it holds for all rule packages that their rules have *incompatible* input conditions and (ii) there are no lexical ambiguities.

### 11.3.5 Definition of syntactically ambiguous LA-grammars

An LA-grammar is syntactically ambiguous if and only if (i) it has at least one rule package containing at least two rules with *compatible* input conditions and (ii) there are no lexical ambiguities.

### 11.3.6 +global syntactic ambiguity

A syntactic ambiguity is called +global if it is a property of the whole sentence.

Example: Flying air planes can be dangerous.

### 11.3.7 –global syntactic ambiguity

A syntactic ambiguity is called -global if it is a property of only part of the sentence.

Example: The horse raced by the barn fell.

### 11.3.8 Role of the $\pm$ global distinction

In LA-grammar, the difference between +global and –global ambiguities consists in whether more than one reading survives to the end of the sentence (example 11.3.6) or not (example 11.3.7). The  $\pm$ global distinction has no impact on complexity in LA-grammar and is made mainly for linguistic reasons.

### 11.3.9 +recursive syntactic ambiguity

An ambiguity is +recursive, if it originates within a recursive loop of rule applications.

Examples: the C-LAGs for  $WW^R$  (cf. 11.5.6) and  $WW$  (cf. 11.5.8), which are –global, and for `SubsetSum` (cf. 11.5.10), which are +global.

### 11.3.10 -recursive syntactic ambiguity

An ambiguity is –recursive, if none of the branches produced in the ambiguity split returns to the state which caused the ambiguity.

Examples: the C-LAG for  $a^k b^k c^m d^m \cup a^k b^m c^m d^k$  (cf. 11.5.3), which is +global, and the C-LAGs for natural language in Chapter 17 and 18, which exhibit both +global and –global ambiguities.

### 11.3.11 Role of the $\pm$ recursive distinction

The  $\pm$ recursive distinction is crucial for the analysis of complexity because it can be shown that in LA-grammars with nonrecursive ambiguities the maximal number of rule applications per combination step is limited by a grammar-dependent constant.

### 11.3.12 Theorem 3

The maximal number of rule applications in LA-grammar with only –recursive ambiguities is

$$(n - (R - 2)) \cdot 2^{(R-2)}$$

for  $n > (R - 2)$ , where  $n$  is the length of the input and  $R$  is the number of rules in the grammar.

*Proof:* Parsing an input of length  $n$  requires  $(n - 1)$  combination steps. If an LA-grammar has  $R$  rules, then one of these rules has to be reapplied after  $R$  combination steps at the latest. Furthermore, the maximal number of rule applications in a combination step for a given reading is  $R$ .

According to the definition of –recursive ambiguity, rules causing a syntactic ambiguity may not be reapplied in a time-linear derivation path (reading). The first ambiguity-causing rule may produce a maximum of  $R-1$  new branches (assuming its rule package contains all  $R$  rules except for itself), the second ambiguity causing rule may produce a maximum of  $R - 2$  new branches, etc. If the different rules of the LA-grammar are defined with their maximally possible rule packages, then after  $R - 2$  combination steps a maximum of  $2^{(R-2)}$  readings is reached.

Q.E.D.

## 11.4 Complexity of grammars and automata

### 11.4.1 Choosing the primitive operation

The Griffith and Petrick data is not in terms of actual time, but in terms of “primitive operations.” They have expressed their algorithms as sets of nondeterministic rewriting rules for a Turing-machine-like device. Each application of one of these is a primitive operation. We have chosen as our primitive operation the act of adding a state to a state set (or attempting to add one which is already there). We feel that this is comparable to their primitive operation because both are in some sense the most complex operation performed by the algorithm whose complexity is independent of the size of the grammar and the input string.

J. Earley 1970, p. 100

### 11.4.2 Primitive operation of the C-LAGs

The primitive operation of C-LAGs is a rule application (also counting unsuccessful attempts).

## 11.5 Subhierarchy of C1-, C2-, and C3-LAGs

### 11.5.1 The subclass of C1-LAGs

A C-LAG is a C1-LAG if it is not recursively ambiguous. The class of C1-languages parses in linear time and contains all deterministic context-free languages which can be recognized by a DPDA without  $\varepsilon$ -moves, plus context-free languages with –recursive ambiguities, e.g.  $a^k b^k c^m d^m \cup a^k b^m c^m d^k$ , as well as many context-sensitive languages, e.g.  $a^k b^k c^k$ ,  $a^k b^k c^k d^k e^k$ ,  $\{a^k b^k c^k\}^*$ ,  $L_{square}$ ,  $L_{hast}^k$ ,  $a^{2^i}$ ,  $a^k b^m c^{k \cdot m}$ , and  $a^{i!}$ , whereby the last one is not even an index language.

### 11.5.2 C1-LAG for context-sensitive $a^{2^i}$

$$LX =_{def} \{[a(a)]\}$$

$$ST_S =_{def} \{[(a) \{r_1\}]\}$$

$$r_1: (a) \quad (a) \Rightarrow (aa) \quad \{r_2\}$$

$$r_2: (aX) \quad (a) \Rightarrow (Xbb) \quad \{r_2, r_3\}$$

$$r_3: (bX) \quad (a) \Rightarrow (Xaa) \quad \{r_2, r_3\}$$

$$ST_F =_{def} \{[(aa) rp_1], [(bXb) rp_2], [(aXa) rp_3]\}.$$

### 11.5.3 C1-LAG for ambiguous $a^k b^k c^m d^m \cup a^k b^m c^m d^k$

$LX =_{def} \{[a(a)], [b(b)], [c(c)], [d(d)]\}$

$ST_S =_{def} \{[(a) \{r_1, r_2, r_5\}]\}$

$r_1: (X) \quad (a) \Rightarrow (a X) \quad \{r_1, r_2, r_5\}$

$r_2: (a X) \quad (b) \Rightarrow (X) \quad \{r_2, r_3\}$

$r_3: (X) \quad (c) \Rightarrow (c X) \quad \{r_3, r_4\}$

$r_4: (c X) \quad (d) \Rightarrow (X) \quad \{r_4\}$

$r_5: (X) \quad (b) \Rightarrow (b X) \quad \{r_5, r_6\}$

$r_6: (b X) \quad (c) \Rightarrow (X) \quad \{r_6, r_7\}$

$r_7: (a X) \quad (d) \Rightarrow (X) \quad \{r_7\}$

$ST_F =_{def} \{[\varepsilon rp_4], [\varepsilon rp_7]\}$



### 11.5.4 The Single Return Principle (SRP)

A +recursive ambiguity is single return, if exactly one of the parallel paths returns into the state resulting in the ambiguity in question.

### 11.5.5 The subclass of C2-LAGs

A C-LAG is a C2-LAG if it is SR-recursively ambiguous. The class of C2-languages parses in polynomial time and contains certain nondeterministic context-free languages like  $WW^R$  and  $L_{\text{fast}}^\infty$ , plus context-sensitive languages like  $WW$ ,  $W^{k \geq 3}$ ,  $\{WWW\}^*$ , and  $W_1W_2W_1^RW_2^R$ .

### 11.5.6 C2-LAG for context-free $WW^R$

$LX =_{def} \{[a(a)], [b(b)], [c(c)], [d(d)] \dots\}$

$ST_S =_{def} \{[(seg_c) \{r_1, r_2\}]\}$ , where  $seg_c \in \{a, b, c, d, \dots\}$

$r_1: (X) (seg_c) \Rightarrow (seg_c X) \{r_1, r_2\}$

$r_2: (seg_c X) (seg_c) \Rightarrow (X) \{r_2\}$

$ST_F =_{def} \{[\varepsilon rp_2]\}$

### 11.5.7 Derivation structure of the worst case in $WW^R$

rules:

analyses:

2	a \$ a
1 2 2	a a \$ a a
1 1 2 2 2	a a a \$ a a a
1 1 1 2 2	a a a a \$ a a
1 1 1 1 2	a a a a a \$ a
1 1 1 1 1	a a a a a a \$

### 11.5.8 C2-LAG for context-sensitive WW

$$\begin{aligned}
 LX &=_{def} \{[a(a)], [b(b)], [c(c)], [d(d)] \dots\} \\
 ST_S &=_{def} \{[(\text{seg}_c) \{r_1, r_2\}]\}, \text{ where } \text{seg}_c \in \{a, b, c, d, \dots\} \\
 r_1: (X) \quad (\text{seg}_c) &\Rightarrow (X \text{ seg}_c) \{r_1, r_2\} \\
 r_2: (\text{seg}_c X) \quad (\text{seg}_c) &\Rightarrow (X) \quad \{r_2\} \\
 ST_F &=_{def} \{[\varepsilon \text{ rp}_2]\}
 \end{aligned}$$

### 11.5.9 C2-LAG for context-sensitive $W_1 W_2 W_1^R W_2^R$

$$\begin{aligned}
 LX &=_{def} \{[a(a)], [b(b)]\} \\
 ST_S &=_{def} \{[(\text{seg}_c) \{r_{1a}\}], [(\text{seg}_c) \{r_{1b}\}]\}, \text{ where } \text{seg}_c, \text{seg}_d \in \{a, b\} \\
 r_{1a}: (\text{seg}_c) \quad (\text{seg}_d) &\Rightarrow (\# \text{seg}_c \text{seg}_d) \{r_2, r_3\} \\
 r_{1b}: (\text{seg}_c) \quad (\text{seg}_d) &\Rightarrow (\text{seg}_d \# \text{seg}_c) \{r_3, r_4\} \\
 r_2: (X) \quad (\text{seg}_c) &\Rightarrow (X \text{ seg}_c) \quad \{r_2, r_3\} \\
 r_3: (X) \quad (\text{seg}_c) &\Rightarrow (\text{seg}_c X) \quad \{r_3, r_4\} \\
 r_4: (X \text{ seg}_c) \quad (\text{seg}_c) &\Rightarrow (X) \quad \{r_4, r_5\} \\
 r_5: (\text{seg}_c X \#) \quad (\text{seg}_c) &\Rightarrow (X) \quad \{r_6\} \\
 r_6: (\text{seg}_c X) \quad (\text{seg}_c) &\Rightarrow (X) \quad \{r_6\} \\
 ST_F &=_{def} \{[\varepsilon \text{ rp}_5], [\varepsilon \text{ rp}_6]\}
 \end{aligned}$$

### 11.5.10 C3-LAG for SubsetSum.

$$LX =_{def} \{[0 (0)], [1 (1)], [# (\#)]\}$$

$$ST_S =_{def} \{[(seg_c) \{r_1, r_2\}]\}, \text{ where } seg_c \in \{0, 1\}$$

$$seg_c \in \{0, 1\}$$

$$r_1: (X) \quad (seg_c) \quad \Rightarrow \quad (seg_c X) \{r_1, r_2\}$$

$$r_2: (X) \quad (\#) \quad \Rightarrow \quad (\# X) \{r_3, r_4, r_6, r_7, r_{12}, r_{14}\}$$

$$r_3: (X seg_c) \quad (seg_c) \quad \Rightarrow \quad (0 X) \{r_3, r_4, r_6, r_7\}$$

$$r_4: (X \#) \quad (\#) \quad \Rightarrow \quad (\# X) \{r_3, r_4, r_6, r_7, r_{12}, r_{14}\}$$

$$r_5: (X seg_c) \quad (seg_c) \quad \Rightarrow \quad (0 X) \{r_5, r_6, r_7, r_{11}\}$$

$$r_6: (X 1) \quad (0) \quad \Rightarrow \quad (1 X) \{r_5, r_6, r_7, r_{11}\}$$

$$r_7: (X 0) \quad (1) \quad \Rightarrow \quad (1 X) \{r_8, r_9, r_{10}\}$$

$$r_8: (X seg_c) \quad (seg_c) \quad \Rightarrow \quad (1 X) \{r_8, r_9, r_{10}\}$$

$$r_9: (X 1) \quad (0) \quad \Rightarrow \quad (0 X) \{r_5, r_6, r_7, r_{11}\}$$

$$r_{10}: (X 0) \quad (1) \quad \Rightarrow \quad (0 X) \{r_8, r_9, r_{10}\}$$

$$r_{11}: (X \#) \quad (\#) \quad \Rightarrow \quad (\# X) \{r_3, r_4, r_6, r_7, r_{12}, r_{14}\}$$

$$r_{12}: (X 0) \quad (seg_c) \quad \Rightarrow \quad (0 X) \{r_4, r_{12}, r_{14}\}$$

$$r_{13}: (X 0) \quad (seg_c) \quad \Rightarrow \quad (0 X) \{r_{11}, r_{13}, r_{14}\}$$

$$r_{14}: (X 1) \quad (seg_c) \quad \Rightarrow \quad (1 X) \{r_{11}, r_{13}, r_{14}\}$$

$$ST_F =_{def} \{[(X) rp_4]\}$$

### 11.5.11 Types of restriction in LA-grammar

0. LA-type A: no restriction
1. LA-type B: The length of the categories of intermediate expressions is limited by  $k \cdot n$ , where  $k$  is a constant and  $n$  is the length of the input ( $R1.1$ , amount).
2. LA-type C3: The form of the category patterns results in a constant limit on the operations required by the categorial operations ( $R1.2$ , amount).
3. LA-type C2: LA-type C3 and the grammar is at most SR-recursively ambiguous ( $R2$ , number).
4. LA-type C1: LA-type C3 and the grammar is at most  $\omega$ -recursively ambiguous ( $R2$ , number).

### 11.5.12 LA-grammar hierarchy of formal languages

restrictions	types of LAG	languages	complexity
LA-type C1	C1-LAGs	C1 languages	linear
LA-type C2	C2-LAGs	C2 languages	polynomial
LA-type C3	C3-LAGs	C3 languages	exponential
LA-type B	B-LAGs	B languages	exponential
LA-type A	A-LAGs	A languages	exponential

## 12. LA- and PS-hierarchies in comparison

### 12.1 Language classes of LA- and PS-grammar

#### 12.1.1 Complexity degrees of the LA- and PS-hierarchy

	<i>LA-grammar</i>	<i>PS-grammar</i>
<i>undecidable</i>	—	recursively enumerable languages
<i>exponential</i>	A-languages B-languages C3-languages	context-sensitive languages
<i>polynomial</i>	C2-languages	context-free languages
<i>linear</i>	C1-languages	regular languages

## 12.2 Subset relations in the two hierarchies

### 12.2.1 Subset relations in the PS-hierarchy

regular lang.  $\subset$  context-free lang.  $\subset$  context-sensitive lang.  $\subset$  rec. enum. languages

### 12.2.2 Subset relations in the LA-hierarchy

C1-languages  $\subseteq$  C2-languages  $\subseteq$  C3-languages  $\subseteq$  B-languages  $\subset$  A-languages



## 12.3 Non-equivalence of the LA- and PS-hierarchy

### 12.3.1 Languages which are in the same class in PS-grammar, but in different classes in LA-grammar

$a^k b^k$  and  $WW^R$  are in the same class in PS-grammar (i.e. context-free), but in different classes in LA-grammar:  $a^k b^k$  is a C1-LAG parsing in linear time, while  $WW^R$  is a C2-LAG parsing in  $n^2$ .

### 12.3.2 Languages which are in the same class in LA-grammar, but in different classes in PS-grammar

$a^k b^k$  and  $a^k b^k c^k$  are in the same class in LA-grammar (i.e. C1-LAGs), but in different classes in PS-grammar:  $a^k b^k$  is context-free, while  $a^k b^k c^k$  is context-sensitive.

### 12.3.3 Inherent complexity

The inherent complexity of a language is based on the number of operations required in the worst case on an abstract machine (e.g. a Turing or register machine). This form of analysis occurs on a very low level corresponding to machine or assembler code.

### 12.3.4 Class assigned complexity

The complexity of artificial and natural languages is usually analyzed at the abstraction level of grammar formalisms, whereby complexity is determined for the grammar type and its language class as a whole.

### 12.3.5 Difference between the two types of complexity

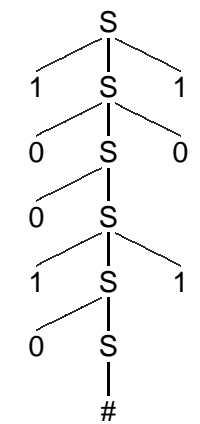
Languages which are inherently of high complexity (e.g. 3SAT and SUBSET SUM) are necessarily in a high complexity class (here exponential) in any possible grammar formalism.

Languages which are inherently of low complexity (e.g.  $a^k b^k c^k$ ) may be assigned high or low class complexity, depending on the formalism.

### 12.3.6 PS-Grammar of $L_{no}$

$$\begin{array}{lll} S \rightarrow 1S1 & S \rightarrow 1S & S \rightarrow \# \\ S \rightarrow 0S0 & S \rightarrow 0S & \end{array}$$

### 12.3.7 PS-grammar derivation of 10010#101 in $L_{no}$

derivation tree:	generated chains:	states:
	1S1 10S01 100S01 1001S101 10010S101 10010#101	1.S1 1S1. 1.S 0.S0 0S0. 0.S 0.S0 0.S 0S. 1.S1 1S1. 1.S 0.S0 0.S 0S. #.

### 12.3.8 C3-LAG for $L_{no}$

$LX =_{def} \{[0 (0)], [1 (1)], [# (\#)]\}$

$ST_S =_{def} \{[(seg_c) \{r_1, r_2, r_3, r_4, r_5\}]\}$ , where  $seg_c, seg_d \in \{0, 1\}$ .

$r_1: (seg_c)(seg_d) \Rightarrow \varepsilon \quad \{r_1, r_2, r_3, r_4, r_5\}$

$r_2: (seg_c)(seg_d) \Rightarrow (seg_d) \quad \{r_1, r_2, r_3, r_4, r_5\}$

$r_3: (X)(seg_c) \Rightarrow (X) \quad \{r_1, r_2, r_3, r_4, r_5\}$

$r_4: (X)(seg_c) \Rightarrow (seg_c X) \quad \{r_1, r_2, r_3, r_4, r_5\}$

$r_5: (X)(\#) \Rightarrow (X) \quad \{r_6\}$

$r_6: (seg_c X)(seg_c) \Rightarrow (X) \quad \{r_6\}$

$ST_F =_{def} \{[\varepsilon rp_6]\}$

## 12.4 Comparing the lower LA- and PS-classes

Context-free PS-grammar has been widely used because it provides the greatest amount of generative capacity within the PS-grammar hierarchy while being computationally tractable.

### 12.4.1 How suitable is context-free grammar for describing natural and programming languages?

There is general agreement in linguistics that context-free PS-grammar does not properly fit the structures characteristic of natural language. The same holds for computer science:

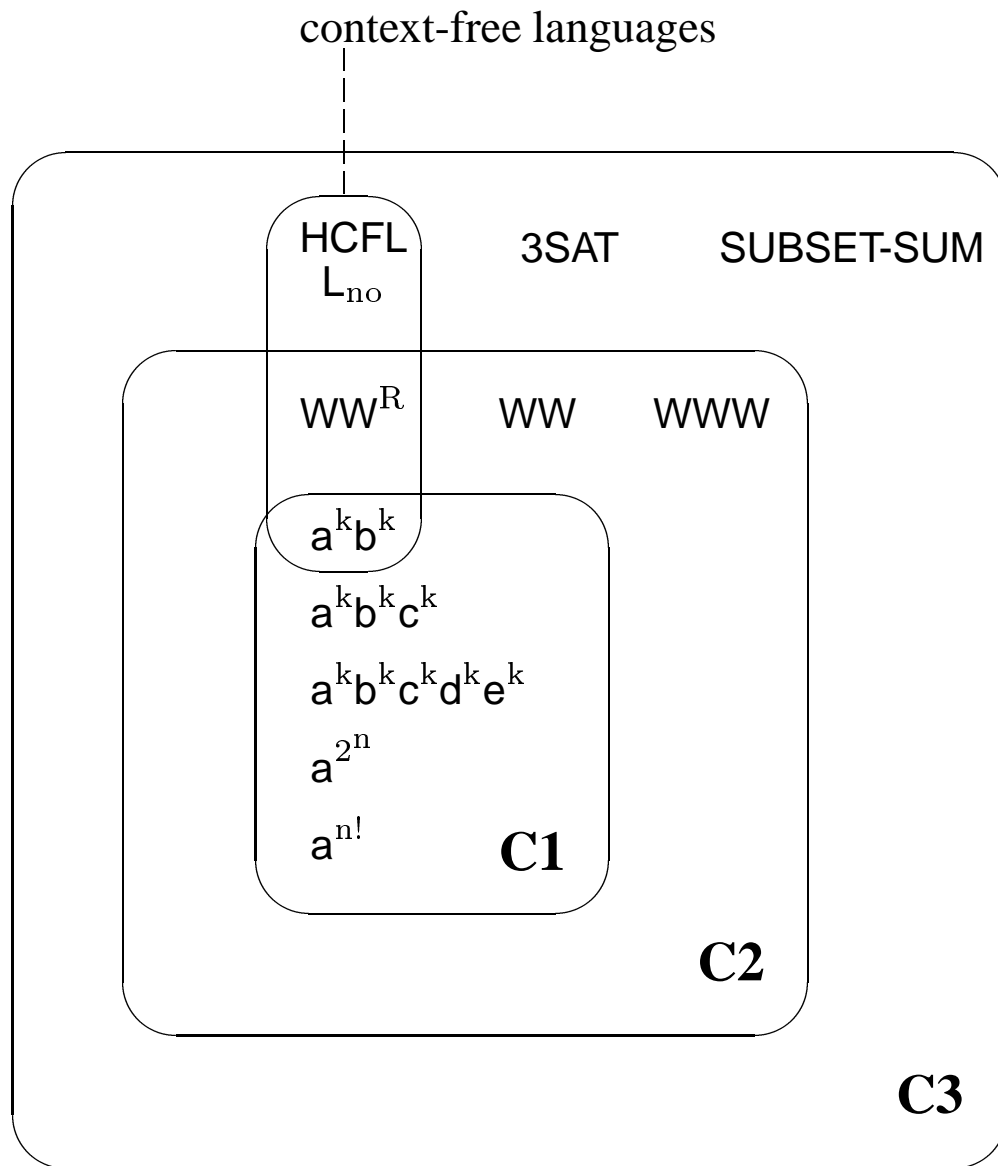
It is no secret that context-free grammars are only a first order approximation to the various mechanisms used for specifying the syntax of modern programming languages.

S. Ginsberg 1980, p.7

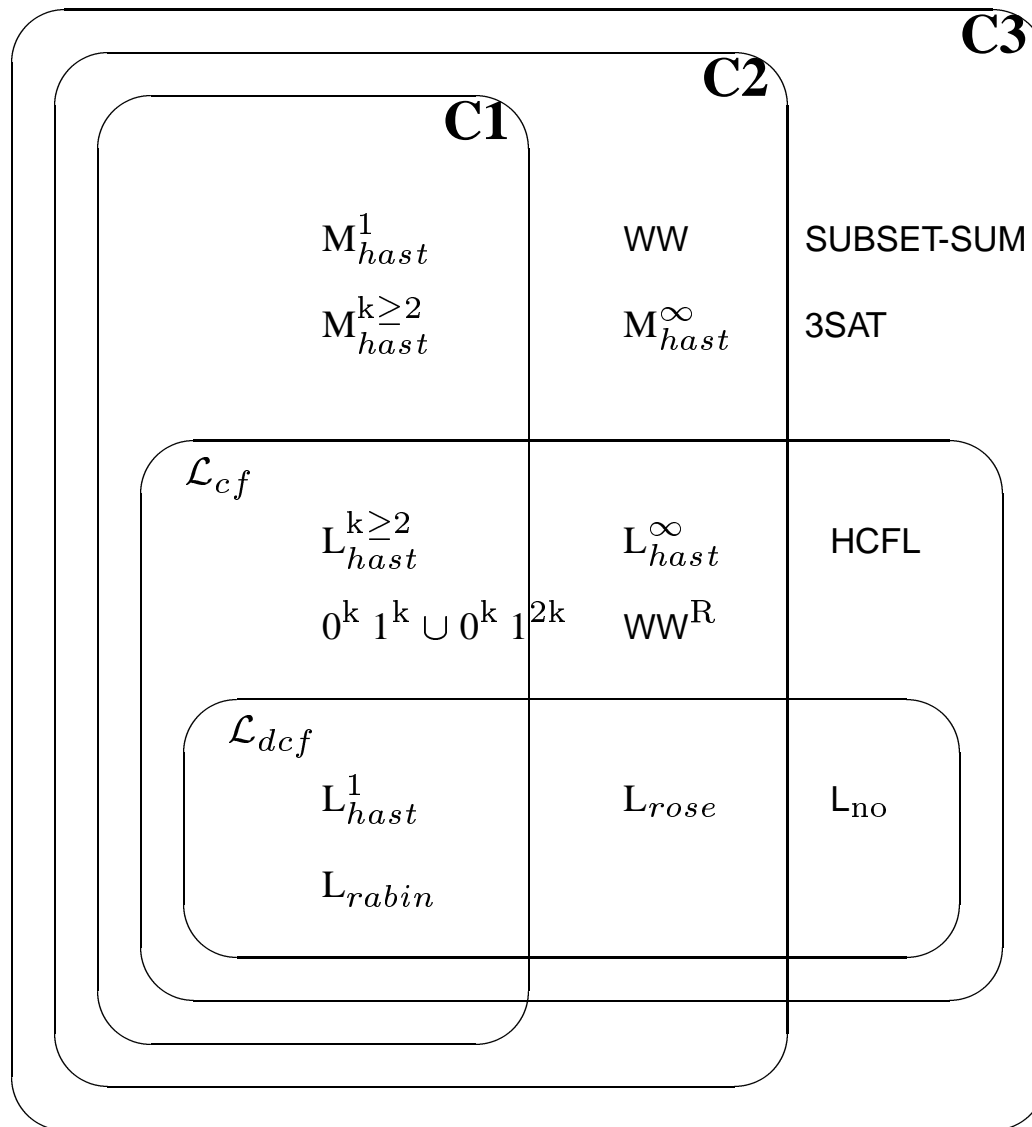
### 12.4.2 Conservative extensions of the PS-grammar hierarchy

regular lang.  $\subset$  context-free lang.  $\subset$  TAL  $\subset$  index lang.  $\subset$  context-sensitive lang.  $\subset$  r.e. languages

### 12.4.3 Orthogonal relation between C- and cf-languages



### 12.4.4 Orthogonal $\mathcal{L}_{dcf}$ , $\mathcal{L}_{cf}$ , $C_1$ , $C_2$ , and $C_3$ classifications



## 12.5 Linear complexity of natural language

### 12.5.1 Why the natural languages are likely to be C-languages

In a context-sensitive language which is not a C-language, the category length would have to grow just within the LBA-definition of context-sensitive languages, but grow faster than the pattern-based categorial operations of the C-LAGs would permit.

That this type of language should be characteristic for the structure of natural language is highly improbable.

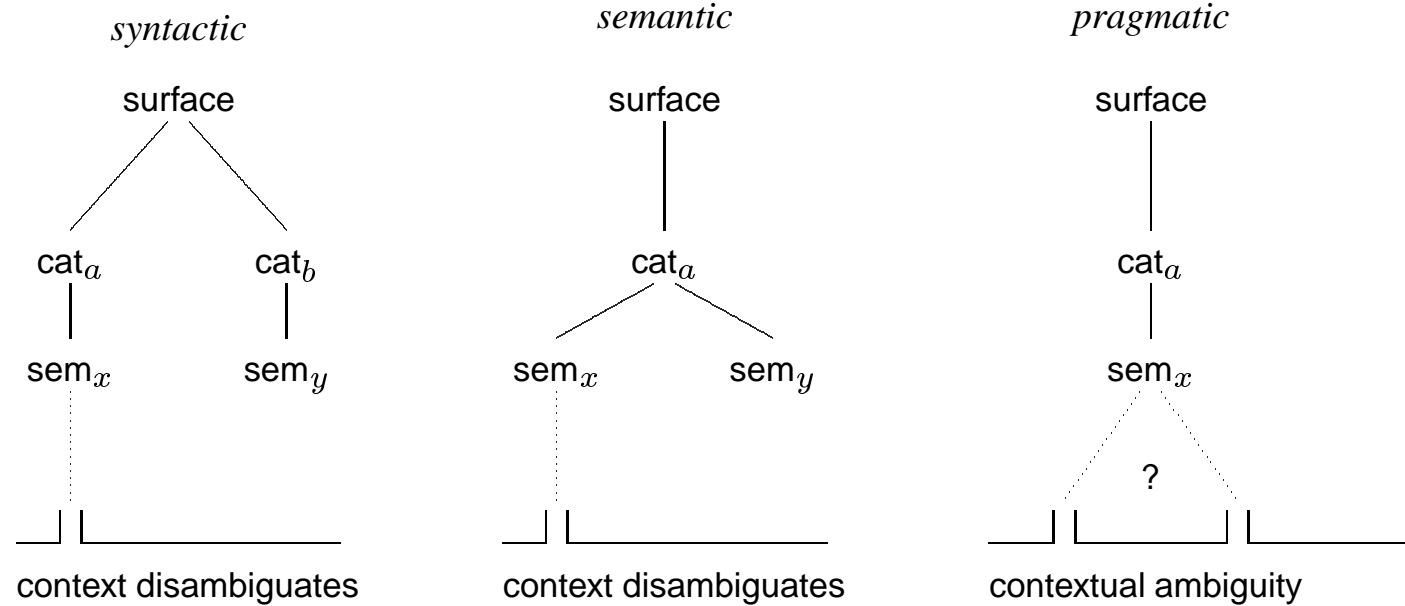
### 12.5.2 If the natural languages are C-LAGs

then the following two questions are equivalent:

- (i) *How complex are the natural languages?*
- (ii) *How ambiguous are the natural languages?*

This is because the C-LAG subclasses differ solely in their degrees of ambiguity.

### 12.5.3 SLIM-theoretic analysis of ambiguity



### 12.5.4 Multiple interpretations of prepositional phrases: a syntactic or a semantic ambiguity?

The man saw the girl with the telescope.

Julia ate the apple on the table behind the tree in the garden.



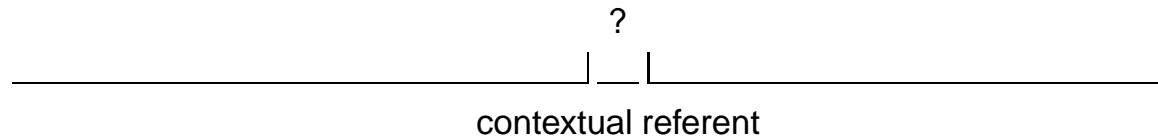
### 12.5.5 Incorrect analysis of a semantic ambiguity

analysis 1:

The osprey is looking for a perch  
|  
[*kind of fish*]

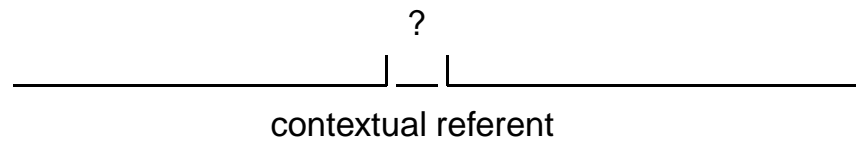
analysis 2:

The osprey is looking for a perch  
|  
[*place to roost*]

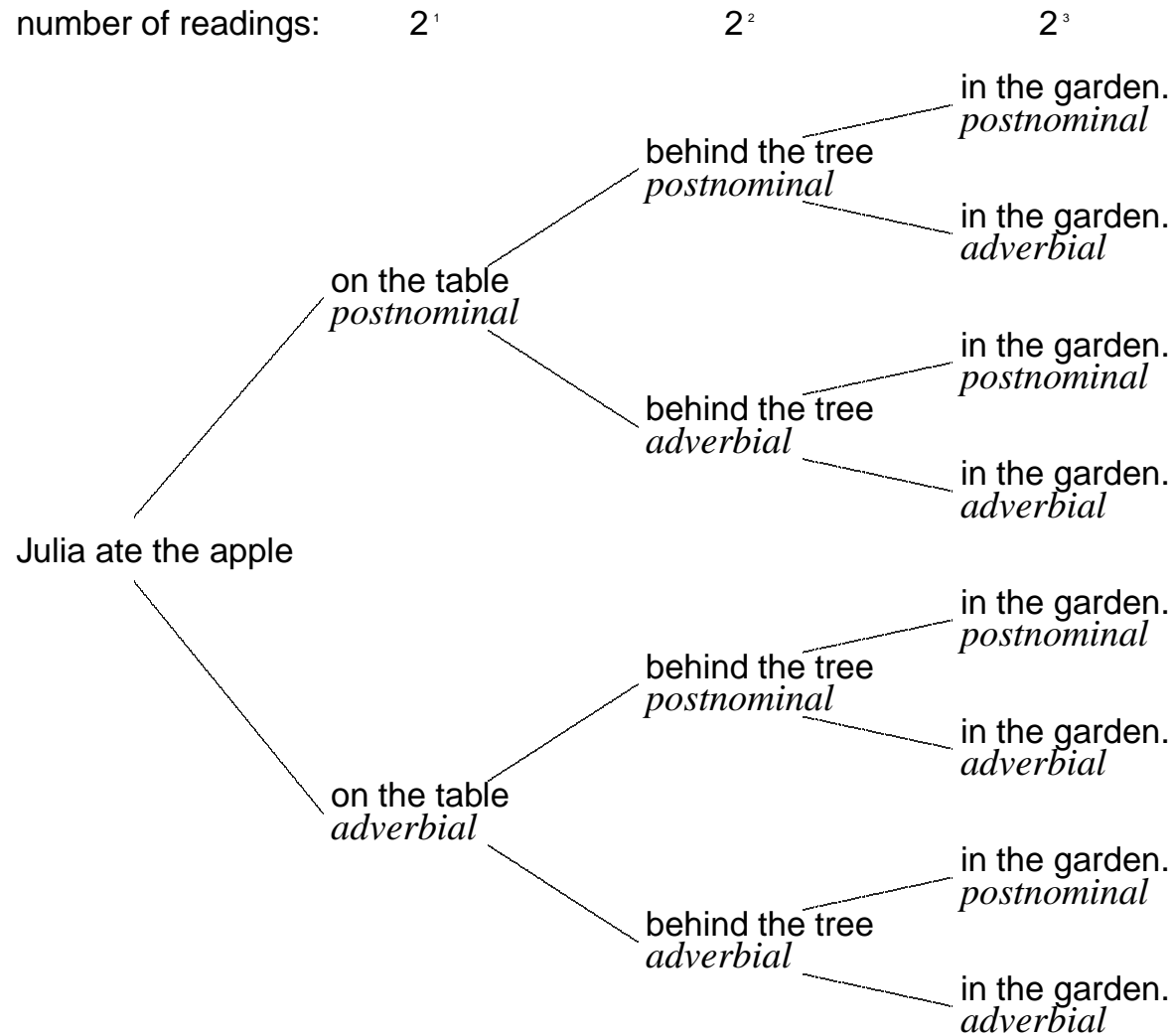


### 12.5.6 Correct analysis of a semantic ambiguity

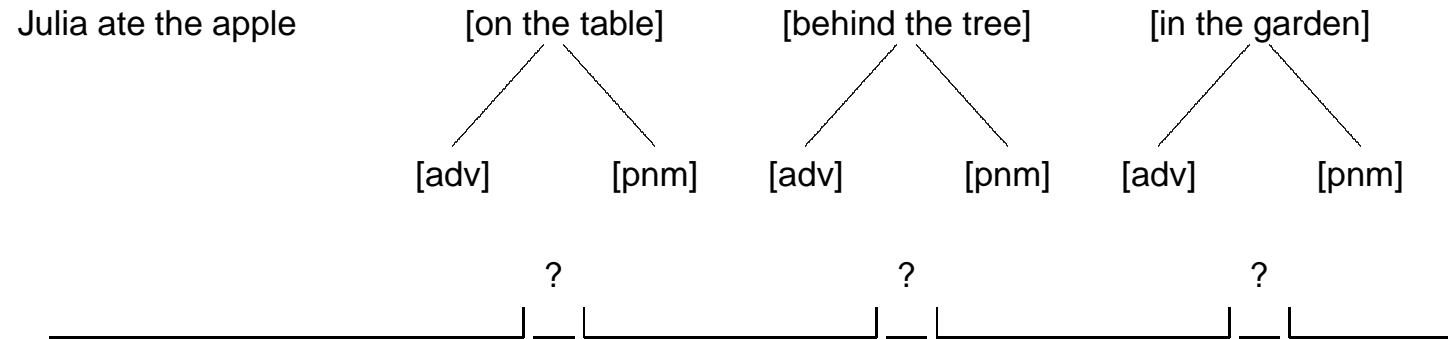
The osprey is looking for a perch  
/      \  
[*kind of fish*]      [*place to roost*]



### 12.5.7 Incorrect analysis: a recursive pseudo-ambiguity



### 12.5.8 Correct analysis with *semantic doubling*



### 12.5.9 CoNSyx hypothesis (Complexity of natural language syntax)

The natural languages are contained in the class of C1-languages and parse in linear time.

This hypothesis holds as long as no recursive ambiguities are found in natural language.

**Part III.**  
**Morphology and Syntax**

## 13. Words and morphemes

### 13.1 Words and word forms

#### 13.1.1 Different syntactic compatibilities of word forms

\*write

\*writes

\*wrote

*John has written a letter.*

\*writing

#### 13.1.2 Francis' & Kučera's 1982 definition of a graphic word

“A word is a string of continuous alphanumeric characters with space on either side; may include hyphens and apostrophes, but no other punctuation marks.”

### 13.1.3 Combination principles of morphology

1. *Inflection* is the systematic variation of a word with which it can perform different syntactic and semantic functions, and adapt to different syntactic environments. Examples are learn, learn/s, learn/ed, and learn/ing.
2. *Derivation* is the combination of a word with an affix. Examples are clear/ness, clear/ly, and un/clear.
3. *Composition* is the combination of two or more words into a new word form. Examples are gas/light, hard/wood, over/indulge, and over-the-counter.

### 13.1.4 Definition of the notion *word*

Word =<sub>def</sub> {associated analyzed word forms}

### 13.1.5 Example of an analyzed word form

[wolves (PN) wolf]

### 13.1.6 Analysis of an inflecting word

<i>word</i>	<i>word forms</i>
wolf = <sub>def</sub>	{ [wolf (SN) wolf], [wolf's (GN) wolf], [wolves (PN) wolf], [wolves' (GN) wolf] }

### 13.1.7 Analysis of a noninflecting word

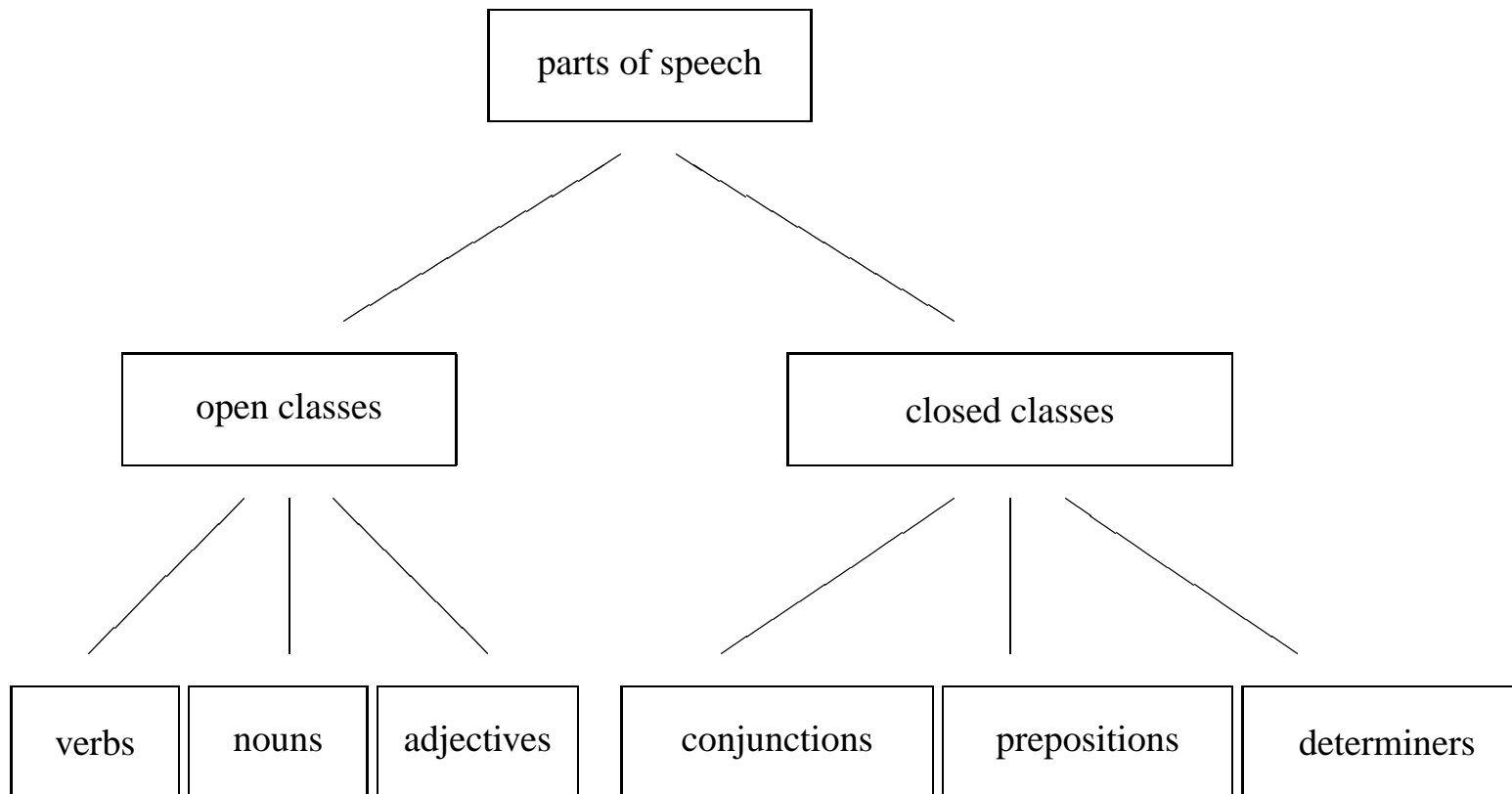
<i>word</i>	<i>word forms</i>
and = <sub>def</sub>	{ [and (cnj) and] }

### 13.1.8 Parts of speech

- *verbs*, e.g., walk, read, give, help, teach, ...
- *nouns*, e.g., book, table, woman, messenger, arena, ...
- *adjective-adverbials*, e.g., quick, good, low, ...
- *conjunctions*, e.g., and, or, because, after, ...
- *prepositions*, e.g., in, on, over, under, before, ...
- *determiners*, e.g., a, the, every, some, all, any, ...
- *particles*, e.g., only, already, just. . .



### 13.1.9 Classification of the parts of speech into open and closed classes



### 13.1.10 Comparison of the open and the closed classes

- The open classes comprise several 10 000 elements, while the closed classes contain only a few hundred words.
- The morphological processes of inflection, derivation, and composition are productive in the open classes, but not in the closed classes.
- In the open classes, the use of words is constantly changing, with new ones entering and obsolete ones leaving the current language, while the closed classes do not show a comparable fluctuation.

### 13.1.11 Parts of speech and types of signs

The elements of the open classes are also called *content words*, while the elements of the closed classes are also called *function words*. In this distinction, however, the sign type must be taken into consideration besides the category.

This is because only the *symbols* among the nouns, verbs, and adjective-adverbials are content words in the proper sense. *Indices*, on the other hand, e.g. the personal pronouns he, she, it etc., are considered function words even though they are of the category noun. Indexical adverbs like here or now do not even inflect, forming no comparatives and superlatives. The sign type *name* is also a special case among the nouns.

## 13.2 Segmentation and concatenation

### 13.2.1 Relation of words and their inflectional forms in German

	base forms	inflectional forms
nouns:	23 000	92 000
verbs:	6 000	144 000
adjective-adverbials:	11 000	198 000
<hr/>		
	40 000	434 000

### 13.2.2 Number of noun-noun compositions

- length two:  $n^2$   
Examples Haus/schuh, Schuh/haus, Jäger/jäger. This means that from 20 000 nouns 400 000 000 possible compounds of length 2 can be derived (base forms).
- length three:  $n^3$   
Examples: Haus/schuh/sohle, Sport/schuh/haus, Jäger/jäger/jäger. This means that an additional 8 000 000 000 000 000 (eight thousand trillion) possible words may be formed.

### 13.2.3 Possible words, actual words, and neologisms

- Possible words

Because there is no grammatical limit on the length of noun compounds, the number of possible word forms in German is infinite. These word forms exist potentially because of the inherent productivity of morphology.

- Actual words

The set of words and word forms used by the language community within a certain interval of time is finite.

- Neologisms

Neologisms are coined spontaneously by the language users on the basis of known words and the rules of word formation. Neologisms turn possible words into actual words.

### 13.2.4 Examples of neologisms in English

insurrectionist (inmate)

copper-jacketed (bullets)

cyberstalker

self-tapping (screw)

migraineur

three-player (set)

bad-guyness

trapped-rat (frenzy)

dismissiveness

extraconstitutional (gimmick)

### 13.2.5 Definition of the notion *morpheme*

morpheme =<sub>def</sub> { associated analyzed allomorphs }

### 13.2.6 Formal analysis of the morpheme wolf

*morpheme*      *allomorphs*  
 wolf =<sub>def</sub>      { [wolf (SN SR) wolf],  
                          [wolv (PN SR) wolf] }

### 13.2.7 Comparing morpheme and word wolf

<i>morpheme</i>	<i>allomorphs</i>	<i>word</i>	<i>word forms</i>
wolf = <sub>def</sub>	{ wolf, wolv }	wolf = <sub>def</sub>	{ wolf, wolf's, wolv/es, wolv/es/ }

### 13.2.8 Alternative forms of segmentation

allomorphs:	learn/ing
syllables:	lear/ning
phonemes:	/e/r/n/i/n/g
letters:	/e/a/r/n/i/n/g

## 13.3 Morphemes and allomorphs

### 13.3.1 The regular morpheme learn

*morpheme*      *allomorphs*  
 learn =<sub>def</sub>    {[learn (N ... V) learn]}

### 13.3.2 The irregular morpheme swim

*morpheme*      *allomorphs*  
 swim =<sub>def</sub>    {[swim (N ... V1) swim],  
                   [swimm (... B) swim],  
                   [swam (N ... V2) swim],  
                   [swum (N ... V) swim]}

### 13.3.3 An example of suppletion

*morpheme*      *allomorphs*  
 good =<sub>def</sub>    {[good (ADV IR) good],  
                   [bett (CAD IR) good],  
                   [b (SAD IR) good]}

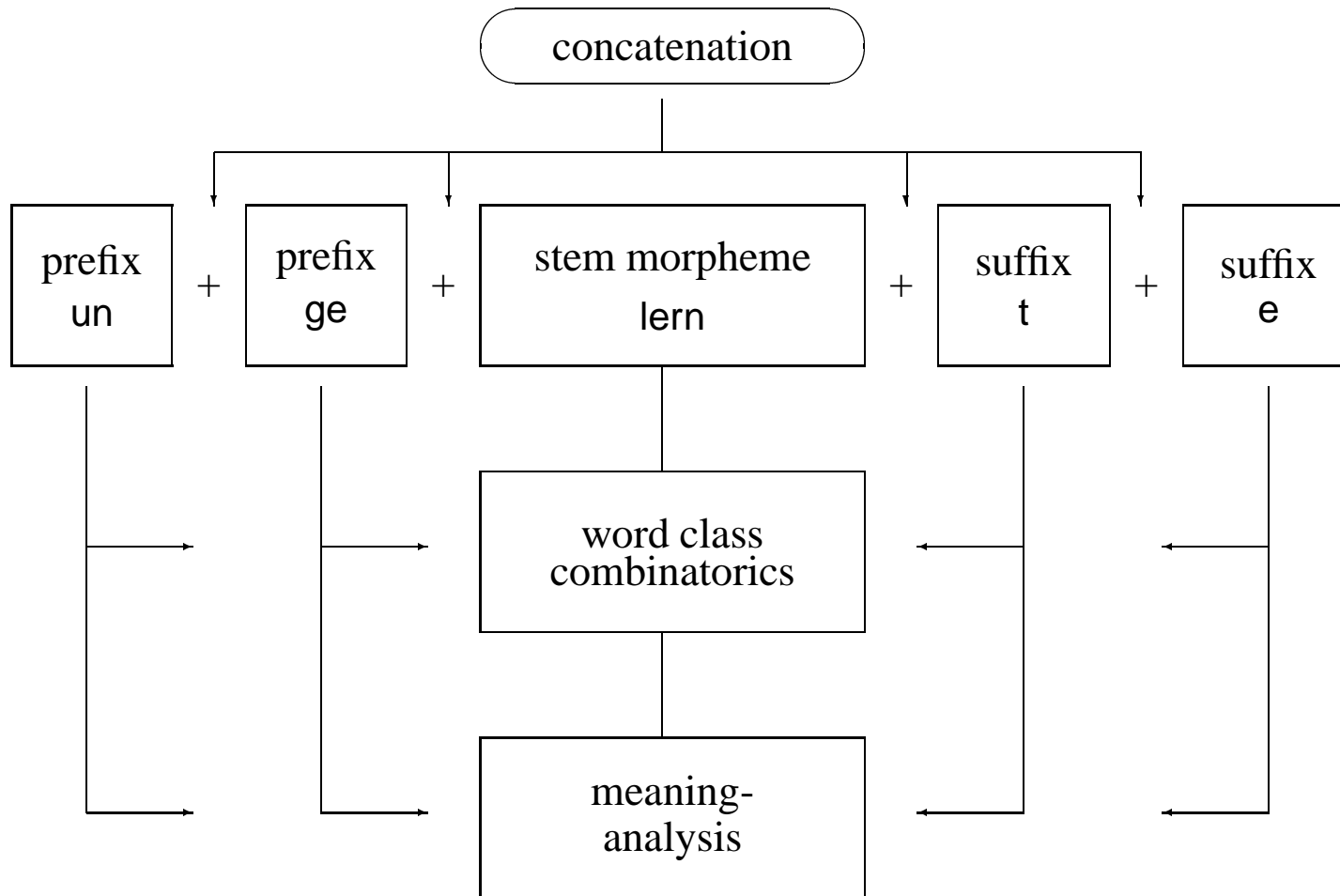
### 13.3.4 Example of a bound morpheme (hypothetical)

*morpheme*      *allomorphs*  
-s =<sub>def</sub> {[s (PL1) plural],  
          [es (PL2) plural],  
          [en (PL3) plural],  
          [# (PL4) plural]}



## 13.4 Categorization and lemmatization

### 13.4.1 Morphological analysis of ungelernte



### 13.4.2 Schematic derivation in LA-grammar

("un" (CAT1) MEAN-a) + ("ge" (CAT2) MEAN-b)  
("un/ge" (CAT3) MEAN-c) + ("lern" (CAT4) MEAN-d)  
("un/ge/lern" (CAT5) MEAN-e) + ("t" (CAT6) MEAN-f)  
("un/ge/lern/t" (CAT7) MEAN-g) + ("e" (CAT8) MEAN-h)  
("un/ge/lern/t/e" (CAT9) MEAN-i)

### 13.4.3 Components of word form recognition

- *On-line lexicon*

For each element (e.g. morpheme) of the natural language there must be defined a lexical analysis which is stored electronically.

- *Recognition algorithm*

Using the on-line lexicon, each unknown word form (e.g. wolves) must be characterized automatically with respect to categorization and lemmatization:

- *Categorization*

consists in specifying the part of speech (e.g. noun) and the morphosyntactic properties of the surface (e.g. plural); needed for syntactic analysis.

- *Lemmatization*

consists in specifying the correct base form (e.g. wolf); provides access to the corresponding lemma in a semantic lexicon.

### 13.4.4 Basic structure of a lemma

[surface (lexical description)]

### 13.4.5 Lemma of a traditional dictionary (*excerpt*)

<sup>1</sup>**wolf** \ 'wʊlf\ *n. pl wolves* \ 'wʊlvz\ *often attributed* [ME, fr. OE *wulf*; akin to OHG *wolf*, L *lupus*, Gk *lykos*] **1 pl also wolf**  
**a:** any of various large predatory mammals (genus *Canis* and esp. *C. lupus*) that resemble the related dogs, are destructive to game and livestock, and may rarely attack man esp. when in a pack – compare COYOTE, JACKAL **b:** the fur of a wolf ...

### 13.4.6 Matching a surface onto a key

word form surface:

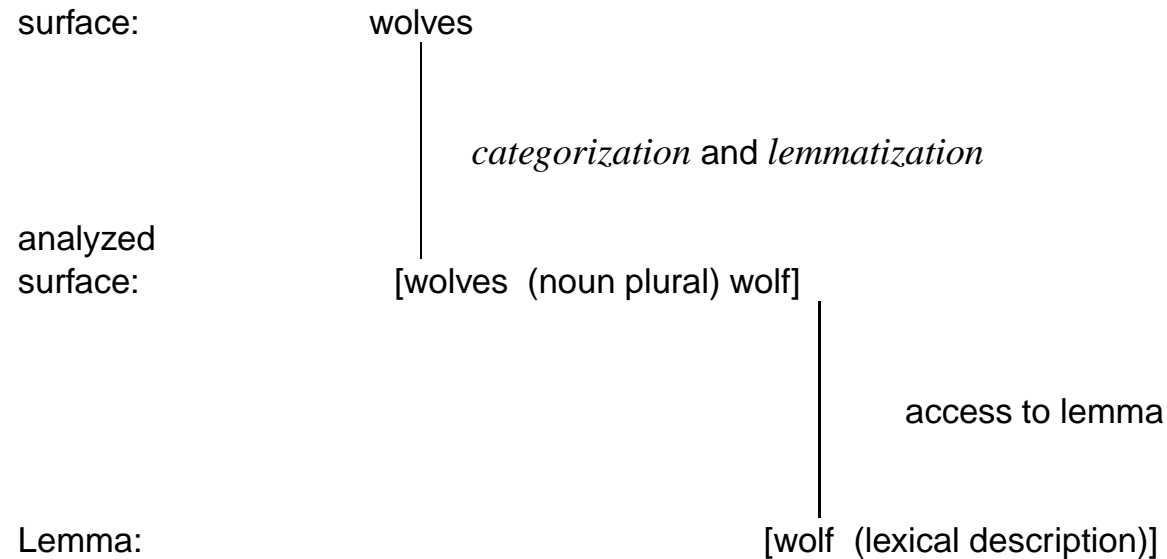
wolf

| *matching*

lemma:

[ wolf (lexical description)]

### 13.4.7 Two-step procedure of word form recognition



### 13.4.8 Reason for the Two-step procedure

In the natural languages

- the number of word forms is considerably larger than the number of words, at least in inflectional and agglutinating languages, and
- the lexical lemmata normally define words rather than word forms,

## 13.5 Methods of automatic word form recognition

### 13.5.1 Word form method

Based on a lexicon of analyzed word forms.

### 13.5.2 Analyzed word form as lexical lemma

[wolves (part of speech: Subst, num: Pl, case: N,D,A, base form: wolf)]

Categorization and lemmatization are not handled by rules, but solely by the lexical entry.

### 13.5.3 Advantages and disadvantages of the word form method

- Advantage  
Allows for the simplest recognition algorithm because the surface of the unknown word form, e.g. wolves, is simply matched whole onto the corresponding key in the analysis lexicon.
- Disadvantages  
The production of the analysis lexicon is costly, its size is extremely large, and there is no possibility to recognize neologisms.

### 13.5.4 Morpheme method

Based on a lexicon of analyzed morphemes.

### 13.5.5 Schema of the morpheme method

surface:	wolves	
		<i>segmentation</i>
allomorphs:	wolv/es	
	↓ ↓	<i>reduction</i>
morphemes:	wolf+s	<i>base form lookup and concatenation</i>

(1) segmentation into allomorphs, (2) reduction of allomorphs to the morphemes, (3) recognition of morphemes using an analysis lexicon, and (4) rule-based concatenation of morphemes to derive analyzed word form.

### 13.5.6 Advantages and disadvantages of the morpheme method

- Advantages

Uses the smallest analysis lexicon. Neologisms may be analyzed and recognized during run-time using a rule-based segmentation and concatenation of complex word forms into their elements (morphemes).

- Disadvantages

A maximally complex recognition algorithm ( $\mathcal{NP}$  complete).

### 13.5.7 Allomorph method

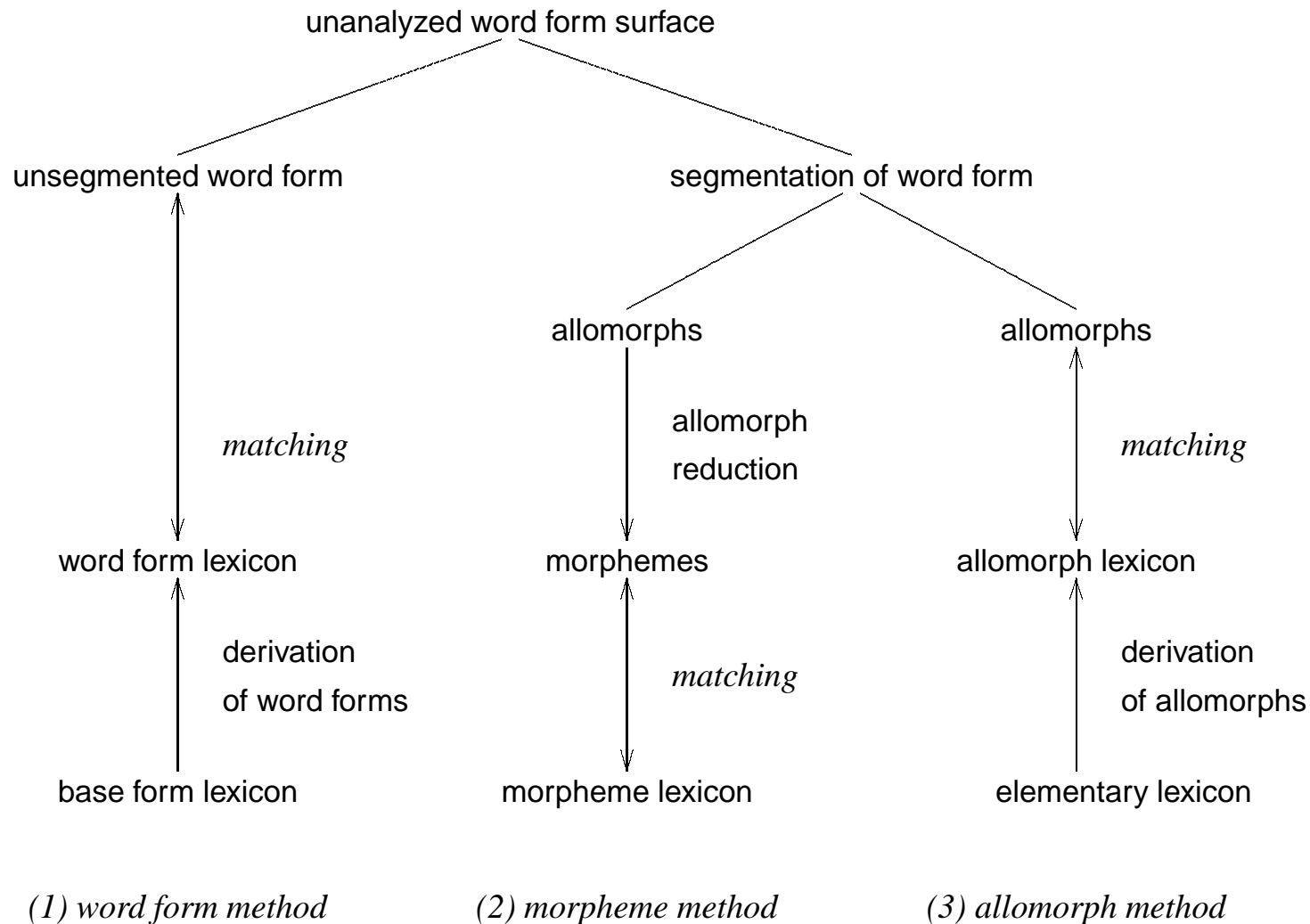
Based on a lexicon of elementary base forms, from which a lexicon of analyzed allomorphs is derived before run by means of allo-rules..

### 13.5.8 Schema of the allomorph method

surface:	wolves	
		<i>segmentation</i>
allomorphs:	wolv/es	<i>allomorph lookup and concatenation</i>
	↑ ↑	<i>derivation of allomorphs before run-time</i>
morphemes & allomorphs:	wolf s	

During run-time, the allomorphs of the allomorph lexicon are available as precomputed, fully analyzed forms, providing the basis for a maximally simple segmentation: the unknown surface is matched from left to right with suitable allomorphs – without any reduction to morphemes. Concatenation takes place on the level of analyzed allomorphs by means of combi-rules.

### 13.5.9 Schematic comparison of the three basic methods

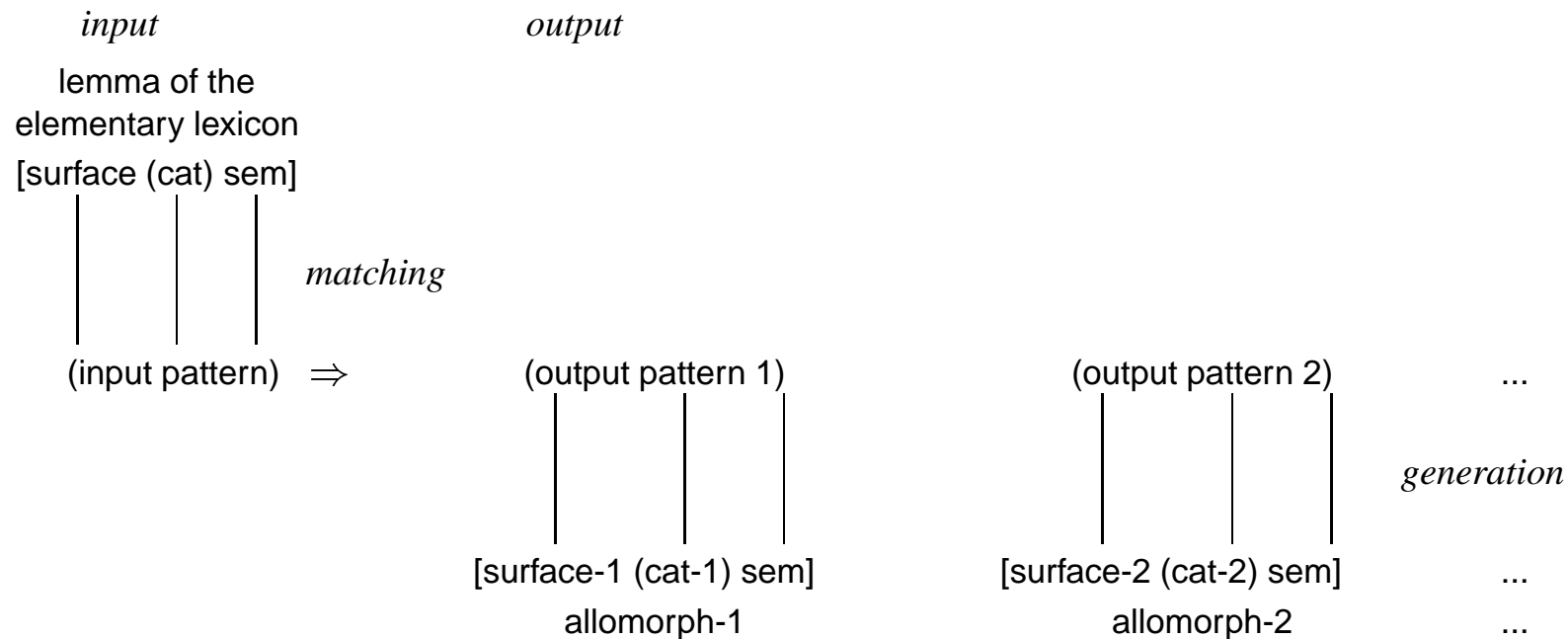




# 14. Word form recognition in LA-Morph

## 14.1 Allo-rules

### 14.1.1 Abstract format of an allo-rule



### 14.1.2 Example of a base form lemma

("derive" (nom a v) derive)

### 14.1.3 Result of applying allo-rules to base form lemma

("derive" (sr nom a v) derive)

("deriv" (sr a v) derive)

### 14.1.4 Base form entry of schlafen

```
("schla2fen" (KV VH N GE {hinueber VS GE } {durch VH A GE }
  {aus VH GE } {ein VS GE } \$ <be VH A GE- >
  <ent VS GE- > <ueber VH A GE- > <ver VH A GE- > )
  schlafen)
```

### 14.1.5 Output of allo-rules for schlafen

```
("schlaf" (IV V1 VH N GE { hinüber VS GE } { durch VH A GE }
  { aus VH GE } { ein VS GE } $ < be VH A GE- >
  < ent VS GE- > < über VH A GE- > < ver VH A GE- > )
  schlafen)
```

```
("schläf" (IV V2 _0 N GE { hinüber VS GE } { durch VH A GE }
  { aus VH GE } { ein VS GE } $ < be VH A GE- >
  < ent VS GE- > < über VH A GE- > < ver VH A GE- > )
  schlafen)
```

```
("schlief" (IV V34 _0 N GE { hinüber VS GE } { durch VH A GE }
  { aus VH GE } { ein VS GE } $ < be VH A GE- >
  < ent VS GE- > < über VH A GE- > < ver VH A GE- > )
  schlafen_i)
```

### 14.1.6 The word forms of schlafen (excerpt)

```

("schlaf/e" (S1 {hinüber}{durch A}{aus}{ein} V) schlafen_p)
("schlaf/e" (S13 {hinüber} {durch A} {aus} {ein} V ) s._k1)
("schlaf/e/n" (P13 {hinüber} {durch A} {aus} {ein} V ) s._pk1)
("schlaf/e/st" (S2 {hinüber} {durch A} {aus} {ein} V ) s._k1)
("schlaf/e/t" (P2 {hinüber} {durch A} {aus} {ein} V ) s._k1)
("schlaf/t" (P2 {hinüber} {durch A} {aus} {ein} V ) s._p)
("schlaf/end" (GER ) schlafen)
("schlaf/end/e" (E ) schlafen)
("schlaf/end/en" (EN ) schlafen)
("schlaf/end/er" (ER ) schlafen)
("schlaf/end/es" (ES ) schlafen)
("schlaf/end/em" (EM ) schlafen)
("schlaf/e/st" (S2 {hinüber} {durch A} {aus} {ein} V ) s._k1)
("schlaf/e/t" (P2 {hinüber} {durch A} {aus} {ein} V ) s._k1)
("schläf/st" (S2 {hinüber} {durch A} {aus} {ein} V ) s._p)
("schläf/t" (S3 {hinüber} {durch A} {aus} {ein} V ) s._p)
("schlief" (S13 {hinüber} {durch A} {aus} {ein} V ) s._i)
("schlief/e" (S13 {hinüber} {durch A} {aus} {ein} V ) s._k2)
("schlief/en" (P13 {hinüber} {durch A} {aus} {ein} V ) s._ik2)

```

("schlief/est" (S2 {hinüber} {durch A} {aus} {ein} V ) s.\_ik2)  
("schlief/et" (P2 {hinüber} {durch A} {aus} {ein} V ) s.\_ik2)  
("schlief/st" (S2 {hinüber} {durch A} {aus} {ein} V ) s.\_ik2)  
("schlief/t" (P2 {hinüber} {durch A} {aus} {ein} V ) s.\_i)  
("ge/schlaf/en" (H) schlafen)  
("ge/schlaf/en/e" (E) schlafen)  
("ge/schlaf/en/en" (EN) schlafen)  
("ge/schlaf/en/es" (ES) schlafen)  
("ge/schlaf/en/er" (ER) schlafen)  
("ge/schlaf/en/em" (EM) schlafen)  
  
("aus/schlaf/e" (S1 V) ausschlafen\_pk1)  
("aus/schlaf/e" (S13 V ) ausschlafen\_k1)  
("aus/schlaf/en" (P13 A V ) ausschlafen\_pk1)  
...  
("aus/schläf/st" (S2 V) ausschlafen\_p)  
("aus/schläf/t" (S3 V) ausschlafen\_p)  
...

### 14.1.7 Four degrees of regularity in LA-Morph

- *Regular* inflectional paradigm

The paradigm is represented by one lemma without any special surface markings, from which one allomorph is derived, e.g. learn  $\Rightarrow$  learn, or book  $\Rightarrow$  book.

- *Semi-regular* inflectional paradigm

The paradigm is represented by one lemma without any special surface markings, from which more than one allomorph is derived, e.g. derive  $\Rightarrow$  derive, deriv, or wolf  $\Rightarrow$  wolf, wolv.

- *Semi-irregular* inflectional paradigm

The paradigm is represented by one lemma with a special surface marker, from which more than one allomorph is derived, e.g. swim  $\Rightarrow$  swim, swimm, swam, swum.

- *Irregular* inflectional paradigm

The paradigm is represented by several lemmata for suppletive allomorphs which pass through the default rule, e.g. go  $\Rightarrow$  go, went  $\Rightarrow$  went, gone  $\Rightarrow$  gone. The allomorphs serve as input to general combi-rules, as in go/ing.

### 14.1.8 Tabular presentation of the degrees of regularity

	one lemma per paradigm	lemma without markings	one allomorph per lemma
regular	yes	yes	yes
semi-regular	yes	yes	no
semi-irregular	yes	no	no
irregular	no	no	yes

## 14.2 Phenomena of allomorphy

### 14.2.1 Allomorphs of semi-regular nouns

LEX	ALLO1	ALLO2
wolf	wolf	wolv
knife	knife	knive
ability	ability	abiliti
academy	academy	academi
agency	agency	agenci
money	money	moni

### 14.2.2 Allomorphs of semi-irregular nouns

LEX	ALLO1	ALLO2
analysis	analysis	analyses
larva	larva	larvae
stratum	stratum	strati
matrix	matrix	matrices
thesis	thesis	theses
criterion	criterion	criteria



tempo	tempo	tempi
calculus	calculus	calculi

### 14.2.3 Allomorphs of semi-regular verbs

LEX	ALLO1	ALLO2
derive	derive	deriv
dangle	dangle	dangl
undulate	undulate	undulat
accompany	accompany	accompani

### 14.2.4 Allomorphs of semi-irregular verbs

LEX	ALLO1	ALLO2	ALLO3	ALLO4
swlm	swim	swimm	swam	swum
rUN	run	runn	ran	run
bET	bet	bett	bet	bet

### 14.2.5 Allomorphs of semi-regular adjective-adverbials

LEX	ALLO1	ALLO2
able	able	abl
happy	happy	happi
free	free	fre
true	true	tru

### 14.2.6 Definition of the allomorph quotient

The allomorph quotient is the percentage of additional allomorphs relative to the number of base form entries.

### 14.2.7 The allomorph quotient of different languages

*Italian: 37%*

*German: 31%*

*English: 8,97%*

### **14.2.8 Compounds with ‘pseudo-’ contained in Webster’s New Collegiate Dictionary**

pseudoclassic  
pseudopregnancy  
pseudosalt  
pseudoscientific  
etc.

### **14.2.9 Compounds with ‘pseudo-’ not contained in Webster’s New Collegiate Dictionary**

pseudogothic  
pseudomigrane  
pseudoscientist  
pseudovegetarian  
etc.

### **14.2.10 Problem for recognition algorithm**

In order to recognize the highly productive compositions involving the prefix pseudo, the LA-Morph system must provide a general rule-based analysis. As a consequence, the word forms in 14.2.8, are analyzed as ambiguous whereby the second reading stems from the compositional analysis based on the known forms, e.g. pseudo and classic.

### **14.2.11 Solution I**

Automatic removal of all non-elementary base forms from the on-line lexicon.

### **14.2.12 Solution II**

Leaving the non-elementary base forms like 14.2.8 in the lexicon, but selecting the most likely reading after the word form analysis.

### **14.2.13 Solution III**

Using two lexica. One is an elementary lexicon which does not contain any non-elementary base forms. It is used for the categorization and lemmatization of word forms.

The other is a base form lexicon of content words. It assigns semantic representations to base forms including composita and derivata established in use. During word form analysis the two lexica are related by matching the result of lemmatization onto a corresponding – if present – key word of the base form lexicon (cf. 13.4.7).

### 14.2.14 Example of solution III

The compositional analysis of kin/ship would be matched onto kinship in the non-elementary base form lexicon, accessing the proper semantic description. In this way, (i) maximal data coverage – including neologisms – is ensured by a rule based analysis, (ii) the possibility of noncompositional meanings is accounted for, and (iii) unnecessary ambiguities are avoided.

## 14.3 Left-associative segmentation into allomorphs

### 14.3.1 Left-associative letter by letter matching

attempt 1:	W	O	L	F
				×
surface:	W	O	L	V
attempt 2:	W	O	L	V

b14.3.1.pictex

### 14.3.2 Hypothetical examples of English allowing alternative segmentations

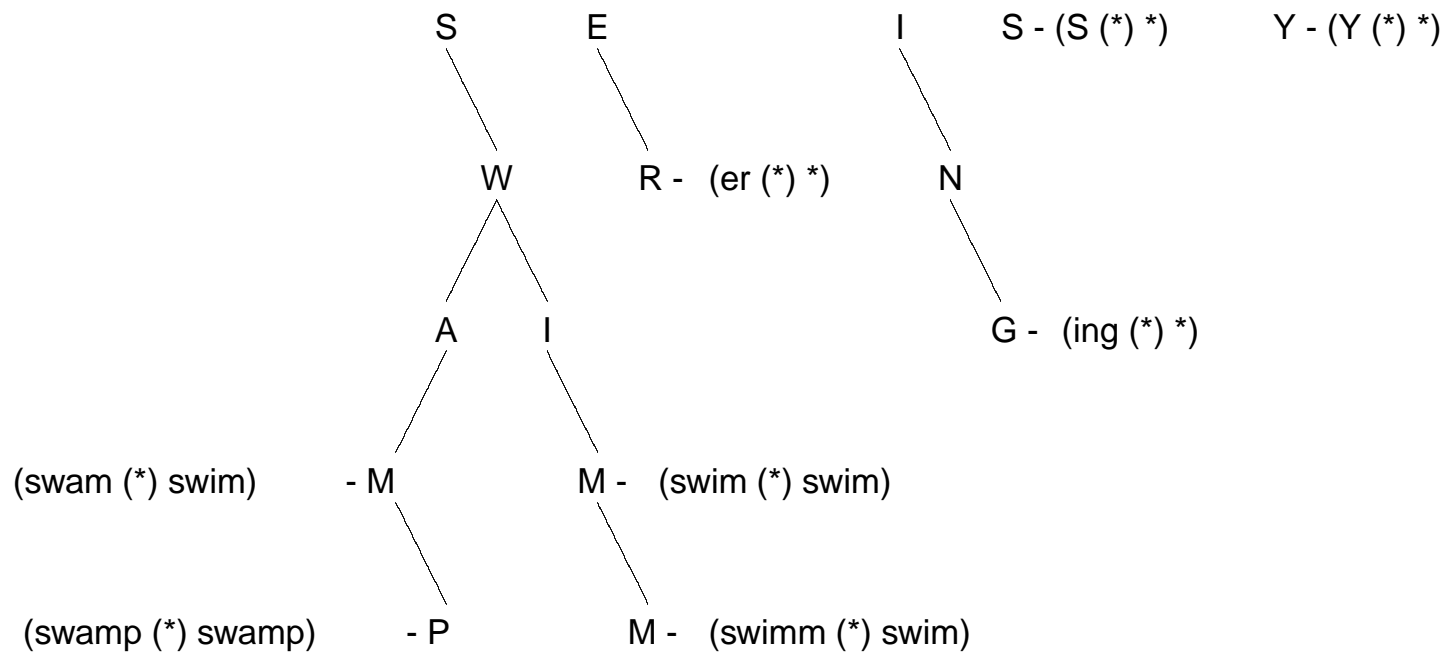
coverage grandparent history lamp/light land/s/end  
 cover/age grandpa/rent hi/story lam/plight land/send  
 cove/rage his/tory

rampage rampart scar/face sing/able war/plane  
 ramp/age ramp/art scarf/ace sin/gable warp/lane  
 ram/page ram/part

### 14.3.3 Alternative segmentations of a word form in German

<i>surface:</i>	Staubecken	Staubecken
<i>segmentation:</i>	Stau/becke/n	Staub/ecke/n
<i>translation:</i>	<i>reservoir</i>	<i>dust corners</i>

### 14.3.4 Storing allomorphs in a trie structure



### 14.3.5 Possibilities after finding an entry in the trie structure

- There are no letters left in the surface of the unknown word form, e.g. SWAM. Then the program simply returns the analysis stored at the last letter, here M.
- There are still letters left in the surface of the unknown word form. Then one of the following alternatives applies:
  - The allomorph found so far *is part* of the word form, as swim in SWIMS. Then the program (i) gives the lexical analysis of swim to the combi-rules of the system and (ii) looks for the next allomorph (here s), starting again from the top level of the trie structure.
  - The allomorph found so far *is not part* of the word form, as swam in SWAMPY. In this case the program continues down the trie structure provided there are continuations. In our example, it will find swamp.

Because it becomes apparent only at the very end of a word form which of these two possibilities applies – or whether they apply simultaneously in the case of an ambiguity – they are pursued simultaneously by the program.



## 14.4 Combi-rules

### 14.4.1 Structure of combi-rules

$$r_n: \begin{array}{cc} \textit{input} & \textit{output} \\ \text{(pattern of start) (pattern of next)} & \Rightarrow \text{rp}_n \text{ (pattern of new start)} \end{array}$$

### 14.4.2 Difference between allo- and combi-rules

Combi-rules differ from allo-rules in that they are defined for different domains and different ranges:

An *allo-rule* takes a lexical entry as input and maps it into one or more allomorphs.

A *combi-rule* takes a word form start and a next allomorph as input and maps it into a new word form start.

### 14.4.3 Tasks of combi-rules

The combi-rules ensure that

1. the allomorphs found in the surface are not combined into ungrammatical word forms, e.g. \*swam+ing or \*swimm+s (input condition),
2. the surfaces of grammatical allomorph combinations are properly concatenated, e.g. swim+s  $\Rightarrow$  swims,
3. the categories of the input pair are mapped into the correct result category, e.g. (NOM V) + (SX S3)  $\Rightarrow$  (S3 V),
4. the correct result is formed on the level of semantic interpretation, and
5. after a successful rule application the correct rule package for the next combination is activated.

### 14.4.4 Derivation of unduly in LA-Morph

```

1 +u [NIL . NIL]
2 +n [NIL . (un (PX PREF) UN)]
RP: {V-START N-START A-START P-START}; fired: P-START
3 +d [(un (PX PREF) UN) . (d (GG) NIL)]
  +d [NIL . NIL]
4 +u [(un (PX PREF) UN) . (du (SR SN) DUE (SR ADJ-V) DUE)]
RP: {PX+A UN+V}; fired: PX+A
  +u [NIL . NIL]
5 L [(un+du (SR ADJ) DUE) . (l (GG) NIL (ABBR) LITER)]
RP: {A+LY}; fired: none
  +l [(un (PX PREF) UN) . NIL]
  +l [NIL . NIL]
6 +y [(un+du (SR ADJ) DUE) . (ly (SX ADV) LY)]
RP: {A+LY}; fired: A+LY
("un/du/ly" (ADV) due)

```

### 14.4.5 Handling of ungrammatical input in LA-Morph

1 +a [NIL . (a (SQ) A)]

2 +b [NIL . NIL]

3 +l [NIL . (abl (SR ADJ-A) ABLE)]

RP: {V-START N-START A-START P-START}; fired: A-START

4 +e [(abl (SR ADJ) ABLE) . NIL]

+e [NIL . (able (ADJ) ABLE)]

RP: {V-START N-START A-START P-START}; fired: none

5 +l [(abl (SR ADJ) ABLE) . NIL]

ERROR

Unknown word form: "ablely"

NIL

## 14.4.6 Parsing the simplex undulate

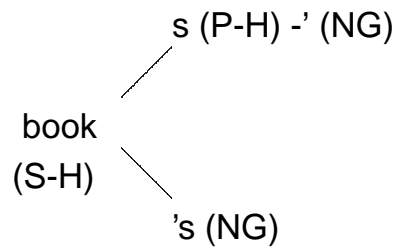
```

1 +u [NIL . NIL]
2 +n [NIL . (un (PX PREF) UN)]
RP:{V-START N-START A-START P-START}; fired: P-START
3 +d [(un (PX PREF) UN) . (d (GG) NIL)]
  +d [NIL . NIL]
4 +u [(un (PX PREF) UN) . (du (SR SN) DUE (SR ADJ-V) DUE)]
RP:{PX+A UN+V}; fired: PX+A
  +u [NIL . NIL]
5 +l [(un+du (SR ADJ) DUE) . (l (GG) NIL (ABBR) LITER)]
RP:{A+LY}; fired: none
  +l [(un (PX PREF) UN) . NIL]
  +l [NIL . NIL]
6 +a [(un+du (SR ADJ) DUE) . NIL]
  +a [NIL . NIL]
7 +t [(un+du (SR ADJ) DUE) . NIL]
  +t [NIL . (undulat (SR A V) UNDULATE)]
RP:{V-START N-START A-START P-START}; fired: V-START
8 +e [(un+du (SR ADJ) DUE) . (late (ADJ-AV) LATE (ADV) LATE)]
RP:{A+LY}; fired: none
  +e [(undulat (SR A V) UNDULATE) . NIL]
  +e [NIL . (undulate (SR NOM A V) UNDULATE)]
RP:{V-START N-START A-START P-START}; fired: V-START
("undulate" (NOM A V) UNDULATE)

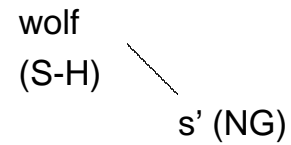
```

## 14.5 Concatenation patterns

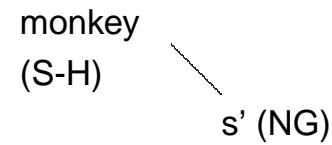
### 14.5.1 Concatenation patterns of English nouns



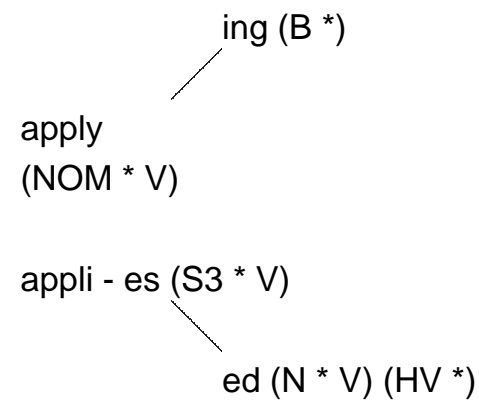
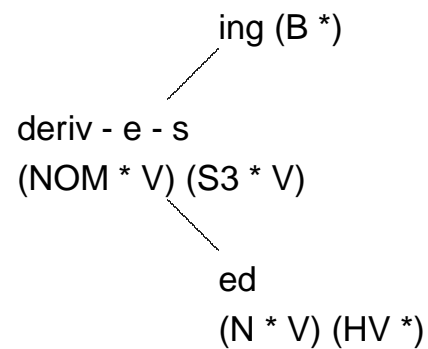
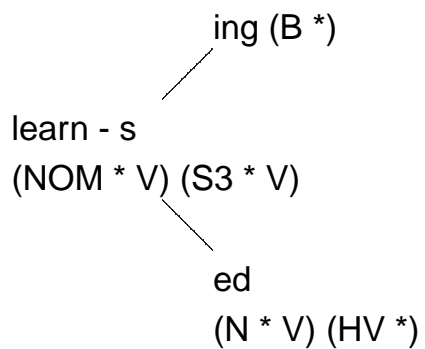
wolv - es - '  
(P-H) (NG)



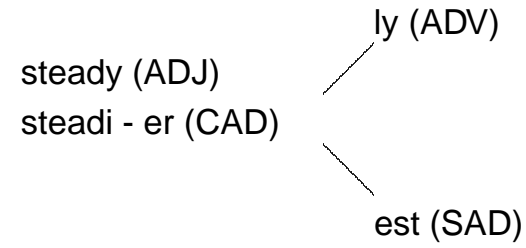
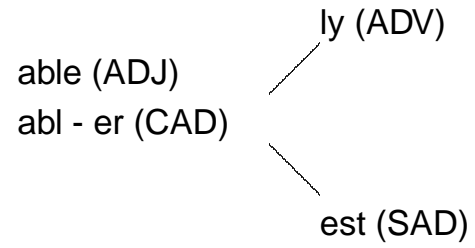
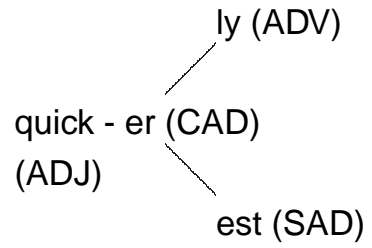
monki - es - '  
(P-H) (NG)



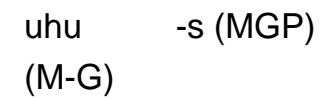
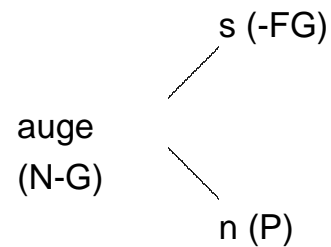
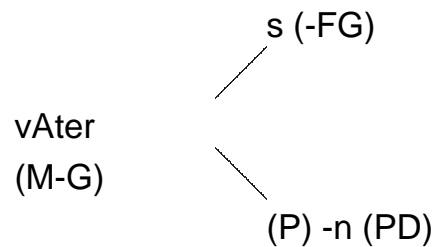
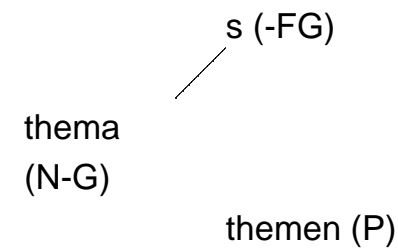
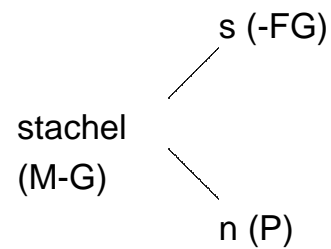
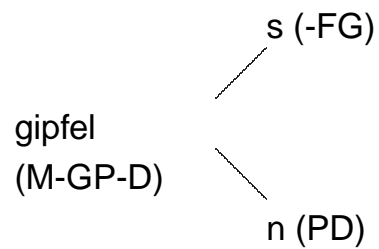
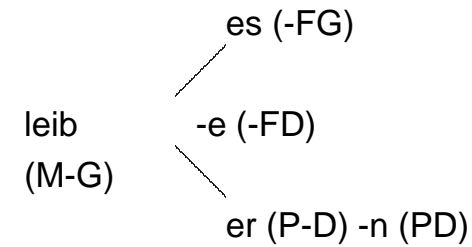
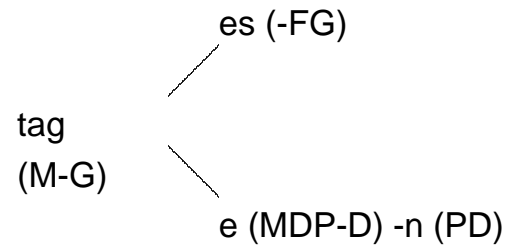
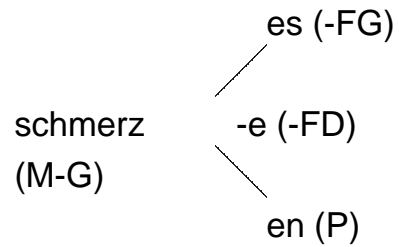
### 14.5.2 Concatenation patterns of English verbs



### 14.5.3 Concatenation patterns of adjective-adverbs



## 14.5.4 Concatenation patterns of German nouns





---

braten (M-GP)	-s (-FG)	hAnd (F)	-e (P-D) -n (PD)	frau (F)	-en (P)
drangsal (F)	-e (P-D) -n (PD)	kenntnis (F)	-se (P-D) -n (PD)	mUtter (F)	- (P-D) -n (PD)

### 14.5.5 Category segments of German noun forms

MN	= Masculinum Nominativ	(Bote)
M-G	= Masculinum no Genitiv	(Tag)
-FG	= no Femininum Genitiv	(Tages, Kindes)
-FD	= no Femininum Dativ	(Schmerze, Kinde)
M-NP	= Masculinum no Nominativ or Plural	(Boten)
M-GP	= Masculinum no Genitiv or Plural	(Braten)
MGP	= Masculinum Genitiv or Plural	(Uhus)
M-GP-D	= Masculinum no Genitiv or Plural no Dativ	(Gipfel)
F	= Femininum	(Frau)
N-G	= Neutrum no Genitiv	(Kind)
NG	= Neutrum Genitiv	(Kindes)
ND	= Neutrum Dativ	(Kinde)
N-GP	= Neutrum no Genitiv or Plural	(Leben)
N-GP-D	= Neutrum no Genitiv or Plural no Dativ	(Wasser)
NDP-D	= Neutrum Dativ or Plural no Dativ	(Schafe)
P	= Plural	(Themen)
P-D	= Plural no Dativ	(Leiber)
PD	= Plural Dativ	(Leibern)

# 15. Corpus analysis

## 15.1 Implementation and application of grammar systems

### 15.1.1 Parts of a grammar system

- Formal algorithm
- Linguistic method

### 15.1.2 Options for grammar system of word form recognition

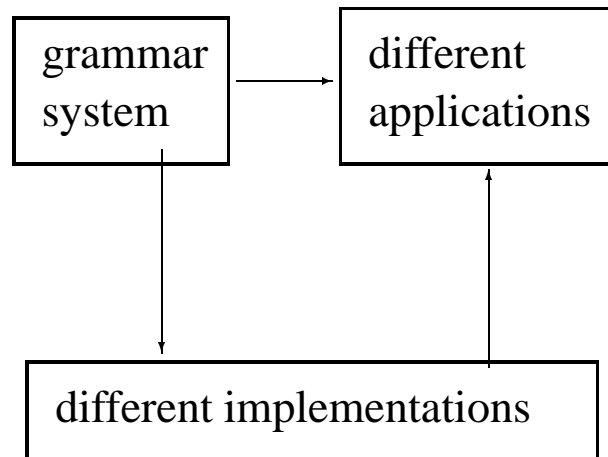
- Formal algorithm:  
C- (Section 7.4), PS- (Section 8.1), or LA-grammar (Section 10.2).
- Linguistic method:  
Word form, morpheme, or allomorph method (cf. Section 13.5).

### 15.1.3 Minimal standard of well-defined grammar systems

A grammar system is well-defined only if it simultaneously allows

1. different *applications* in a given *implementation*, and
2. different *implementations* in a given *application*.

### 15.1.4 Modularity of a grammar system



### 15.1.5 Different implementations of LA-morphology

1988 in LISP (Hausser & Todd Kaufmann)

1990 in C (Hausser & Carolyn Ellis)

1992 in C, 'LAMA' (Norbert Bröker)

1994 in C, 'LAP' (Gerald Schüller)

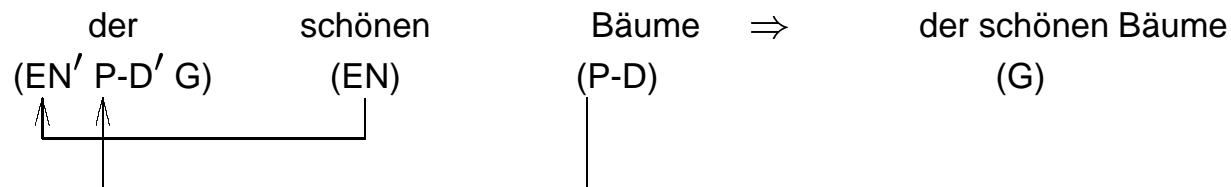
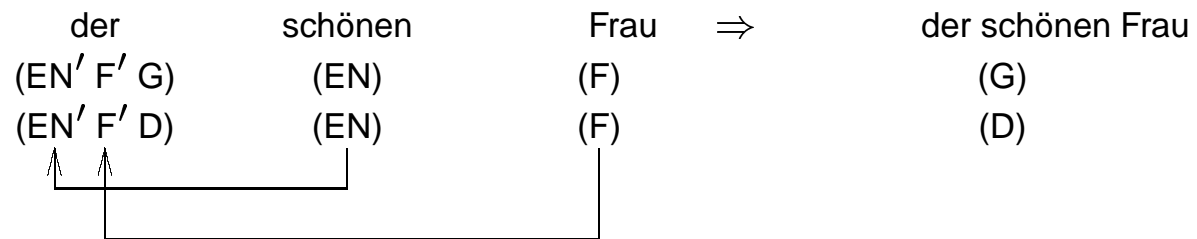
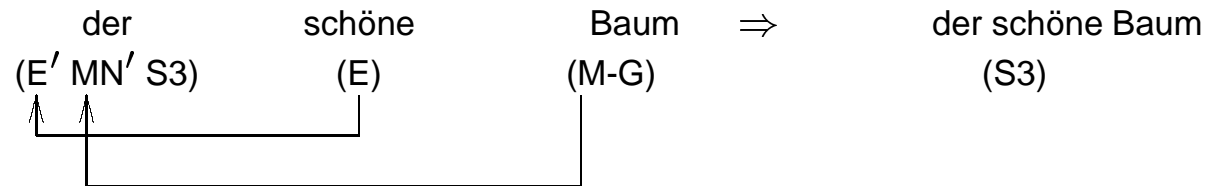
1995 in C, 'Malaga' (Björn Beutel)

### 15.1.6 Structural principles common to different LA-Morph implementations

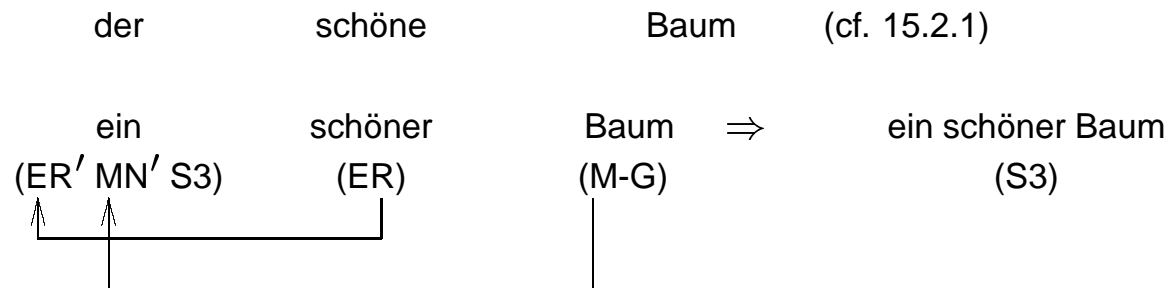
- Specification of the allo- (cf. 14.1.1) and the combi-rules (cf. 14.4.1) on the basis of patterns which are matched onto the input.
- Storage of the analyzed allomorphs in a trie structure and their left-associative lookup with parallel pursuit of alternative hypotheses (cf. Section 14.3).
- Modular separation of motor, rule components, and lexicon, permitting a simple exchange of these parts, for example in the application of the system to new domains or languages.
- Use of the same motor and the same algorithm for the combi-rules of the morphological, syntactic, and semantic components during analysis.
- Use of the same rule components for analysis and generation in morphology, syntax, and semantics.

## 15.2 Subtheoretical variants

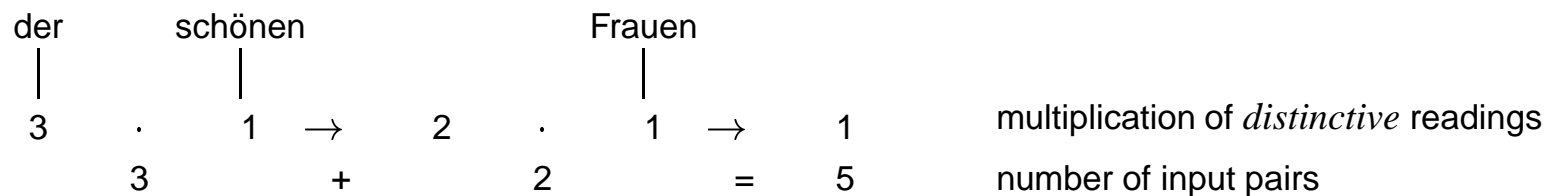
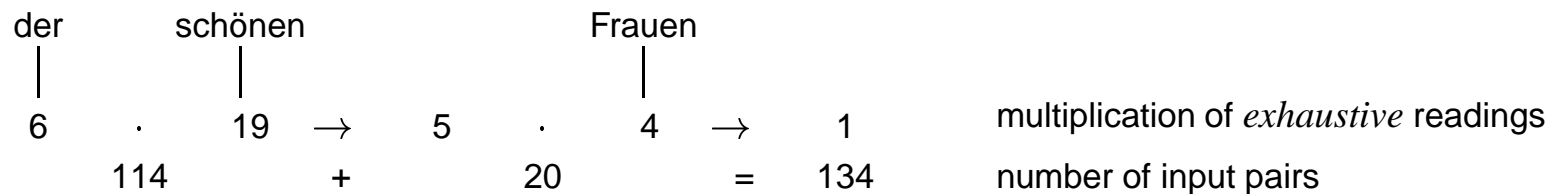
### 15.2.1 Combinatorics of the German determiner der



### 15.2.2 Agreement of adjective-ending with determiner



### 15.2.3 Exhaustive versus distinctive categorization in deriving der schönen Frauen



### 15.2.4 Representing lexical readings via different entries

[der (E' MN' S3) DEF-ART]

[der (EN' F' G&D) DEF-ART]

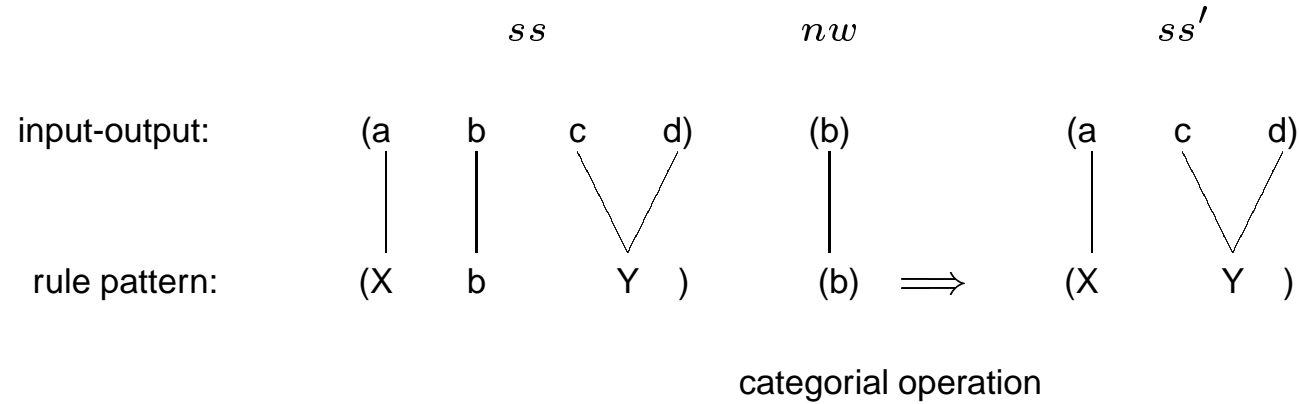
[der (EN' P-D' G) DEF-ART]

### 15.2.5 Representing lexical readings via multcats

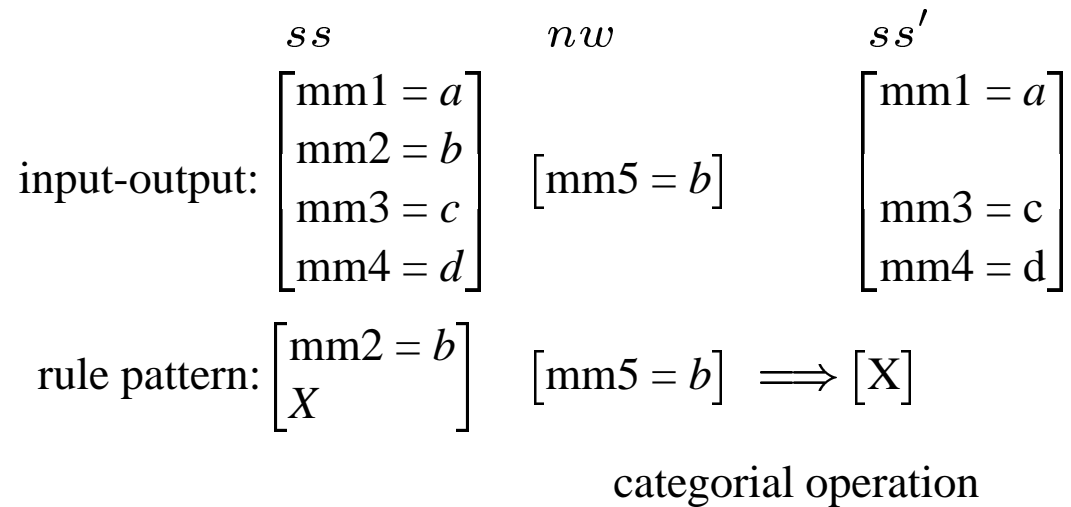
[der ((E' MN' S3) (EN' F' G&D) (EN' P-D' G)) DEF-ART]



## 15.2.6 List-based matching (LAP)



## 15.2.7 Feature-based matching (Malaga)



## 15.3 Building corpora

### 15.3.1 Text genres of the Brown and the LOB corpus

	Brown	LOB
A Press: reportage	44	44
B Press: editorial	27	27
C Press: reviews	17	17
D Religion	17	17
E Skills, trade, and hobbies	36	38
F Popular lore	48	44
G Belle lettres, biography, essays	75	77
H Miscellaneous (government documents, foundation records, industry reports, college catalogues, industry house organ)	30	38
J Learned and scientific writing	80	80
K General fiction	29	29
L Mystery and detective fiction	24	24
M Science fiction	6	6
N Adventure and western fiction	29	29
P Romance and love story	29	29
R Humour	9	9
<hr/>		
Total	500	500

### 15.3.2 Kučera & Francis' desiderata for the construction of corpora

1. Definite and specific delimitation of the language texts included, so that scholars using the Corpus may have a precise notion of the composition of the material.
2. Complete synchronicity; texts published in a single calendar year only are included.
3. A predetermined ratio of the various genres represented and a selection of individual samples through a random sampling procedure.
4. Accessibility of the Corpus to automatic retrieval of all information contained in it which can be formally identified.
5. An accurate and complete description of the basic statistical properties of the Corpus and of several subsets of the Corpus with the possibility of expanding such analysis to other sections or properties of the Corpus as may be required.

### 15.3.3 Difficulties with achieving a representative and balanced corpus

'Genre' is not a well-defined concept. Thus genres that have been distinguished so far have been identified on a purely intuitive basis. No empirical evidence has been provided for any of the genre distinctions that have been made.

N. Oostdijk 1988

## 15.4 Distribution of word forms

### 15.4.1 Definition of rank

The position of a word form in the frequency list

### 15.4.2 Definition of frequency class (F-class)

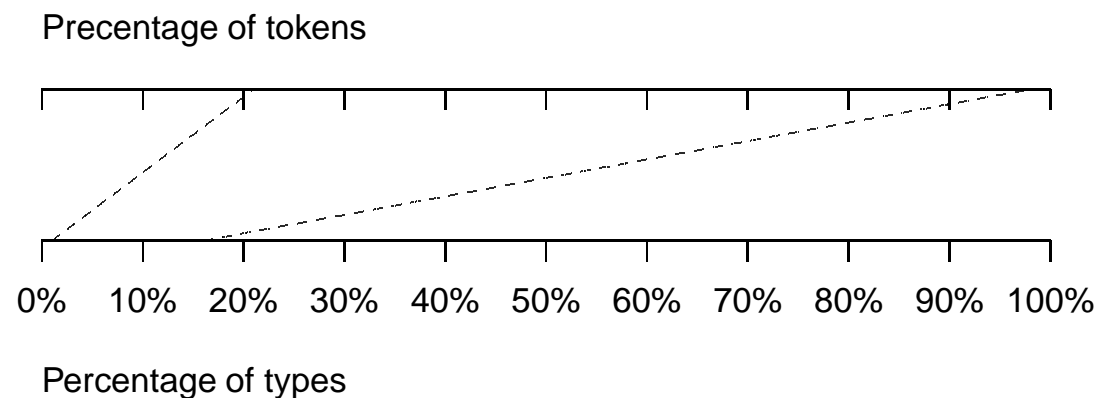
F-class =<sub>def</sub> [frequency of types # number of types]

There are much fewer F-classes in a corpus than ranks. In the BNC, for example, 655 270 ranks result in 5 301 F-classes. Thus, the number of the F-classes is only 0.8% of the number of ranks. Because of their comparatively small number the F-classes are well suited to bring the type-token correlation into focus.

### 15.4.3 Type-token distribution in the BNC (*surface-based*)

F-class	start_r	end_r	types	tokens	types-%	tokens-%	
beginning (the first 9 F-classes)							
1 (the)	1	1	1	5776399	0.000152	6.436776	
2 (of)	2	2	1	2789563	0.000152	3.108475	
3 (and)	3	3	1	2421306	0.000152	2.698118	
4 (to)	4	4	1	2332411	0.000152	2.599060	
5 (a)	5	5	1	1957293	0.000152	2.181057	
6 (in)	6	6	1	1746891	0.000152	1.946601	
7 (is)	7	7	1	893368	0.000152	0.995501	
8 (that)	8	8	1	891498	0.000152	0.993417	
9 (was)	9	9	1	839967	0.000152	0.935995	
sums			9	19648 696	0.001368 %	21.895 %	
middle (9 samples)							
1000	1017	1017	1	9608	0.000152	0.010706	
2001	2171	2171	1	4560	0.000152	0.005081	tokens
3000	3591	3591	1	2521	0.000152	0.002809	per
3500	4536	4536	1	1857	0.000152	0.002069	type:
4000	5907	5910	4	5228	0.000607	0.005826	1307
4500	8332	8336	5	4005	0.000758	0.004463	801
4750	10842	10858	17	9367	0.002579	0.010438	551
5000	16012	16049	38	11438	0.005764	0.012746	301
5250	44905	45421	517	26367	0.078420	0.029381	51
end (the last 9 F-classes)							
5292	108154	114730	6577	59193	0.997620	0.065960	9
5293	114731	122699	7969	63752	1.208763	0.071040	8
5294	122700	132672	9973	69811	1.512736	0.077792	7
5295	132673	145223	12551	75306	1.903775	0.083915	6
5296	145224	161924	16701	83505	2.533260	0.093052	5
5297	161925	186302	24378	97512	3.697732	0.108660	4
5298	186303	225993	39691	119073	6.020456	0.132686	3
5299	225994	311124	85131	170262	12.912938	0.189727	2
5300	311125	659269	348145	348145	52.807732	0.387946	1
sums			551 116	1 086 559	83.595012 %	1.210778 %	

### 15.4.4 Correlation of type and token frequency



### 15.4.5 Semantic significance

The higher the frequency, the lower the semantic significance.

Examples: the, of, and, to, a, in, that, was

The lower the frequency, the higher the semantic significance.

Examples: audiophile, butternut, customhouse, dustheap

### 15.4.6 Hapaxlegomena

Word forms in a corpus which occur only once.

### 15.4.7 Zipf's law

frequency · rank = constant

### 15.4.8 Illustration of Zipf's law

word form	rank	·	frequency	=	constant
the	1	·	5 776 399	=	5 776 399
and	2	·	2 789 563	=	5 579 126
...					
was	9	·	839 967	=	7 559 703
...					
holder	3 251	·	2 870	=	9 330 370

## 15.5 Statistical tagging

### 15.5.1 Top of Brown corpus frequency list

69971-15-500	THE	21341-15-500	IN
36411-15-500	OF	10595-15-500	THAT
28852-15-500	AND	10099-15-485	IS
26149-15-500	TO	9816-15-466	WAS
23237-15-500	A	9543-15-428	HE

The entry 9543-15-428 HE, for example, indicates that the word form HE occurs 9 543 times in the Brown corpus, in all 15 genres, and in 428 of the 500 sample texts.

### 15.5.2 Statistical tagging

is based on categorizing by hand – or half automatically with careful post-editing – a small part of the corpus, called the *core corpus*. The categories used for the classification are called *tags* or *labels*. After hand-tagging the core corpus, the probabilities of the transitions from one word form to the next are computed by means of *Hidden Markov Models* (HMMs).



### 15.5.3 Subset of the *basic (C5) tagset*

AJ0 Adjective (general or positive) (e.g. good, old, beautiful)

CRD Cardinal number (e.g. one, 3, fifty-five, 3609)

NN0 Common noun, neutral for number (e.g. aircraft, data, committee)

NN1 Singular common noun (e.g. pencil, goose, time, revelation)

NN2 Plural common noun (e.g. pencils, geese, times, revelations)

NP0 Proper noun (e.g. London, Michael, Mars, IBM)

UNC Unclassified items

VVB The finite base form of lexical verbs (e.g. forget, send, live, return)

VVD The past tense form of lexical verbs (e.g. forgot, sent, lived, returned)

VVG The -ing form of lexical verbs (e.g. forgetting, sending, living, returning)

VVI The infinitive form of lexical verbs (e.g. forget, send, live, return)

VVN The past participle form of lexical verbs (e.g. forgotten, sent, lived, returned)

VVZ The -s form of lexical verbs (e.g. forgets, sends, lives, returns)

### 15.5.4 Sample from the alphabetical word form list of the BNC

1 activ nn1-np0 1	8 activating aj0-nn1 6
1 activ np0 1	47 activating aj0-vvg 22
2 activa nn1 1	3 activating nn1-vvg 3
3 activa nn1-np0 1	14 activating np0 5
4 activa np0 2	371 activating vvg 49
1 activatd nn1-vvb 1	538 activation nn1 93
21 activate np0 4	3 activation nn1-np0 3
62 activate vvb 42	2 activation-energy aj0 1
219 activate vvi 116	1 activation-inhibition aj0 1
140 activated aj0 48	1 activation-synthesis aj0 1
56 activated aj0-vvd 26	1 activation. nn0 1
52 activated aj0-vvn 34	1 activation/ unc 1
5 activated np0 3	282 activator nn1 30
85 activated vvd 56	6 activator nn1-np0 3
43 activated vvd-vvn 36	1 activator/ unc 1
312 activated vvn 144	1 activator/ unc 1
1 activatedness nn1 1	7 activator/tissue unc 1
88 activates vvz 60	61 activators nn2 18
5 activating aj0 5	1 activators np0 1

Each entry consists (i) of a number detailing the frequency of the tagged word form in the whole corpus, (ii) the surface of the word form, (iii) the label, and (iv) the number of texts in which the word form was found under the assigned label.

### 15.5.5 Error rates in statistical tagging

The error rate of CLAWS4 is quoted by Leech 1995 at 1.7%, which may seem very good. However, given that the last 1.2% of the low frequency tokens requires 83.6% of the types (cf. 15.4.4), an error rate of 1.7% may also represent a very bad result – namely that about 90% of the types are not analyzed or not analyzed correctly. This conclusion is born out by a closer inspection of sample 15.5.4.

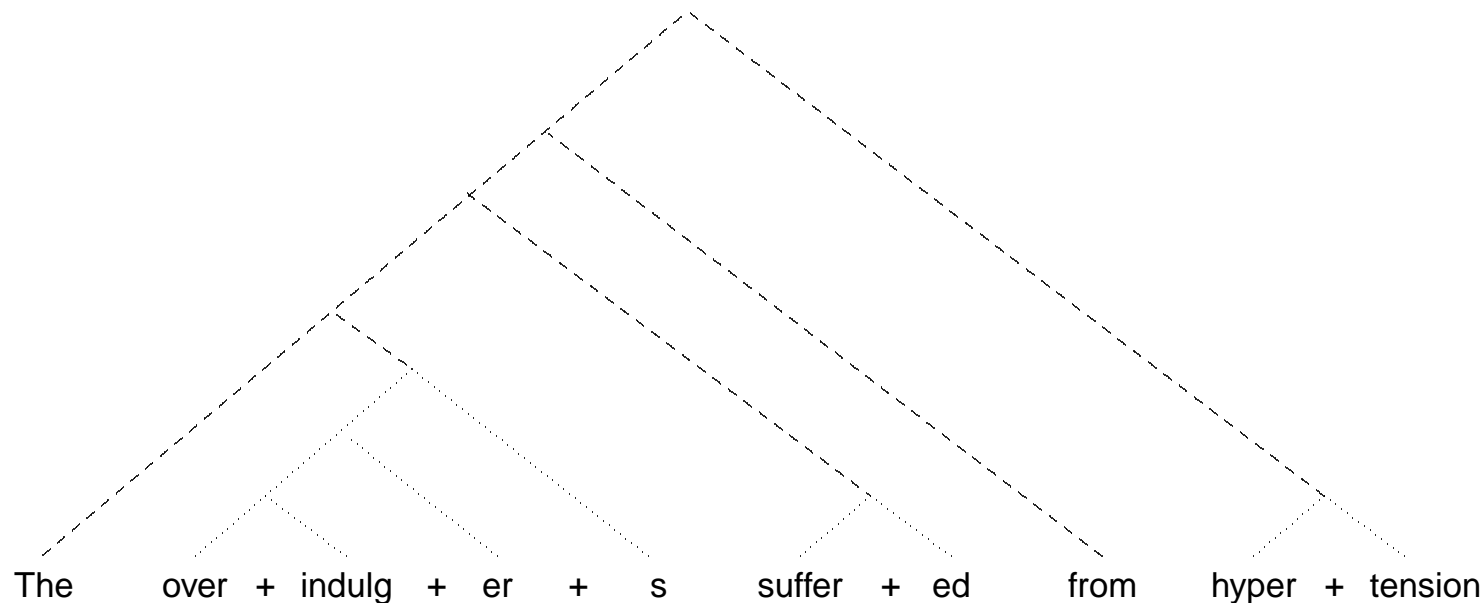
### 15.5.6 Weaknesses of statistical tagging

1. The categorization is too unreliable to support rule-based syntactic parsing.
2. Word forms can be neither reduced to their base forms (lemmatization) nor segmented into their allomorphs or morphemes.
3. The overall frequency distribution analysis of a corpus is distorted by an artificial inflation of types (e.g., 37.5% in the BNC).
4. Even if the tagger is successfully improved as a whole, its results can never be more than probabilistically-based conjectures.

## 16. Basic concepts of syntax

### 16.1 Delimitation of morphology and syntax

#### 16.1.1 Correlation of LA-morphology and LA-syntax



The tree structures of LA-morphology and LA-syntax both satisfy the SLIM-theoretic principles of surface compositionality (S) and time-linear composition (L). However, their respective time-linear compositions occur in different phases.

### 16.1.2 Treatment of idioms in morphology or syntax?

A syntactic treatment is generally motivated in idioms which (i) retain their compositional meaning as an option, (ii) are subject to normal variations of word order, and (iii) exhibit internal inflectional variation. Otherwise idioms should be handled in the lexicon (e.g. over-the-counter).

### 16.1.3 Correlation of morphology and syntax in different types of language

Some natural languages compose meaning<sub>1</sub> mainly in the syntax (e.g. Chinese) and others mainly in morphology (e.g. Eskimo in which long chains of morphemes are concatenated into a single word form such as [a:wlis-ut-iss?ar-si-niarpu-na] *I am looking for something suitable for a fish-line*). This alternative exists also within a given natural language. For example, in English the complex concept denoted by the word form *overindulgers* may roughly be expressed analytically as *people who eat and drink too much*.

### 16.1.4 Combination principles of syntax

1. *Valency*
2. *Agreement*
3. *Word order*

## 16.2 Valency

### 16.2.1 The notions valency carrier and valency filler

go back to the French linguist L. TESNIÈRE 1959, who borrowed them from chemistry. The valency positions of a carrier must be filled, or canceled, by compatible fillers in order for an expression to be syntactically and semantically complete.

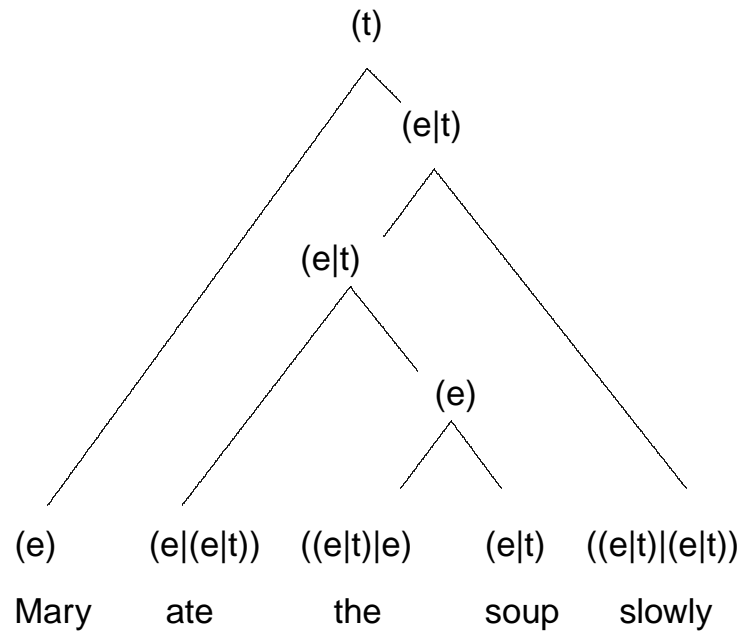
### 16.2.2 Coding the structure of valency carriers in LA-grammar

Composite syntactic categories are defined as lists of category segments. For example, the English verb form *ate* is analyzed as follows.

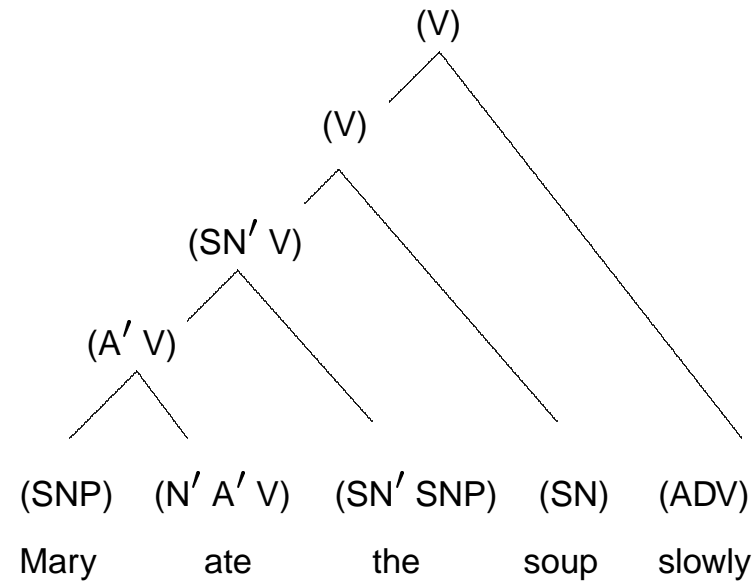
[ate (N' A' V) eat]

## 16.2.3 Carriers, fillers, and modifiers in CG and LAG

*C-grammar analysis*



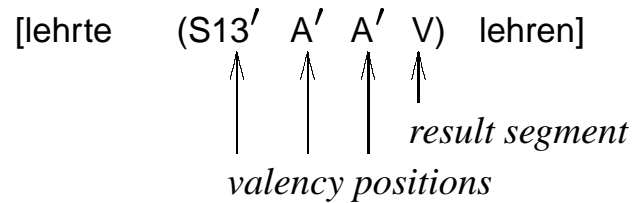
*LA-grammar analysis*



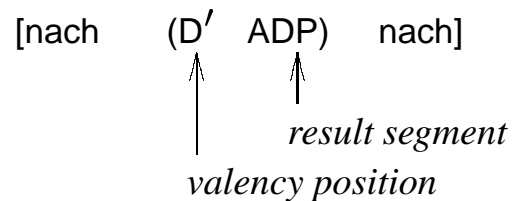




- The three-place verb form *lehrte* (*taught*):



- The one-place preposition *nach* (*after*):



### 16.2.5 Category structure of valency fillers and modifiers

[Bücher (P-D) buch]

(*books*)

[ihm (D) er]

(*him*)

[gestern (ADV) gestern]

(*yesterday*)

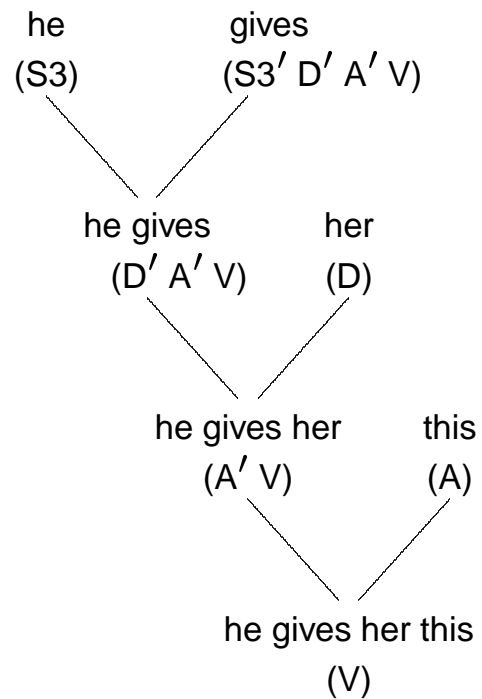
Valency carriers may also function as valency fillers using their result segment, e.g V, as the filler segment. In this case, the segments representing valency positions are attached at the beginning of the category resulting from the composition.

## 16.3 Agreement

### 16.3.1 Agreement violation in English

\*Every girls need a mother.

### 16.3.2 Identity-based agreement in a simple LA-syntactic analysis



### 16.3.3 An LA-grammar for 6.3.2 (*LA-plaster*)

$$LX =_{def} \{ [\text{he } (S3) *], [\text{her } (D) *], [\text{this } (A) *], [\text{gives } (S3' D' A' V) *] \}$$

$$ST_S =_{def} \{ [(S3) \{MAIN+FV\} ] \}$$

$$MAIN+FV: (S3) (S3' D' A' V) \Rightarrow (D' A' V) \{FV+MAIN1\}$$

$$FV+MAIN1: (D' A' V) (D) \Rightarrow (A' V) \{FV+MAIN2\}$$

$$FV+MAIN2: (A' V) (A) \Rightarrow (V) \{ \}$$

$$ST_F =_{def} \{ [(V) rp_{FV+MAIN2} ] \}$$

### 16.3.4 Example of an error in identity-based agreement

I (S1)	+	gives (S3' D' A' V)	⇒	Error: ungrammatical continuation
-----------	---	------------------------	---	-----------------------------------

## 16.4 Free word order in German (*LA-D1*)

### 16.4.1 Word order variations in a declarative main clause of German

Der Mann gab der Frau den Strauß.

(the man gave the woman the bouquet.)

Der Mann gab den Strauß der Frau.

(the man gave the bouquet the woman.)

Der Frau gab der Mann den Strauß.

(the woman gave the man the bouquet.)

Der Frau gab den Strauß der Mann.

(the woman gave the bouquet the man.)

Den Strauß gab der Mann der Frau.

(the bouquet gave the man the woman.)

Den Strauß gab der Frau der Mann.

(the bouquet gave the woman the man.)

### 16.4.2 Word order violation in German

\*Der Mann der Frau gab einen Strauß.

(the man the woman gave the bouquet.)

### 16.4.3 Free canceling of valency positions in a carrier of German

Der Mann + gab  $\Rightarrow$  Der Mann gab  
 (S3) (S3' D' A' V) (D' A' V)

Der Frau + gab  $\Rightarrow$  Der Frau gab  
 (D) (S3' D' A' V) (S3' A' V)

Den Strauß + gab  $\Rightarrow$  Den Strauß gab  
 (A) (S3' D' A' V) (S3' D' V)

### 16.4.4 German LA-grammar with partial free word order

$LX =_{def} \{ [er (S3) *], [ihr (D) *], [das (A) *], [gab (S3' D' A' V) *] \}$

Variable definition:  $np \in \{D, A\}$ , with  $np'$  correspondingly  $D'$  or  $A'$

$x, y = .?.?.?.?$  (i.e. an arbitrary sequence up to length 4)

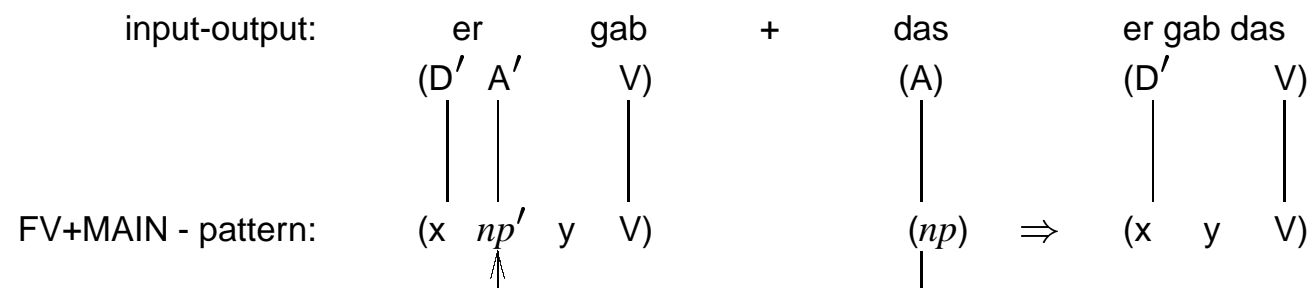
$ST_S =_{def} \{ [(S3) \{MAIN+FV\}] \}$

$MAIN+FV: (S3) (S3' D' A' V) \Rightarrow (D A V) \{FV+MAIN\}$

$FV+MAIN: (x np' y V) (np) \Rightarrow (x y V) \{FV+MAIN\}$

$ST_F =_{def} \{ [(V) rp_{FV+MAIN}] \}$

### 16.4.5 FV+MAIN matching a next word accusative





### 16.4.8 German LA-grammar with free word order (*LA-DI*)

$$LX =_{def} \{ [er (S3) *], [ihr (D) *], [das (A) *], [gab (S3' D' A' V) *] \}$$

Variable definition:  $np \in \{S3, D, A\}$ , with  $np'$  correspondingly  $S3'$ ,  $D'$  or  $A'$   
 $x, y = .?.?.?.?$  (i.e. an arbitrary sequence up to length 4)

$$ST_S =_{def} \{ [(np) \{MAIN+FV\}] \}$$

$$MAIN+FV: (np) (x np' y V) \Rightarrow (x y V) \{FV+MAIN\}$$

$$FV+MAIN: (x np' y V) (np) \Rightarrow (x y V) \{FV+MAIN\}$$

$$ST_F =_{def} \{ [(V) rp_{FV+MAIN}] \}$$

### 16.4.9 Word order variants of *LA-DI*

er gab ihr das

das gab er ihr

ihr gab er das

er gab das ihr

das gab ihr er

ihr gab das er







## 16.5 Fixed word order in English (*LA-E1*)

### 16.5.1 Fixed canceling of valency positions in a carrier of English

Peter + gave  $\Rightarrow$   
 (SNP) (N' D' A' V)

Peter gave  
 (D' A' V)

Peter gave + Mary  $\Rightarrow$   
 (D' A' V) (SNP)

Peter gave Mary  
 (A' V)

Peter gave Mary + books  $\Rightarrow$   
 (A' V) (PN)

Peter gave Mary books  
 (V)

### 16.5.2 English LA-grammar with fixed word order (LA-E1)

$LX =_{def} \{ [\text{Peter (SNP) *}], [\text{Mary (SNP) *}], [\text{books (PN) *}],$   
 $[\text{gave (N' D' A' V) *}] \}$

Variable definition:  $np \in \{\text{SNP, PN}\}$ ,  $np' \in \{\text{N', D', A'}\}$ ,  
 $x = .?.?.?.?$  (i.e. an arbitrary sequence up to length 4)

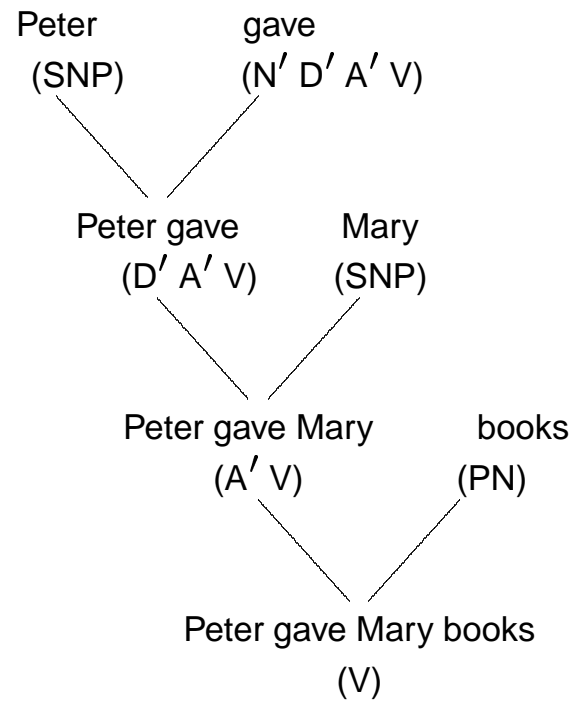
$ST_S =_{def} \{ [(x) \{\text{NOM+FV}\}] \}$

$\text{NOM+FV: } (np) (np' x V) \Rightarrow (y V) \{\text{FV+MAIN}\}$

$\text{FV+MAIN: } (np' x V) (np) \Rightarrow (y V) \{\text{FV+MAIN}\}$

$ST_F =_{def} \{ [(V) rp_{\text{FV+MAIN}}] \}$

### 16.5.3 Derivation in *LA-E1* (definition-based agreement)



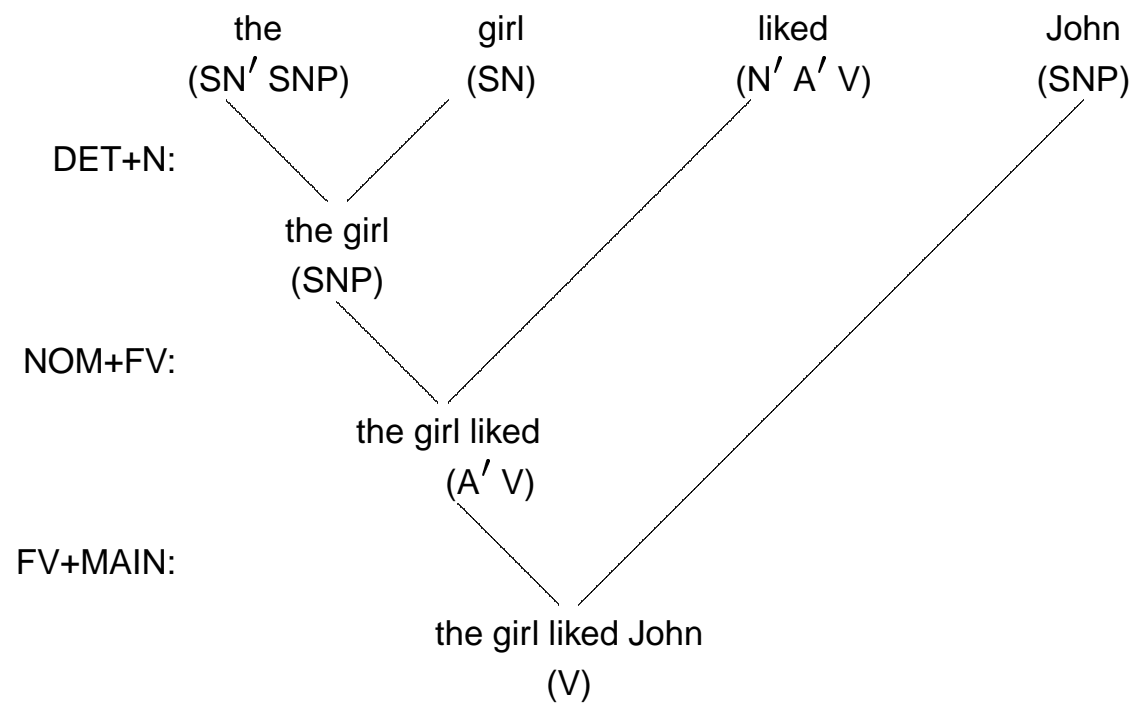
## 17. LA-syntax for English

### 17.1 Complex fillers in pre- and postverbal position

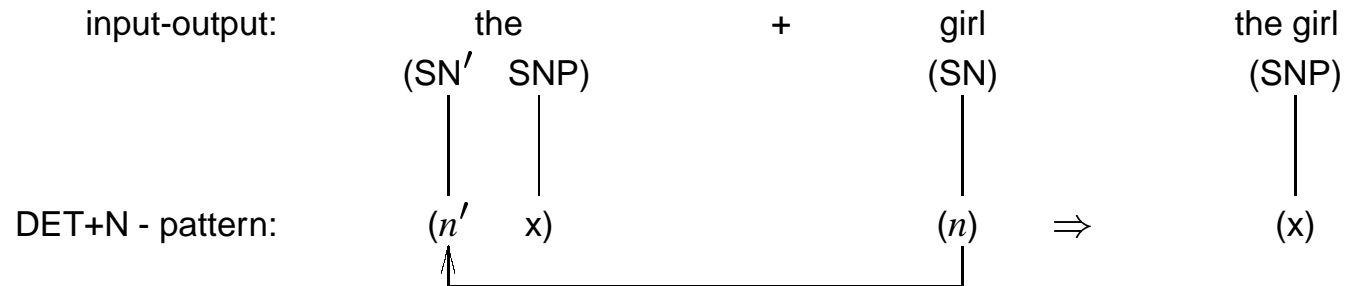
#### 17.1.1 Determiner and noun categories of English

<i>categories</i>	<i>surfaces</i>	<i>examples of lemmata</i>
singular and plural determiners:		
(SN' SNP)	a, an, every, the	[a (SN' SNP) *]
(PN' PNP)	all, several, the	[all (PN' PNP) *]
singular and plural nouns:		
(SN)	man, woman, book, car	[woman (SN) *]
(PN)	men, women, books, cars	[men (PN) *]

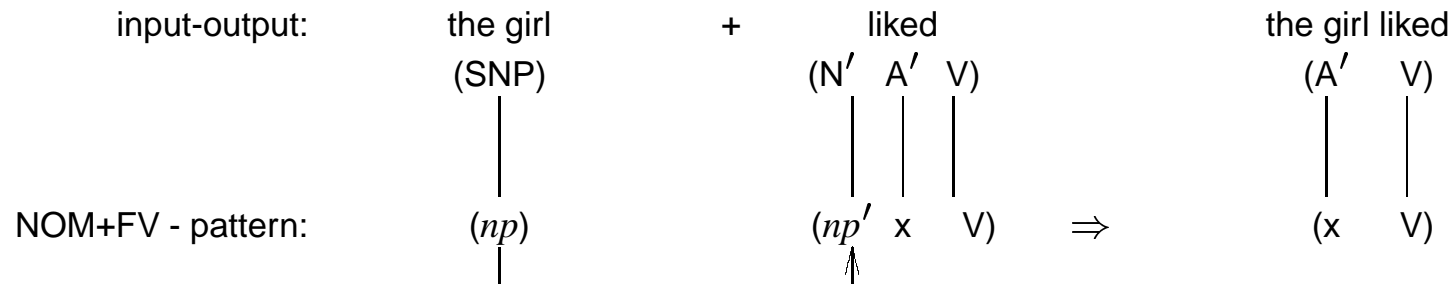
### 17.1.2 Complex noun phrase before the valency carrier



### 17.1.3 Preverbal application of Det+N



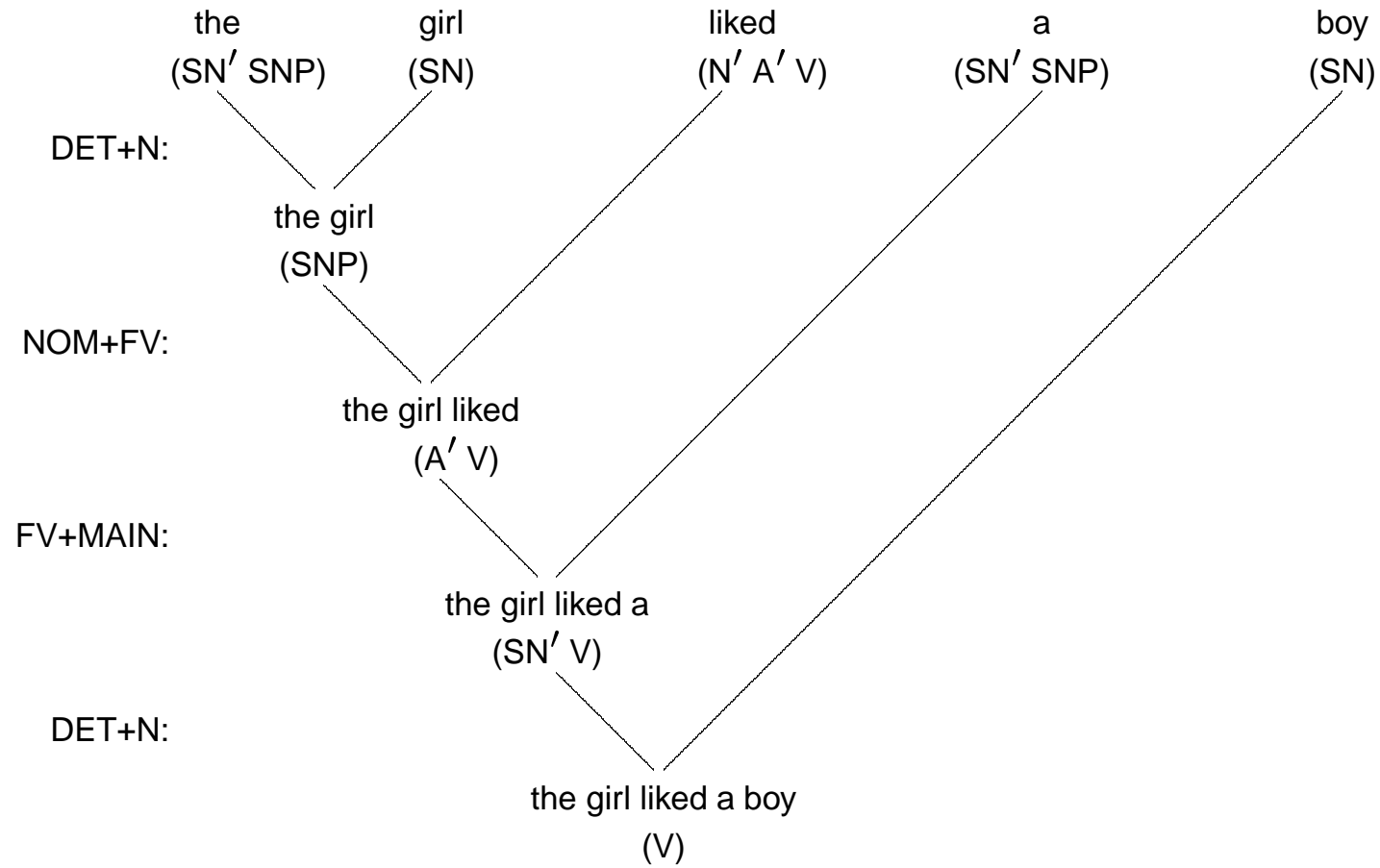
### 17.1.4 Application of NOM+FV to complex nominative NP







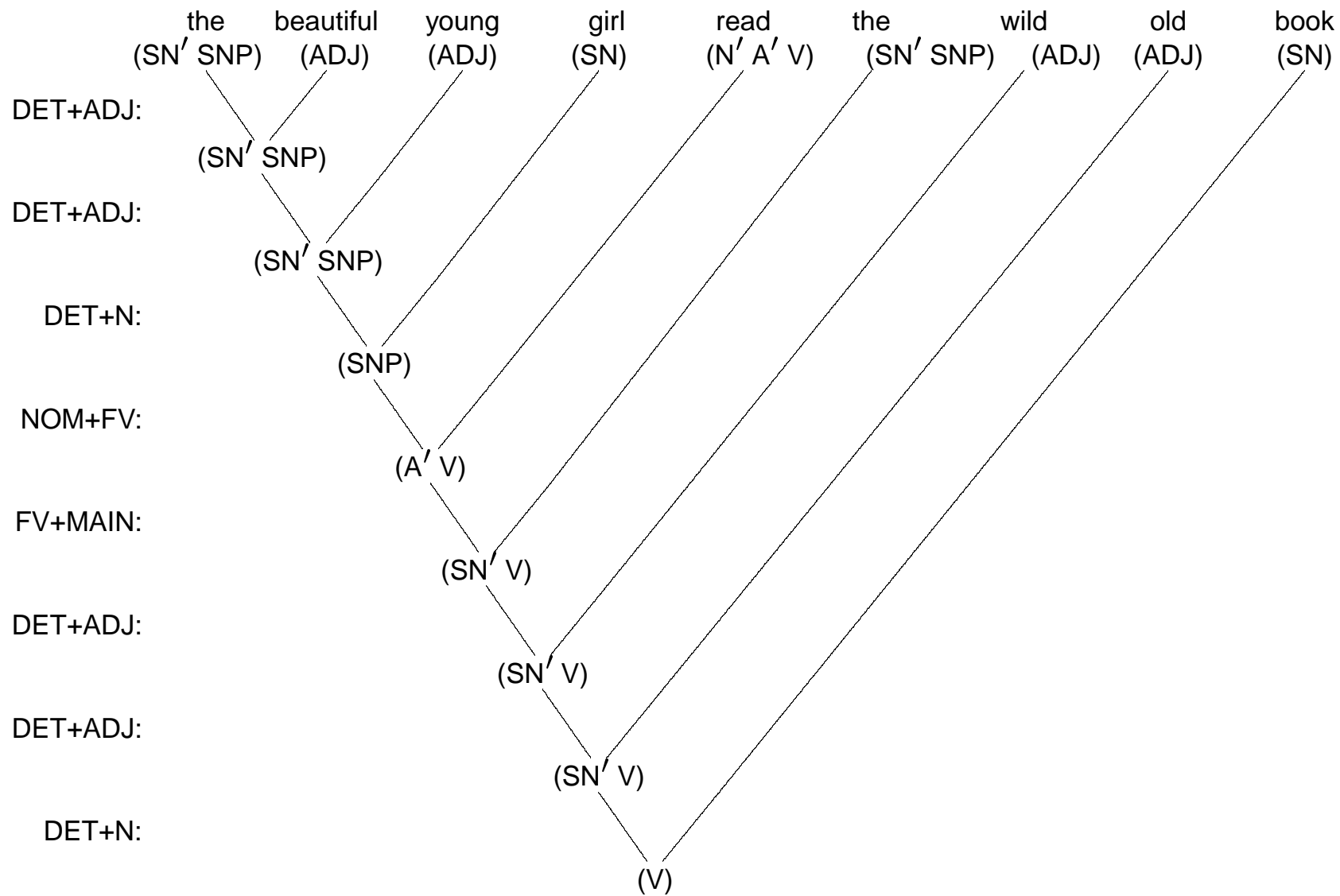
### 17.1.6 Complex noun phrase after valency carrier







### 17.1.10 Complex noun phrases with adjectives



## 17.2 English field of referents

### 17.2.1 Categories of nominal valency fillers in English

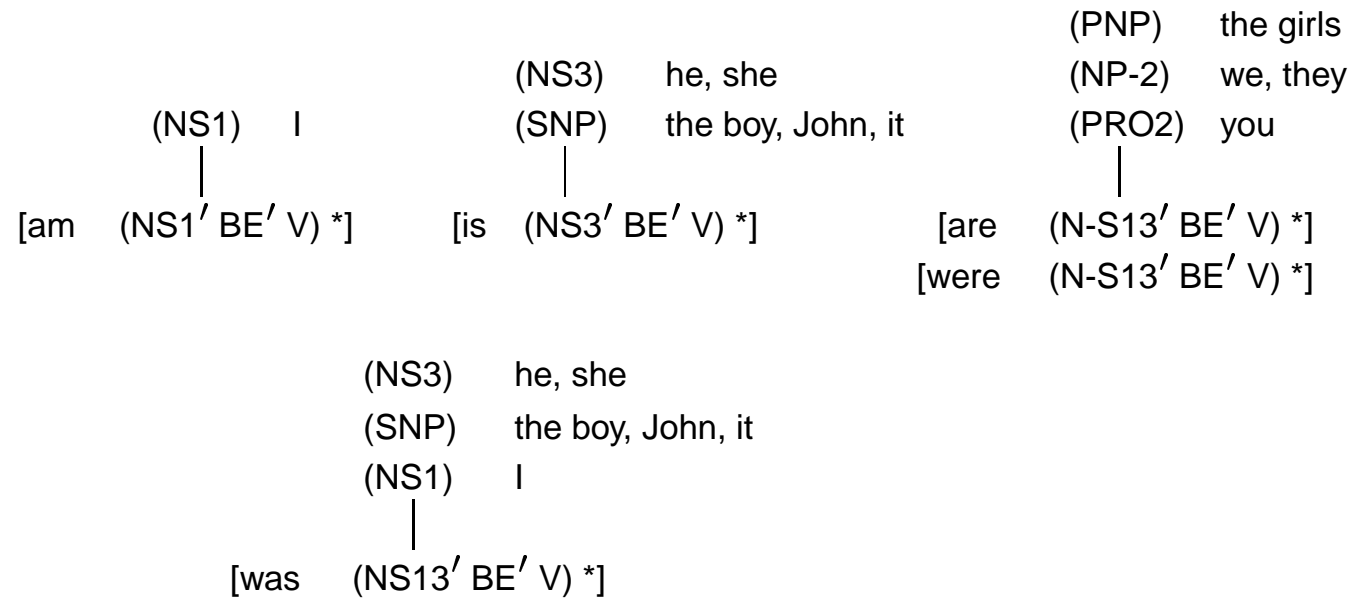
	<i>singular</i>			<i>plural</i>	
<i>nominative</i>	(SNP) the boy	(NS3) he she	(NS1) I	(NP-2) we    they	(PNP) the boys
<i>oblique</i>	John	him	(PRO2) you	us	them
	it	her	me	(OBQ)	

## 17.2.2 Agreement of fillers and valency in main verbs

	(NS1)	I	(SNP)	(SNP)	the boy, John, it	
	(NP-2)	we, they	(OBQ)	(OBQ)	me, him, her, us, them	
	(PNP)	the girls	(PNP)	(PNP)	the girls	
	(PRO2)	you	(PRO2)	(PRO2)	you	
[give	(N-S3')					V) *]
[gave	(N')		D'	A'		V) *]
[gives	(NS3')					V) *]
	(SNP)	the boy, John, it				
	(NS3)	he, she				

## 17.3 Complex verb forms

### 17.3.1 Nominative agreement of the auxiliary be





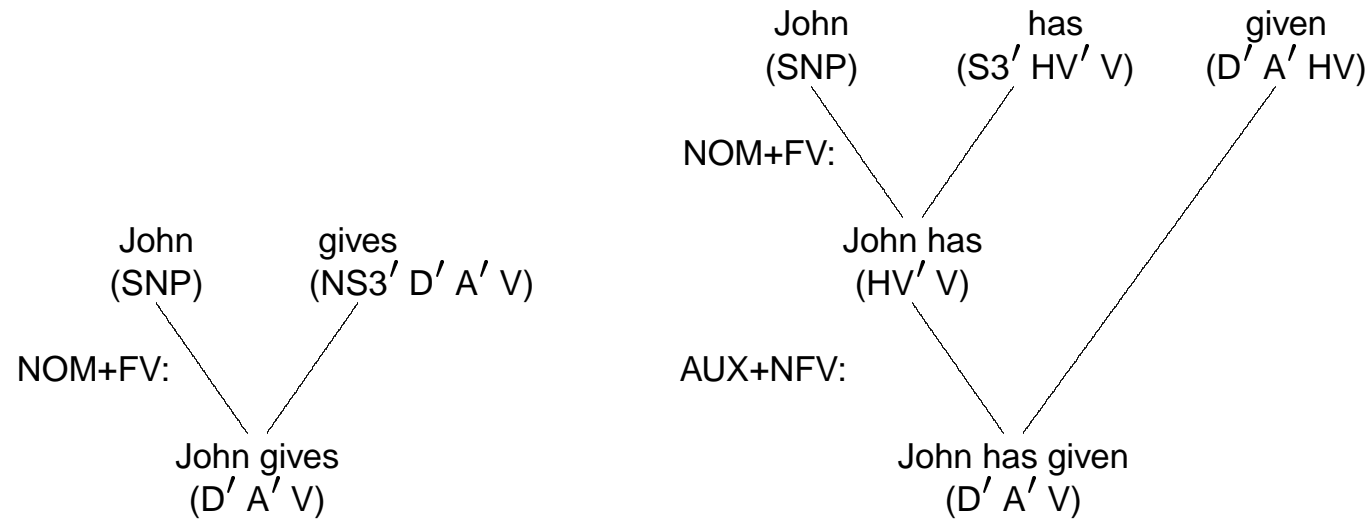
### 17.3.2 Complex verb forms of English

does	give		does give
(NS3' DO' V)	(D' A' DO)	⇒	(NS3' D' A' V)
	┌──────────────────┐ ↑		

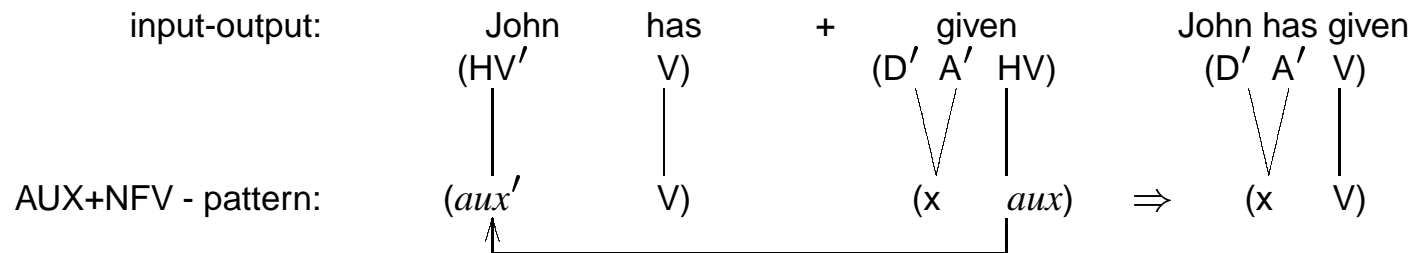
has	given		has given
(NS3' HV' V)	(D' A' HV)	⇒	(NS3' D' A' V)
	┌──────────────────┐ ↑		

is	giving		is giving
(NS3' BE' V)	(D' A' BE)	⇒	(NS3' D' A' V)
	┌──────────────────┐ ↑		

### 17.3.3 Comparing basic and complex verb forms of English



### 17.3.4 AUX+NFV adding a nonfinite verb



## 17.4 Finite state backbone of LA-syntax (*LA-E2*)

### 17.4.1 *LA-E2*: an English LA-syntax with complex NPs

$LX =_{def} \{$  [Julia (SNP) \*], [John (SNP) \*], [Suzy (SNP) \*], [it (SNP) \*],  
 [boy (SN) \*], [boys (PN) \*], [girl (SN) \*], [girls (PN) \*], [book (SN) \*],  
 [books (PN) \*], [a (SN' SNP) \*], [every (SN' SNP) \*], [the (SN' SNP) \*],  
 [all (PN' PNP) \*], [several (PN' PNP) \*], [the (PN' PNP) \*]  
 [I (NS1) \*], [you (PRO2)], [he (NS3) \*], [she (NS3) \*], [it (SNP) \*],  
 [we (NP-2) \*], [they (NP-2) \*], [me (OBQ) \*], [him (OBQ) \*],  
 [her (OBQ) \*], [us (OBQ) \*], [them (OBQ) \*]  
 [am (NS1' BE' V) \*], [is (NS3' BE' V) \*], [are (N-S13' BE' V) \*]  
 [was (NS13' BE' V) \*], [were (N-S13' BE' V) \*]  
 [have (N-S3' HV' V) \*], [has (NS3' HV' V) \*], [had (N' HV' V) \*]  
 [do (N-S3' DO' V) \*], [does (NS3' DO' V) \*], [did (N' DO' V) \*]  
 [give (N-S3' D' A' V) \*], [gives (NS3' D' A' V)], [gave (N' D' A' V) \*],  
 [give (D' A' DO) \*], [given (D' A' HV) \*], [giving (D A BE) \*]  
 [like (N-S3' A' V) \*], [likes (NS3' A' V)], [liked (N' A' V) \*]  
 [like (A' DO) \*], [liked (A' HV) \*], [liking (A' BE) \*]  
 [sleep (N-S3' V) \*], [sleeps (NS3' V) \*], [slept (N' V) \*]  
 [sleep (DO) \*], [slept (HV) \*], [sleeping (BE) \*]  $\}$

## Variable definition:

$np' \in \{N', N-S3', NS1', NS3', NS13', N-S13', D', A'\}$ , (valency positions)

$np \in \{PRO2, NS1, NS3, NP-2, SNP, PNP, PN, OBQ\}$  (valency fillers), and

if  $np = PRO2$ , then  $np' \in \{N', N-S3', N-S13', D', A'\}$ ,

if  $np = NS1$ , then  $np' \in \{N', N-S3', NS1', NS13'\}$ ,

if  $np = NS3$ , then  $np' \in \{NS3', NS13'\}$ ,

if  $np = NP-2$ , then  $np' \in \{N', N-S3'\}$ ,

if  $np = SNP$ , then  $np' \in \{N', NS3', NS13', D', A'\}$ ,

if  $np = PNP$ , then  $np' \in \{N', N-S3', N-S13', D', A'\}$ ,

if  $np = OBQ$ , then  $np' \in \{D', A'\}$ ,

$n \in \{SN, PN\}$  and  $n'$  correspondingly  $SN'$  or  $PN'$ ,

$aux \in \{DO, HV, BE\}$  and  $aux'$  correspondingly  $DO'$ ,  $HV'$  or  $BE'$

$x, y = .??.?.?$  (arbitrary sequence up to length 4)

$ST_S =_{def} \{ [(x) \{1 \text{ DET+ADJ}, 2 \text{ DET+N}, 3 \text{ NOM+FV}\}] \}$

DET+ADJ:  $(n' x) (\text{ADJ}) \Rightarrow (n x) \{4 \text{ DET+ADJ}, 5 \text{ DET+N}\}$

DET+N:  $(n' x) (n) \Rightarrow (x) \{6 \text{ NOM+FV}, 7 \text{ FV+MAIN}\}$

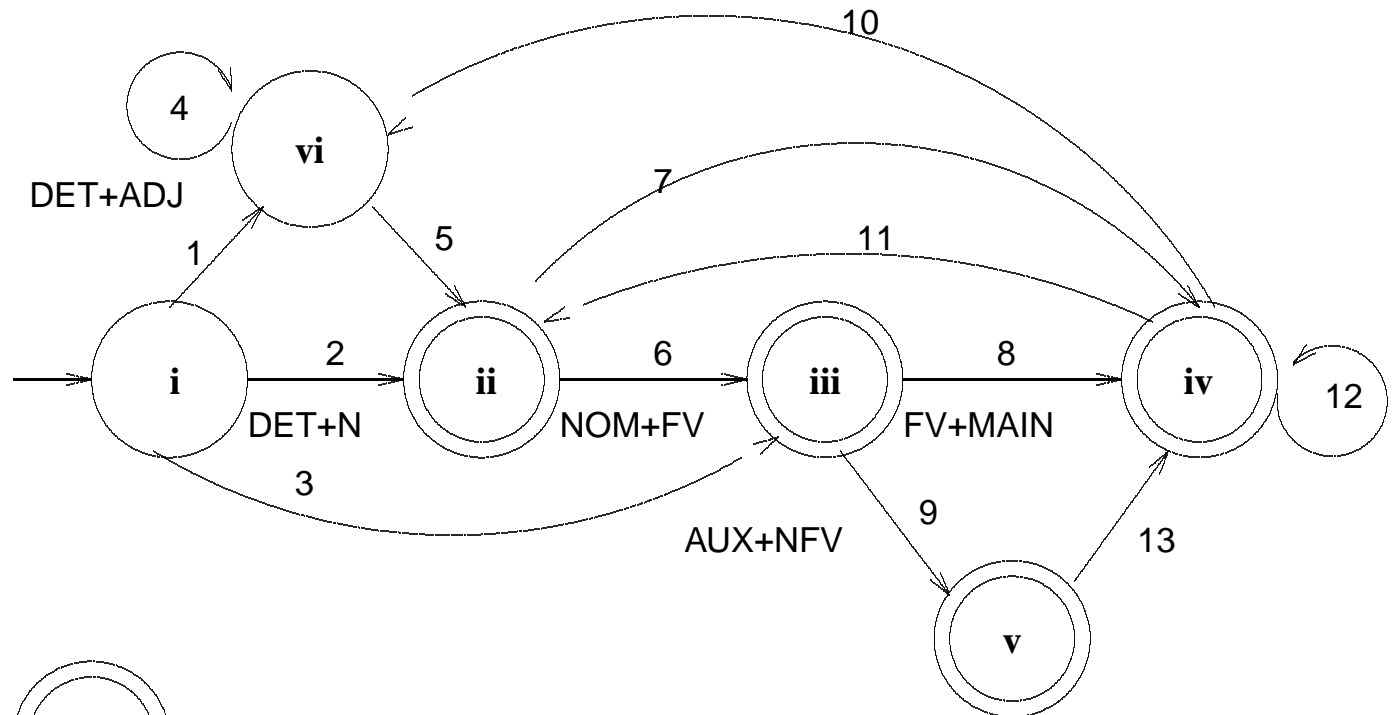
NOM+FV:  $(np) (np' x V) \Rightarrow (x V) \{8 \text{ FV+MAIN}, 9 \text{ AUX+NFV}\}$

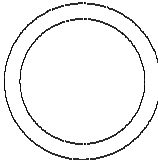
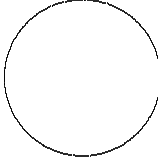
FV+MAIN:  $(np' x V) (y np) \Rightarrow (y x V) \{10 \text{ DET+ADJ}, 11 \text{ DET+N}, 12 \text{ FV+MAIN}\}$

AUX+NFV:  $(aux' V) (x aux) \Rightarrow (x V) \{13 \text{ FV+MAIN}\}$

$ST_F =_{def} \{ [(V) rp_{\text{nom+fv}}], [(V) rp_{\text{aux+nfv}}], [(V) rp_{\text{fv+main}}], [(V) rp_{\text{det+n}}] \}$

### 17.4.2 The finite state backbone of *LA-E2*



 = possible final state  
 = not a possible final state

(ii)	2, 5, 11	DET+N
(iii)	3, 6	NOM+FV
(iv)	7, 8, 12, 13	FV+MAIN
(v)	9	AUX+NFV
(vi)	1, 4, 10	DET+ADJ

### 17.4.3 Specifying the transition numbers in the input

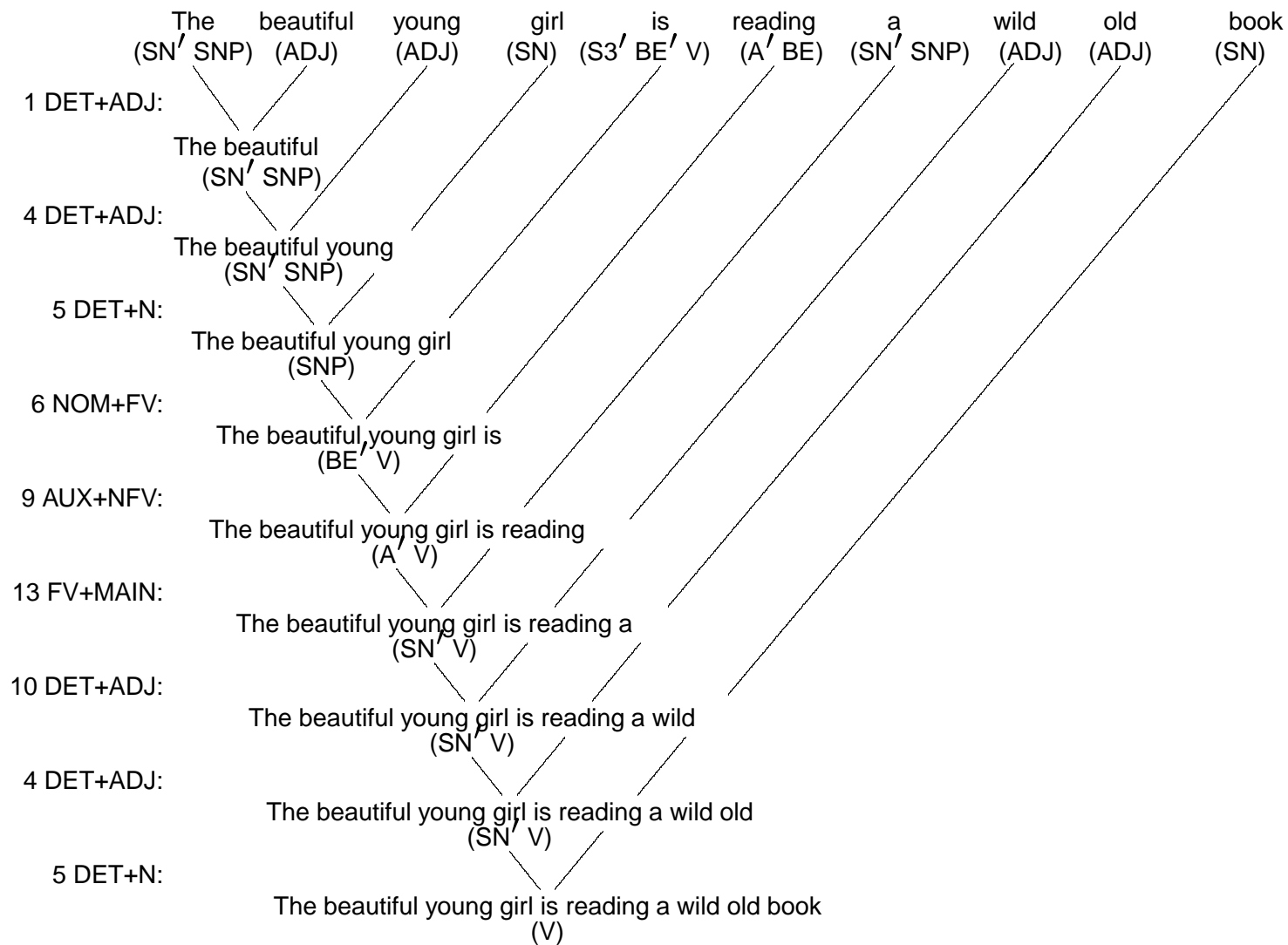
Peter 3 gave 8 Mary 12 a 11 book

the 1 beautiful 4 young 5 girl 6 is 9 reading 13 a 10 wild 4 old 5 book

the 2 boy 6 gave 8 the 11 girl 7 a 11 book

Peter 3 gave 8 Mary 12 Suzy

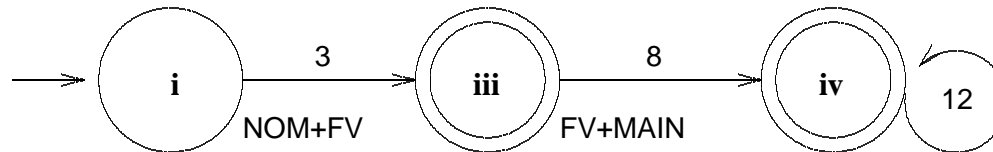
### 17.4.4 Syntactic analysis with transition numbers



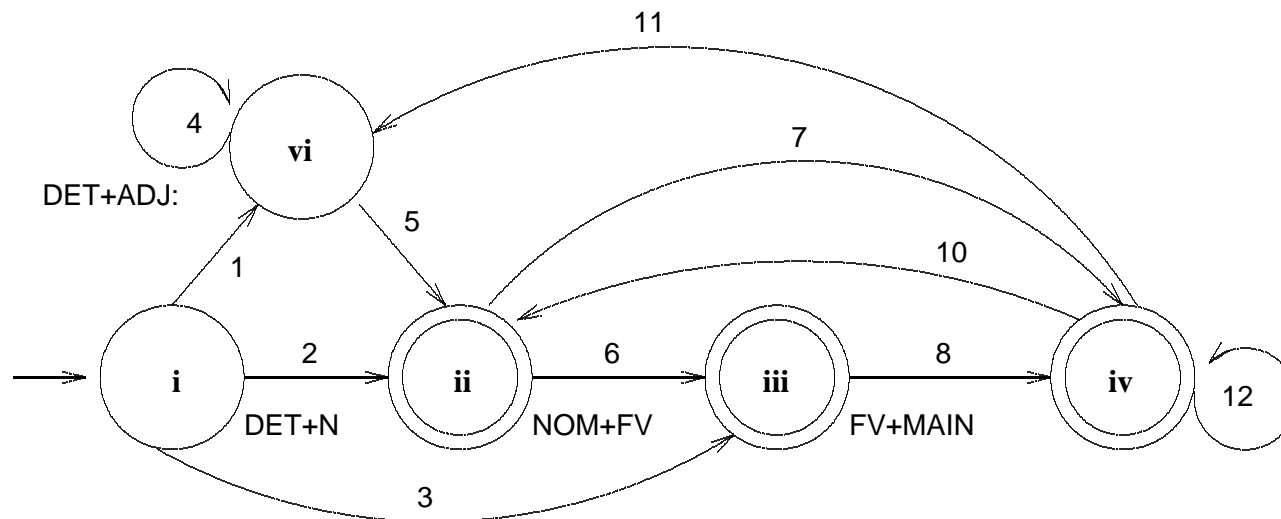
## 17.5 Yes/no-interrogatives (*LA-E3*) and grammatical perplexity

### 17.5.1 Expanding *LA-E1* to *LA-E1.5* handling complex NPs

LA-E1



LA-E1.5







### 17.5.5 LA-E3 for English yes/no-interrogatives

LX = LX of LA-E2 plus  $\{[. (V' \text{ decl}) *], [? (V' \text{ interrog}) *], [? (VI' \text{ interrog}) *]\}$

Variable definitions = that of LA-E2 plus  $vt \in \{V, VI\}$ ,

$ST_S =_{def} \{ [(x) \{1 \text{ DET+ADJ}, 2 \text{ DET+N}, 3 \text{ NOM+FV}, 4 \text{ AUX+MAIN}\}] \}$

DET+ADJ:  $(n' x) (\text{ADJ}) \Rightarrow (n' x) \{5 \text{ DET+ADJ}, 6 \text{ DET+N}\}$

DET+N:  $(n' x) (n) \Rightarrow (x) \{7 \text{ NOM+FV}, 8 \text{ FV+MAIN}, 9 \text{ AUX+NFV}, 10 \text{ IP}\}$

NOM+FV:  $(np) (np' x V) \Rightarrow (x V) \{11 \text{ FV+MAIN}, 12 \text{ AUX+NFV}, 13 \text{ IP}\}$

FV+MAIN:  $(np' x V) (y np) \Rightarrow (y x V) \{14 \text{ DET+ADJ}, 15 \text{ DET+N}, 16 \text{ FV+MAIN}, 17 \text{ IP}\}$

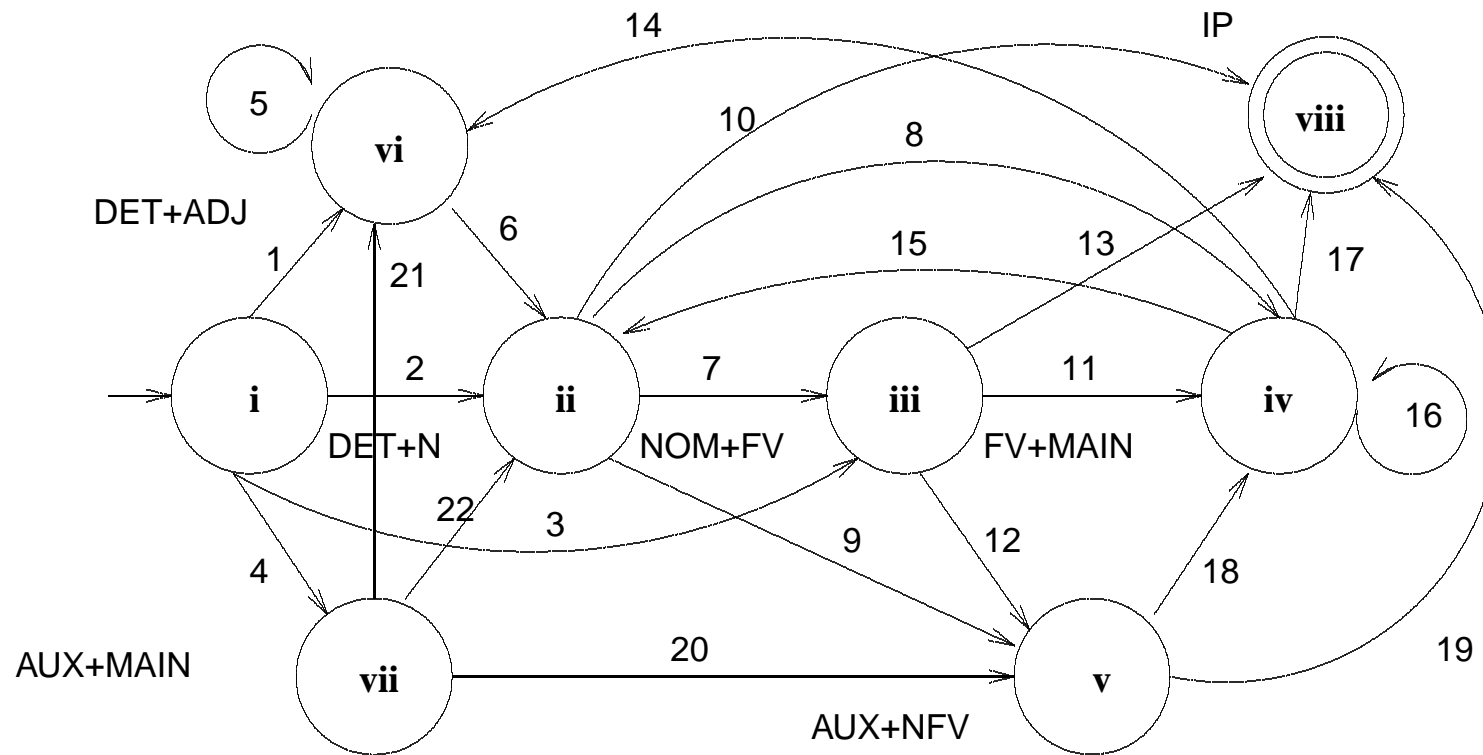
AUX+NFV:  $(aux' V) (x aux) \Rightarrow (x V) \{18 \text{ FV+MAIN}, 19 \text{ IP}\}$

AUX+MAIN:  $(np' aux' V) (x np) \Rightarrow (x aux' VI) \{20 \text{ AUX+NFV}, 21 \text{ DET+ADJ}, 22 \text{ DET+N}\}$

IP:  $(vt) (vt' x) \Rightarrow (x) \{\}$

$ST_F =_{def} \{ [(decl) rp_{ip}], [(interrog) rp_{ip}] \}$

### 17.5.6 The finite state backbone of *LA-E3*



ii	2, 6, 15, 22	DET+N	v	9, 12, 20	AUX+NFV
iii	3, 7	NOM+FV	vi	1, 5, 14, 21	DET+ADJ
iv	8, 11, 16, 18	FV+MAIN	vii	4	AUX+MAIN
			viii	10, 13, 17, 19	IP

## 17.5.7 Perplexity

Perplexity is, crudely speaking, a measure of the size of the set of words from which the next word is chosen given that we observe the history of the spoken words.

S. Roukos 1995

## 18. LA-syntax for German

### 18.1 Standard procedure of syntactic analysis

#### 18.1.1 Differences between natural languages

Natural languages are all based on the same time-linear derivation order. They differ only in their language specific handling of

- *agreement*
- *word order*
- *valency structure* (lexicalization)

#### 18.1.2 Phase I of standard procedure

1. Formal treatment of declarative main clauses with elementary finite verbs and elementary nominal fillers determines the basic typological properties of the natural language,
2. Extension to complex nominal fillers requires treatment of the internal and the external agreement restrictions of derived noun phrases, and the time-linear derivation of complex fillers in pre- and postverbal position.
3. The extension to complex verb phrases treats complex tenses and modalities.

### **18.1.3 What has to be done before Phase II**

A theoretically well-founded semantic and pragmatic interpretation for the syntactic analysis developed so far.

### **18.1.4 Phase II of the standard procedure**

The syntactic analyses of the second phase should be developed directly out of the semantic and pragmatic interpretation, and be provided for both, the speaker and the hearer mode.

Topics: (i) addition of basic and derived modifiers ranging from adverbs over prepositional phrases to subordinate clauses, (ii) treatment of sentential subjects and objects including infinitive constructions, (iii) handling of different syntactic moods like interrogative and different verbal moods like passive, and (iv) treatment of conjunctions including gapping constructions.

### 18.1.5 Distinctive categorization of determiners

#### *definite article*

[der	(E' MN' S3)	
	(EN' F' G&D)	
	(EN' P-D' G)	DEF-ART]
[des	(EN' -FG' G)	DEF-ART]
[dem	(EN' -FD' D)	DEF-ART]
[den	(EN' M-N' A)	
	(EN' PD' D)	DEF-ART]
[das	(E' N-G' S3&A)	DEF-ART]
[die	(E' F' S3&A)	
	(EN' P-D' P3&A)	DEF-ART]

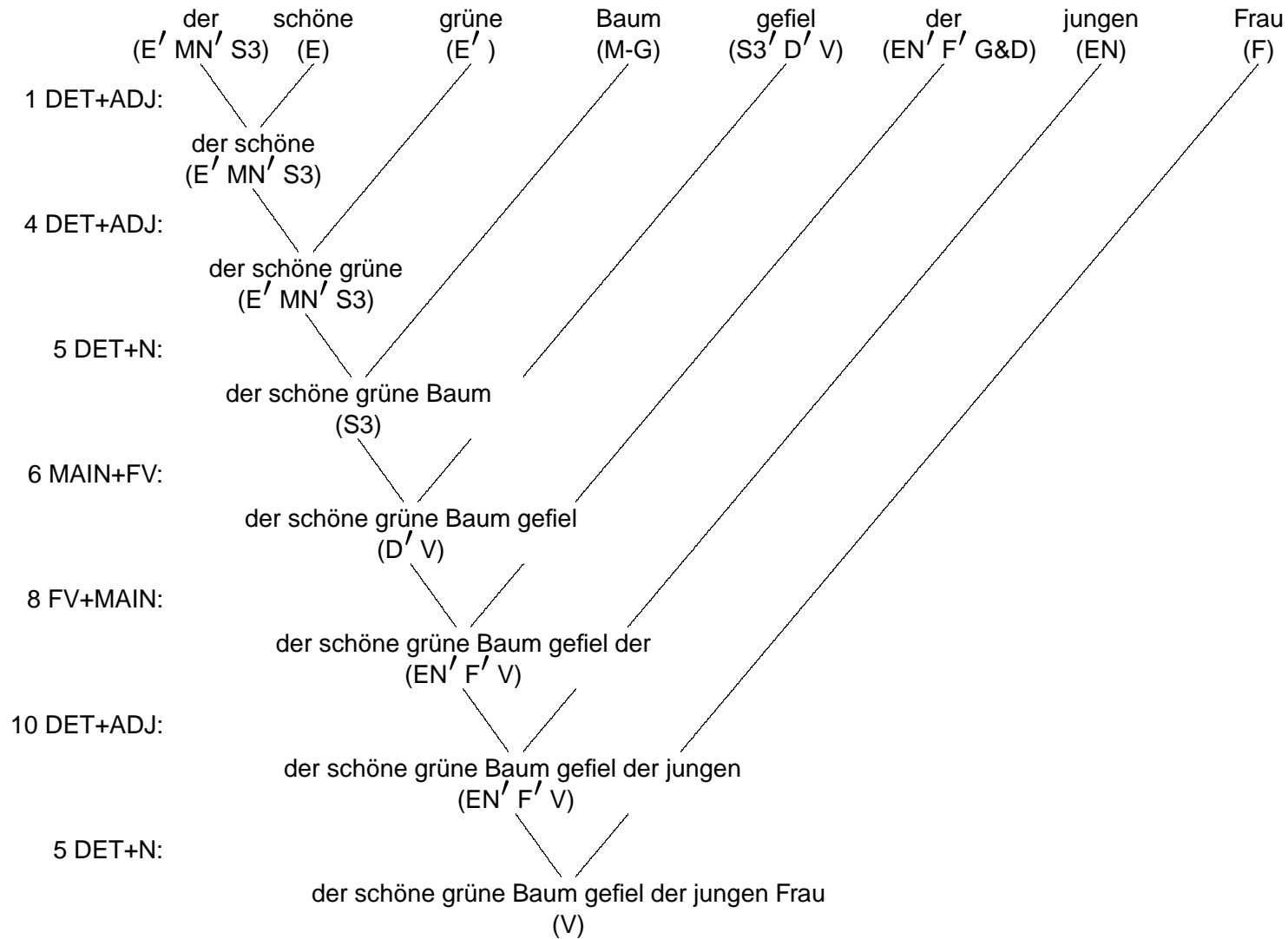
#### *indefinite article*

[ein	(ER' MN' S3)	
	(ES' N-G' S3&A)	INDEF-ART]
[eines	(EN' -FG' G)	INDEF-ART]
[einem	(EN' -FD' D)	INDEF-ART]
[einen	(EN' M-N' A)	INDEF-ART]
[eine	(E' F' S3&A)	INDEF-ART]
[einer	(EN' F' G&D)	INDEF-ART]





## 18.1.8 Pre- and postverbal derivation of noun phrases



## 18.2 German field of referents (*LA-D2*)

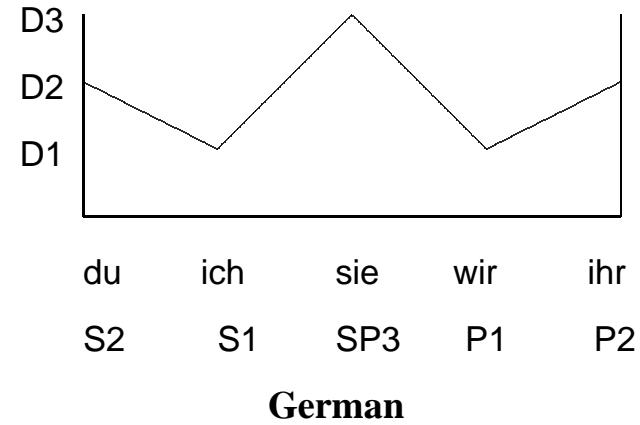
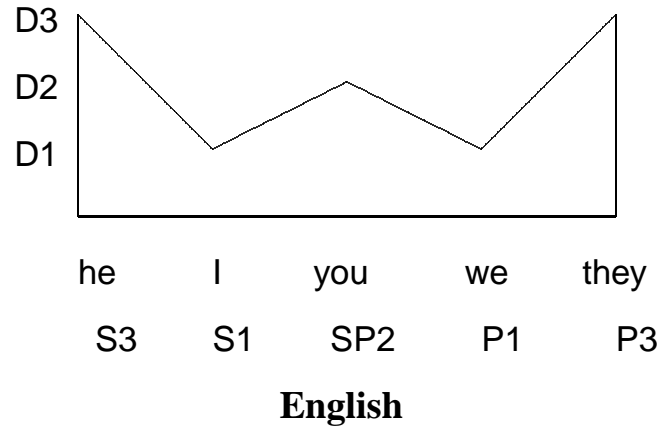
### 18.2.1 Traditional paradigms of German noun phrases

	<i>Masculinum</i>	<i>Femininum</i>	<i>Neutrum</i>	<i>Plural</i>
<i>Nominative</i>	der Mann	die Frau	das Kind	die Männer, etc.
<i>Genitive</i>	des Mannes	der Frau	des Kindes	der Männer, etc.
<i>Dative</i>	dem Mann	der Frau	dem Kind	den Männern, etc.
<i>Accusative</i>	den Mann	die Frau	das Kind	die Männer, etc.

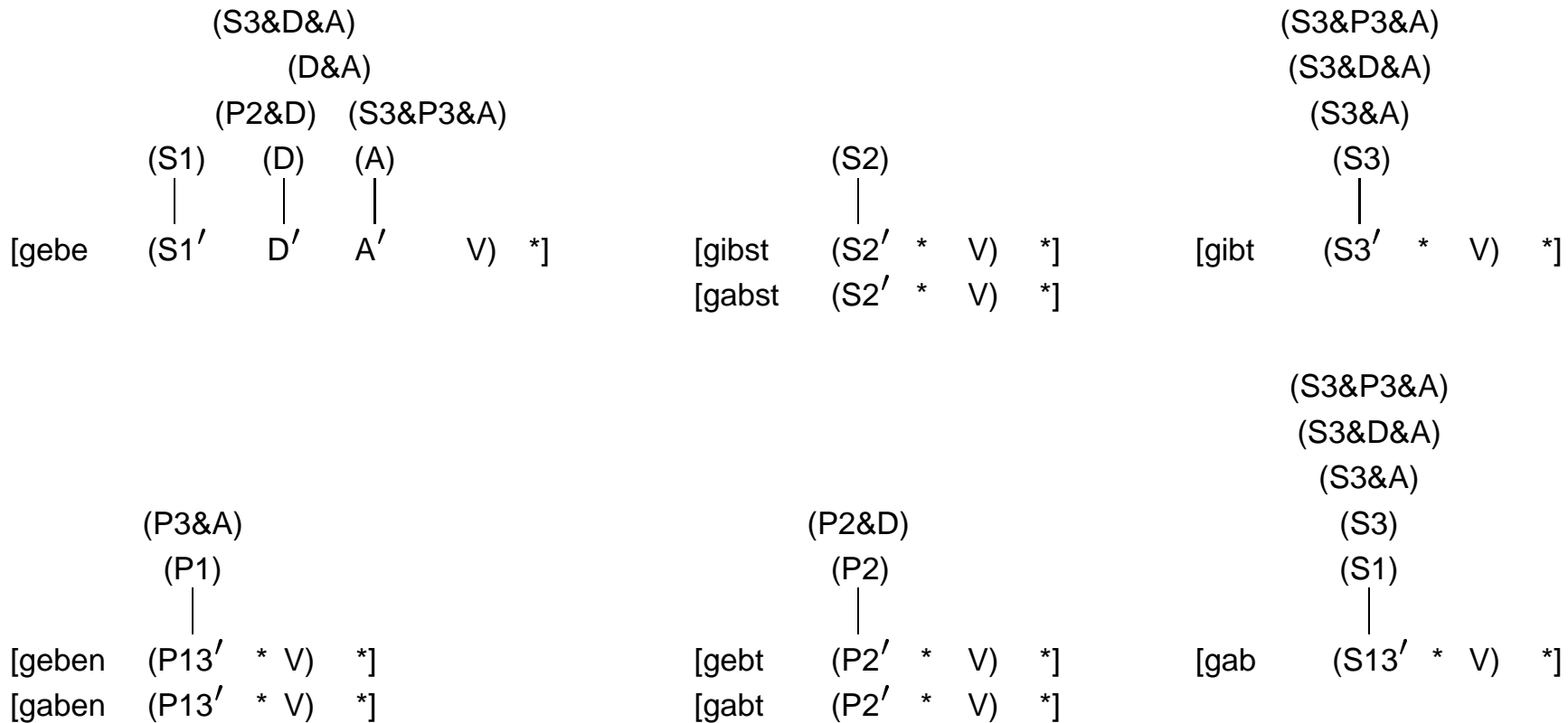
## 18.2.2 Distinctive categories of nominal fillers (German)

		Singular				Plural						
N		du (S2)	ich (S1)	er (S3)	Peter (S3&A&D)	das Kind es die Frau (S3&A)	sie (S3&P3 &A)	die Männer die Frauen die Kinder (P3&A)	wir (P1)	<i>ihr</i> (P2)		
	A	dich (A)	mich (A)	ihn (A)	den Mann (A)				uns (D&A)	euch (D&A)		
D		dir (D)	mir (D)	ihm (D)	dem Mann (D)	dem Kind (D)	<i>ihr</i> (D)	ihnen (D)	den Männern (D)	den Frauen (D)	den Kindern (D)	
	G	deiner (G)	meiner (G)	seiner (G)	des Kindes (G&D)	des Mannes (G&D)	ihrer (G)	(G)	der Männer (G)	der Frauen (G)	der Kinder (G)	unserer (G)

### 18.2.3 Centering and distance in fields of reference



## 18.2.4 Agreement of nominal fillers and verbal valencies



### 18.2.5 German LA-grammar handling complex fillers (LA-D2)

LX = LX of LA-D1 plus the determiners defined in 18.1.5, the nouns defined in 14.5.1, 14.5.2, and the following pronouns

[ich (S1) \*], [du (S2) \*], [er (S3) \*], [es (S3&A) \*], [wir (P1) \*],  
 [ihr (P2&D) \*], [sie (S3&P3&A) \*], [deiner (G) \*], [uns (D&A) \*],  
 [euch (D&A) \*], [mir (D) \*], [dir (D) \*], [ihm (D) \*], [mich (A) \*],  
 [dich (A) \*], [ihn (A) \*]

plus adjectives with comparison

[schöne (E) *]	[schönere (E) *]	[schönste (E) *]
[schönen (EN) *]	[schöneren (EN) *]	[schönsten (EN) *]
[schöner (ER) *]	[schönerer (ER) *]	[schönster (ER) *]
[schönes (ES) *]	[schöneres (ES) *]	[schönstes (ES) *]

plus finite main verb forms of differing valency structures

[gebe (S1' D' A' V) *]	[lese (S1' A' V) *]	[schlafe (S1' V) *]
[gibst (S2' D' A' V) *]	[liest (S23' A' V) *]	[schläfst (S2' V) *]
[gibt (S3' D' A' V) *]	[lesen (P13' A' V) *]	[schläft (S3' V) *]
[geben (P13' D' A' V) *]	[lest (P2' A' V) *]	[schlafen (P13' V) *]
[gebt (P2' D' A' V) *]	[las (S13' A' V) *]	[schläft (P2' V) *]
[gab (S13' D' A' V) *]	[last (S2P2' A' V) *]	[schlief (S13' V) *]
[gabst (S2' D' A' V) *]	[lasen (P13' A' V) *]	[schliefst (S2' V) *]

[gaben (P13' D' A' V) \*]

[gabt (P2' D' A' V) \*]

[schließen (P13' V) \*]

[schließt (P2' V) \*]

variable definition

$$np \in \{S1, S2, S3, P1, P2, P2\&D, G, G\&D, D, A, S3\&A, S3\&D\&A, D\&A, P3\&A, S3\&P3\&A\}$$

$$np' \in \{S1', S13', S2', S23', S2P2', S3', P13', P2', G', D', A'\}$$
and if  $np \in \{G, D, A\}$ , then  $np'$  is correspondingly  $G'$ ,  $D'$ , or  $A'$ if  $np = P1$ , then  $np' = P13'$ if  $np = S1$ , then  $np' \in \{S1', S13'\}$ if  $np = S2$ , then  $np' \in \{S2', S23'\}$ if  $np = S3$ , then  $np' \in \{S3', S23'\}$ if  $np = P3\&A$ , then  $np' \in \{P13', A'\}$ if  $np = P2\&D$ , then  $np' \in \{P2', D'\}$ if  $np = G\&D$ , then  $np' \in \{G', D'\}$ if  $np = D\&A$ , then  $np' \in \{D', A'\}$ if  $np = S3\&A$ , then  $np' \in \{S3', S23', A'\}$ if  $np = S3\&D\&A$ , then  $np' \in \{S3', S23', D', A'\}$ if  $np = S3\&P3\&A$ , then  $np' \in \{S3', S23', P13', A'\}$ 

$$n \in \{MN, M-G, M-NP, M-GP, MGP, M-GP-D, F, N-G, -FG, -FD, N-GP, N-GP-D,$$

NDP-D, P, P-D, PD},

$n' \in \{MN', M-N', F', N-G', -FG', -FD', P-D', PD'\}$ , and

if  $n \in \{MN, -FG, -FD, F, P-D, PD\}$ , then  $n'$  is corresponding

if  $n = M-G$ , then  $n' \in \{MN', M-N'\}$

if  $n = M-NP$ , then  $n' \in \{-FG', -FD', P-D', PD'\}$

if  $n = M-GP$ , then  $n' \in \{MN', -FD', M-N', P-D', PD'\}$

if  $n = MGP$ , then  $n' \in \{-FG', P-D', PD'\}$

if  $n = M-GP-D$ , then  $n' \in \{MN', -FD', M-N', P-D'\}$

if  $n = N-G$ , then  $n' \in \{N-G', -FG', -FD'\}$

if  $n = N-GP$ , then  $n' \in \{N-G', -FG', -FD', P-D', PD'\}$

if  $n = N-GP-D$ , then  $n' \in \{N-G', -FG', -FD', P-D'\}$

if  $n = NDP-D$ , then  $n' \in \{-FD', P-D'\}$

if  $n = P$ , then  $n' \in \{P-D', PD'\}$

$adj \in \{e, en, es, er\}$  and  $adj'$  is corresponding

$ST_S =_{def} \{ [(x) \{1 \text{ DET+ADJ}, 2 \text{ DET+N}, 3 \text{ MAIN+FV}\}] \}$

DET+ADJ:  $(adj' x) (adj) \Rightarrow (adj' x) \{4 \text{ DET+ADJ}, 5 \text{ DET+N}\}$

DET+N:  $(adj' n' x) (n) \Rightarrow (x) \{6 \text{ MAIN+FV}, 7 \text{ FV+MAIN}\}$

MAIN+FV:  $(np) (x np' y V) \Rightarrow (x y V) \{8 \text{ FV+MAIN}\}$

FV+MAIN:  $(x np' y V) (z np) \Rightarrow (z x y V) \{9 \text{ FV+MAIN}, 10 \text{ DET+ADJ}, 11 \text{ DET+N}\}$

$ST_F =_{def} \{ [(V) rp_{\text{MAIN+FV}}], [(V) rp_{\text{FV+MAIN}}], [(V) rp_{\text{DET+N}}] \}$



## 18.3 Verbal positions in English and German

### 18.3.1 Finite verb position in declarative main clauses

*English:* post-nominative

1. Julia *read* a book
2. \*a book *read* Julia
3. Yesterday Julia *read* a book
4. \*Yesterday *read* Julia a book
5. Julia yesterday *read* a book
6. \*While Mary slept, *read* Julia a book
7. While Mary slept, Julia *read* a book

*German:* verb-second

- Julia *las* ein Buch  
Ein Buch *las* Julia  
\*Gestern Julia *las* ein Buch  
Gestern *las* Julia ein Buch  
\*Julia gestern *las* ein Buch  
Als Maria schlief, *las* Julia ein Buch  
\*Als Maria schlief, Julia *las* ein Buch

### 18.3.2 Nonfinite main verb position in declarative main clauses

*English: contact position*

1. *Julia has slept*
2. *Julia has read a book*
3. \**Julia has a book read*
4. *Yesterday Julia has read a book*
5. \**Yesterday has Julia a book read*
6. *Julia has given M. a book yesterday*
7. \**Julia has M. yesterday a book given*

*German: distance position*

- Julia hat geschlafen*  
 \**Julia hat gelesen ein Buch*  
*Julia hat ein Buch gelesen*  
 \**Gestern Julia hat gelesen ein Buch*  
*Gestern hat Julia ein Buch gelesen*  
 \**Julia hat gegeben M. ein Buch gestern*  
*Julia hat M. gestern ein Buch gegeben*

### 18.3.3 Satzklammer in German

Julia has the offer of the opposing party yesterday afternoon

Julia hat das Angebot der Gegenseite gestern nachmittag	<i>abgelehnt.</i>	declined
	<i>verworfen.</i>	refused
	<i>kritisiert.</i>	criticized
	<i>zurückgewiesen.</i>	rejected

### 18.3.4 Verb position in subordinate clauses

*English*: post-nominative

1. before Julia *slept*
2. before Julia *had slept*
3. \*before Julia *slept had*
4. before Julia *bought* the book
5. \*before Julia the book *bought*
6. before Julia *had bought* the book
7. \*before the book a man *bought*

*German*: clause final

- bevor Julia *schlief*  
\*bevor Julia *hatte geschlafen*  
bevor Julia *geschlafen hatte*  
\*bevor Julia *kaufte* das Buch  
bevor Julia das Buch *kaufte*  
\*bevor Julia *hatte gekauft* das Buch  
bevor das Buch ein Mann *kaufte*

## 18.4 Complex verbs and elementary adverbs (*LA-D3*)

### 18.4.1 LA-paradigms of German auxiliaries and modals

[bin (S1' S' V) \*]

[bist (S2' S' V) \*]

[ist (S3' S' V) \*]

[sind (P13' S' V) \*]

[seid (P2' S' V) \*]

[war (S13' S' V) \*]

[warst (S2' S' V) \*]

[waren (P13' S' V) \*]

[wart (P2' S' V) \*]

[habe (S1' H' V) \*]

[hast (S2' H' V) \*]

[hat (S3' H' V) \*]

[haben (P13' H' V) \*]

[habt (P2' H' V) \*]

[hatte (S13' H' V) \*]

[hattest (S2' H' V) \*]

[hatten (P13' H' V) \*]

[hattet (P2' H' V) \*]

[kann (S13' M' V) \*]

[kannst (S2' M' V) \*]

[können (P13' M' V) \*]

[könnt (P2' M' V) \*]

[konnte (S13' M' V) \*]

[konntest (S2' M' V) \*]

[konnten (P13' M' V) \*]

[konntet (P2' M' V) \*]

## 18.4.2 Complex verb forms of German



## 18.4.3 Declarative main clauses with a finite main verb

Die Frau *gab* dem Kind den Apfel

Dem Kind *gab* die Frau den Apfel

## 18.4.4 Declarative main clauses with an auxiliary construction

Die Frau *hat* dem Kind den Apfel *gegeben*

Dem Kind *hat* die Frau den Apfel *gegeben*

### 18.4.5 +FV alternatives of adding the auxiliary

1. input-output: Die Frau (S3&A) + hat (S3' H' V) ⇒ Die Frau hat (H' V)
- rule pattern +FV: (nom) (nom' aux' V) ⇒ (aux' V)
2. input-output: Dem Kind (D) + hat (S3' H' V) ⇒ Dem Kind hat (D S3' H' V)
- rule pattern +FV: (obq) (x aux' V) ⇒ (obq' x aux' V)
3. input-output: Gegeben (D' A' H) + hat (S3' H' V) ⇒ Gegeben hat (S3' D' A' V)
- rule pattern +FV: (x aux) (nom' aux' V) ⇒ (nom' x V)
5. input-output: Gestern (ADV) + hat (S3' H' V) ⇒ Gestern hat (S3' H' V)
- rule pattern +FV: (y ADV) (x V) ⇒ (y x V)

### 18.4.6 Extending MAIN+FV into +FV using clauses

- +FV:**
1.  $(nom)(nom' aux' V) \Rightarrow (aux' V)$
  2.  $(obq)(x aux' V) \Rightarrow (obq x aux' V)$
  3.  $(x aux)(nom' aux' V) \Rightarrow (nom' x V)$
  4.  $(np)(x np' y V) \Rightarrow (x y V)$
  5.  $(y ADV)(x V) \Rightarrow (y x V) \{+MAIN, +NFV, +FV, +IP\}$

### 18.4.7 +MAIN Alternatives after the auxiliary

1. input-output: Dem Kind hat + die  
 (D S3' H' V) (E' F' S3&A) (E' F' D H' V)
- rule pattern +MAIN: (x np' y aux' V) (z np) ⇒ (z x y aux' V)
2. input-output: Die Frau hat + dem  
 (H' V) (EN' -FD' D) (EN' -FD' D H' V)
- rule pattern +MAIN: (aux' V) (y obq) ⇒ (y obq aux' V)
4. input-output: Dem Kind hat + gestern  
 (D S3' H' V) (ADV) (D S3' H' V)
- rule pattern +MAIN: (x V) (y ADV) ⇒ (y x V)





### 18.4.10 German grammar handling complex verb forms (*LA-D3*)

LX = LX of *LA-D2* plus auxiliaries defined in 18.4.1, plus

nonfinite main verb form of 18.4.2, plus adverbials

[gestern (ADV) \*], [hier (ADV) \*], [jetzt (ADV) \*], plus punctuation signs

[. (V' DECL) \*], [? (VI' INTERROG) \*], [? (V' INTERROG) \*]

variable definition = variable definition of *LA-D2* plus

$nom \in np \setminus \{D, A, D\&A\}$  nominative filler<sup>1</sup>

$nom' \in np \setminus \{D, A\}$  nominative valency positions

$obq \in \{D, A, D\&A\}$  oblique filler

$aux \in \{H, B, M\}$ , auxiliaries and modals

$vt \in \{V, VI\}$ , mood marker

$sm \in \{DECL, INTERROG\}$ , sentence mood

$ST_S =_{def} \{ [(x) \{1 +ADJ, 2 +N, 3 +FV, 4 +NFV\}] \}$

**+ADJ:**  $(adj' x) (adj) \Rightarrow (adj x) \{5 +ADJ, 6 +N\}$

**+N:**  $(adj' n' x) (n) \Rightarrow (x) \{7 +FV, 8 +MAIN, 9 +NFV, 10 +IP\}$

**+FV:**  $(nom)(nom' aux' V) \Rightarrow (aux' V)$

$(obq)(x aux' V) \Rightarrow (obq x aux' V)$

$(x aux)(nom' aux' V) \Rightarrow (nom' x V)$

$(np)(x np' y V) \Rightarrow (x y V)$

$(ADV)(x V) \Rightarrow (x V) \{11 +MAIN, 12 +NFV, 13 +IP\}$

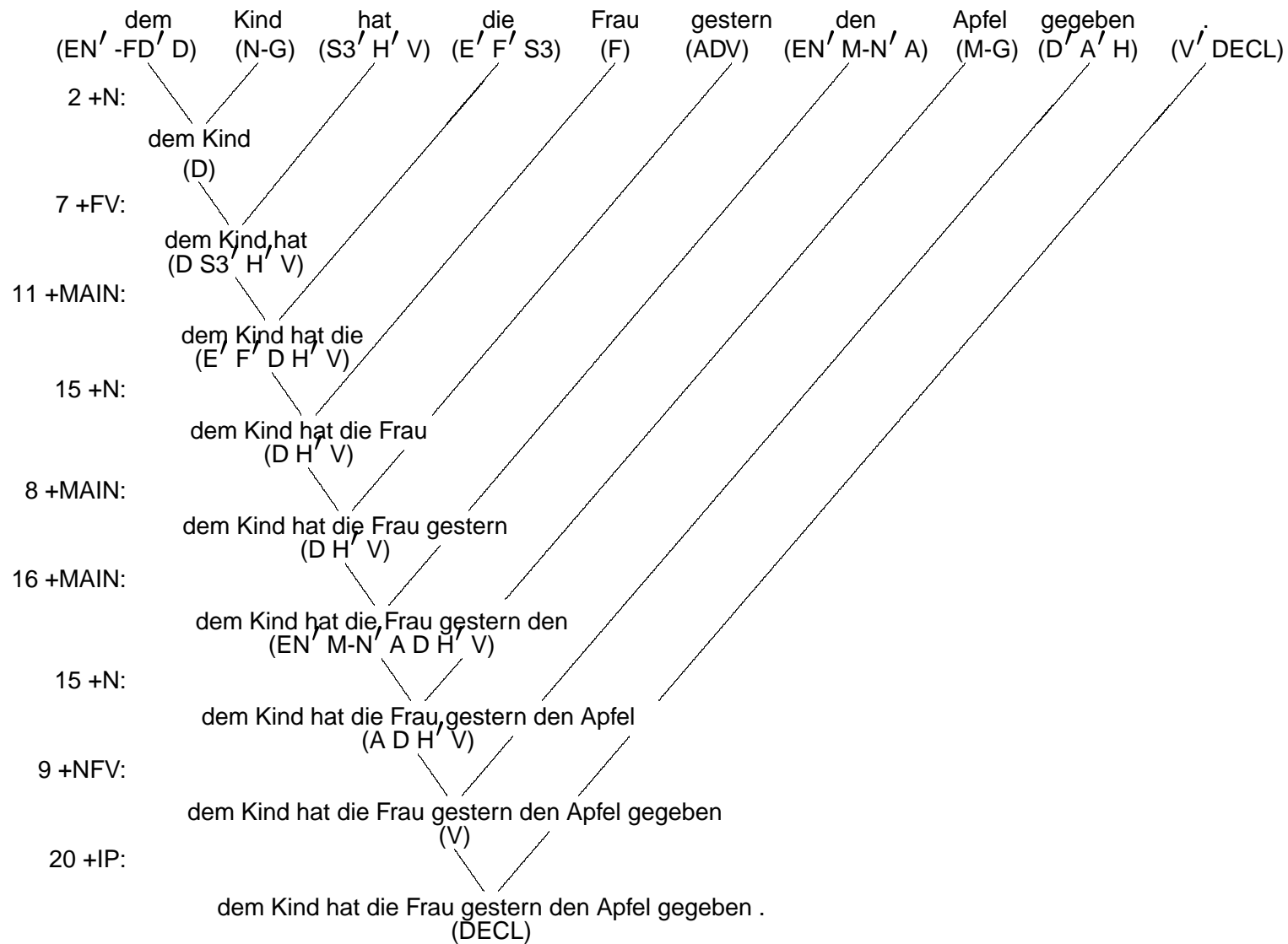
**+MAIN:**  $(x \text{ nom}' y \text{ aux}' V)(z \text{ nom}) \Rightarrow (z x y \text{ aux}' V)$   
 $(x \text{ aux}' V)(y \text{ obq}) \Rightarrow (y \text{ obq } x \text{ aux}' V)$   
 $(x \text{ np}' y V)(z \text{ np}) \Rightarrow (z x y V)$   
 $(x V)(y \text{ ADV}) \Rightarrow (y x V) \quad \{14 +\text{ADJ}, 15 +\text{N}, 16 +\text{MAIN}, 17 +\text{NFV},$   
 $18 +\text{FV}, 19 +\text{IP}\}$

**+NFV:**  $(x \text{ aux}' V)(x^{\sim} \text{ aux})$   
 $(x = x^{\sim}) \Rightarrow (V) \quad \{20 +\text{IP}\}$

**+IP:**  $(vt) (vt' sm) \Rightarrow (sm) \quad \{\}$

$\text{ST}_F =_{def} \{ [(sm) \text{ rp}_{+ipt}] \}$

### 18.4.11 Declarative with dative preceding auxiliary



## 18.5 Interrogatives and subordinate clauses (*LA-D4*)

### 18.5.1 Interrogative with and without auxiliary

1. *Hat* die Frau dem Kind gestern den Apfel *gegeben* ?  
(*Has the woman the child yesterday the apple given* ?)
2. *Hat* dem Kind gestern die Frau den Apfel *gegeben*?
3. *Hat* gestern die Frau dem Kind den Apfel *gegeben*?
4. *Gab* die Frau dem Kind gestern den Apfel ?  
(*Gave the woman the child yesterday the apple* ?)
5. *Gab* gestern die Frau dem Kind den Apfel?

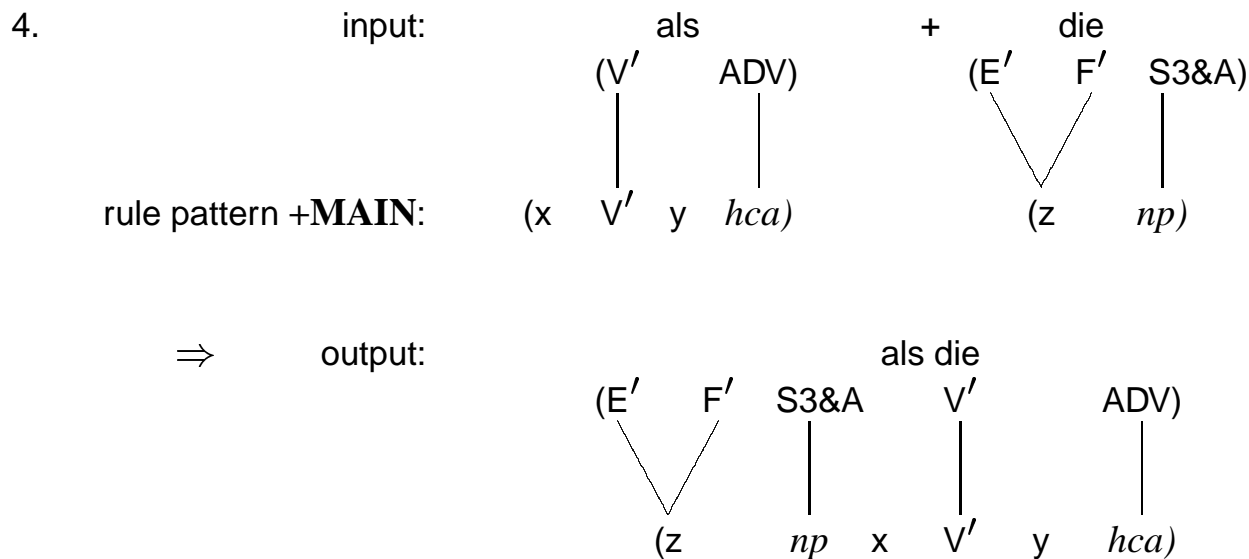
### 18.5.2 ?+MAIN starting an interrogative main clause

1. input-output:
- |       |      |    |   |     |      |    |       |     |     |         |    |    |     |
|-------|------|----|---|-----|------|----|-------|-----|-----|---------|----|----|-----|
| (S3'  | hat  | V) | + | (E' | die  | F' | S3&A) | ⇒   | (E' | hat die | F' | H' | VI) |
|       |      |    |   | \   | /    |    |       |     | \   | /       |    |    |     |
| (nom' | aux' | V) |   | (z  | nom) | (z | aux'  | VI) |     |         |    |    |     |
- rule pattern ?+MAIN:

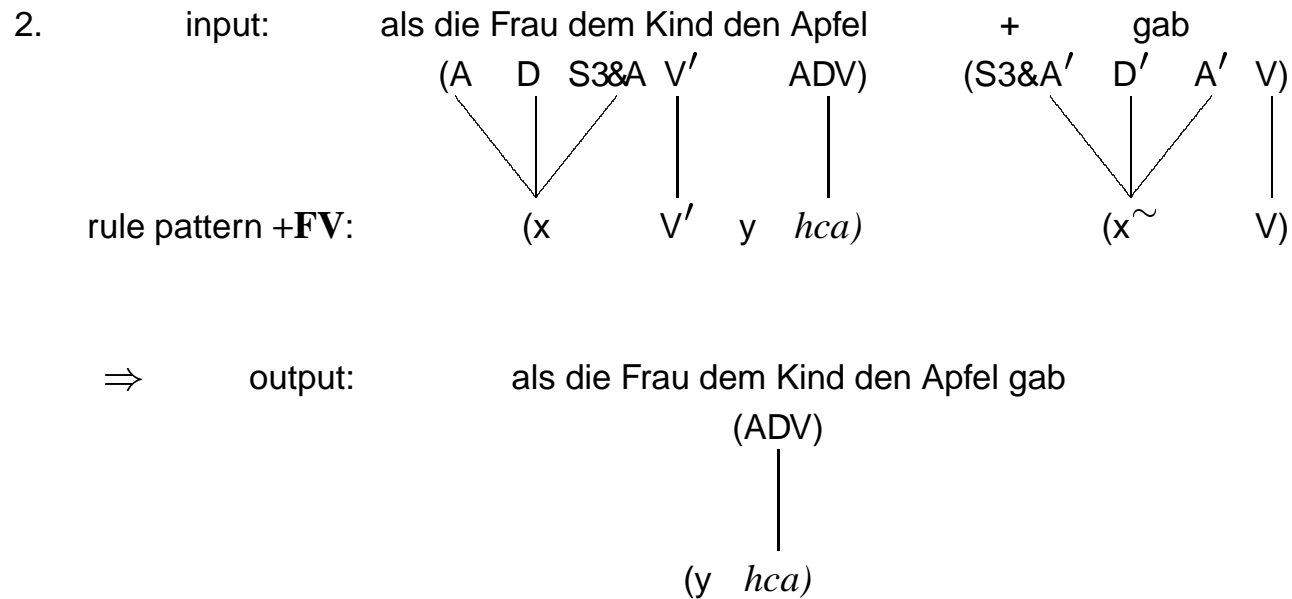
### 18.5.3 Subordinate clauses with and without auxiliary

1. *Als die Frau dem Kind gestern den Apfel gegeben hat*  
(When the woman the child yesterday the apple given has)
2. *Als dem Kind gestern die Frau den Apfel gegeben hat*
3. *Als gestern die Frau dem Kind den Apfel gegeben hat*
4. *Als die Frau dem Kind gestern den Apfel gab*  
(When the woman the child yesterday the apple gave)
5. *Als gestern die Frau dem Kind den Apfel gab*

### 18.5.4 +MAIN starting an adverbial subclause



### 18.5.5 +FV concluding subclause with finite main verb



### 18.5.6 Beginning of an adverbial subclause in postverbal position

Julia las, + als Julia las, als + Maria Julia las, als Maria  
 (A' V) (V' ADV)  $\Rightarrow$  (V' A' V) (S3&D&A)  $\Rightarrow$  (S3&D&A V' A' V)

### 18.5.7 Completion of an adverbial subclause in postverbal position

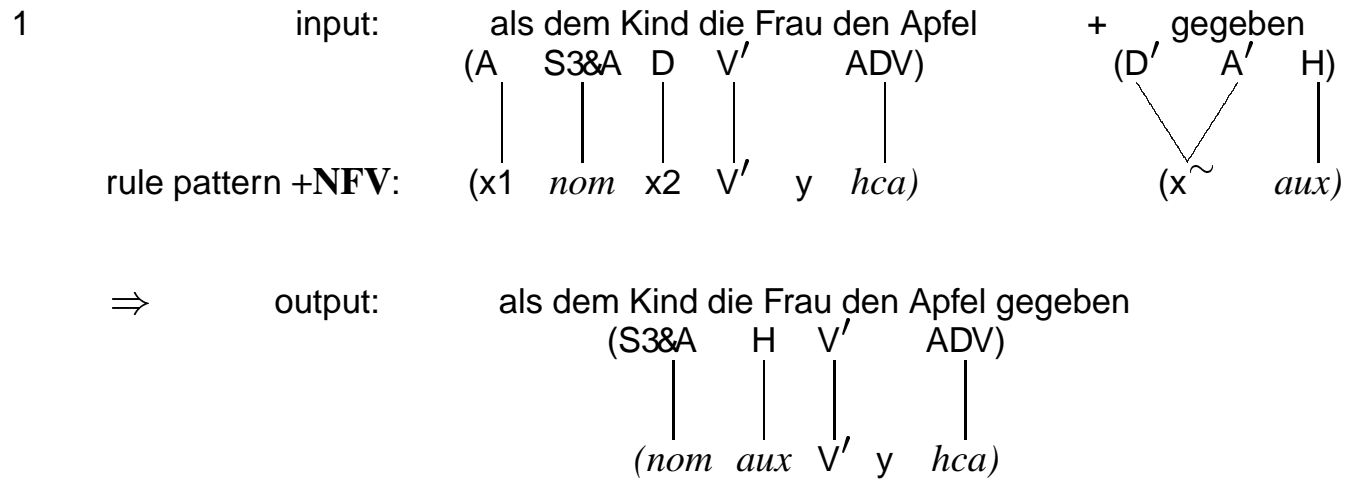
Julia las, als Maria + schief Julia las, als Maria schief,  
 (S3&D&A V' A' V) (S3' V)  $\Rightarrow$  (A' V)

### 18.5.8 Nesting of adverbial subclauses in preverbal position

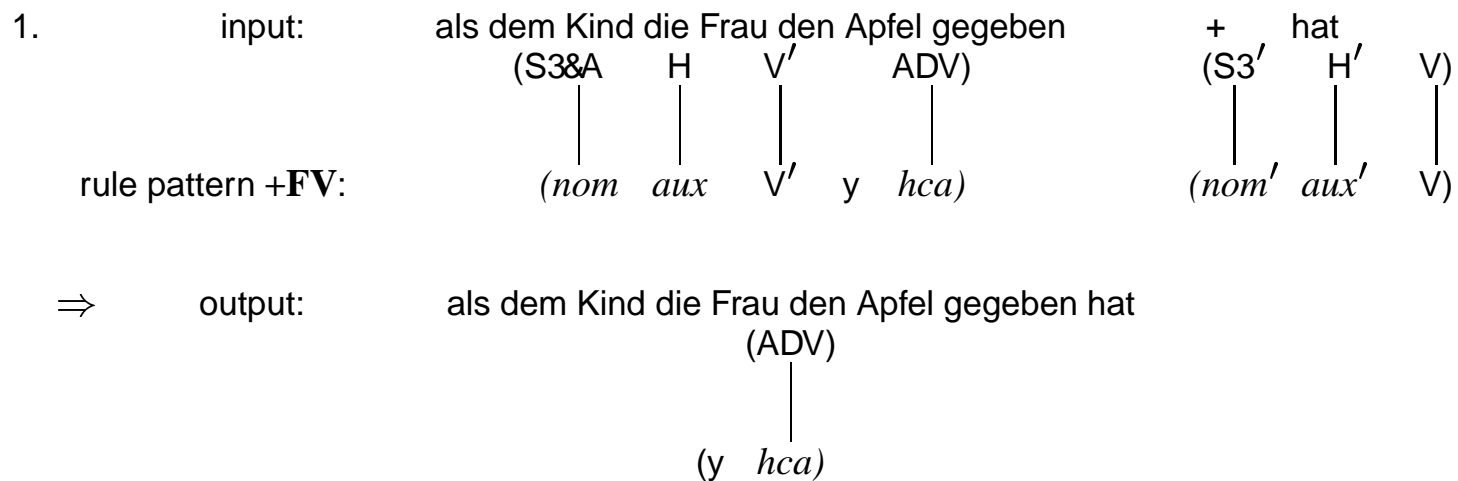
Als Maria, obwohl Julia die Zeitung + las Als Maria, obwohl Julia die Zeitung las,  
 (A S3&D&A V' S3&D&A V' ADV) (S3' A' V)  $\Rightarrow$  (S3&D&A V' ADV)



### 18.5.9 +NFV adds nonfinite main verb to subclause



### 18.5.10 +FV concludes subclause with finite auxiliary



### 18.5.11 LAG handling interrogative and adverbial clauses (*LA-D4*)

LX = LX of *LA-D3* plus subordinating conjunctions

[als (V' ADV) \*], [nachdem (V' ADV) \*], [obwohl (V' ADV) \*]

variable definition = variable definition of *LA-D3* plus  $hca \in \{V, VI, ADV\}$

$ST_S =_{def} \{ [(x) \{1 +ADJ, 2 +N, 3 +FV, 4 +MAIN, 5 ?+MAIN\}] \}$

**+N:**  $(adj' n' x) (n) \Rightarrow (x) \quad \{6 +FV, 7 +MAIN, 8 +NFV, 9 +IP\}$

**+ADJ:**  $(adj' x) (adj) \Rightarrow (adj' x) \quad \{10 +ADJ, 11 +N\}$

**?+MAIN:**  $(nom' aux' V)(z nom) \Rightarrow (z aux' VI)$

$(nom' aux' V)(y obq) \Rightarrow (y obq nom' aux' VI)$

$(x np' y V)(z np) \Rightarrow (z x y VI)$

$(x V)(y ADV) \Rightarrow (y x VI) \quad \{12 +ADJ, 13 +N, 14 +MAIN, 15 +NFV, 16 +IP\}$

**+FV:**  $(nom aux V' y hca) (nom' aux' V) \Rightarrow (y hca)$

$(x V' y hca)(x \sim V)$

$[x = x \sim] \Rightarrow (y hca)$

$(nom)(nom' aux' V) \Rightarrow (aux' V)$

$(obq)(x aux' V) \Rightarrow (obq x aux' V)$

$(x aux)(np' aux' V) \Rightarrow (x np' V)$

$(np)(x np' y V) \Rightarrow (x y V)$

$(ADV)(x V) \Rightarrow (x V) \quad \{17 +MAIN, 18 +NFV, 19 +FV, 20 +IP\}$

**+MAIN:**  $(x nom' y aux' V)(z nom) \Rightarrow (z x y aux' V)$

$$(x \text{ aux}' V)(y \text{ obq}) \Rightarrow (y \text{ obq } x \text{ aux}' V)$$

$$(x \text{ np}' y V)(z \text{ np}) \Rightarrow (z \text{ x } y V)$$

$$(x \text{ V}' y \text{ hca})(z \text{ np}) \Rightarrow (z \text{ np } x \text{ V}' y \text{ hca})$$

$$(x \text{ V})(y \text{ ADV}) \Rightarrow (y \text{ x } V) \quad \{21 +\text{ADJ}, 22 +\text{N}, 23 +\text{MAIN}, 24 +\text{NFV}, 25 +\text{FV}, 26 +\text{IP}\}$$
**+NFV:**

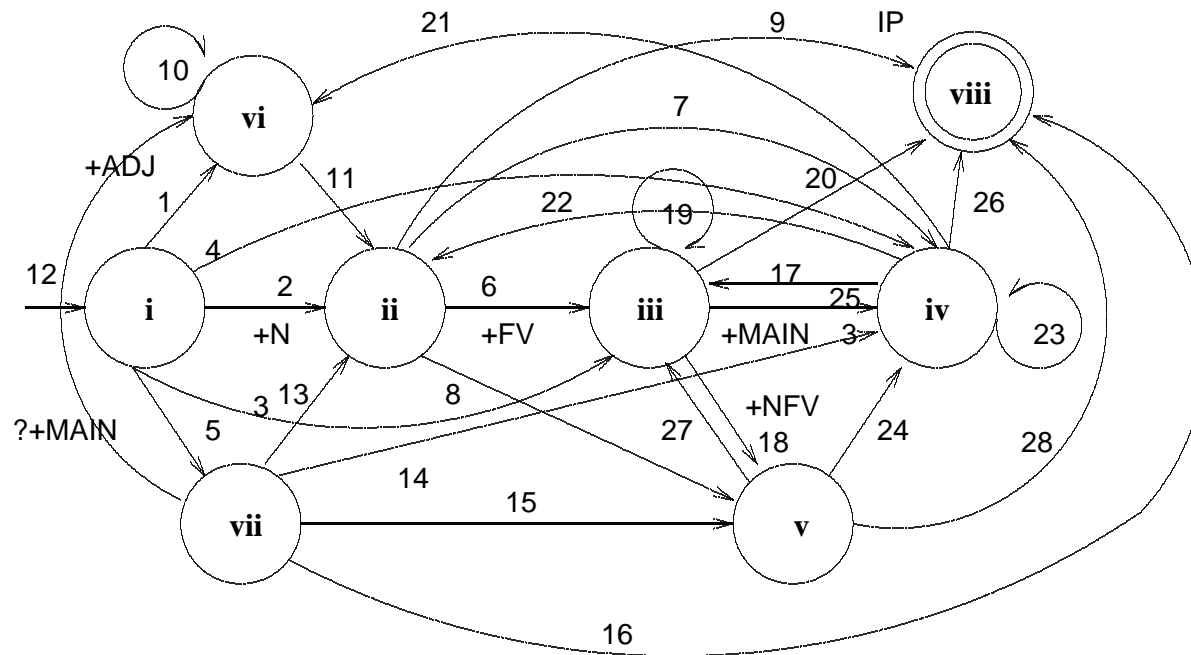
$$(x1 \text{ nom } x2 \text{ V}' y \text{ hca})(x \sim \text{ aux})$$

$$[(x1 \circ x2) = x \sim] \Rightarrow (\text{nom } \text{aux } V' y \text{ hca})$$

$$(x \text{ aux}' V)(x \sim \text{ aux}) \Rightarrow (V) \quad \{27 +\text{FV}, 28 +\text{IP}\}$$
**+IP:**

$$(vt) (vt' \text{ sm}) \Rightarrow (\text{sm}) \quad \{\}$$

$$\text{ST}_F =_{def} \{ [(V) \text{ rp}_{+\text{ipt}}], [(VI) \text{ rp}_{+\text{ipt}}] \}$$

18.5.12 The finite state backbone of *LA-D4*

ii	2, 11, 13, 19, 22,	+N	vi.	1, 10, 12, 21,	+ADJ
iii	3, 6, 8, 17, 19, 27	+FV	vii:	5,	+MAIN
iv	4, 14, 7, 23, 24, 25	+MAIN	viii:	9, 16, 20, 26,	+IP
v.	8, 15, 18,	+NFV			

### 18.5.13 Verification of grammars

#### 1. *Syntactic verification*

The formal grammars for English and German developed so far should be implemented as parsers and tested automatically on increasing sets of positive and negative test sentences.

#### 2. *Morphological and lexical verification*

The word form recognition of these grammars should be changed from the preliminary full form lexica LX to suitable applications of LA-Morph and be tested on corpus-based word lists in order to provide extensions with sufficient data coverage of the lexicon and the morphology.

#### 3. *Functional verification in communication*

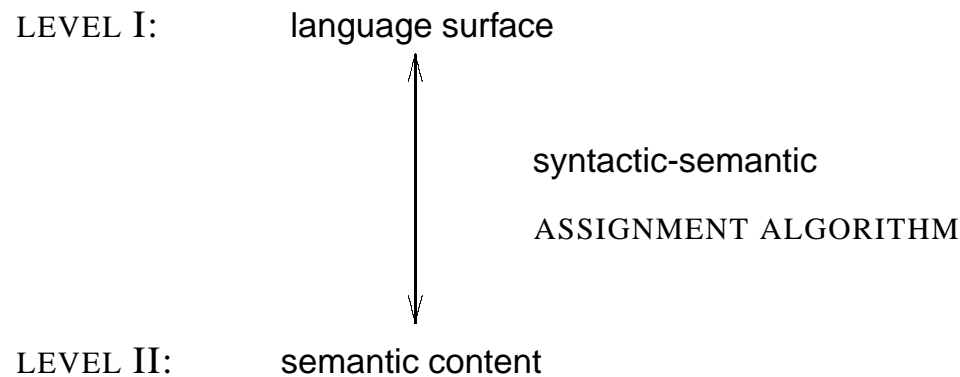
The formal grammars and parsers for natural languages should be supplemented with an automatic semantic and pragmatic interpretation that is (i) in line with the basis assumptions of the SLIM theory of language and (ii) demonstrated to be functional in automatic applications.

**Part IV.**  
**Semantics and Pragmatics**

## 19. Three system types of semantics

### 19.1 Basic structure of semantic interpretation

#### 19.1.1 The 2-level structure of semantic interpretation



#### 19.1.2 The function of semantic interpretation

For purposes of transmission and storage, semantic content is coded into surfaces of language (representation). When needed, the content may be decoded by analyzing the surface (reconstruction).

The expressive power of semantically interpreted languages resides in the fact that representing and reconstructing are realized *automatically*: a semantically interpreted language may be used correctly without the user having to be conscious of these procedures, or even having to know or understand their details.

## 19.2 Logical, programming, and natural languages

### 19.2.1 Three different types of semantic systems

1. *Logical languages*

Designed to determine the truth value of arbitrary propositions relative to arbitrary models. The correlation between the two levels is based on *metalanguage definitions*.

2. *Programming languages*

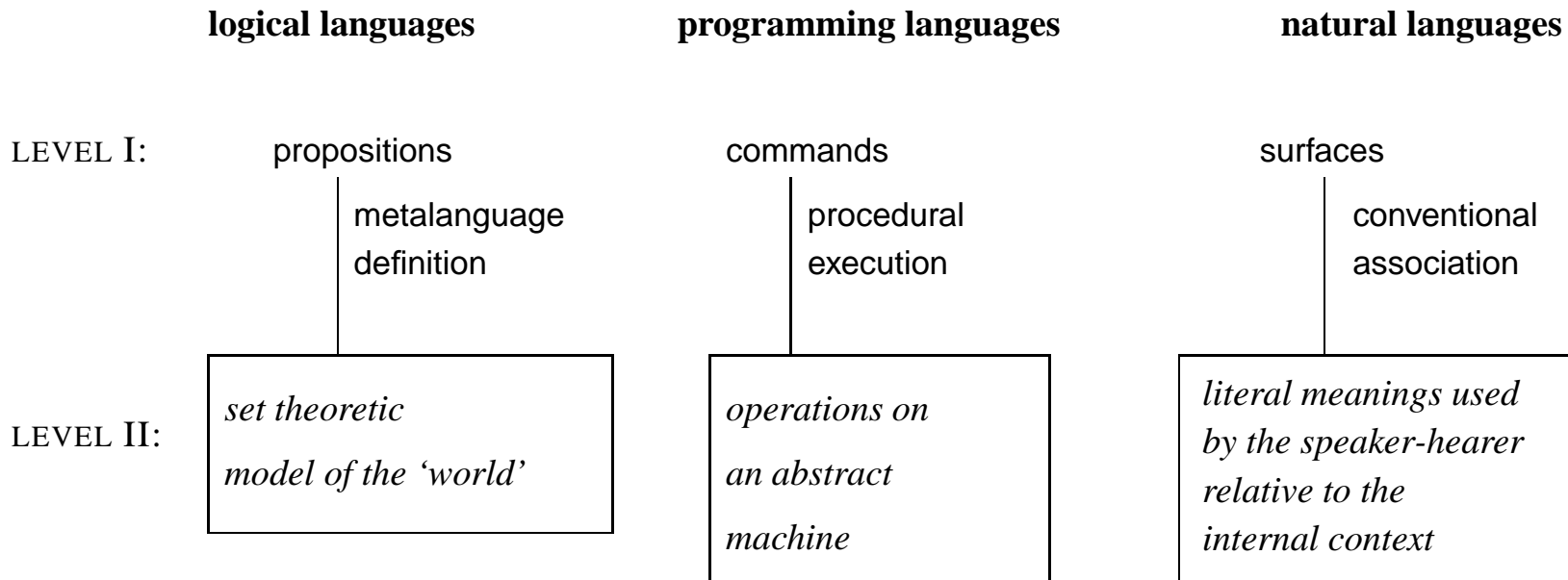
Designed to simplify the interaction with computers and the development of software. The correlation between the two levels is based on the *procedural execution* on an abstract machine, usually implemented electronically.

3. *Natural languages*

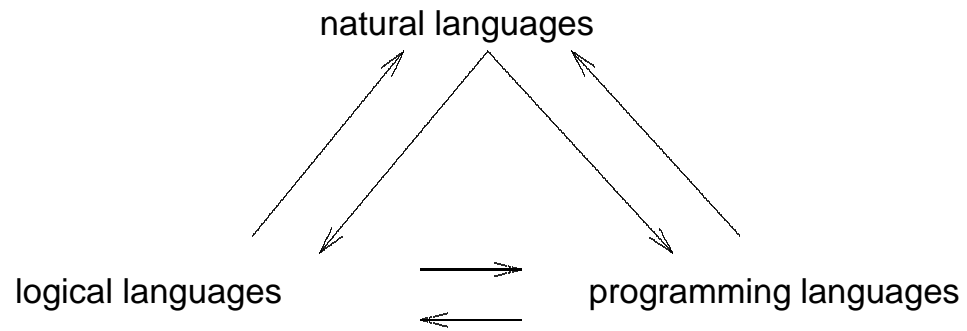
Preexisting in the language community, they are analyzed syntactically by reconstructing the combinatorics of their surfaces. The associated semantic representations have to be deduced via the general principles of natural communication. The correlation between the two levels is based on *conventional association*.



## 19.2.2 Three types of semantic interpretation



### 19.2.3 Mapping relations between the three types of semantics



## 19.2.4 Characterizing the mapping relations: Replication, Reconstruction, Transfer, and Combination

- *Replication*

Selected natural language phenomena are replicated in logical languages ( $N \rightarrow L$ ). Selected aspects of logical languages are replicated procedurally in programming languages like LISP and Prolog ( $L \rightarrow P$ ). The programming languages also replicate natural language concepts directly, e.g. ‘command’ ( $N \rightarrow P$ ).

- *Reconstruction*

Theoretical linguistics attempts to reconstruct fragments of natural language in terms of logic ( $L \rightarrow N$ ). Computational linguistics aims at reconstructing natural languages by means of programming languages ( $P \rightarrow N$ ). One may also imagine a reconstruction of programming concepts in a new logical language ( $P \rightarrow L$ ).

- *Transfer*

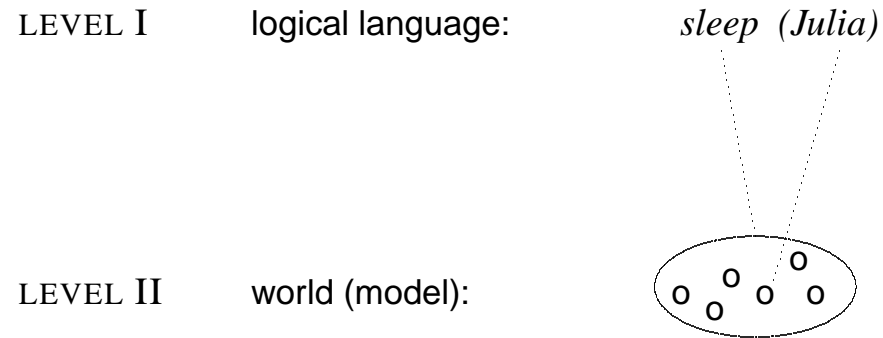
Computer science attempts to transfer methods and results of logical proof theory into the programming languages ( $L \rightarrow P$ ). Philosophy of language attempts to transfer the model-theoretic method to the semantic analysis of natural language ( $L \rightarrow N$ ).

- *Combination*

Computational linguistics aims at modeling natural communication with the help of programming languages ( $P \rightarrow N$ ). Thereby methods and results of the logical languages play a role in both, the construction of programming languages ( $L \rightarrow P$ ) and the analysis of natural language ( $L \rightarrow N$ ). This requires a functional overall framework for combining the three types of language in a way that utilizes their different properties while avoiding redundancy as well as conflict.

## 19.3 Functioning of logical semantics

### 19.3.1 Interpretation of a proposition



## 19.3.2 Definition of a minimal logic

### 1. Lexicon

Set of one-place predicates: {sleep, sing}

Set of names: {Julia, Susanne}

### 2. Model

A model  $\mathcal{M}$  is a two-tuple  $(A, F)$ , where  $A$  is a non-empty set of entities and  $F$  a denotation function (see 3).

### 3. Possible Denotations

- (a) If  $P_1$  is a one-place predicate, then a possible denotation of  $P_1$  relative to a model  $\mathcal{M}$  is a subset of  $A$ . Formally,  $F(P_1)\mathcal{M} \subseteq A$ .
- (b) If  $\alpha$  is a name, then the possible denotations of  $\alpha$  relative to a model  $\mathcal{M}$  are elements of  $A$ . Formally,  $F(\alpha)\mathcal{M} \in A$ .
- (c) If  $\phi$  is a sentence, then the possible denotations of  $\phi$  relative to a model  $\mathcal{M}$  are the numbers 0 and 1, interpreted as the truth values ‘true’ and ‘false.’ Formally,  $F(\phi)\mathcal{M} \in \{0,1\}$ .

Relative to a model  $\mathcal{M}$  a sentence  $\phi$  is a true sentence, if and only if the denotation  $\phi$  in  $\mathcal{M}$  is the value 1.

### 4. Syntax

- (a) If  $P_1$  is a one-place predicate and  $\alpha$  is a name, then  $P_1(\alpha)$  is a sentence.
- (b) If  $\phi$  is a sentence, then  $\neg\phi$  is a sentence.
- (c) If  $\phi$  is a sentence and  $\psi$  is a sentence, then  $\phi \ \& \ \psi$  is a sentence.

- (d) If  $\phi$  is a sentence and  $\psi$  is a sentence, then  $\phi \vee \psi$  is a sentence.
- (e) If  $\phi$  is a sentence and  $\psi$  is a sentence, the  $\phi \rightarrow \psi$  is a sentence.
- (f) If  $\phi$  is a sentence and  $\psi$  is a sentence, then  $\phi = \psi$  is a sentence.

## 5. Semantics

- (a) ' $P_1(\alpha)$ ' is a true sentence relative to a model  $\mathcal{M}$  if and only if the denotation of  $\alpha$  in  $\mathcal{M}$  is element of the denotation of  $P_1$  in  $\mathcal{M}$ .
- (b) ' $\neg \phi$ ' is a true sentence relative to a model  $\mathcal{M}$  if and only if the denotation of  $\phi$  is 0 relative to  $\mathcal{M}$ .
- (c) ' $\phi \& \psi$ ' is a true sentence relative to a model  $\mathcal{M}$  if and only if the denotations of  $\phi$  and of  $\psi$  are 1 relative to  $\mathcal{M}$ .
- (d) ' $\phi \vee \psi$ ' is a true sentence relative to a model  $\mathcal{M}$  if and only if the denotation of  $\phi$  or  $\psi$  is 1 relative to  $\mathcal{M}$ .
- (e) ' $\phi \rightarrow \psi$ ' is a true sentence relative to a model  $\mathcal{M}$  if and only if the denotation of  $\phi$  relative to  $\mathcal{M}$  is 0 or the denotation of  $\psi$  is 1 relative to  $\mathcal{M}$ .
- (f) ' $\phi = \psi$ ' is a true sentence relative to a model  $\mathcal{M}$  if- and only if the denotation of  $\phi$  relative to  $\mathcal{M}$  equals the denotation of  $\psi$  relative to  $\mathcal{M}$ .

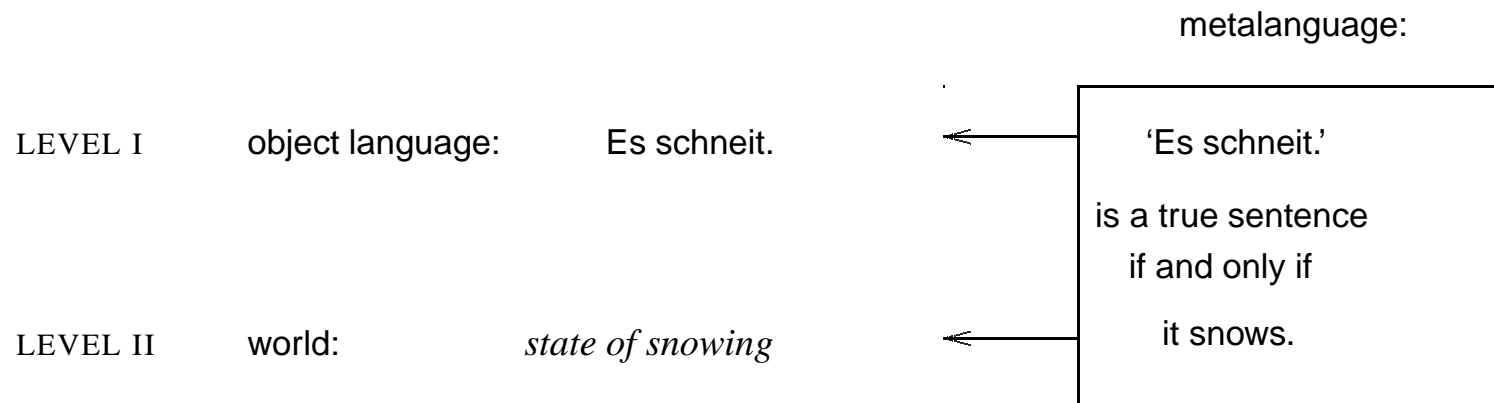
### 19.3.3 Schema of Tarski's T-condition

T: x is a true sentence if and only if p.

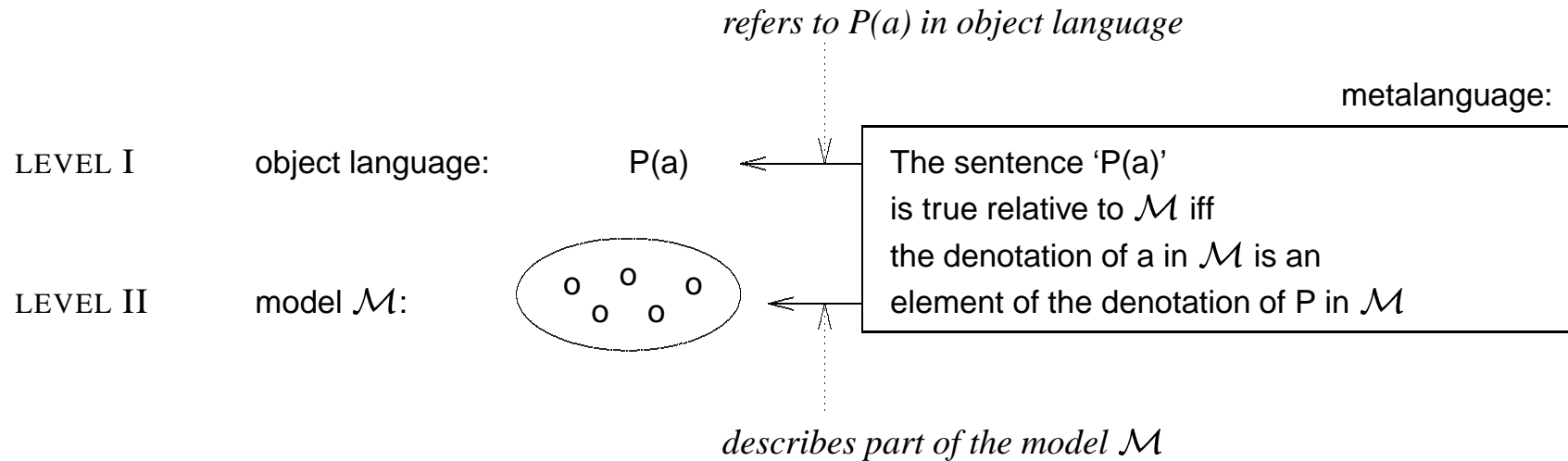
### 19.3.4 Instantiation of Tarski's T-condition

'Es schneit' is a true sentence if and only if it snows.

### 19.3.5 Relation between object and metalanguage



### 19.3.6 T-condition in a logical definition



### 19.3.7 The appeal to immediate obviousness in mathematics

En l'un les principes sont palpables mais éloignés de l'usage commun de sorte qu'on a peine à tourner late tête de ce côte-la, manque d'habitude : mais pour peu qu'on l'y tourne, on voit les principes à peine; et il faudrait avoir tout à fait l'esprit faux pour mal raisonner sur des principes si gros qu'il est presque impossible qu'ils échappent.

[In [the mathematical mind] the principles are obvious, but remote from ordinary use, such that one has difficulty to turn to them for lack of habit : but as soon as one turns to them, one can see the principles in full; and it would take a thoroughly unsound mind to reason falsely on the basis of principles which are so obvious that they can hardly be missed.]

B. PASCAL (1623 -1662), *Pensées*, 1951:340



## 19.4 Metalanguage-based versus procedural semantics

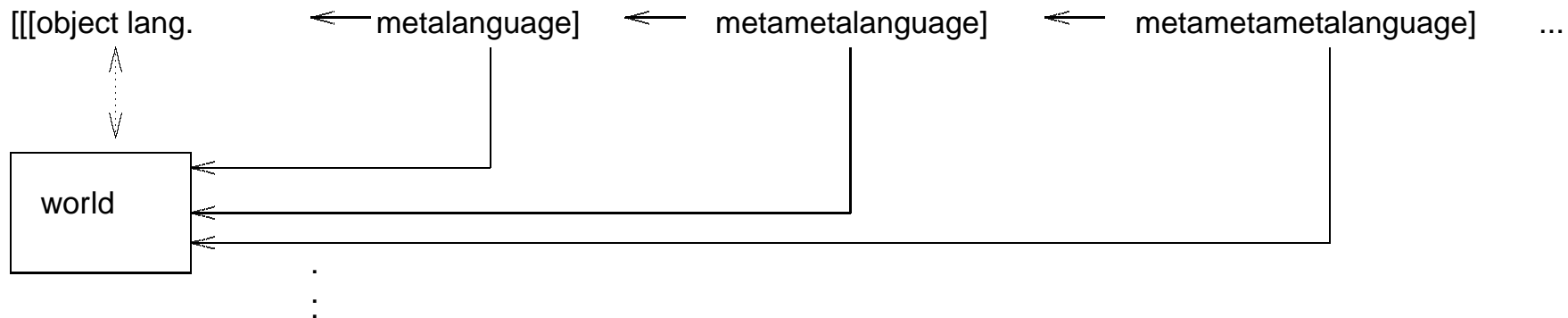
### 19.4.1 Example of a vacuous T-condition

‘A is red’ is a true sentence if and only if A is red.

### 19.4.2 Improved T-condition for red

‘A is red’ is a true sentence if and only if A refracts light in the electromagnetic frequency interval between  $\alpha$  and  $\beta$ .

### 19.4.3 Hierarchy of metalanguages



### **19.4.4 Autonomy from the metalanguage**

Autonomy from the metalanguage does not mean that computers would be limited to uninterpreted, purely syntactic deduction systems, but rather that Tarski's method of semantic interpretation is not the only one possible. Instead of assigning semantic representations to an object language by means of a metalanguage, computers use an operational method in which the notions of the programming language are realized automatically as machine operations.

### **19.4.5 Example of autonomy from metalanguage**

There is no problem to provide an adequate metalanguage definition for the rules of basic addition, multiplication, etc. However, the road from such a metalanguage definition to a working calculator is quite long and in the end the calculator will function mechanically – without any reference to these metalanguage definitions and without any need to understand the metalanguage.

### **19.4.6 Programming logical systems**

There exist many logical calculi which have not been and never will be realized as computer programs. The reason is that their metalanguage translations contain parts which may be considered immediately obvious by their designers (e.g. quantification over infinite sets of possible worlds in modal logic), but which are nevertheless unsuitable to be realized as empirically meaningful mechanical procedures.

## 19.5 Tarski's problem for natural language semantics

### 19.5.1 Logical semantics for natural language?

The attempt to set up a structural definition of the term 'true sentence' – applicable to colloquial language – is confronted with insuperable difficulties.

A. Tarski 1935, p. 164.

### 19.5.2 Tarski's proof

For the sake of greater perspicuity we shall use the symbol 'c' as a typological abbreviation of the expression 'the sentence printed on page 355, line 8 from the bottom.' Consider now the following sentence:

c is not a true sentence

Having regard to the meaning of the symbol 'c', we can establish empirically:

(a) 'c is not a true sentence' is identical with c.

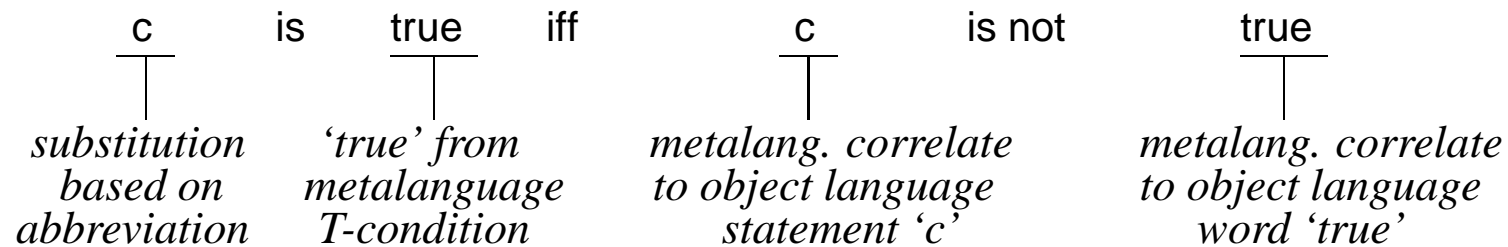
For the quotation-mark name of the sentence c we set up an explanation of type (2) [i.e. the T-condition 19.3.3]:

(b) 'c is not a true sentence' is a true sentence if and only if c is not a true sentence.

The premise (a) and (b) together at once give a contradiction:

c is a true sentence if and only if c is not a true sentence.

### 19.5.3 Inconsistent T-condition using Epimenides paradox



### 19.5.4 Three options for avoiding Tarski's contradiction in logical semantics

1. Forbidding the abbreviation and the substitution based on it. This possibility is rejected by Tarski because "no rational ground can be given why substitution should be forbidden in general."
2. Distinguishing between the truth predicate  $true^m$  of the metalanguage and  $true^o$  of the object language. On this approach
 

$c$  is  $true^m$  if and only if  $c$  is not  $true^o$

is not contradictory because  $true^m \neq true^o$ .
3. This option, chosen by Tarski, consists in forbidding the use of truth predicates in the object language.

### 19.5.5 Reasons for the third option

If the goal is to characterize scientific theories like physics as true relations between logical propositions and states of affairs, then the vagueness and contradictions of the natural languages must be avoided – as formulated by G. Frege 1896:

Der Grund, weshalb die Wortsprachen zu diesem Zweck [d.h. Schlüsse nur nach rein logischen Gesetzen zu ziehen] wenig geeignet sind, liegt nicht nur an der vorkommenden Vieldeutigkeit der Ausdrücke, sondern vor allem in dem Mangel fester Formen für das Schließen. Wörter wie >also<, >folglich<, >weil< deuten zwar darauf hin, daß geschlossen wird, sagen aber nichts über das Gesetz, nach dem geschlossen wird, und können ohne Sprachfehler auch gebraucht werden, wo gar kein logisch gerechtfertigter Schluß vorliegt.

[The reason why the word languages are suited little for this purpose [i.e., draw inferences based on purely logical laws] is not only the existing ambiguity of the expressions, but mainly the lack of clear forms of inference. Even though words like ‘therefore,’ ‘consequently,’ ‘because’ indicate inferencing, they do not specify the rule on which the inference is based and they may be used without violating the well-formedness of the language even if there is no logically justified inference.]

The goal of characterizing scientific truth precludes reconstructing the object language as a natural language. Therefore there is no need for a truth predicate in the object language – which is in line with Tarski’s option.

### 19.5.6 Reasons against the third option

If the goal is to apply logical semantics to natural language, then the third option poses a serious problem. This is because natural language as the pretheoretical metalanguage *must* contain the words true and false. Therefore a logical semantic interpretation of a natural (object-)language in its entirety will unavoidably result in a contradiction.

### 19.5.7 Montague's choice: Ignoring the problem

I reject the contention that an important theoretical difference exists between formal and natural languages. ... Like Donald Davidson I regard the construction of a theory of truth – or rather the more general notion of truth under an arbitrary interpretation – as the basic goal of serious syntax and semantics.

R. Montague 1970

### 19.5.8 Davidson's choice: Suspending the problem

Tarski's ... point is that we should have to reform natural language out of all recognition before we could apply formal semantic methods. If this is true, it is fatal to my project.

D. Davidson 1967

## 20. Truth, meaning, and ontology

### 20.1 Analysis of meaning in logical semantics

#### 20.1.1 The meaning principle of logical semantics

If a speaker-hearer knows the meaning of a sentence, (s)he can say for any state of affairs whether the sentence is true or false with respect to it.

#### 20.1.2 Existential generalization

The truth of a proposition  $F(a,b)$  implies that  $a$  exists and that  $b$  exists. For example, the sentence Julia kissed Richard is analyzed semantically as a *kiss*-relation between the entities Julia and Richard. If Julia kissed Richard is true, then it must be true that Julia exists and Richard exists.

#### 20.1.3 Substitutivity of identicals

The premises  $F(b)$  and  $b = c$ ,  $F(b)$  implies  $F(c)$ . For example, if Richard = Prince of Burgundy, then the truth of the sentence Julia kissed Richard implies the truth of the sentence Julia kissed the Prince of Burgundy. This substitutivity of Richard and Prince of Burgundy *salva veritate*, i.e. preserving the truth-value, is based on the fact that these two different expressions denote the same object.

### 20.1.4 Valid and invalid instances of existential generalization

- 1) Julia finds a unicorn.  $\supset$  A unicorn exists.
- 2) Julia seeks a unicorn.  $\not\supset$  A unicorn exists.

The premises in these two examples have exactly the same syntactic structure, namely  $F(a,b)$ . The only difference consists in the choice of the verb. Yet in (1) the truth of the premise implies the truth of the consequent, in accordance with the rule of existential generalization, while in (2) this implication does not hold.

### 20.1.5 First problem for extensional ontology

How a relation can be established between a subject and an object if the object does not exist. How can Julia seeks a unicorn be grammatically well-formed, meaningful, and even true under realistic circumstances?

### 20.1.6 Isolating the first problem

Part of the solution consisted in specifying certain environments in natural sentences in which the rule of existential generalization does not apply, e.g., in the scope of a verb like *seek*. These environments are called the *uneven* (Frege 1892), *opaque* (Quine 1960), or *intensional* (Montague 1974) *contexts*.



### 20.1.7 Second problem for extensional ontology

How should the difference in the meaning of different expressions for non-existing objects, such as square circle, unicorn, and Pegasus, be explained? This is necessary because of the second inference rule, the substitutivity of identicals.

For example, if we were to use the empty set as the referent of square circle, unicorn, and Pegasus in order to express that no real objects correspond to these terms, then the truth of Julia seeks a unicorn would imply the truth of Julia seeks Pegasus and Julia seeks the square circle because of the substitutivity of identicals.

### 20.1.8 Frege's solution to the second problem

Frege 1892 concluded from the non-equivalence of, e.g., Julia seeks a unicorn and Julia seeks a square circle that in addition to the real objects in the world there also exist natural language meanings, called *sense* (Sinn), which are independent of their referents *referents* (Bedeutung).

### 20.1.9 Ontological status of meaning (sense)

Frege attributed a similar form of existence to the meanings of natural language as to the numbers and their laws in mathematical realism. Mathematical realism proceeds on the assumption that the laws of mathematics exist even if no one knows about them; mathematicians *discover* laws which have extemporal validity. Frege supposed the meanings of natural language to exist in the same way, i.e., independently of whether there are speakers-hearers who have discovered them and use them more or less correctly.

## 20.2 Intension and extension

### 20.2.1 Examples of Carnap's *Intensions*


proposition:  $I \times J \rightarrow \{0,1\}$   
*intension*  
*extension*

proper name:  $I \times J \rightarrow a \in A$   
*intension*  
*extension*


1-pl. predicate:  $I \times J \rightarrow \{a_1, a_2, \dots\} \subseteq A$   
*intension*  
*extension*

## 20.2.2 Two approaches to meaning

### Frege's theory: [+sense]

1. surface of expression
  2. meaning (*sense*)
  3. referent
- 

### Carnap's theory: [-sense]

1. surface of expression
  2. function: index
  3. extension
- 

## 20.3 Propositional attitudes

### 20.3.1 Two basic problems of logical semantics for natural language

- the Epimenides paradox and
- the problem of propositional attitudes.

### 20.3.2 Example of a propositional attitude

Suzanne believes that Cicero denounced Catiline.

### 20.3.3 Assumption of modal logic regarding proper names: rigid designators

According to the intuitions of modal logic, a proper name denotes the same individual in all possible worlds (rigid designator). For example, because Cicero and Tullius are names for one and the same person it holds necessarily (i.e, in all possible worlds) that Cicero = Tullius. Therefore, it follows necessarily from the truth of Cicero denounced Catiline that Tullius denounced Catiline.

### 20.3.4 Problem for propositional attitudes

Even though the referents of Cicero and Tullius are necessarily identical, this could be unknown to Suzanne. Therefore, a valid substitution *salva veritate* would require the truth of Suzanne believes that Cicero is Tullius.

Determining what an individual believes depends on what the individual chooses to report. Because it cannot be checked objectively whether this is true or not, individual ‘belief-worlds’ have justly been regarded as a prime example of what lies outside any scientific approach to truth.

### 20.3.5 Fundamental question of logical semantics: Formulation I

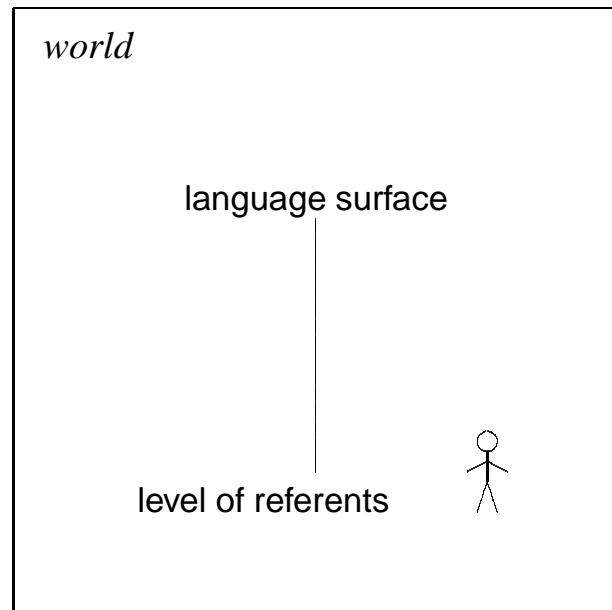
Definition of truth (conditions) by means of meaning or  
definition of meaning in terms of truth (conditions)?

### 20.3.6 Fundamental question of logical semantics: Formulation II

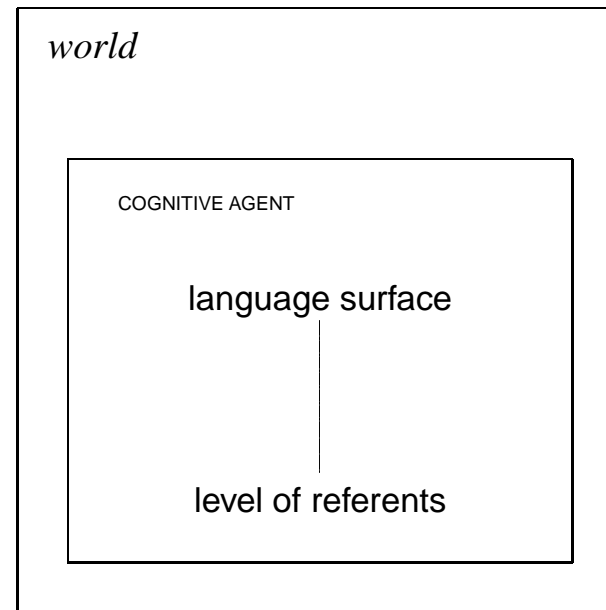
Is the speaker-hearer part of the model structure or  
is the model structure part of the speaker-hearer?

### 20.3.7 Two ontological interpretations of model theory

[-constructive]



[+constructive]



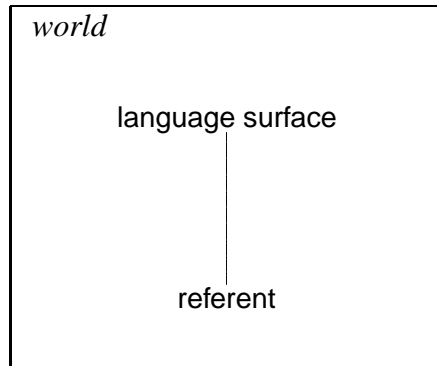
### 20.3.8 The most fundamental difference between [ $\pm$ constructive] ontologies

- Any system based on a [-constructive] ontology must have a metalanguage-based semantics.
- Any system based on a [+constructive] ontology must have a procedural semantics.

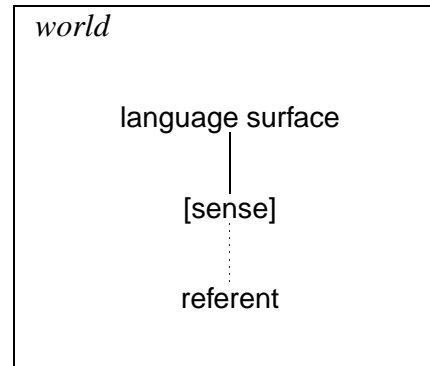
## 20.4 Four basic ontologies

### 20.4.1 Ontologies of semantic interpretation

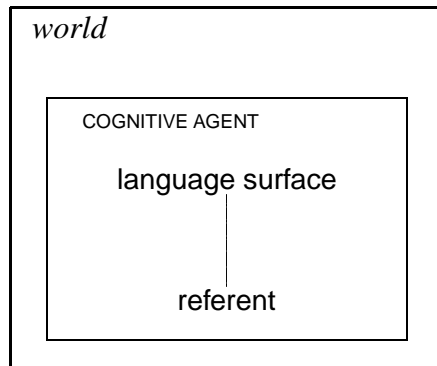
**i** [-sense, -constructive]  
Russell, Carnap, Quine, Montague



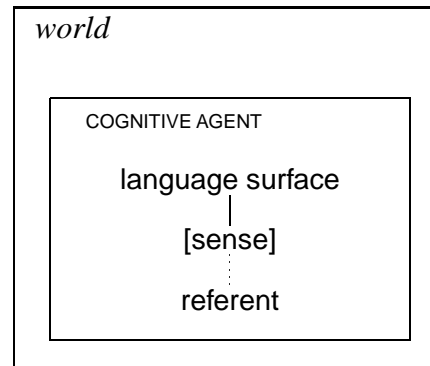
**ii** [+sense, -constructive]  
Frege



**iii** [-sense, +constructive]  
Newell & Simon, Winograd, Shank



**iv** [+sense, +constructive]  
Anderson, CURIOUS, SLIM-machine



### **20.4.2 The [-sense,-constructive] ontology (i) of logical semantics**

Concerned with a solid foundation for truth, logical semantics uses only referents which are considered to be real. Given its ontological foundations, logical semantics is in principle unsuitably for a complete analysis of natural language meaning.

### **20.4.3 The [+sense,-constructive] ontology (ii) of Frege**

Attempt to analyze uneven (opaque, intensional) readings in natural language. As a theory of truth, any [-constructive], metalanguage-based semantics is necessarily incompatible with representing cognitive states.

### **20.4.4 The [-sense,+constructive] ontology (iii) of programming languages.**

Designed to control electronic procedures via the commands of a programming language. A direct, fixed connection between language expressions and their referents prevents autonomous classification of new objects in principle. Therefore, [-sense, +constructive] systems are limited to closed worlds created by the programmer.

### **20.4.5 The [+sense,+constructive] ontology (iv) of the SLIM theory of language**

The [+sense] property is the structural basis for matching of meaning<sub>1</sub> and the context of use, while the [+constructive] property allows the matching to occur inside the cognitive agent.



## 20.5 Sorites paradox and the treatment of vagueness

### 20.5.1 Sorites paradox or paradox of the heap

One grain of sand does not make a heap. Adding an additional grain still doesn't make a heap. If  $n$  grains do not form a heap, then adding another single grain will not make a heap either. However, if this process of adding a grain is continued long enough, there will eventually result a genuine heap.

### 20.5.2 Vagueness as motivation for non-bivalent logic

Sensitive students of language, especially psychologists and linguistic philosophers, have long been attuned to the fact that natural language concepts have vague boundaries and fuzzy edges and that, consequently, natural-language sentences will very often be neither true, nor false, nor nonsensical, but rather true to a certain extent and false to a certain extent, true in certain respects and false in other respects.

G. Lakoff 1972, p. 183

### 20.5.3 Future-contingent propositions as motivation for non-bivalent logic

Throughout the orthodox mainstream of the development of logic in the West, the prevailing view was that every proposition is either true or else false - although which of these is the case may well neither be *necessary* as regards the matter itself nor *determinable* as regards our knowledge of it. This thesis, now commonly called the “Law of Excluded Middle”, was, however, already questioned in antiquity. In Chap. 9 of his treatise *On Interpretation (de interpretatione)*, Aristotle discussed the truth status of alternatives regarding “future-contingent” matters, whose occurrence – like that of the sea battle tomorrow – is not yet determinable by us and may indeed actually be undetermined.

N. Rescher, 1969, p. 1

### 20.5.4 The basic problem of three-values logics and the many-valued logics

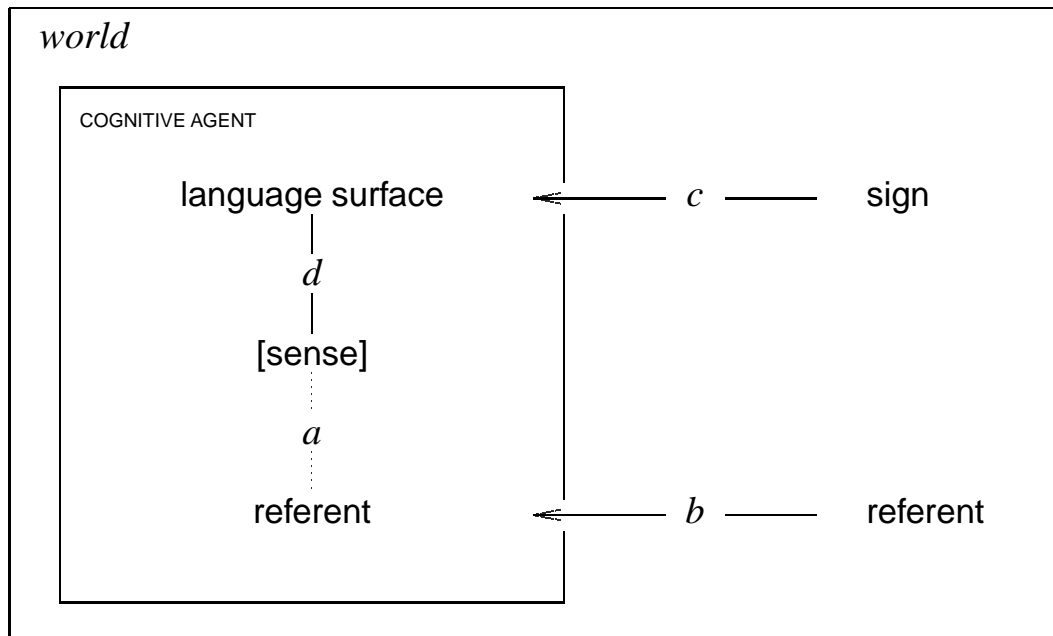
Which truth-value should be assigned to complex propositions based on component propositions with non-bivalent truth-values?

For example: What should be the value of, e.g.,  $A \& B$  if  $A$  has the value 1 and  $B$  has the value #? Similarly in a many-valued system: if the component proposition  $A$  has the truth-value 0.615 and  $B$  has the value 0.423, what value should be assigned to  $A \& B$ ?

### 20.5.5 Vagueness in [-sense,-constructive] semantics

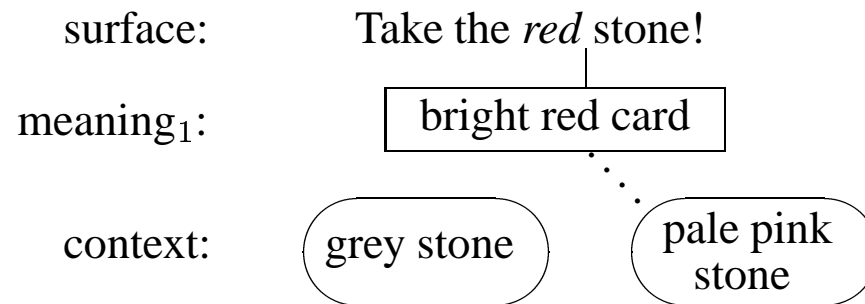
<i>world</i>				
language surfaces:	[the door is open]	and	[the door is red]	
referents:	0,615		0,423	

### 20.5.6 Vagueness in [+sense,+constructive] semantics

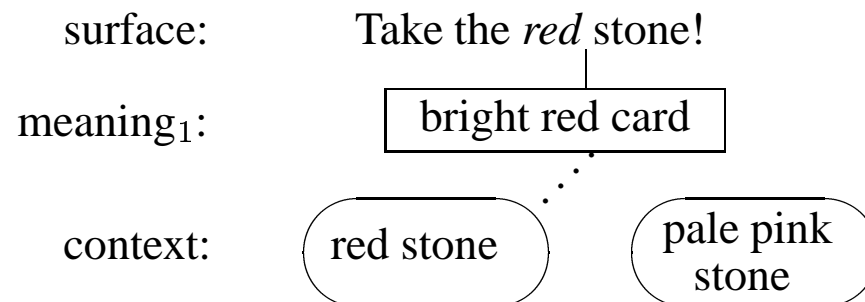


### 20.5.7 Why vagueness is not a property of language meaning

The hearer is faced with a context consisting of a grey stone and a pale pink stone. Responding to the utterance Take the red stone, the cooperative hearer will pick the pale pink stone. For simplicity, the meaning<sub>1</sub> of red is represented by a bright red card.



If the grey stone is replaced by a dark red one, the pale pink stone ceases to be the best match. Responding to Take the red stone, the cooperative hearer will not pick the pale pink stone, but the red one.



It is not the meaning<sub>1</sub> of red which changed, but the context.

## 21. Absolute and contingent propositions

### 21.1 Absolute and contingent truth

#### 21.1.1 Notion of proposition in logic

Specialized use, representing sentences which do not require knowledge of the utterance situation for semantic interpretation. This use is problematic because it constitutes a hybrid between an *utterance* and an *expression*.

#### 21.1.2 Absolute propositions

Express scientific or mathematical contents. These are special in that they make the interpretation largely independent from the usual role of the speaker. For example, in

In a right-angled triangle, it holds for the hypotenuse A and the cathetes B and C that  $A^2 = B^2 + C^2$ .  
the circumstances of the utterance have no influence on interpretation and truth value.

#### 21.1.3 Logical truth for absolute propositions

Logical truth is represented by the metalanguage words false and true referring to the set-theoretic objects  $\emptyset$  und  $\{\emptyset\}$ , respectively. These serve as model-theoretic fix points into which the denotations of propositions are mapped by the metalanguage rules of interpretation.

### 21.1.4 Contingent propositions

Express everyday contents such as Your dog is doing well.

Can only be interpreted – and thereby evaluated with respect to their truth value – if relevant circumstances of utterance situation (STAR point) are known.

### 21.1.5 Natural truth for contingent proposition

Represented by the truth values  $\text{true}^c$  and  $\text{false}^c$ . A contingent proposition such as

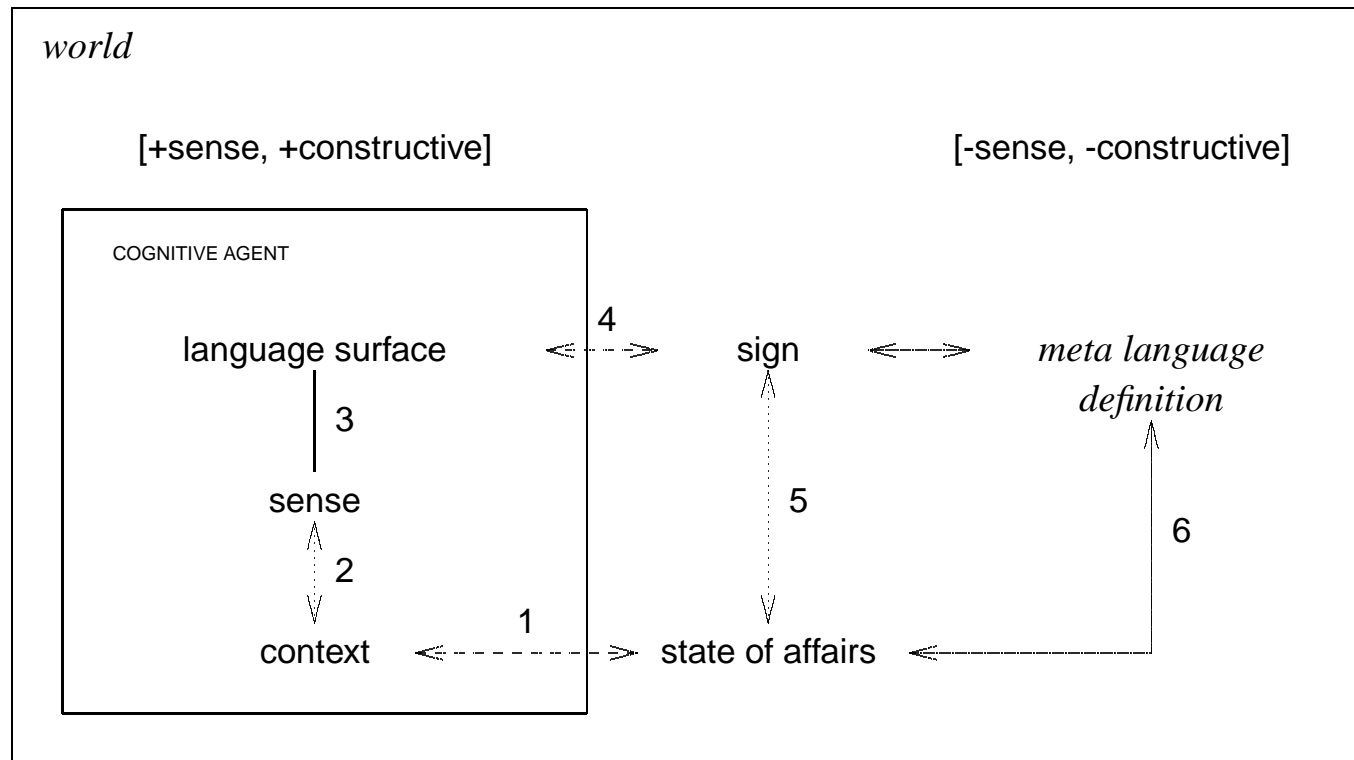
The Persians have lost the battle

is  $\text{true}^c$ , if the speaker is an eye witness who is able to correctly judge and communicate the facts, or if there exists a properly functioning chain of communication between the speaker and a reliable eye witness.

### 21.1.6 Procedural definition of the natural truth values $\text{true}^c$ and $\text{false}^c$

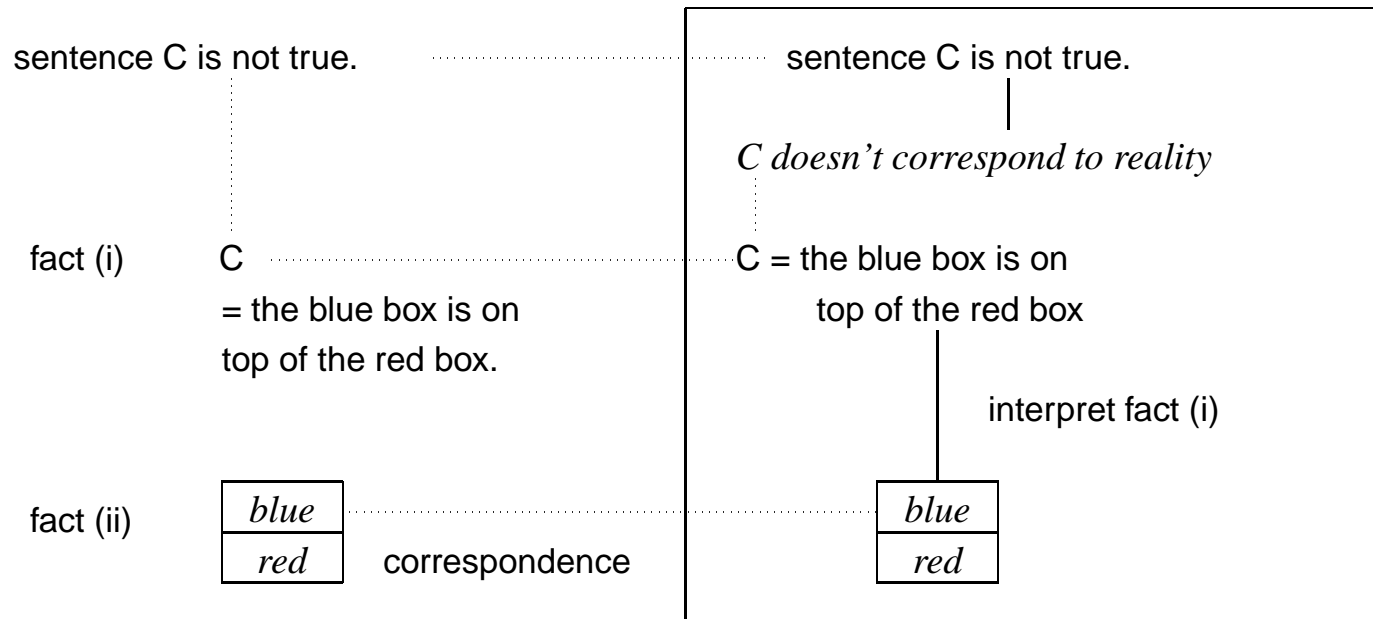
A proposition – or rather a statement – uttered by, e.g., a robot is evaluated as  $\text{true}^c$ , if all procedures contributing to communication work correctly. Otherwise it is evaluated as  $\text{false}^c$ .

## 21.1.7 Comparing natural and logical truth



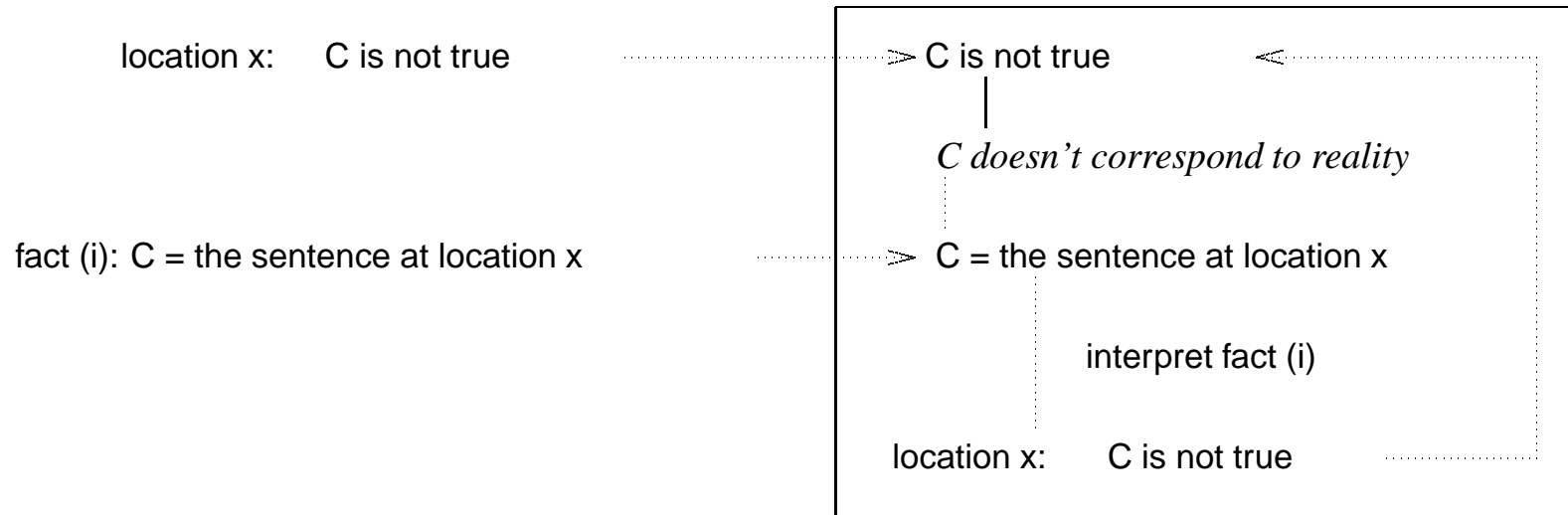
## 21.2 Epimenides in a [+sense,+constructive] system

### 21.2.1 Benign case of a language-based abbreviation





### 21.2.2 A [+constructive,+sense] reanalysis of the Epimenides paradox



### 21.2.3 How the [+constructive,+sense] reanalysis disarms the Epimenides paradox

- the words true<sup>c</sup> and false<sup>c</sup> may be part of the object language without causing a logical contradiction in its semantics, and
- the recursion caused by the Epimenides paradox can be recognized in the pragmatics and taken care of without adversely affecting the communicative functioning of the system.

### 21.2.4 Basis of the reanalysis of the Epimenides paradox

The distinction between (i) the logical truth values 1 and 0 from the T-condition and (ii) the natural truth values  $\text{true}^c$  and  $\text{false}^c$  from the object language replaces Tarski's logical contradiction

*a.* C is 1 if and only if C is not 1

by the contingent statement

*b.* C is 1 if and only if C is not  $\text{true}^c$ .

### 21.2.5 Why the reanalysis is not open to logical semantics

The procedural notion of natural truth – essential for avoiding Tarski's contradiction – can be neither motivated nor implemented outside a [+constructive,+sense] ontology.

## 21.3 Frege's principle as homomorphism

### 21.3.1 The communicative function of natural syntax

is the composition of semantic representations by means of composing the associated surfaces. Montague formalized this structural correlation between syntax and semantics mathematically as a *homomorphism*.

### 21.3.2 Intuitive notion of a homomorphism

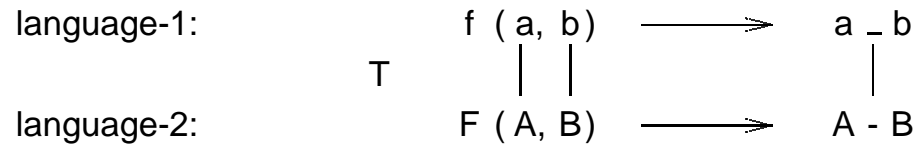
A structural object  $so$  is homomorphic to another structural object  $SO$ , if for each basic element of  $so$  there is a (not necessarily basic) counterpart in  $SO$ , and for each relation between elements in  $so$  there is a corresponding relation between corresponding elements in  $SO$ .

### 21.3.3 Homomorphism as a relation between two (uninterpreted) languages

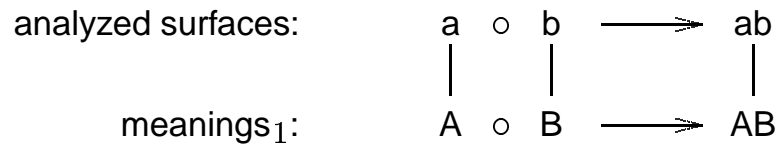
Language-2 is homomorphic to language-1 if there is a function  $T$  which

- assigns to each word of category  $a$  in language-1 a corresponding expression of category  $A$  in language-2, and
- assigns to each  $n$ -place composition  $f$  in language-1 a corresponding  $n$ -place composition  $F$  in language-2, such that
- $T(f(a,b)) = F((T(a))(T(b)))$

### 21.3.4 Schematic representation of Montague's homomorphism



### 21.3.5 Syntactic composition with homomorphic semantics

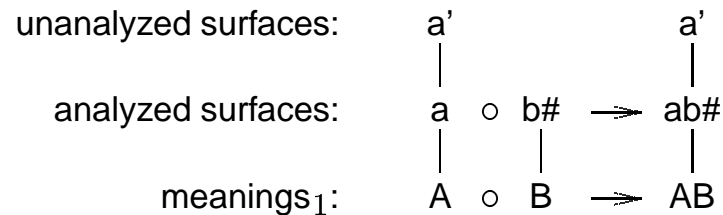


### 21.3.6 Why the homomorphism condition by itself is not sufficient as a formalization of Frege's principle

Frege's principle is defined for *analyzed* surfaces, whereas natural language communication is based on *unanalyzed* surfaces. The problem is that the transition from unanalyzed to analyzed surfaces (interpretation) and vice versa (production) has been misused to enrich the levels of the analyzed surface and/or the meaning<sub>1</sub> by means of zero elements or identity mappings.

### 21.3.7 Use of zero element (illegal)

#### 1. Smuggling in during interpretation ( $\downarrow$ ) – Filtering out during production ( $\uparrow$ )

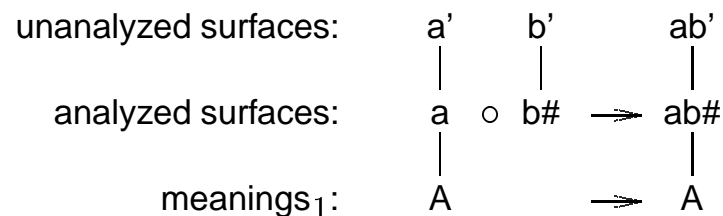


Postulated whenever the unanalyzed surface does not contain what the grammar theory would like to find.

Peter drank DET# wine

YOU# help me!

#### 2. Filtering out during interpretation ( $\downarrow$ ) – Smuggling in during production ( $\uparrow$ )



Postulated whenever the surface contains something which the grammar theory would not like to find.

Peter believes THAT# Jim is tired.

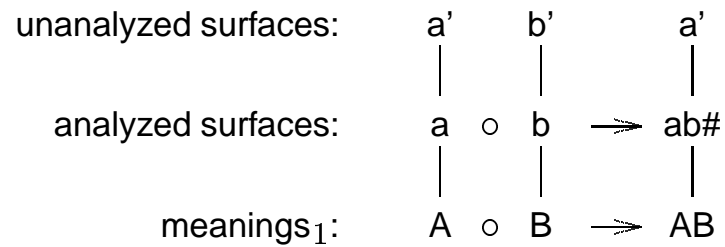
mixed:DET# wine WAS# ordered BY# Peter

mixed: Peter promised Jim TO# Peter# sleep

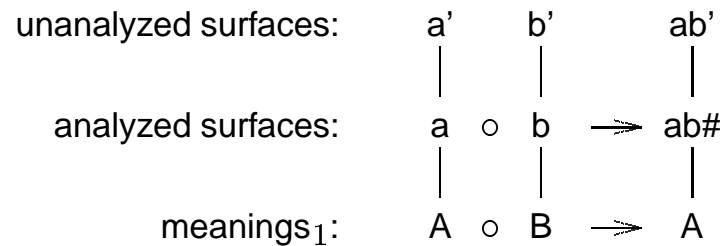
mixed: Peter persuaded Jim TO# Jim# sleep.

### 21.3.8 Use of identity mapping (illegal)

#### 1. Filtering out during production ( $\uparrow$ ) – Smuggling in during interpretation ( $\downarrow$ )



#### 2. Smuggling in during production ( $\uparrow$ ) – Filtering out during interpretation ( $\downarrow$ )



### 21.3.9 Surface compositionality II (SC-II principle)

A semantically interpreted grammar is surface compositional if and only if

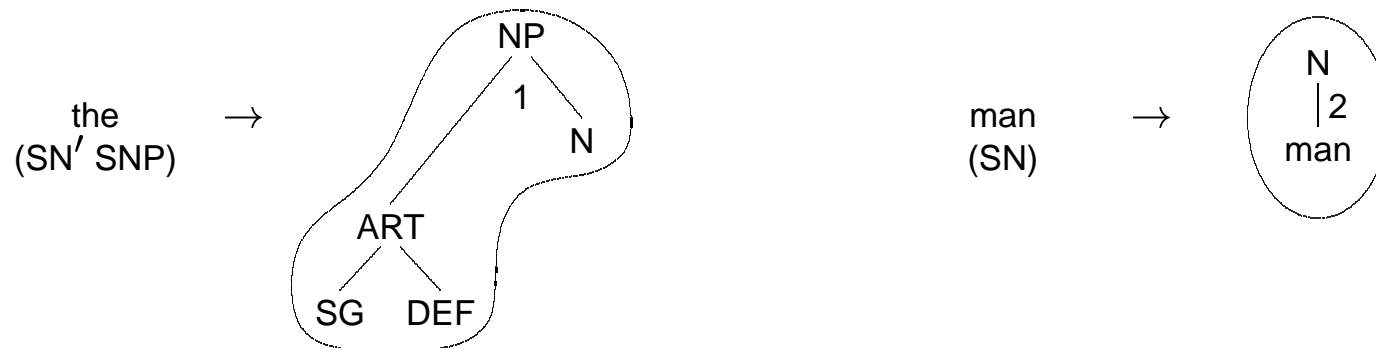
- the syntax is restricted to the composition of concrete word forms (i.e. no zero elements and no identity mappings),
- the semantics is homomorphic to the syntax, and
- objects and operations on the level of semantics which correspond to the syntax in accordance with the homomorphism condition may not be realized by zero elements or identity mappings.

## 21.4 Time-linear syntax with homomorphic semantics

### 21.4.1 Time-linear build-up of semantic hierarchies

- *Step 1: Translation of word forms into component hierarchies*  
Each word form is mapped into a semantic component hierarchy (tree). The structure of the tree is determined by the syntactic category of the word form.
- *Step 2: Left-associative combination of component hierarchies*  
For each combination of the left-associative syntax there is defined a corresponding combination of component hierarchies on the level of the semantics.

### 21.4.2 Derivation of component hierarchies from word forms





---

### **21.4.3 Time-linear composition with homomorphic semantics**

#### **21.4.4 Why 21.4.3 is not a constituent structure**

A constituent structure analysis would proceed on the assumption that *gave* is semantically closer to the woman and the book than to the man.

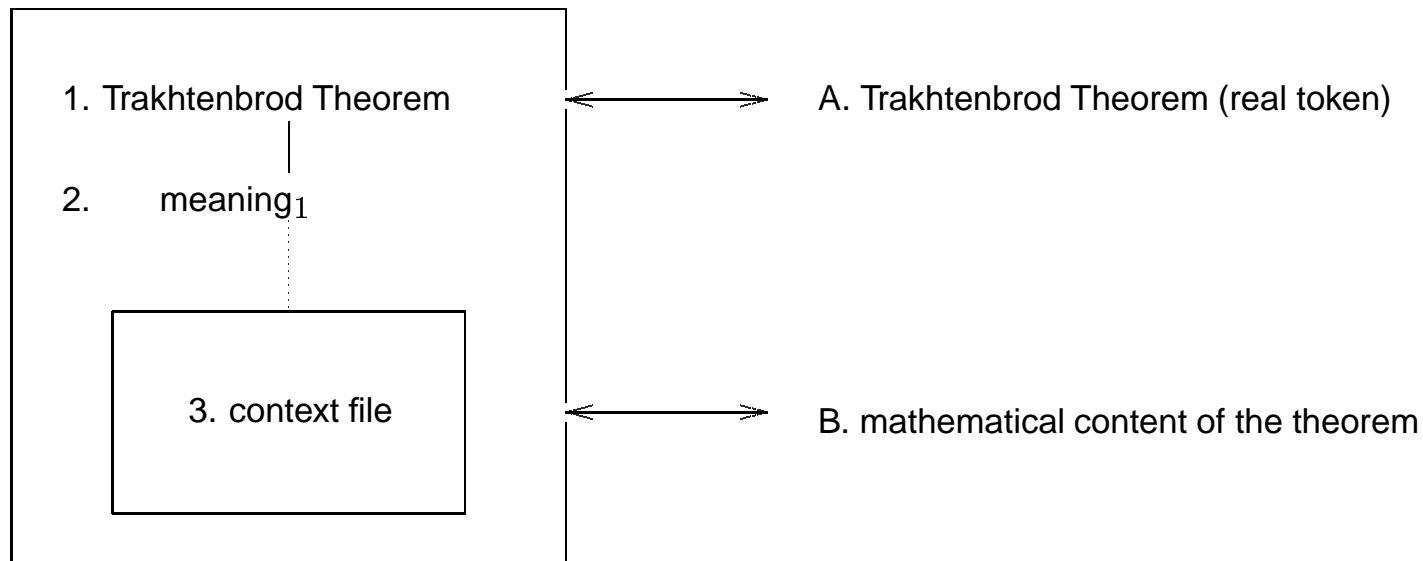
## 21.5 Complexity of natural language semantics

### 21.5.1 Low complexity of syntactic system may be pushed sky high by semantic interpretation

(a)  $\pi$   
|  
3.14159265...

(b) 1:3  
|  
1' : 3' = 0.333...

### 21.5.2 Interpretation of 'Trakhtenbrod Theorem' within SLIM theory



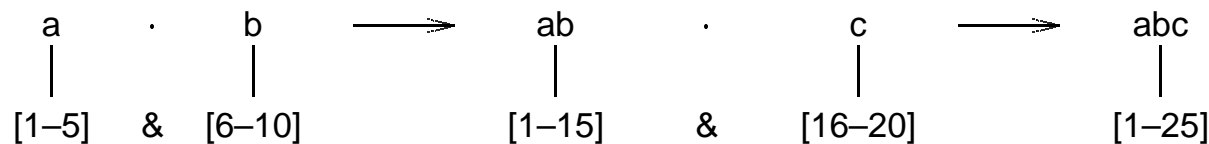
### 21.5.3 CoNSem hypothesis (Complexity of Natural language Semantics)

The interpretation of a natural language syntax within the C-LAGs is empirically adequate only if there is a finite constant  $k$  such that

- it holds for each elementary word form in the syntax that the associated semantic representation consists of at most  $k$  elements, and
- it holds for each elementary composition in the syntax that the associated semantic composition increases the number of elements introduced by the two semantic input expressions by maximally  $k$  elements in the output.

This means that the semantic interpretation of syntactically analyzed input of length  $n$  consists of maximally  $(2n - 1) \cdot k$  elements.

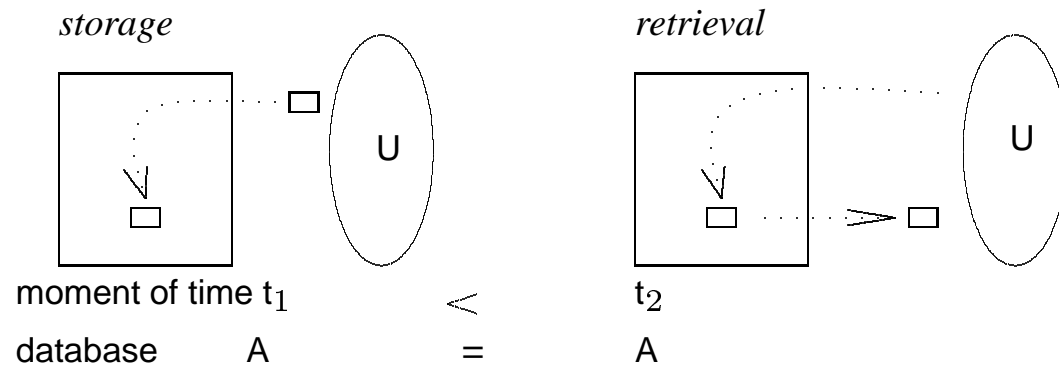
### 21.5.4 Illustration of CoNSem hypothesis with $k = 5$



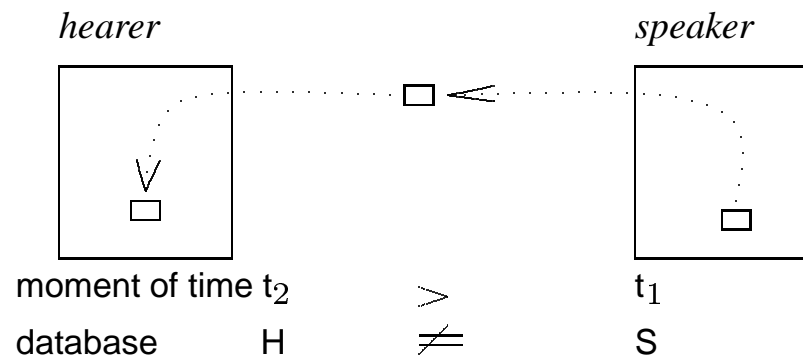
## 22. Database semantics

### 22.1 Database metaphor of natural communication

#### 22.1.1 Interaction with a conventional database



#### 22.1.2 Interaction between speaker and hearer



### 22.1.3 DB interaction and NL communication

- ENTITIES INVOLVED

*Database interaction:* takes place between two different entities, the user and the database.

*NL communication:* takes place between two similar and equal cognitive agents, the speaker and the hearer.

- ORIGIN OF CONTROL

*Database interaction:* operations of input and output are controlled by the user.

*NL communication:* there is no user. Instead, the cognitive agents control each other by alternating in the speaker- and the hearer-mode (*turn taking*).

- METHOD OF CONTROL

*Database interaction:* user controls the operations of the database with a programming language the commands of which are executed as electronic procedures.

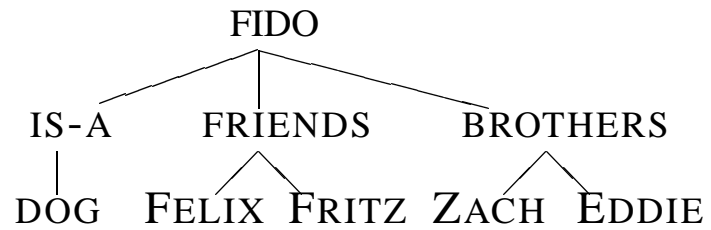
*NL communication:* speaker controls language production as an autonomous agent, coding the parameters of the utterance situation into the output expressions. The hearer's interpretation is controlled by the incoming language expression.

- TEMPORAL ORDER

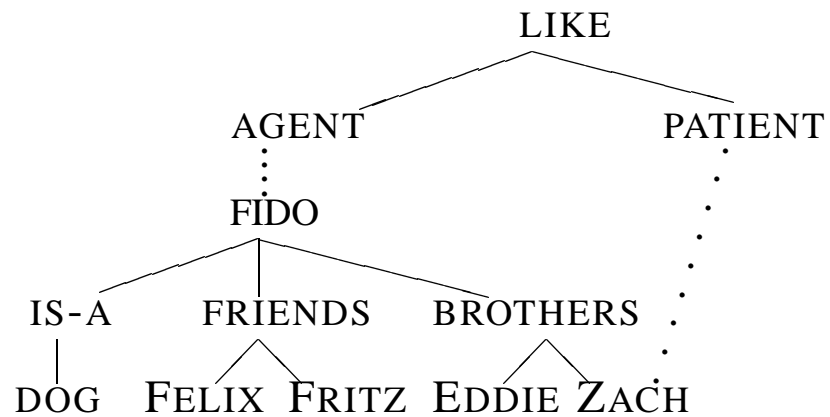
*Database interaction:* output (database as 'speaker') occurs necessarily *after* the input (database as 'hearer').

*NL communication:* production (output procedure of the speaker) occurs necessarily *before* interpretation (input procedure of the hearer).

### 22.1.4 Sketch of a simple subcontext



### 22.1.5 Pragmatic interpretation of 22.1.1



## 22.2 Descriptive aporia and embarrassment of riches

### 22.2.1 Model-theoretic definition of a context

Let  $\mathcal{MS}$  be a model structure  $(A, I, J, \leq, F)$ , where  $A, I, J$  are sets,  $\leq$  is a simple ordering on  $J$ , and  $F$  is a denotation function.

$A, I, J$ , and  $F$  have the following definition:

$$A = \{a_0, a_1, a_2, a_3, a_4\}, I = \{i_1\}, J = \{j_1\}$$

$$F(\text{fido}')(i_1, j_1) = a_0$$

$$F(\text{felix}')(i_1, j_1) = a_1$$

$$F(\text{fritz}')(i_1, j_1) = a_2$$

$$F(\text{zach}')(i_1, j_1) = a_3$$

$$F(\text{eddie}')(i_1, j_1) = a_4$$

$$F(\text{dog}')(i_1, j_1) = \{a_0\}$$

$$F(\text{fido-friends}')(i_1, j_1) = \{a_1, a_2\}$$

$$F(\text{fido-brothers}')(i_1, j_1) = \{a_3, a_4\}$$

### 22.2.2 Extending the hearer context to the meaning of a new sentence such as Fido likes Zach

Requires automatic addition of ' $F(\text{like})(i_1, j_1) = \{(a_0, a_3)\}$ ' to 22.2.1



### 22.2.3 Creating a *frame*

```
(make-frame
  fido
  (is-a (value dog))
  (friends (value felix fritz))
  (brothers (value zach eddie))
)
```

### 22.2.4 Definition of 22.4.2 as a *frame*

```
(fido
  (is-a (value dog))
  (friends (value felix fritz))
  (brothers (value zach eddie))
)
```

### 22.2.5 Retrieving information

```
(get-values 'FIDO 'FRIENDS)
(FELIX FRITZ)
```

## 22.2.6 Extending the hearer context to Fido likes Zach

Requires deriving

```
(fido  
  (like (value Zach)  
  )
```

and automatically adding the part

```
(like (value Zach)
```

as a new slot into 22.2.4.

## 22.3 Propositions as sets of coindexed proplets

### 22.3.1 Proposition 3.4.2 as a set of proplets (preliminary format)

<i>Type:</i> [M-concept: field role: argument ] <i>Token:</i> [I-concept <sub>loc</sub> : x1 functor: contain prn: 23 id: 7 ]	<i>Type:</i> [M-concept: contain role: functor ] <i>Token:</i> [I-concept <sub>loc</sub> : x2 argument 1: field argument 2: triangle prn: 23 epr: 23 and 24 ]	<i>Type:</i> [M-concept: triangle role: argument ] <i>Token:</i> [I-concept <sub>loc</sub> : x3 functor: contain prn: 23 id: 8 ]
--	---	---

<i>Type:</i> [M-concept: field role: argument ] <i>Token:</i> [I-concept <sub>loc</sub> : x4 functor: contain prn: 24 id: 7 ]	<i>Type:</i> [M-concept: contain role: functor ] <i>Token:</i> [I-concept <sub>loc</sub> : x5 argument 1: field argument 2: square prn: 24 epr: 23 and 24 ]	<i>Type:</i> [M-concept: square role: argument ] <i>Token:</i> [I-concept <sub>loc</sub> : x6 functor: contain prn: 24 id: 9 ]
--	---	---

## 22.4 Proplets in a classic database

### 22.4.1 Types of databases

classic: record based

non-classic: based on the principle of slot and filler

### 22.4.2 Types of classic databases

Relational database, hierarchical database, network database

### 22.4.3 Relations between proplet features

type  $\leftrightarrow$  token

token  $\leftrightarrow$  prn

prn  $\leftrightarrow$  epr

token  $\leftrightarrow$  id

functor  $\leftrightarrow$  argument

modifier  $\leftrightarrow$  modified

## 22.4.4 Propositions 3.4.2 as a word bank

TYPES

SIMPLIFIED PROPLETS

[M-concept: contain]  
role: functor

[I-concept<sub>loc</sub>: x2]  
argument 1:field  
argument 2:triangle  
prn: 23  
epr: 23 and 24

[I-concept<sub>loc</sub>: x5]  
argument 1:field  
argument 2:square  
prn: 24  
epr: 23 and 24

[M-concept: field]  
role: argument

[I-concept<sub>loc</sub>: x1]  
functor: contain  
prn: 23  
id: 7

[I-concept<sub>loc</sub>: x4]  
functor: contain  
prn: 24  
id:7

[M-concept: square]  
role: argument

[I-concept<sub>loc</sub>: x6]  
functor: contain  
prn: 24  
id: 9

[M-concept: triangle]  
role: argument

[I-concept<sub>loc</sub>: x3]  
functor: contain  
prn: 23  
id: 8

### 22.4.5 Example of a network database

<i>owner records</i>	<i>member records</i>			
Comp.Sci.	Riedle	Schmidt	Stoll	...
Mathematics	Müller	Barth	Jacobs	...
Physics	Weber	Meier	Miele	...

### 22.4.6 Types of continuations

*intrapositional:*

from argument to functor, functor to argument, from modifier to modified and vice versa

*extrapositional:*

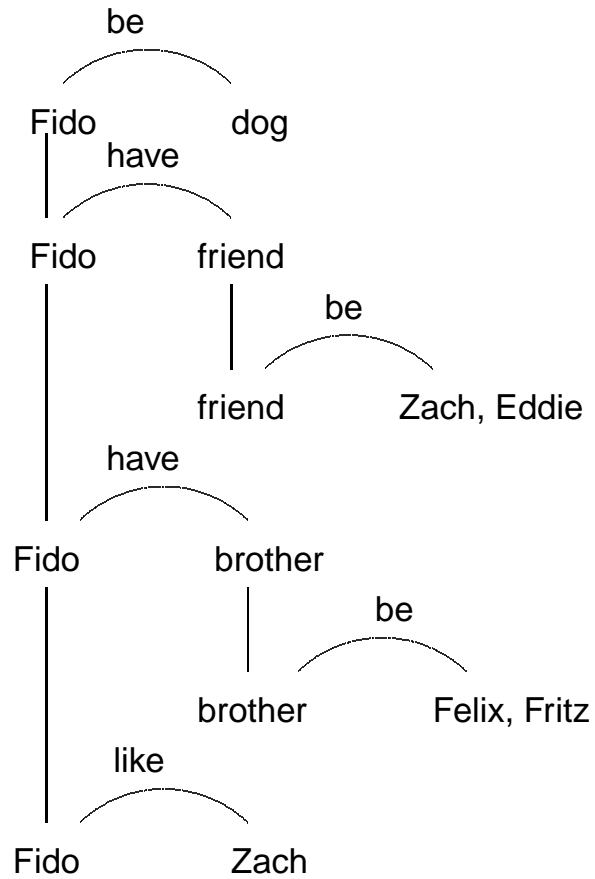
epr from verb to verb, id from noun to noun

## 22.5 Example of a word bank

### 22.5.1 Propositional presentation of subcontext 22.1.4

1. Fido is a dog.
2. Fido has friends.
3. The friends are Zach and Eddie.
4. Fido has brothers.
5. The brothers are Felix and Fritz.
6. Fido likes Zach.

## 22.5.2 Graphical presentation of the propositions in 22.5.1





### 22.5.3 Subcontext 22.1.1 as a word bank

#### TYPES

#### PROPLETS

[M-concept: be]  
role: functor ]

[I-concept<sub>loc</sub>: x1]  
arg1: Fido  
arg2: dog  
prn: 1  
epr: 1 and 2 ]

[I-concept<sub>loc</sub>: x2]  
arg1: friend  
arg2: Zach, Eddie  
prn: 3  
epr: 2 and 3  
3 and 4 ]

[I-concept<sub>loc</sub>: x3]  
arg1: brother  
arg2: Felix, Fritz  
prn: 5  
epr: 4 and 5  
5 and 6 ]

[M-concept: brother]  
role: argument ]

[I-concept<sub>loc</sub>: x4]  
functor: have  
prn: 4  
id: ]

[I-concept<sub>loc</sub>: x5]  
functor: be  
prn: 5  
id: ]

[M-concept: dog]  
role: argument ]

[I-concept<sub>loc</sub>: x6]  
functor: be  
prn: 4  
id: ]

[M-concept: Eddie]  
role: argument ]

[I-concept<sub>loc</sub>: x7]  
functor: be  
prn: 3  
id: 3 ]

[M-concept: Felix role: argument]	[I-concept <sub>loc</sub> : x8 functor: be prn: 5 id: 4]				
[M-concept: Fritz role: argument]	[I-concept <sub>loc</sub> : x9 functor: be prn: 5 id: 5]				
[M-concept: Fido role: argument]	[I-con. <sub>loc</sub> : x10 functor: be prn: 1 id: 1]	[I-con. <sub>loc</sub> : x11 functor: have prn: 2 id: 1]	[I-con. <sub>loc</sub> : x12 functor: have prn: 4 id: 1]	[I-con. <sub>loc</sub> : x13 functor: like prn: 6 id: 1]	&
[M-concept: friend role: argument]	[I-concept <sub>loc</sub> : x14 functor: have prn: 2 id: ]	[I-concept <sub>loc</sub> : x15 functor: be prn: 3 id: ]			
[M-concept: have role: functor]	[I-concept <sub>loc</sub> : x16 arg1: Fido arg2: friend prn: 2 epr: 1 and 2 2 and 3]	[I-concept <sub>loc</sub> : x17 arg1: Fido arg2: brother prn: 4 epr: 3 and 4 4 and 5]			

[M-concept: like] [role: functor]	[I-concept <sub>loc</sub> : x18] arg1: Fido arg2: Zach prn: 6 epr: 5 and 6	&	
[M-concept: Zach] [role: argument]	[I-concept <sub>loc</sub> : x19] functor: be prn: 3 id: 2	[I-concept <sub>loc</sub> : x20] functor: like prn: 6 id: 2	&

## 22.5.4 Semantic representation of proposition 6

TYPES

PROPLETS

[M-concept: Fido]  
[role: argument]

[I-concept<sub>loc</sub>: x13]  
functor: like  
prn: 6  
id: ?]

[M-concept: like]  
[role: functor]

[I-concept<sub>loc</sub>: x18]  
arg1: Fido  
arg2: Zach  
prn: 6  
epr: ?]

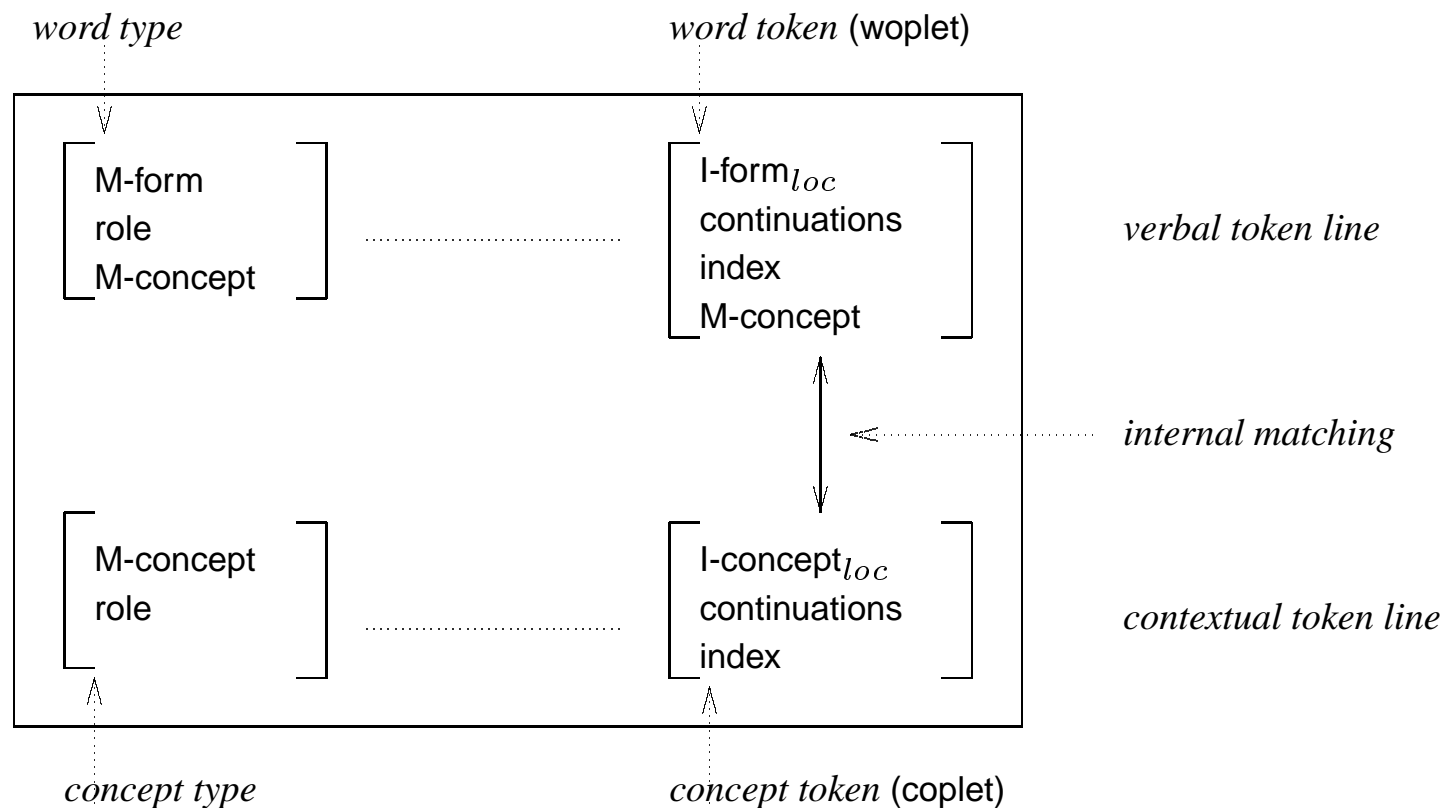
[M-concept: Zach]  
[role: argument]

[I-concept<sub>loc</sub>: x20]  
functor: like  
prn: 6  
id: ?]

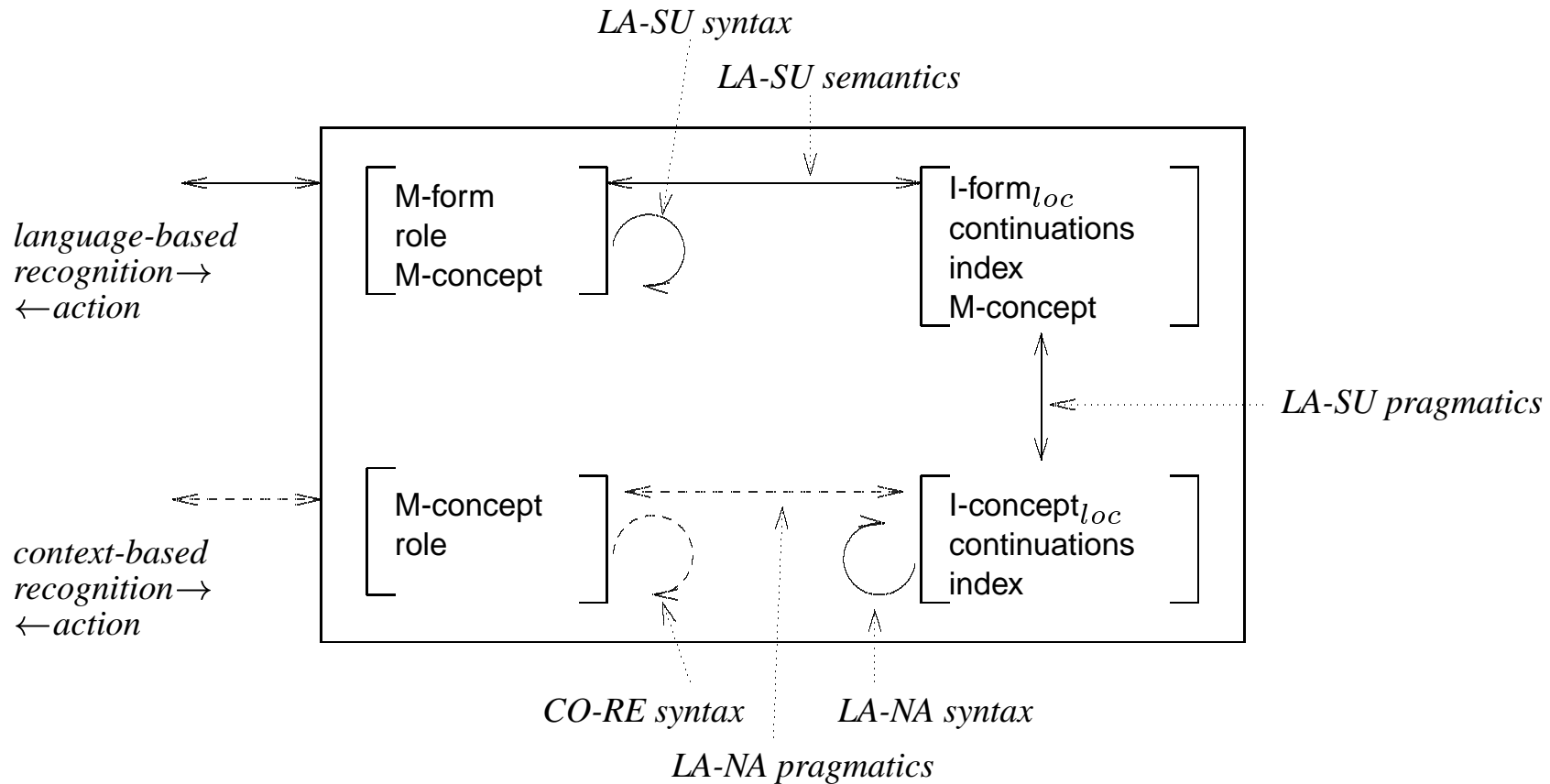
## 23. SLIM machine in the hearer mode

### 23.1 External connections and motor algorithms

#### 23.1.1 Static structures of the SLIM machine

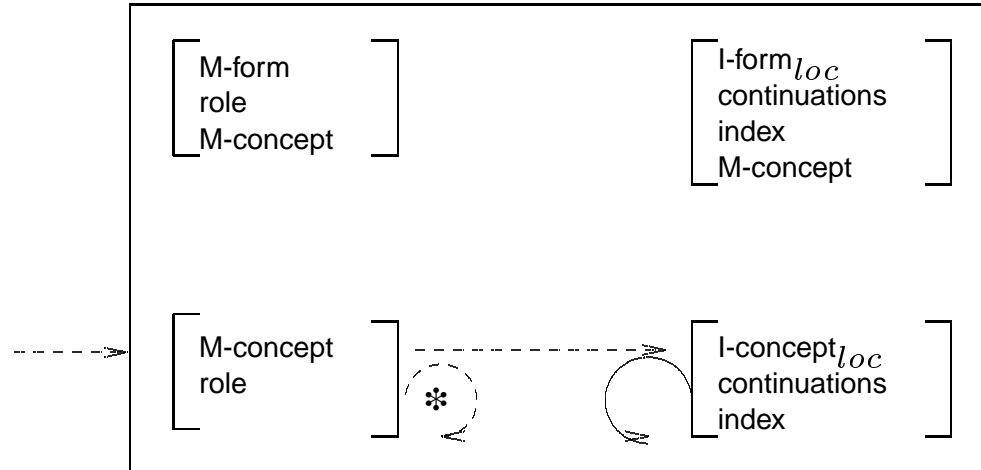


## 23.1.2 External connections and motor algorithms of the SLIM machine

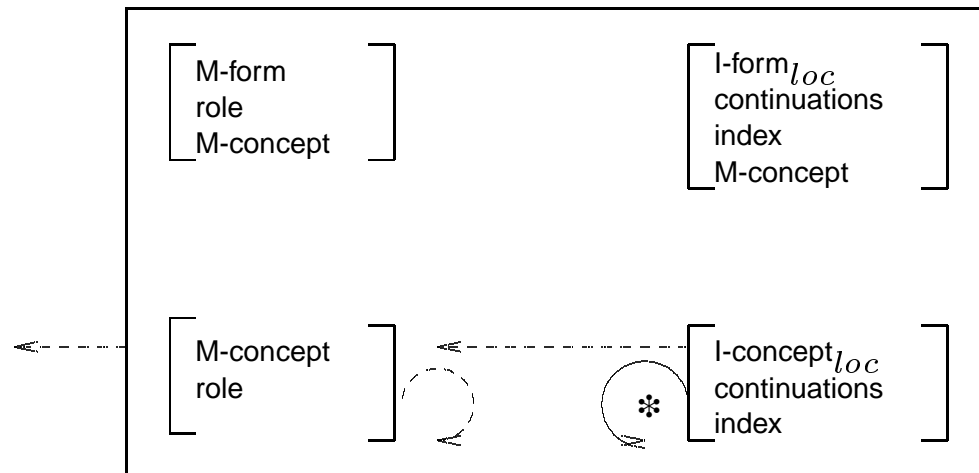


## 23.2 Ten SLIM states of cognition

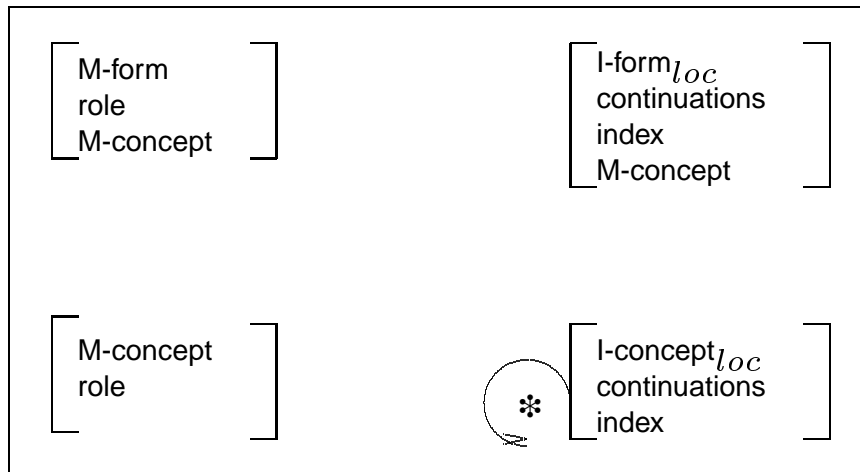
### 23.2.1 SLIM 1: Recognition (contextual)



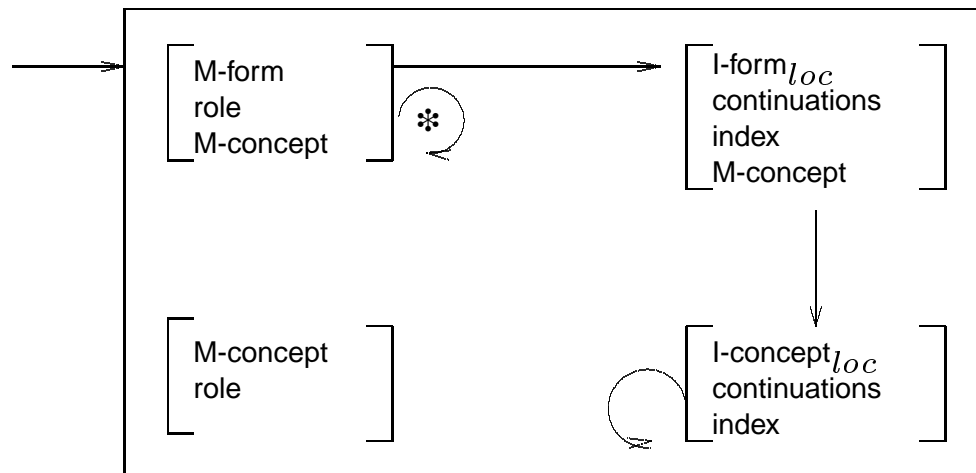
### 23.2.2 SLIM 2: Action (contextual)



### 23.2.3 SLIM 3: Inference (contextual)

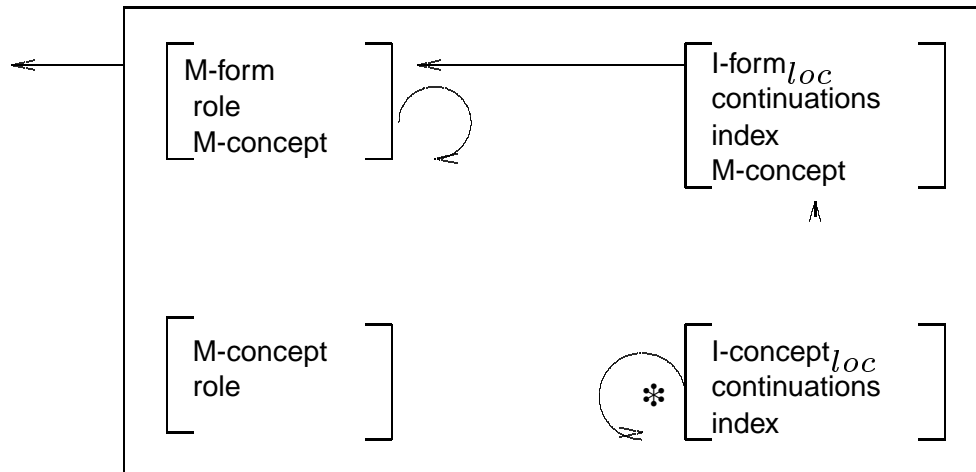


### 23.2.4 SLIM 4: Interpretation of language (mediated reference)

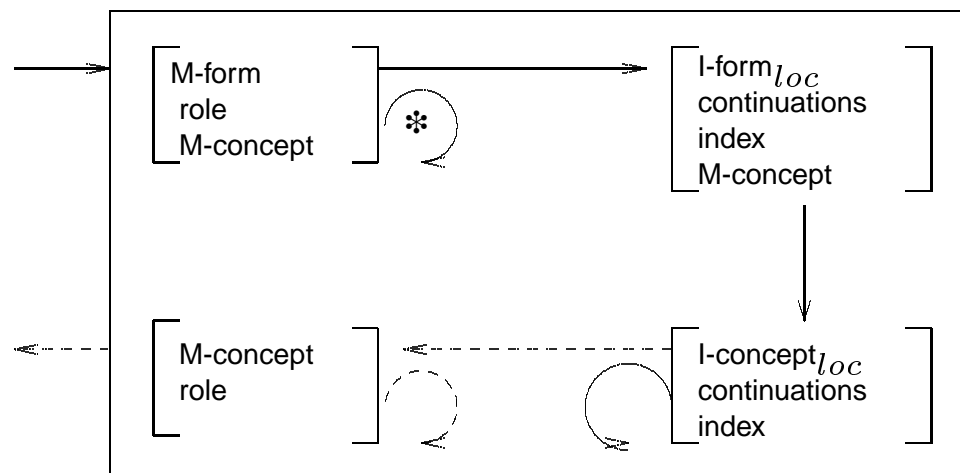




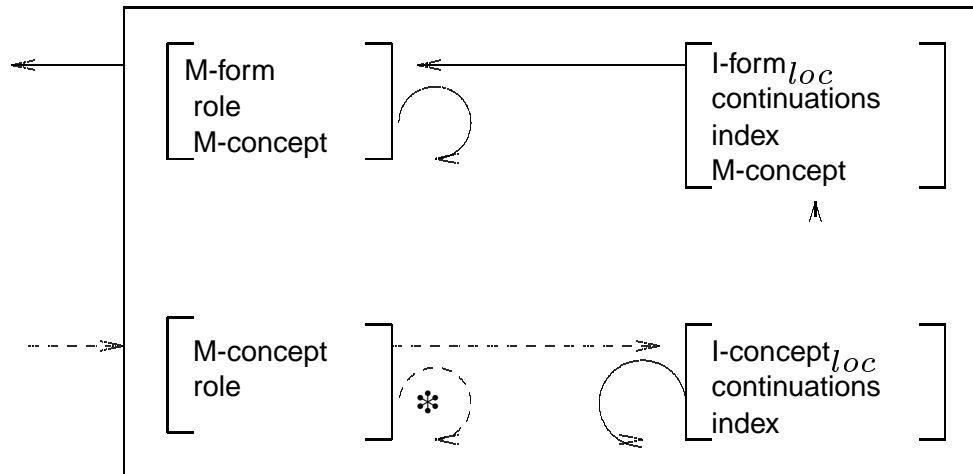
### 23.2.5 SLIM 5: Production of language (mediated reference)



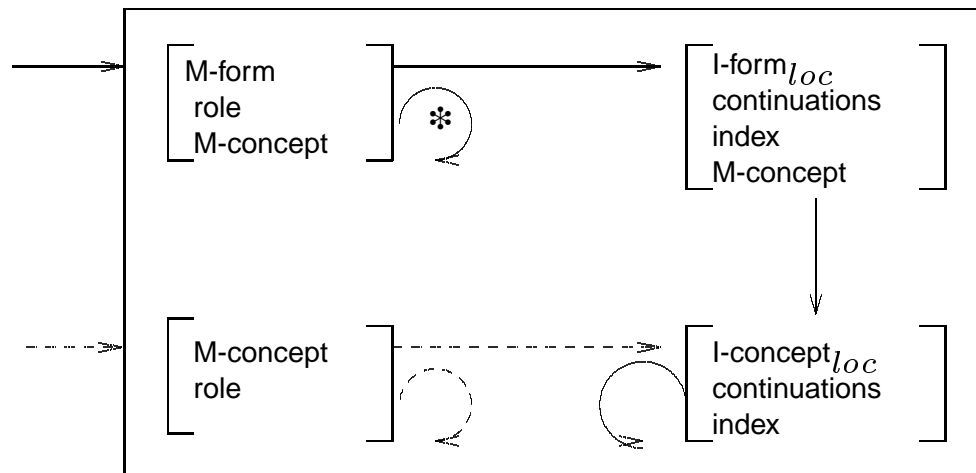
### 23.2.6 SLIM 6: Language-controlled action (immediate reference)



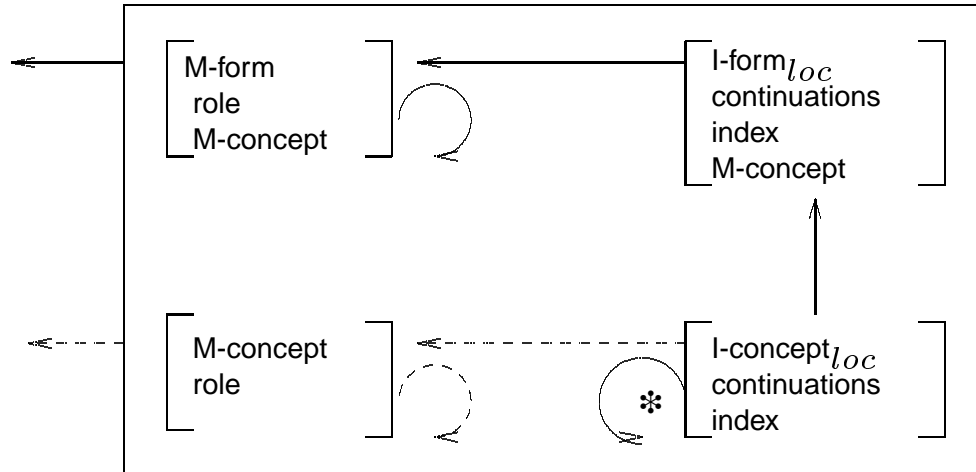
### 23.2.7 SLIM 7: Commented recognition (immediate reference)



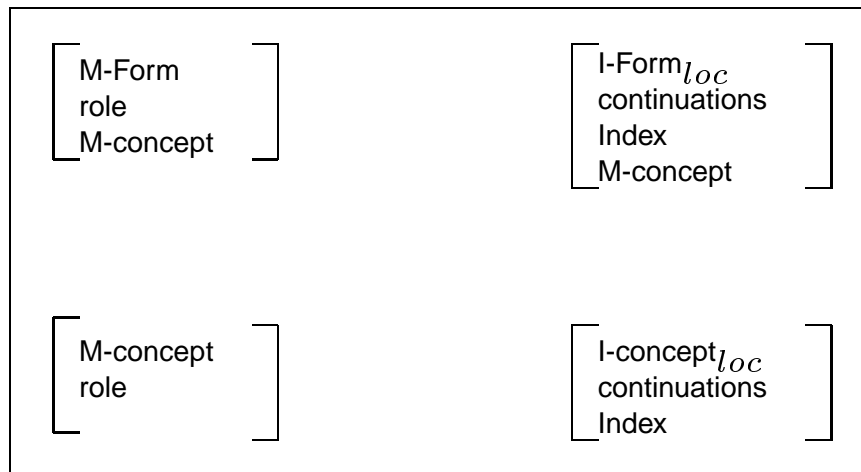
### 23.2.8 SLIM 8: Language-controlled recognition (immediate reference)



### 23.2.9 SLIM 9: Commented action (immediate reference)



### 23.2.10 SLIM 10: Cognitive stillstand



### 23.2.11 Notions grounded in the ten SLIM states

*Context-based* cognition is represented by SLIM 1 to SLIM 3,

*Language-based* cognition is represented by SLIM 4 and SLIM 5,

*Simultaneous context- and language-based* cognition is represented by SLIM 6 to SLIM 9.

*Context-based* cognition distinguishes between *recognition* (SLIM 1), *action* (SLIM 2), and *inferencing* (SLIM 3).

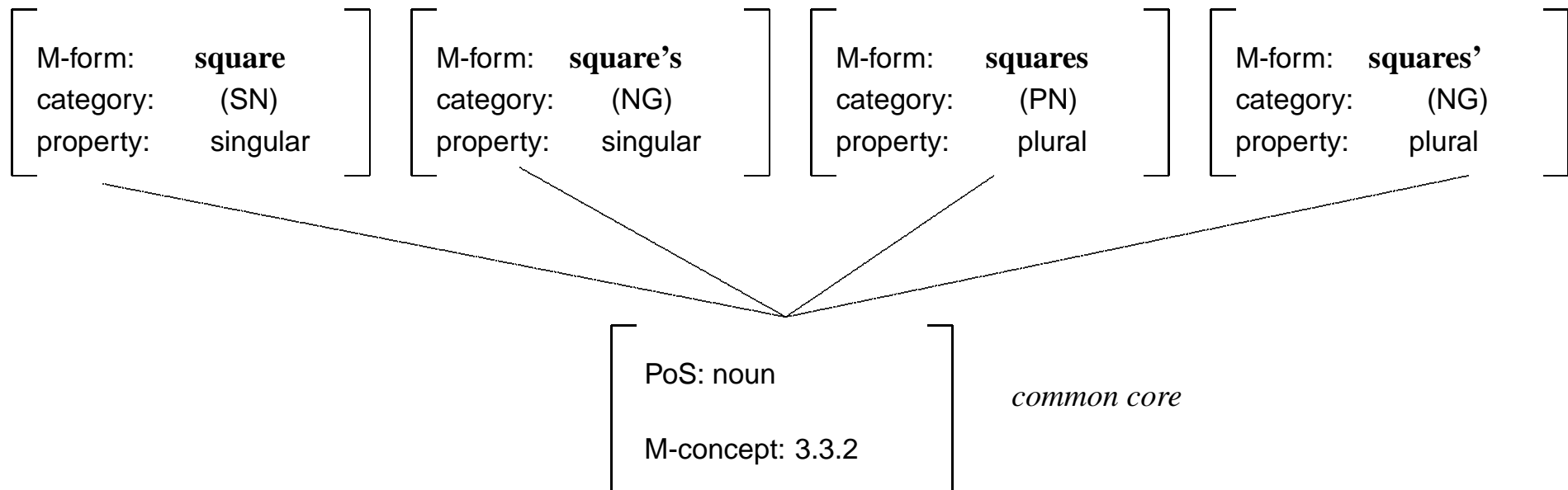
*Language-based* cognition distinguishes between the *hearer mode* (SLIM 4, SLIM 6, SLIM 8), and the *speaker mode* (SLIM 5, SLIM 7, SLIM 9).

*Mediated reference* (SLIM 6 to SLIM 9) is distinguished from *immediate* reference (SLIM 4, SLIM 5).

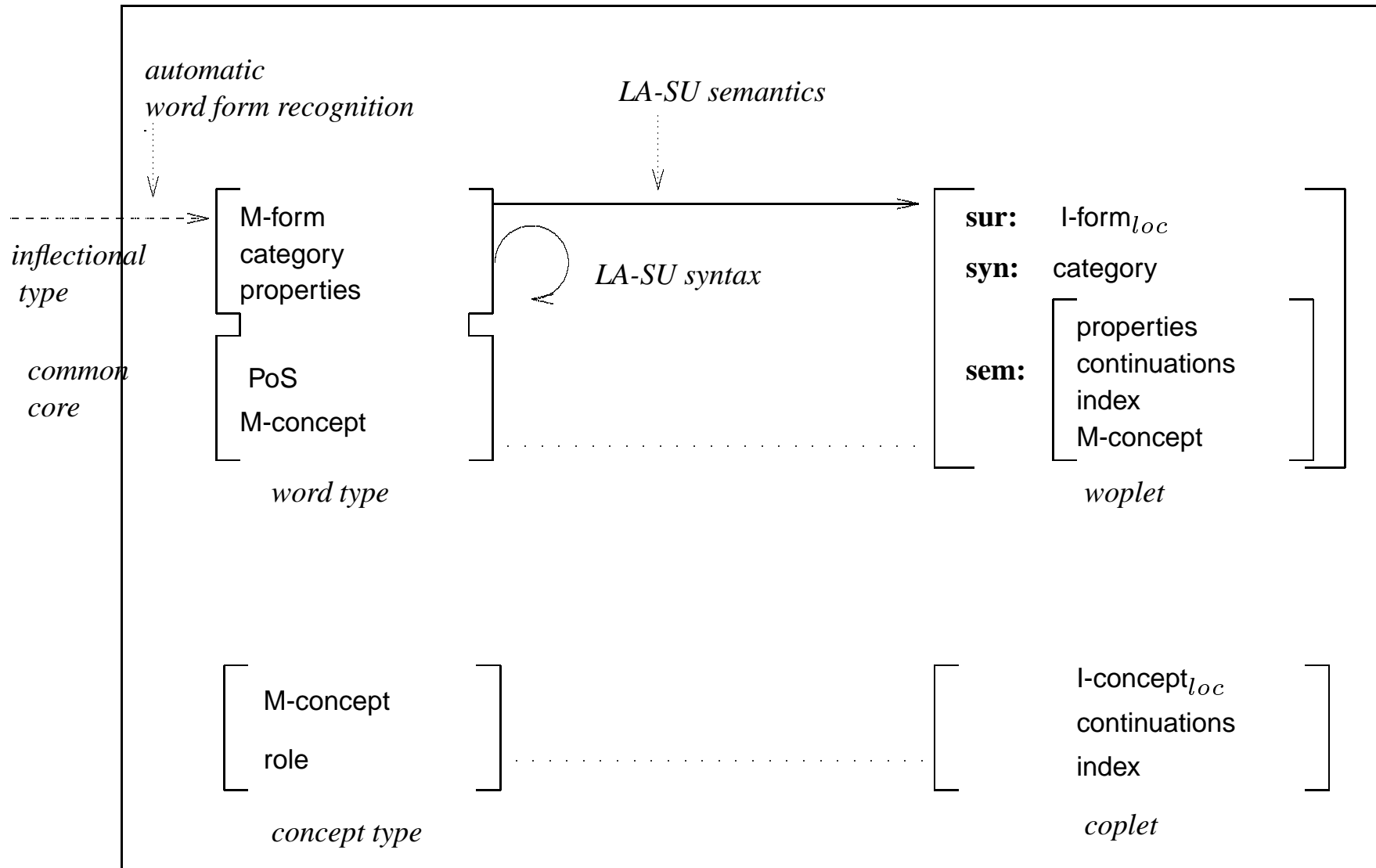
In *immediate reference*, language-based *control* (SLIM 6, SLIM 8) is distinguished from context-based *commenting* (SLIM 7, SLIM 9).

## 23.3 Semantic interpretation of LA-SU syntax

### 23.3.1 Representing inflectional variants in a word type



### 23.3.2 Word form recognition and derivation of a woplet



### 23.3.3 Nominal, verbal, and adjectival woplet structures

*nominal woplet*

$$\left[ \begin{array}{l} \text{sur:} \\ \text{syn:} \\ \\ \text{sem:} \end{array} \left[ \begin{array}{l} \text{properties:} \\ \text{cont.:} \left[ \begin{array}{l} \text{MOD:} \\ \text{VERB:} \end{array} \right] \\ \text{index:} \left[ \begin{array}{l} \text{prn:} \\ \text{id:} \end{array} \right] \\ \text{M-concept:} \end{array} \right] \right]$$

*verbal woplet*

$$\left[ \begin{array}{l} \text{sur:} \\ \text{syn:} \\ \\ \text{sem:} \end{array} \left[ \begin{array}{l} \text{properties:} \\ \text{cont...:} \left[ \begin{array}{l} \text{MOD:} \\ \text{NP:} \end{array} \right] \\ \text{index:} \left[ \begin{array}{l} \text{prn:} \\ \text{epr:} \end{array} \right] \\ \text{M-concept:} \end{array} \right] \right]$$

*adjectival woplet*

$$\left[ \begin{array}{l} \text{sur:} \\ \text{syn:} \\ \\ \text{sem:} \end{array} \left[ \begin{array}{l} \text{properties:} \\ \text{cont.:} \left[ \begin{array}{l} \text{NP:} \\ \text{VERB:} \end{array} \right] \\ \text{index:} \left[ \text{prn:} \right] \\ \text{M-concept:} \end{array} \right] \right]$$

### 23.3.4 Schema of semantically interpreted LA-SU rule

rule:

syn:  $\langle ss\text{-pattern} \rangle$                        $\langle nw\text{-pattern} \rangle \implies \langle ss'\text{-pattern} \rangle$   
 sem:    semantic operations

input:

$$\begin{bmatrix} \text{sur:} \\ \text{syn: } \langle a \rangle \\ \text{sem: } b \end{bmatrix}_1 \dots \begin{bmatrix} \text{sur: } m \\ \text{syn: } \langle c \rangle \\ \text{sem: } d \end{bmatrix}_i + \begin{bmatrix} \text{sur: } n \\ \text{syn: } \langle e \rangle \\ \text{sem: } f \end{bmatrix}_{i+1}$$

output:

$$\begin{bmatrix} \text{sur:} \\ \text{syn: } \langle a \rangle \\ \text{sem: } b \end{bmatrix}_1 \dots \begin{bmatrix} \text{sur: } m+n \\ \text{syn: } \langle g \rangle \\ \text{sem: } h \end{bmatrix}_{i+1}$$



### 23.3.5 The six basic operations of the LA-SU semantics

1.  $\text{copy}_{ss}$ : include the woplets of the sentence start in the result.
2.  $\text{copy}_{nw}$ : include the woplet of the next word in the result.
3.  $n_1.x \text{ --- } \boxed{a} \rightarrow n_2.y$ : copy the values of the source feature  $x$  in  $n_1$  additively into the goal feature  $y$  in  $n_2$ , whereby  $n_1$  and  $n_2$  may be the woplets of the sentence start or the next word.
4.  $n_1.x \text{ --- } \boxed{e} \rightarrow n_2.y$ : copy the values of the source feature  $x$  in  $n_1$  exclusively into the goal feature  $y$  in  $n_2$ , whereby the value of  $y$  must be NIL (empty value).
5.  $n_1.x \text{ --- } \boxed{r} \rightarrow n_2.\textcircled{1}$ : substitute all occurrences of the variable  $\textcircled{1}$  in  $n_2$  simultaneously with the value of the source feature  $x$  in  $n_1$ .
6.  $n.x \text{ --- } \boxed{m} \rightarrow n.x$ : mark the first value of the source feature  $x$  in  $n$ , whereby the value of  $x$  must be a list.

### 23.3.6 Comparison of additive and exclusive copying

Additive:  $\text{nw.y} - \boxed{\text{a}} \rightarrow \text{ss.x}$   
 $\text{copy}_{ss}$

$$[x: a]_1 [x:]_2 + [y: b]_3 \Longrightarrow [\star x: a b]_1 [\star x: b]_2$$

Exclusive:  $\text{nw.y} - \boxed{\text{e}} \rightarrow \text{ss.x}$   
 $\text{copy}_{ss}$

$$[x: a]_1 [x:]_2 + [y: b]_3 \Longrightarrow [x: a]_1 [\star x: b]_2$$

## 23.4 Example of syntactic-semantic derivation (*LA-E4*)

23.4.1 The man gave Mary a flower because he loves her.

### 23.4.2 *LA-E4* for adverbial subclauses of English

LX = LX of *LA-E3* plus {(slowly (ADP) \*), (because (# ADP) \*)}

Variable definitions = those of *LA-E3* plus  $mn \in \{np \cup \{V, VI\}\}$

$ST_S =_{def} \{ [(x) \{ 1 \text{ DET+ADJ}, 2 \text{ DET+N}, 3 \text{ NOM+FV}, 4 \text{ AUX+MAIN}, 5 \text{ STRT-SBCL} \}] \}$

DET+ADJ:  $(n \ x) \ (\text{ADJ}) \Rightarrow (n \ x) \ \{ 6 \text{ DET+ADJ}, 7 \text{ DET+N} \}$

DET+N:  $(n \ x) \ (n) \Rightarrow (x) \ \{ 8 \text{ NOM+FV}, 9 \text{ FV+MAIN}, 10 \text{ AUX+NFV}, 11 \text{ ADD-ADP}, 12 \text{ IP} \}$

NOM+FV:  $(np \ \# \ x) \ (np' \ y \ V) \Rightarrow (y \ \# \ x)$   
 $(np) \ (np' \ x \ V) \Rightarrow (x \ V) \ \{ 13 \text{ FV+MAIN}, 14 \text{ AUX+NFV}, 15 \text{ ADD-ADP}, 16 \text{ IP} \}$

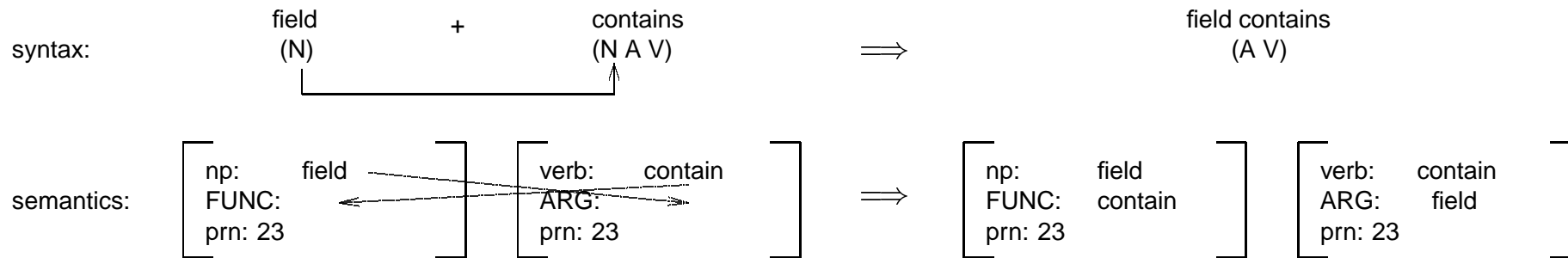
FV+MAIN:  $(np' \ \# \ x) \ (y \ np) \Rightarrow (y \ x)$   
 $(np' \ x \ \# \ y) \ (z \ np) \Rightarrow (z \ x \ \# \ y)$   
 $(np' \ x \ V) \ (y \ np) \Rightarrow (y \ x \ V) \ \{ 17 \text{ DET+ADJ}, 18 \text{ DET+N}, 19 \text{ FV+MAIN}, 20 \text{ IP} \}$

AUX+NFV:  $(aux \ \# \ x \ V) \ (aux) \Rightarrow (x \ V)$   
 $(aux \ \# \ x \ V) \ (y \ aux) \Rightarrow (y \ \# \ x \ V)$

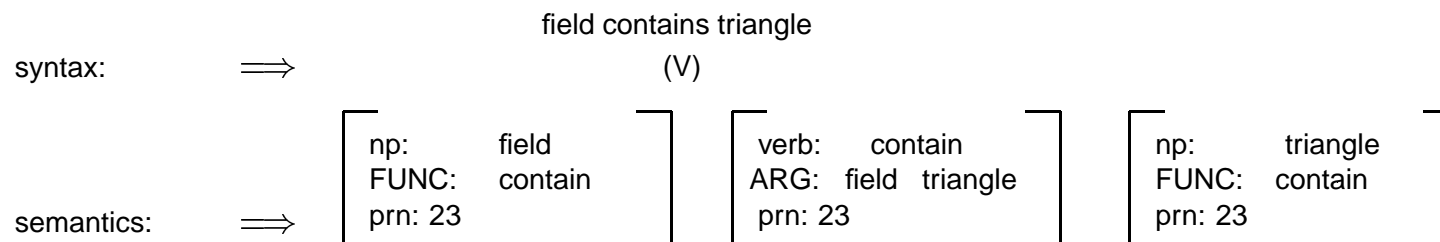
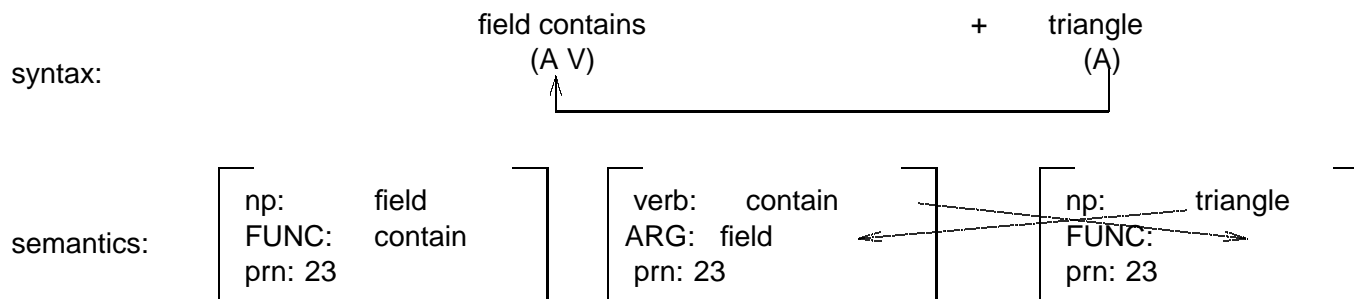


### 23.4.3 SYNTACTICO-SEMANTIC ANALYSIS OF field contains triangle

combination step 1:



combination step 2:

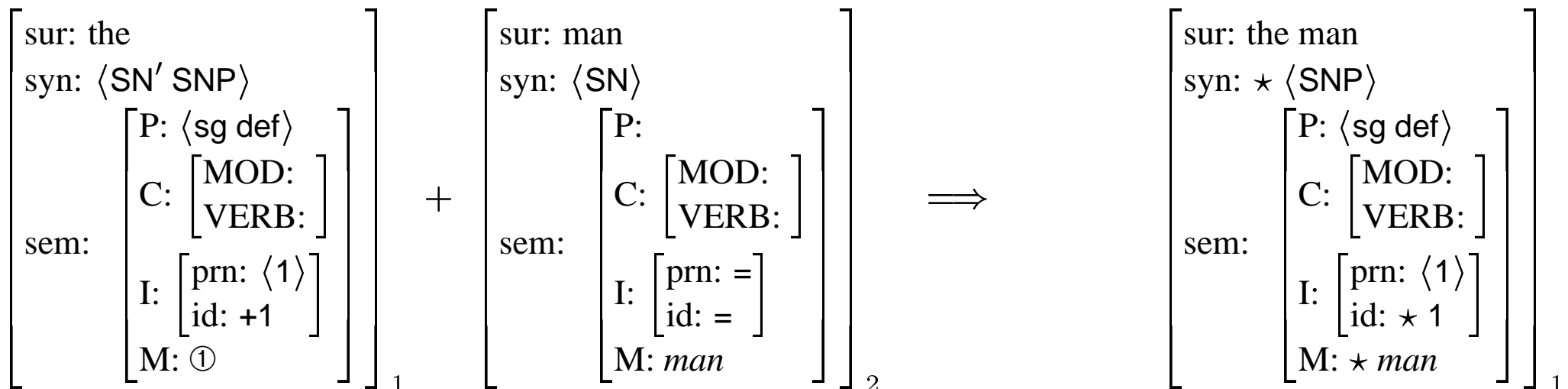


### 23.4.4 The man gave Mary a flower because he loves her.

#### 23.4.5 Applying DET+N to *the + man*

syn:  $\langle n \ x \rangle$  $\langle n \rangle$ 

sem:

 $\Rightarrow$   $\langle x \rangle$ nw.M  $\xrightarrow{\boxed{r}}$  ss.①copy<sub>ss</sub>

### 23.4.6 Applying NOM+FV to *the man* + *gave*

syn: ⟨np⟩

⟨np' x V⟩

⇒

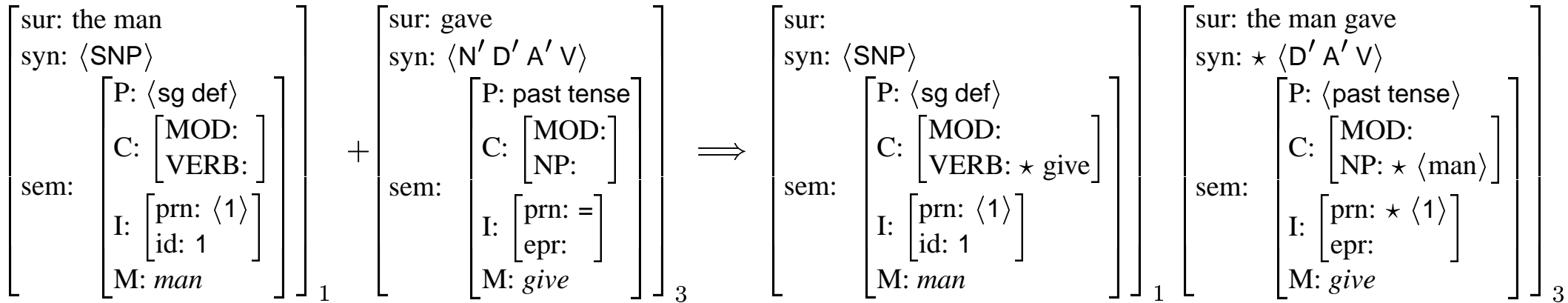
⟨x V⟩

sem:

nw.M — e → ss.VERB

ss.M — a → nw.NP

copy<sub>ss</sub> copy<sub>nw</sub>



### 23.4.7 Applying FV+MAIN to *the man gave + Mary*

syn:  $\langle np' \times V \rangle$  $\langle y \ np \rangle$  $\implies$  $\langle y \times V \rangle$ 

sem:

nw.M — a → ss.NPss.M — e → nw.VERBcopy<sub>ss</sub> copy<sub>nw</sub>

$$\left[ \begin{array}{l} \text{sur: the man gave} \\ \text{syn: } \star \langle D' A' V \rangle \\ \text{sem: } \left[ \begin{array}{l} \text{P: } \langle \text{past tense} \rangle \\ \text{C: } \left[ \begin{array}{l} \text{MOD:} \\ \text{NP: } \langle \text{man} \rangle \end{array} \right] \\ \text{I: } \left[ \begin{array}{l} \text{prn: } \langle 1 \rangle \\ \text{epr:} \end{array} \right] \\ \text{M: } \textit{give} \end{array} \right] \end{array} \right]_3$$

+

$$\left[ \begin{array}{l} \text{sur: Mary} \\ \text{syn: } \langle \text{SNP} \rangle \\ \text{sem: } \left[ \begin{array}{l} \text{P: } \langle \text{sg name} \rangle \\ \text{C: } \left[ \begin{array}{l} \text{MOD:} \\ \text{VERB:} \end{array} \right] \\ \text{I: } \left[ \begin{array}{l} \text{prn: } = \\ \text{id: } +1 \end{array} \right] \\ \text{M: } \textit{Mary} \end{array} \right] \end{array} \right]_4$$

$$\implies \left[ \begin{array}{l} \text{sur: the man gave Mary} \\ \text{syn: } \star \langle A' V \rangle \\ \text{sem: } \left[ \begin{array}{l} \text{P: } \langle \text{past tense} \rangle \\ \text{C: } \left[ \begin{array}{l} \text{MOD:} \\ \text{NP: } \langle \text{man}, \star \textit{Mary} \rangle \end{array} \right] \\ \text{I: } \left[ \begin{array}{l} \text{prn: } \langle 1 \rangle \\ \text{epr:} \end{array} \right] \\ \text{M: } \textit{give} \end{array} \right] \end{array} \right]_3 \left[ \begin{array}{l} \text{sur:} \\ \text{syn: } \langle \text{SNP} \rangle \\ \text{sem: } \left[ \begin{array}{l} \text{P: } \langle \text{sg name} \rangle \\ \text{C: } \left[ \begin{array}{l} \text{MOD:} \\ \text{VERB: } \star \textit{give} \end{array} \right] \\ \text{I: } \left[ \begin{array}{l} \text{prn: } \star \langle 1 \rangle \\ \text{id: } \star 2 \end{array} \right] \\ \text{M: } \textit{Mary} \end{array} \right] \end{array} \right]_4$$



### 23.4.8 Applying FV+MAIN to *the man gave Mary + a*

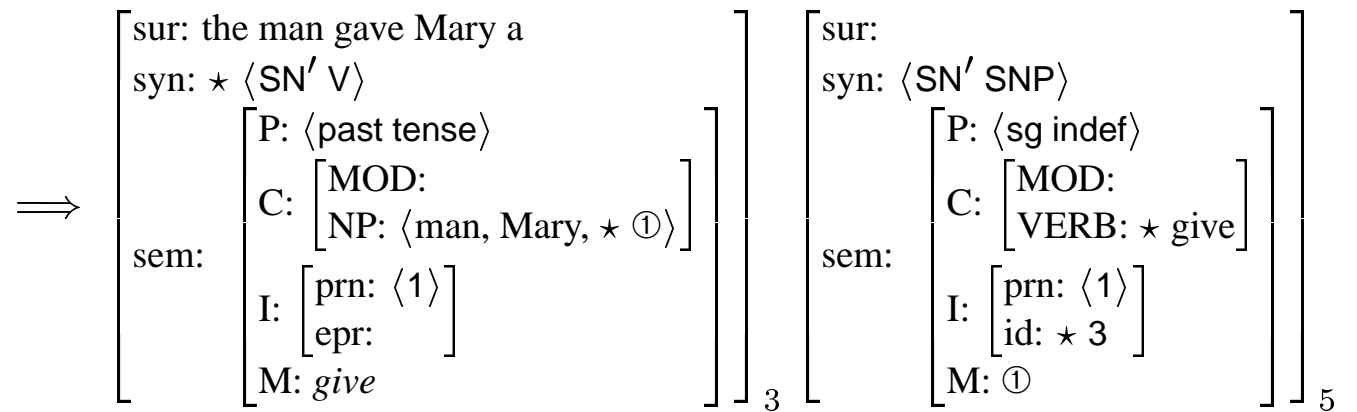
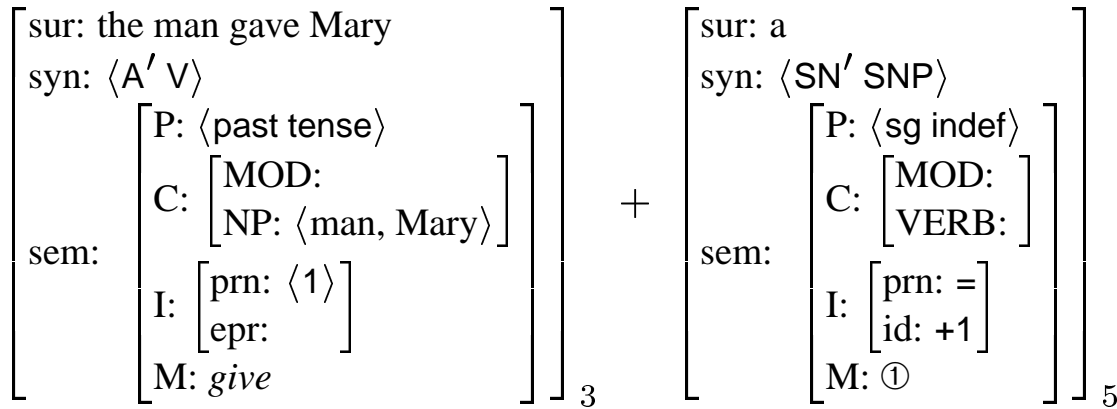
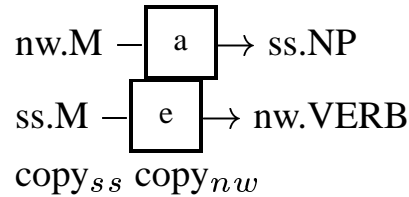
syn:  $\langle np' \times V \rangle$

$\langle y \ np \rangle$

$\implies$

$\langle y \ x \ V \rangle$

sem:



### 23.4.9 Applying DET+N to *The man gave Mary a + flower*

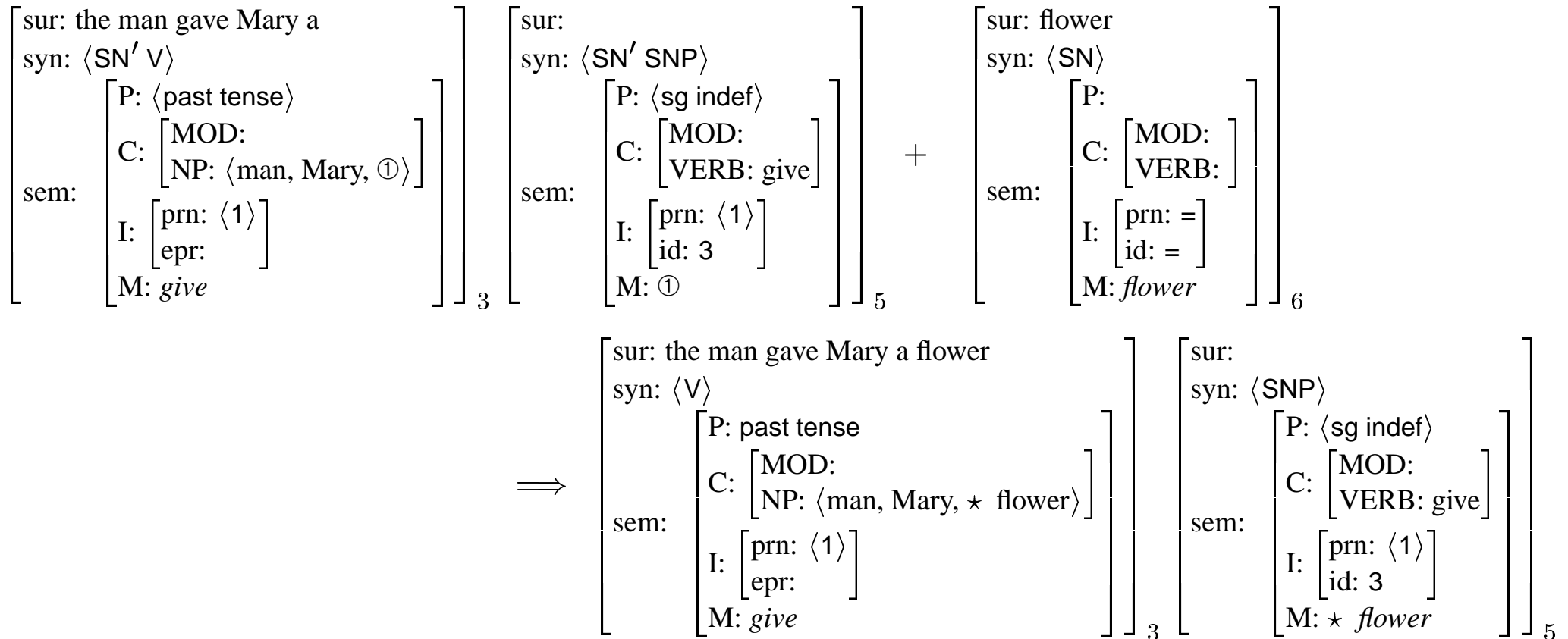
syn:  $\langle n \ x \rangle$ 

sem:

 $\langle n \rangle$  $\Rightarrow$  $\langle x \rangle$ 

$$\text{nw.M} \xrightarrow{\boxed{r}} \text{ss.}\textcircled{1}$$

copy<sub>ss</sub>



### 23.4.10 Applying ADD-ADP to *The man gave Mary a flower + because*

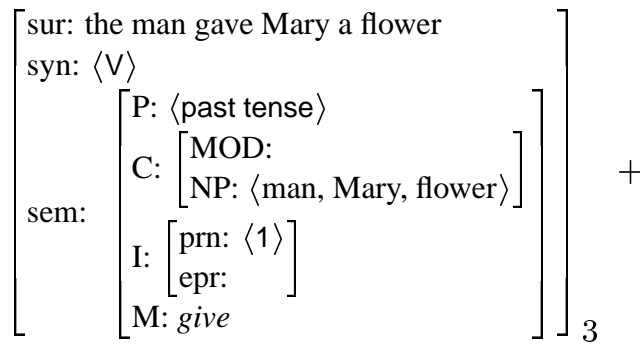
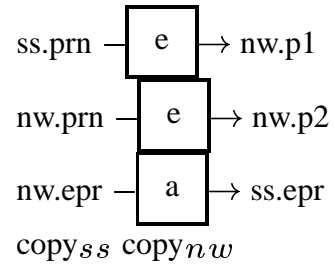
syn: <mn y>

<x ADP>

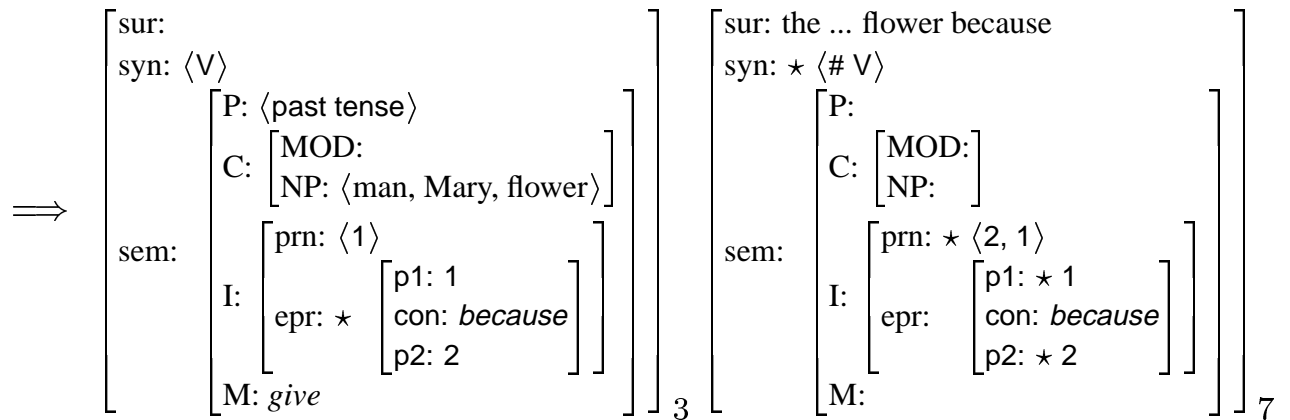
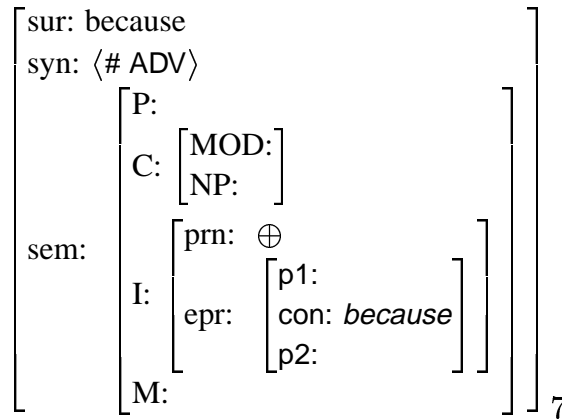
⇒

<x mn y>

sem:



+



### 23.4.11 The epr feature structure introduced by the The conjunction because

$$\left[ \begin{array}{l} \text{epr:} \\ \left[ \begin{array}{l} \text{p1:} \\ \text{con: } \textit{because} \\ \text{p2:} \end{array} \right] \end{array} \right]$$

### 23.4.12 Applying START-SUBCL to *The man gave Mary a flower because + he*

$$\begin{array}{l} \text{syn: } \langle \# \text{ x} \rangle \\ \text{sem:} \end{array} \quad \langle \text{y np} \rangle \quad \Longrightarrow \quad \langle \text{y np } \# \text{ x} \rangle$$

$\text{copy}_{ss} \text{ copy}_{nw}$

$$\begin{array}{l} \left[ \begin{array}{l} \text{sur: the man gave M. a f. because} \\ \text{syn: } \langle \# \text{ V} \rangle \\ \text{sem:} \\ \left[ \begin{array}{l} \text{P:} \\ \text{C:} \left[ \begin{array}{l} \text{MOD:} \\ \text{NP:} \end{array} \right] \\ \text{I:} \left[ \begin{array}{l} \text{prn: } \langle 2, 1 \rangle \\ \text{epr: } 1 \textit{ bec } 2 \end{array} \right] \\ \text{M:} \end{array} \right] \end{array} \right]_7 \end{array} + \begin{array}{l} \left[ \begin{array}{l} \text{sur: he} \\ \text{syn: } \langle \text{SNP} \rangle \\ \text{sem:} \\ \left[ \begin{array}{l} \text{P: } \langle \text{nom sg} \rangle \\ \text{C:} \left[ \begin{array}{l} \text{MOD:} \\ \text{VERB:} \end{array} \right] \\ \text{I:} \left[ \begin{array}{l} \text{prn: } = \\ \text{id: } +1 \end{array} \right] \\ \text{M: } \textit{pro-I} \end{array} \right] \end{array} \right]_8 \end{array} \end{array} \Longrightarrow \begin{array}{l} \left[ \begin{array}{l} \text{sur: the man gave M. a f. because he} \\ \text{syn: } \star \langle \text{SNP } \# \text{ V} \rangle \\ \text{sem:} \\ \left[ \begin{array}{l} \text{P:} \\ \text{C:} \left[ \begin{array}{l} \text{MOD:} \\ \text{NP:} \end{array} \right] \\ \text{I:} \left[ \begin{array}{l} \text{prn: } \langle 2, 1 \rangle \\ \text{epr: } 1 \textit{ bec } 2 \end{array} \right] \\ \text{M:} \end{array} \right] \end{array} \right]_7 \end{array} \left[ \begin{array}{l} \text{sur:} \\ \text{syn: } \langle \text{SNP} \rangle \\ \text{sem:} \\ \left[ \begin{array}{l} \text{P: } \langle \text{nom sg} \rangle \\ \text{C:} \left[ \begin{array}{l} \text{MOD:} \\ \text{VERB:} \end{array} \right] \\ \text{I:} \left[ \begin{array}{l} \text{prn: } \star \langle 2, 1 \rangle \\ \text{id: } \star 1 \end{array} \right] \\ \text{M: } \textit{pro-I} \end{array} \right] \end{array} \right]_8 \end{array}$$

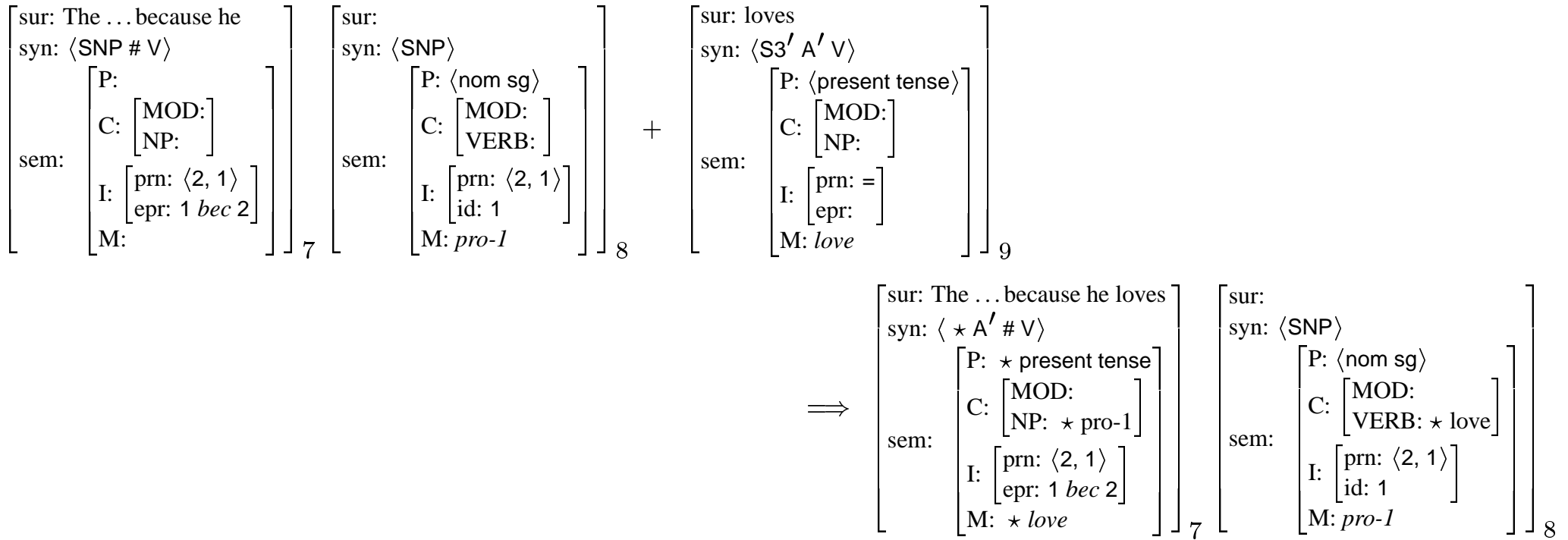
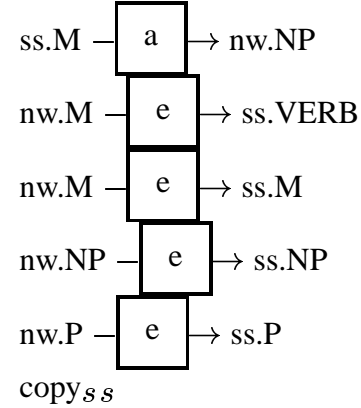
### 23.4.13 Application of NOM+FV to *The man g. M. a f. because he + loves*

syn:  $\langle np \# x \rangle$

$\langle np' y V \rangle$

$\Rightarrow \langle y \# x \rangle$

sem:



### 23.4.14 Application of FV+MAIN to *The m. g. M. a f. because he loves + her*

syn:  $\langle np' \# x \rangle$ 

sem:

 $\langle y \ np \rangle \implies \langle y \ x \rangle$ nw.np — a → ss.NPss.verb — e → nw.VERBss.prn — m → ss.prncopy<sub>ss</sub> copy<sub>nw</sub>

[	sur: The ... because he loves	]	+	[	sur: her	]	]	10
sem:	syn: $\langle A' \# V \rangle$	]		sem:	syn: $\langle SNP \rangle$	]		
	P: present tense	]			P: $\langle obl \ sg \rangle$	]		
	C: [	MOD:	]		C: [	MOD:	]	
	NP: pro-1	]			VERB:	]		
	I: [	prn: $\langle 2-, 1 \rangle$	]		I: [	prn: =	]	
	epr: 1 bec 2	]			id: +1	]		
	M: love	]			M: pro-2	]		
	]	7			]	10		

[	sur: The ... because he loves her	]	⇒	[	sur:	]	]	10
sem:	syn: $\langle \star V \rangle$	]		sem:	syn: $\langle SNP \rangle$	]		
	P: present tense	]			P: $\langle obl \ sg \rangle$	]		
	C: [	MOD:	]		C: [	MOD:	]	
	NP: pro-1 $\star$ pro-2	]			VERB: $\star$ love	]		
	I: [	prn: $\star \langle 2-, 1 \rangle$	]		I: [	prn: $\star \langle 2-, 1 \rangle$	]	
	epr: 1 bec 2	]			id: $\star$ 2	]		
	M: love	]			M: pro-2	]		
	]	7			]	10		

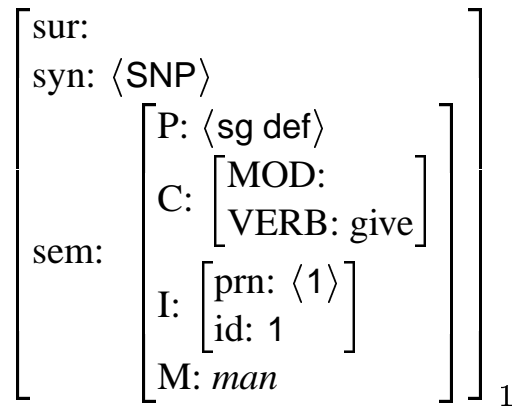
### 23.4.15 Proposition number of embedded subclause

the man,	gave her a flower.
prn: ⟨1⟩ because he loves Mary	prn: ⟨2-, 1⟩
prn: ⟨2, 1⟩	

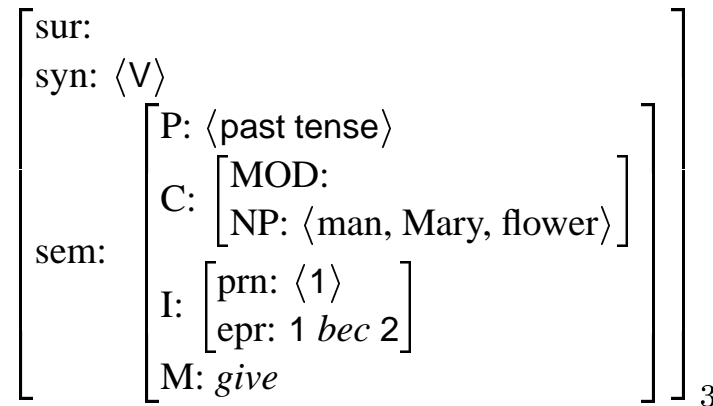
## 23.5 From SLIM semantics to SLIM pragmatics

### 23.5.1 SLIM semantic representation of example 23.4.1

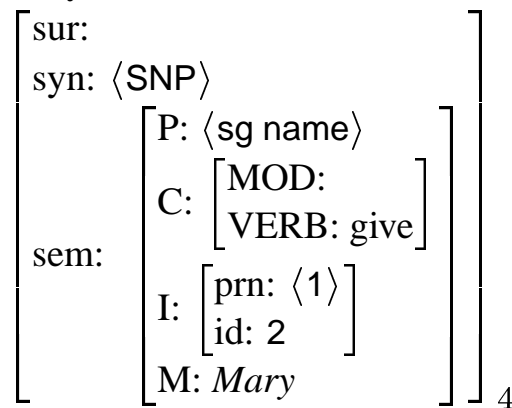
the man



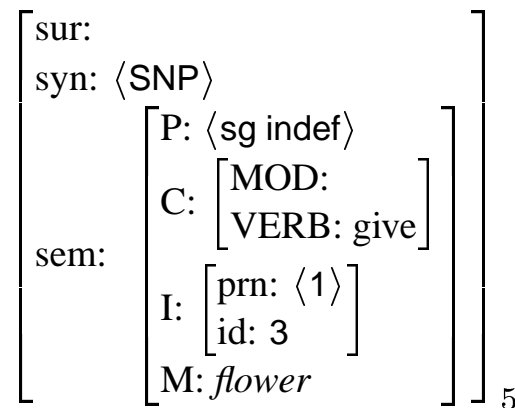
gave



Mary



a flower





because loves

$$\left[ \begin{array}{l} \text{sur:} \\ \text{syn: } \langle V \rangle \\ \\ \text{sem:} \left[ \begin{array}{l} \text{P: } \langle 3\text{sg, present tense} \rangle \\ \text{C: } \left[ \begin{array}{l} \text{MOD:} \\ \text{NP: } \text{pro-1, pro-2} \end{array} \right] \\ \text{I: } \left[ \begin{array}{l} \text{prn: } \langle 2-, 1 \rangle \\ \text{epr: } 1 \text{ bec } 2 \end{array} \right] \\ \text{M: } \textit{love} \end{array} \right] \end{array} \right]_7$$

he

$$\left[ \begin{array}{l} \text{sur:} \\ \text{syn: } \langle \text{SNP} \rangle \\ \\ \text{sem:} \left[ \begin{array}{l} \text{P: } \langle \text{nom sg} \rangle \\ \text{C: } \left[ \begin{array}{l} \text{MOD:} \\ \text{VERB: } \textit{love} \end{array} \right] \\ \text{I: } \left[ \begin{array}{l} \text{prn: } \langle 2-, 1 \rangle \\ \text{id: } 1 \end{array} \right] \\ \text{M: } \textit{pro-1} \end{array} \right] \end{array} \right]_8$$

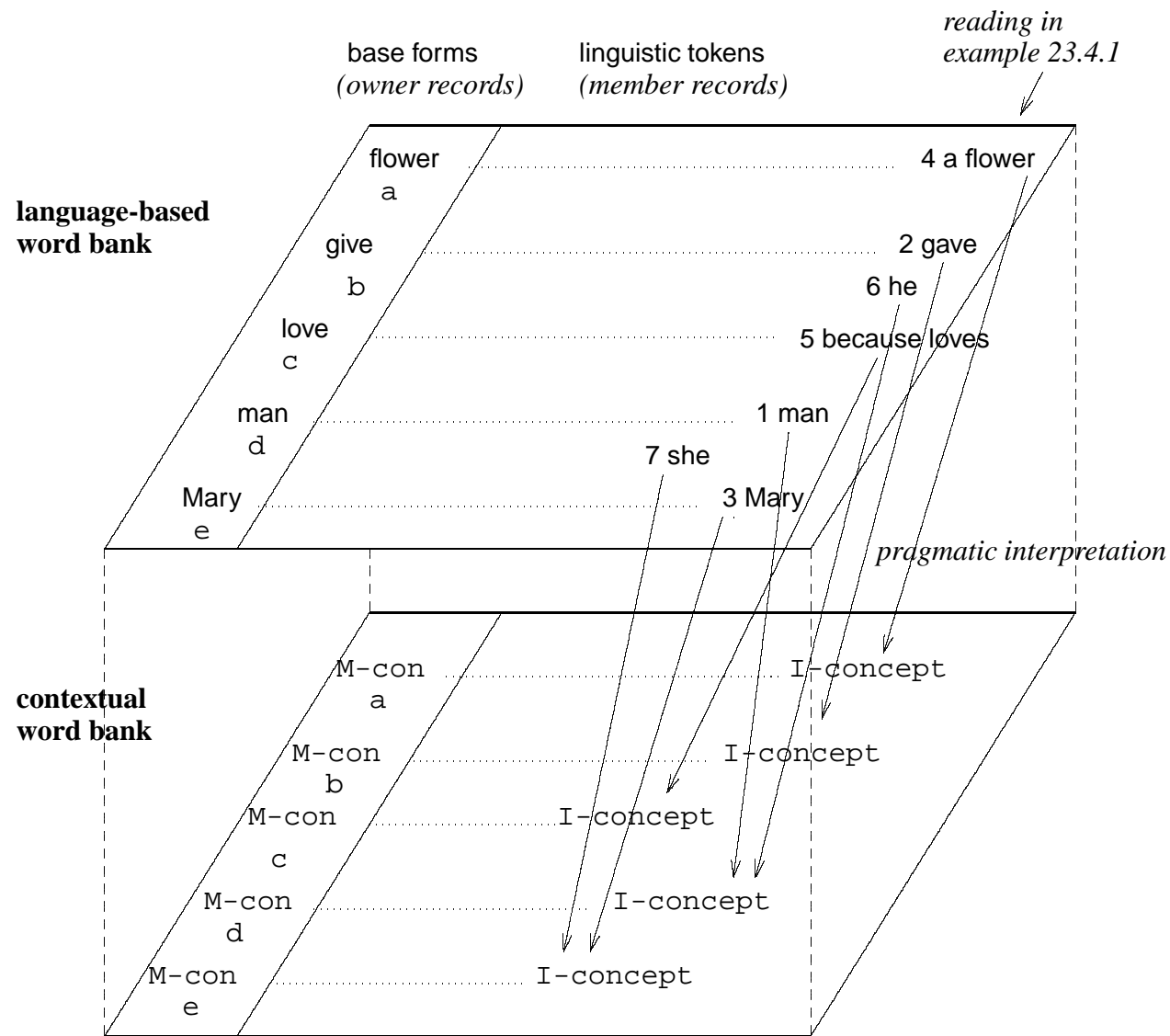
her

$$\left[ \begin{array}{l} \text{sur:} \\ \text{syn: } \langle \text{SNP} \rangle \\ \\ \text{sem:} \left[ \begin{array}{l} \text{P: } \langle \text{obl sg} \rangle \\ \text{C: } \left[ \begin{array}{l} \text{MOD:} \\ \text{VERB: } \textit{love} \end{array} \right] \\ \text{I: } \left[ \begin{array}{l} \text{prn: } \langle 2-, 1 \rangle \\ \text{id: } 2 \end{array} \right] \\ \text{M: } \textit{pro-2} \end{array} \right] \end{array} \right]_9$$

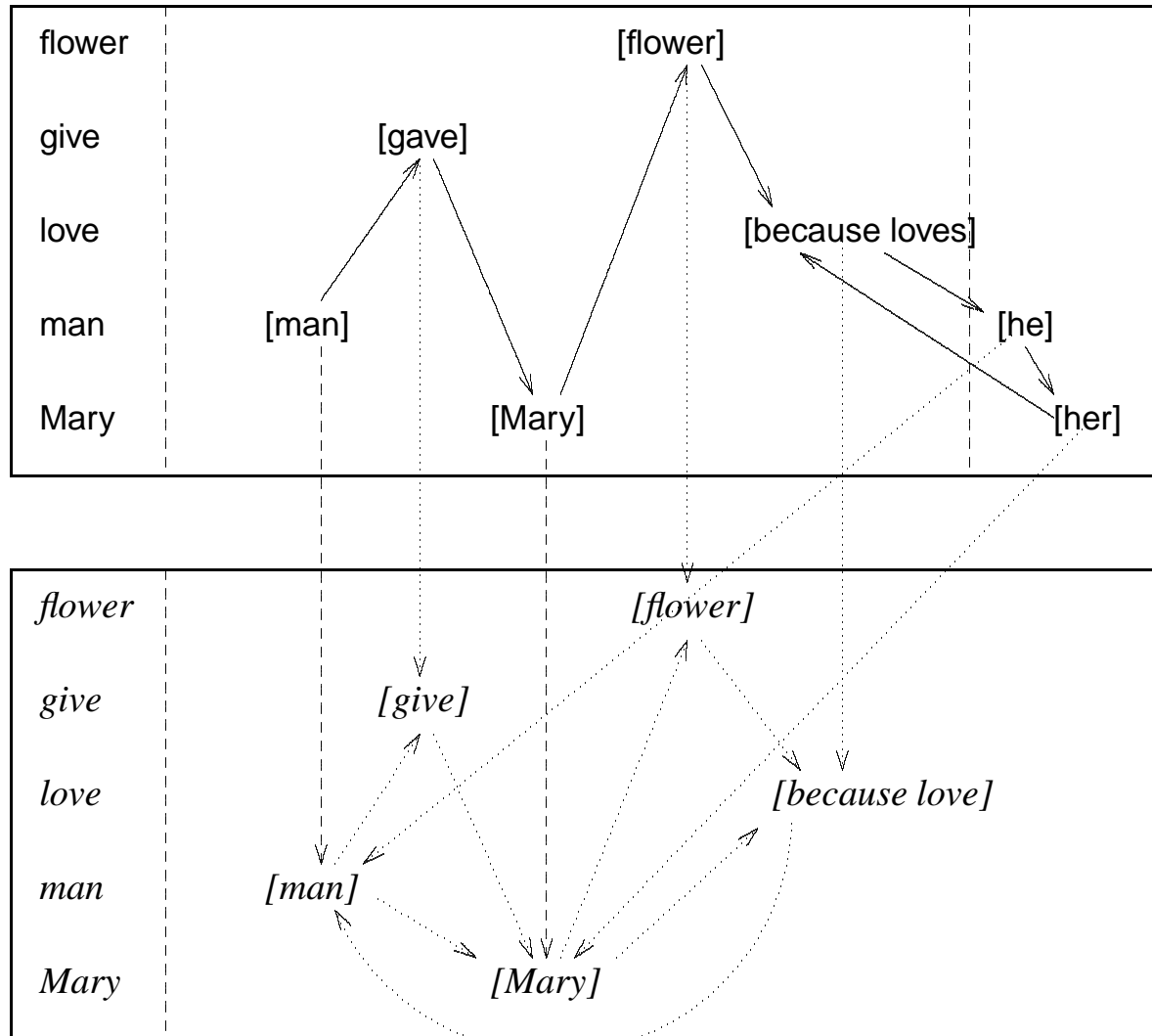
### 23.5.2 Components of meaning<sub>1</sub>

- Compositional semantics (sentence semantics)
  1. Decomposition of input into elementary propositions.
  2. Functor-argument structure within an elementary proposition.
  3. Extrapositional relations among elementary propositions.
- Lexical semantics (word semantics)
  1. Properties and M-concepts of woplets.
  2. Extrapositional relations between word types by means of *absolute propositions*.

### 23.5.3 Embedding 23.5.1 into the contextual word bank



### 23.5.4 Contextual reconstruction of language information



## **24. SLIM machine in the speaker mode**

### **24.1 Subcontext as concatenated propositions**

#### **24.1.1 Immediate vs. mediated subcontexts**

In immediate subcontexts, the coherence of the content follows directly from the coherence of the external world which they reflect, i.e., the temporal and spatial sequence of events, the part-whole relations of objects, etc. In contrast, mediated subcontexts have the special property that the elements familiar from direct recognition may be reordered and reconnected by the author at will.

#### **24.1.2 Comparing coherence and incoherence, Example I**

The representation of a swimmer standing at the pool side, diving into the water, and disappearing with a splash is coherent. In contrast, a representation in which a pair of feet appears in the foaming water and a swimmer flies feet first into the air landing on the pool side, would be incoherent – unless it is specified in addition that the representation happens to be, e.g., a backward running movie.

#### **24.1.3 Comparing coherence and incoherence, Example II**

A representation of people talking with each other would be coherent. In contrast, a similar representation of a deer conversing with a skunk in English would be incoherent – unless it is specified in addition that the representation happens to be fictional.

## 24.1.4 Mediated subcontexts reflecting the coherence of the external world

world → speaker context → language → hearer context → world

## 24.1.5 A sequence of propositions forming a subcontext

1. Peter leaves the house. 2. Peter crosses the street. 3. Peter enters a restaurant. 4. Peter orders a salad.
5. Peter eats the salad. 6. Peter pays the salad. 7. Peter leaves the restaurant. 8. Peter crosses the street.
9. Peter enters the house.

## 24.1.6 Equivalent representation of 24.1.1 as a word bank

CONCEPT TYPES:

COPLETS:

$\left[ \begin{array}{l} \text{M-concept: cross} \\ \text{role: T-verb} \end{array} \right]$	$\left[ \begin{array}{l} \text{I-concept}_{loc}: \textit{cross} \\ \text{P: indicative} \\ \text{C: } \left[ \begin{array}{l} \text{MOD:} \\ \text{NP: Peter, street} \end{array} \right] \\ \text{I: } \left[ \begin{array}{l} \text{prn: 2} \\ \text{epr: } \left[ \begin{array}{l} 2 \text{ then } 3 \\ 1 \text{ then } 2 \end{array} \right] \end{array} \right] \end{array} \right]$	$\left[ \begin{array}{l} \text{I-concept}_{loc}: \textit{cross} \\ \text{P: indicative} \\ \text{C: } \left[ \begin{array}{l} \text{MOD:} \\ \text{NP: Peter, street} \end{array} \right] \\ \text{I: } \left[ \begin{array}{l} \text{prn: 8} \\ \text{epr: } \left[ \begin{array}{l} 8 \text{ then } 9 \\ 7 \text{ then } 8 \end{array} \right] \end{array} \right] \end{array} \right]$
--	---	---

$$\left[ \begin{array}{l} \text{M-concept: eat} \\ \text{role: T-verb} \end{array} \right] \left[ \begin{array}{l} \text{I-concept}_{loc}: \textit{eat} \\ \text{P: indicative} \\ \text{C: } \left[ \begin{array}{l} \text{MOD:} \\ \text{NP: Peter, salad} \end{array} \right] \\ \text{I: } \left[ \begin{array}{l} \text{prn: 5} \\ \text{epr: } \left[ \begin{array}{l} 5 \text{ then } 6 \\ 4 \text{ then } 5 \end{array} \right] \end{array} \right] \end{array} \right]$$

$$\left[ \begin{array}{l} \text{M-concept: enter} \\ \text{role: T-verb} \end{array} \right] \left[ \begin{array}{l} \text{I-concept}_{loc}: \textit{enter} \\ \text{P: indicative} \\ \text{C: } \left[ \begin{array}{l} \text{MOD:} \\ \text{NP: Peter, restaurant} \end{array} \right] \\ \text{I: } \left[ \begin{array}{l} \text{prn: 3} \\ \text{epr: } \left[ \begin{array}{l} 3 \text{ then } 4 \\ 2 \text{ then } 3 \end{array} \right] \end{array} \right] \end{array} \right] \left[ \begin{array}{l} \text{I-concept}_{loc}: \textit{enter} \\ \text{P: indicative} \\ \text{C: } \left[ \begin{array}{l} \text{MOD:} \\ \text{NP: Peter, house} \end{array} \right] \\ \text{I: } \left[ \begin{array}{l} \text{prn: 9} \\ \text{epr: } \left[ \begin{array}{l} 8 \text{ then } 9 \end{array} \right] \end{array} \right] \end{array} \right]$$

$$\left[ \begin{array}{l} \text{M-concept: house} \\ \text{role: noun} \end{array} \right] \left[ \begin{array}{l} \text{I-concept}_{loc}: \textit{house} \\ \text{P: A sg def} \\ \text{C: } \left[ \begin{array}{l} \text{MOD:} \\ \text{VERB: leave} \end{array} \right] \\ \text{I: } \left[ \begin{array}{l} \text{prn: 1} \\ \text{id: 2} \end{array} \right] \end{array} \right] \left[ \begin{array}{l} \text{I-concept}_{loc}: \textit{house} \\ \text{P: A sg def} \\ \text{C: } \left[ \begin{array}{l} \text{MOD:} \\ \text{VERB: enter} \end{array} \right] \\ \text{I: } \left[ \begin{array}{l} \text{prn: 9} \\ \text{id: 2} \end{array} \right] \end{array} \right]$$

[M-concept: leave role: T-verb]	[I-concept <sub>loc</sub> : leave P: indicative C: [MOD: NP: Peter, house] I: [prn: 1 epr: [1 then 2]]]	[I-concept <sub>loc</sub> : leave P: indicative C: [MOD: NP: Peter, restaurant] I: [prn: 7 epr: [7 then 8 6 then 7]]]
------------------------------------	--	---

[M-concept: order role: T-verb]	[I-concept <sub>loc</sub> : order P: indicative C: [MOD: NP: Peter, salad] I: [prn: 4 epr: [4 then 5 3 then 4]]]
------------------------------------	--

[M-concept: pay role: T-verb]	[I-concept <sub>loc</sub> : pay P: indicative C: [MOD: NP: Peter, salad] I: [prn: 6 epr: [6 then 7 5 then 6]]]
----------------------------------	--

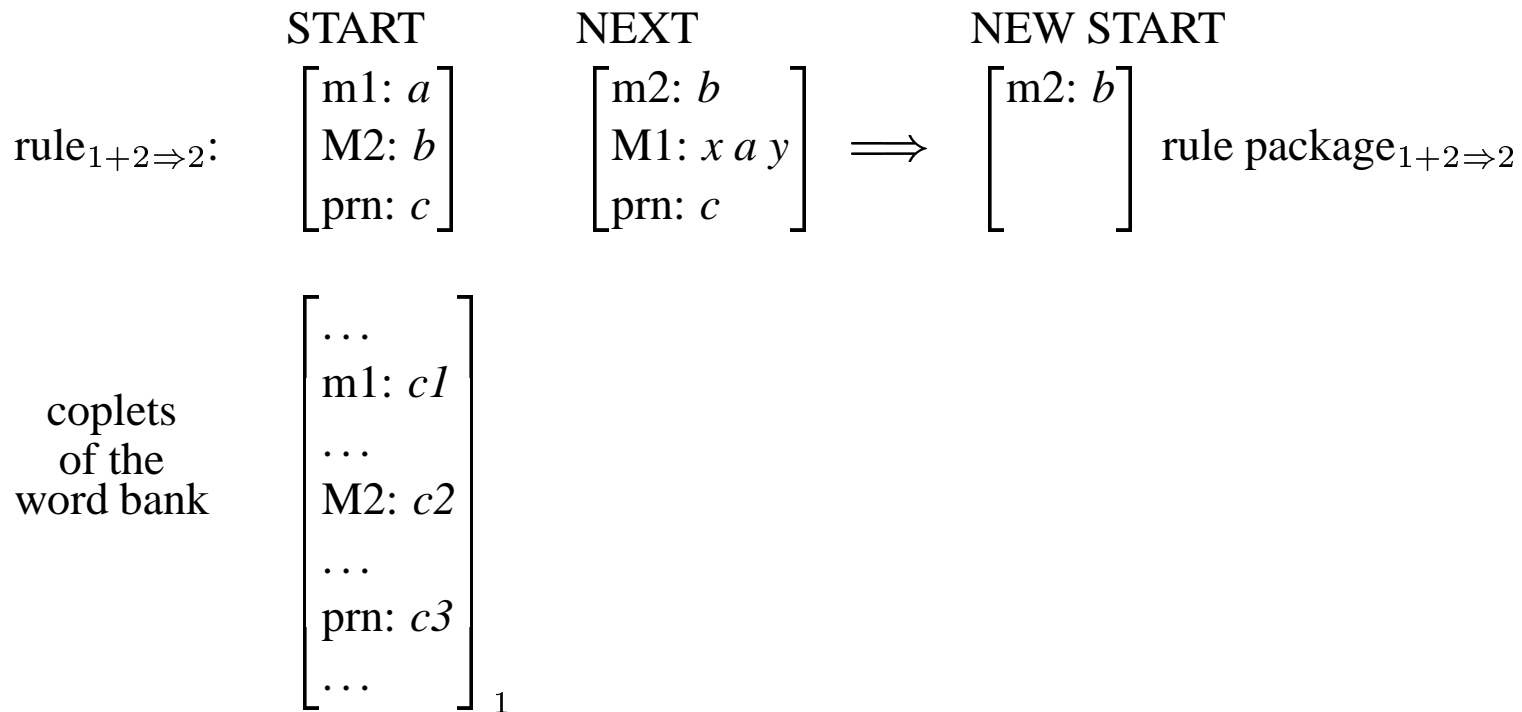
[M-concept: Peter role: name]	[I-concept <sub>loc</sub> : Peter P: Nom C: [MOD: VERB: leave] I: [prn: 1 id: 1]	[I-concept <sub>loc</sub> : Peter P: Nom C: [MD: VB: cross] I: [prn: 2 id: 1]	[I-concept <sub>loc</sub> : Peter P: Nom C: [MD: VB: enter] I: [prn: 3 id: 1]
	[I-concept <sub>loc</sub> : Peter P: Nom C: [MOD: VERB: order] I: [prn: 4 id: 1]	[I-concept <sub>loc</sub> : Peter P: Nom C: [MOD: VERB: eat] I: [prn: 5 id: 1]	[I-concept <sub>loc</sub> : Peter P: Nom C: [MOD: VERB: pay] I: [prn: 6 id: 1]
	[I-concept <sub>loc</sub> : Peter P: Nom C: [MOD: VERB: leave] I: [prn: 7 id: 1]	[I-concept <sub>loc</sub> : Peter P: Nom C: [MD: VB: cross] I: [prn: 8 id: 1]	[I-concept <sub>loc</sub> : Peter P: Nom C: [MD: VB: enter] I: [prn: 9 id: 1]



[M-concept: restaurant] role: noun	[I-concept <sub>loc</sub> : restaurant] P: A sg indef C: [MOD: VERB: enter] I: [prn: 3 id: 4]	[I-concept <sub>loc</sub> : restaurant] P: A sg def C: [MOD: VERB: leave] I: [prn: 7 id: 4]	
[M-concept: salad] role: noun	[I-concept <sub>loc</sub> : salad] P: A sg indef C: [MOD: VERB: order] I: [prn: 4 id: 5]	[I-concept <sub>loc</sub> : salad] P: A sg def C: [MOD: VERB: eat] I: [prn: 5 id: 5]	[I-concept <sub>loc</sub> : salad] P: A sg def C: [MOD: VERB: pay] I: [prn: 6 id: 5]
[M-concept: street] role: noun	[I-concept <sub>loc</sub> : street] P: A sg def C: [MOD: VERB: cross] I: [prn: 2 id: 3]	[I-concept <sub>loc</sub> : street] P: A sg def C: [MOD: VERB: cross] I: [prn: 8 id: 3]	

## 24.2 Tracking principles of LA-navigation

### 24.2.1 Step 1 of a LA-NA rule application



### 24.2.2 Step 2 of an LA-NA rule application

$$\begin{array}{ccc}
 \text{rule}_{1+2 \Rightarrow 2}: & \begin{array}{c} \text{START} \\ \left[ \begin{array}{l} \text{m1: } a \\ \text{M2: } b \\ \text{prn: } c \end{array} \right] \end{array} & \begin{array}{c} \text{NEXT} \\ \left[ \begin{array}{l} \text{m2: } b \\ \text{M1: } x a y \\ \text{prn: } c \end{array} \right] \end{array} \Rightarrow \begin{array}{c} \text{NEW START} \\ \left[ \begin{array}{l} \text{m2: } b \end{array} \right] \end{array} \text{rule package}_{1+2 \Rightarrow 2} \\
 \\
 \text{coplets} & \begin{array}{c} \left[ \begin{array}{l} \dots \\ \text{m1: } c1 \\ \dots \\ \text{M2: } c2 \\ \dots \\ \text{prn: } c3 \\ \dots \end{array} \right]_1 & + & \begin{array}{c} \left[ \begin{array}{l} \dots \\ \text{m2: } c2 \\ \dots \\ \text{M1: } ..c1.. \\ \dots \\ \text{prn: } c3 \\ \dots \end{array} \right]_2
 \end{array}
 \end{array}
 \end{array}$$

### 24.2.3 Step 3 of a LA-NA rule application

$$\begin{array}{ccc}
 \text{rule}_{1+2 \Rightarrow 2}: & \begin{array}{c} \text{START} \\ \left[ \begin{array}{l} \text{m1: } a \\ \text{M2: } b \\ \text{prn: } c \end{array} \right] \\ \end{array} & \begin{array}{c} \text{NEXT} \\ \left[ \begin{array}{l} \text{m2: } b \\ \text{M1: } x a y \\ \text{prn: } c \end{array} \right] \\ \end{array} \Rightarrow \begin{array}{c} \text{NEW START} \\ \left[ \begin{array}{l} \text{m2: } b \\ \end{array} \right] \\ \end{array} \text{rule package}_{1+2 \Rightarrow 2}
 \end{array}$$
  

$$\begin{array}{ccc}
 \text{coplets} & & \\
 \text{of the} & & \\
 \text{word bank} & & \\
 \left[ \begin{array}{l} \dots \\ \text{m1: } c1 \\ \dots \\ \text{M2: } c2 \\ \dots \\ \text{prn: } c3 \\ \dots \end{array} \right]_1 & + & \left[ \begin{array}{l} \dots \\ \text{m2: } c2 \\ \dots \\ \text{M1: } ..c1.. \\ \dots \\ \text{prn: } c3 \\ \dots \end{array} \right]_2 \Rightarrow \left[ \begin{array}{l} \dots \\ \text{m2: } c2 \\ \dots \\ \text{M1: } ..c1.. \\ \dots \\ \text{prn: } c3 \\ \dots \end{array} \right]_2
 \end{array}$$

## 24.2.4 Tracking principles of LA-navigation

### 1. *Completeness*

Within an elementary proposition those coplets are preferred which have not yet been traversed during the current navigation.

### 2. *Uniqueness*

If several START or NEXT coplets are available, no more than one of each are selected whereby the choice may be at random or – if activated – based on a specific navigation pattern.

### 3. *Recency*

In extrapositional navigations, propositions which have been least recently traversed are preferred.

### 4. *Frequency*

When entering a new subcontext, the navigation prefers paths most frequently traversed in previous navigations.

### 24.2.5 Definition of universal LA-NA syntax

$ST_S: \{([M-np: a] \{1 V+NP1, 2 V+NP2\})\}$

V+NP1:  $\begin{bmatrix} M\text{-verb: } a \\ NP: x b y \\ prn: m \end{bmatrix} \begin{bmatrix} M\text{-np: } b \\ VERB: a \\ prn: m \end{bmatrix} \Rightarrow \begin{bmatrix} M\text{-verb: } a \end{bmatrix} \{3 V+NP1, 4 V+NP2, 5 V+epr\}$

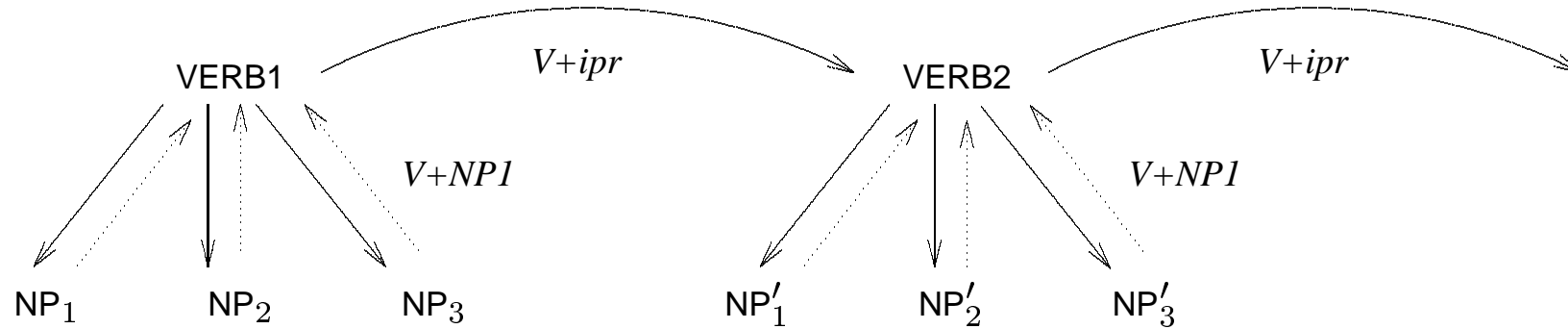
V+NP2:  $\begin{bmatrix} M\text{-verb: } a \\ NP: x b y \\ prn: m \end{bmatrix} \begin{bmatrix} M\text{-np: } b \\ VERB: a \\ prn: m \end{bmatrix} \Rightarrow \begin{bmatrix} M\text{-np: } b \end{bmatrix} \{6 NP+id\}$

V+epr:  $\begin{bmatrix} M\text{-verb: } a \\ NP: x \\ prn: m \\ epr: m C n \end{bmatrix} \begin{bmatrix} M\text{-verb: } b \\ NP: y \\ prn: n \\ epr: m C n \end{bmatrix} \Rightarrow \begin{bmatrix} M\text{-verb: } b \end{bmatrix} \{7 V+NP1, 8 V+NP2\}$

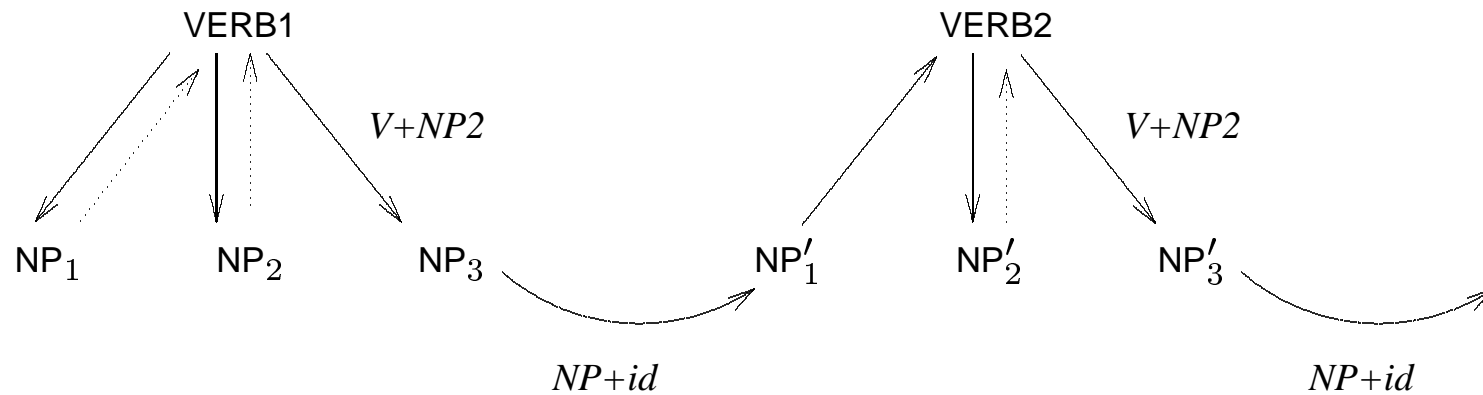
NP+id:  $\begin{bmatrix} M\text{-np: } a \\ VERB: b \\ prn: k \\ id: m \end{bmatrix} \begin{bmatrix} M\text{-np: } a \\ VERB: c \\ prn: l \\ id: m \end{bmatrix} \Rightarrow \begin{bmatrix} M\text{-verb: } c \\ NP: x a y \\ prn: l \end{bmatrix} \{9 V+NP1, 10 V+NP2\}$

$ST_F: \{([M\text{-verb: } x] rp V+NP1)\}$

## 24.2.6 Extrapositional epr-navigation



## 24.2.7 Extrapositional id-navigation



### 24.2.8 First Application of V+NP1 in the word bank 24.1.2

$$\text{V+NP1: } \begin{bmatrix} \text{M-verb: } a \\ \text{NP: } x \ b \ y \\ \text{prn: } c \end{bmatrix} \quad \begin{bmatrix} \text{M-np: } b \\ \text{VERB: } a \\ \text{prn: } c \end{bmatrix} \implies \begin{bmatrix} \text{M-verb: } a \end{bmatrix} \{ 3 \text{ V+NP1, } 4 \text{ V+NP2, } 5 \text{ V+epr} \}$$

$$\begin{bmatrix} \text{I-concept}_{loc}: \textit{eat} \\ \text{P: indicative} \\ \text{C: } \begin{bmatrix} \text{MOD:} \\ \text{NP: Peter, salad} \end{bmatrix} \\ \text{I: } \begin{bmatrix} \text{prn: } 5 \\ \text{epr: } \begin{bmatrix} 5 \text{ then } 6 \\ 4 \text{ then } 5 \end{bmatrix} \end{bmatrix} \end{bmatrix} \quad \begin{bmatrix} \text{I-concept}_{loc}: \textit{salad} \\ \text{P: A sg def} \\ \text{C: } \begin{bmatrix} \text{MOD:} \\ \text{VERB: eat} \end{bmatrix} \\ \text{I: } \begin{bmatrix} \text{prn: } 5 \\ \text{id: } 2 \end{bmatrix} \end{bmatrix} \quad \begin{bmatrix} \text{I-concept}_{loc}: \textit{eat} \\ \text{P: indicative} \\ \text{C: } \begin{bmatrix} \text{MOD:} \\ \text{NP: Peter, salad} \end{bmatrix} \\ \text{I: } \begin{bmatrix} \text{prn: } 5 \\ \text{epr: } \begin{bmatrix} 5 \text{ then } 6 \\ 4 \text{ then } 5 \end{bmatrix} \end{bmatrix} \end{bmatrix}$$



### 24.2.9 Second application of V+NP1 in the word bank 24.1.2

$$\text{V+NP1: } \begin{bmatrix} \text{M-verb: } a \\ \text{NP: } x \ b \ y \\ \text{prn: } c \end{bmatrix} \quad \begin{bmatrix} \text{np: } b \\ \text{VERB: } a \\ \text{prn: } c \end{bmatrix} \implies \begin{bmatrix} \text{M-verb: } a \end{bmatrix} \{ 3 \text{ V+NP1, } 4 \text{ V+NP2, } 5 \text{ V+epr} \}$$

$$\begin{bmatrix} \text{I-concept}_{loc}: \textit{eat} \\ \text{P: indicative} \\ \text{C: } \begin{bmatrix} \text{MOD:} \\ \text{NP: Peter, salad} \end{bmatrix} \\ \text{I: } \begin{bmatrix} \text{prn: } 5 \\ \text{epr: } \begin{bmatrix} 5 \text{ then } 6 \\ 4 \text{ then } 5 \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

$$\begin{bmatrix} \text{I-concept}_{loc}: \textit{Peter} \\ \text{P: Nom} \\ \text{C: } \begin{bmatrix} \text{MOD:} \\ \text{VERB: eat} \end{bmatrix} \\ \text{I: } \begin{bmatrix} \text{prn: } 5 \\ \text{id: } 1 \end{bmatrix} \end{bmatrix}$$

$$\begin{bmatrix} \text{I-concept}_{loc}: \textit{eat} \\ \text{P: indicative} \\ \text{C: } \begin{bmatrix} \text{MOD:} \\ \text{NP: Peter, salad} \end{bmatrix} \\ \text{I: } \begin{bmatrix} \text{prn: } 5 \\ \text{epr: } \begin{bmatrix} 5 \text{ then } 6 \\ 4 \text{ then } 5 \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

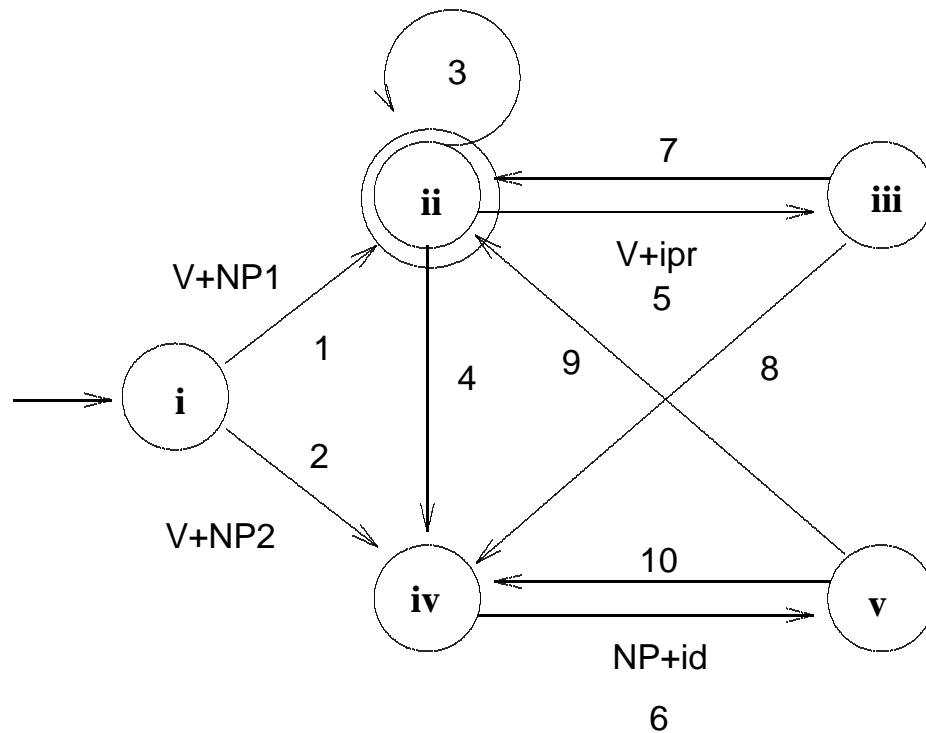
### 24.2.10 Application of V+epr in the word bank 24.1.2

$$\text{V+epr: } \left[ \begin{array}{l} \text{M-verb: } a \\ \text{NP: } x \\ \text{prn: } m \\ \text{epr: } m \ C \ n \end{array} \right] \quad \left[ \begin{array}{l} \text{M-verb: } b \\ \text{NP: } y \\ \text{prn: } n \\ \text{epr: } m \ C \ n \end{array} \right] \Rightarrow \left[ \begin{array}{l} \text{M-verb: } b \\ \\ \\ \end{array} \right] \{7 \text{ V+NP1, } 8 \text{ V+NP2}\}$$

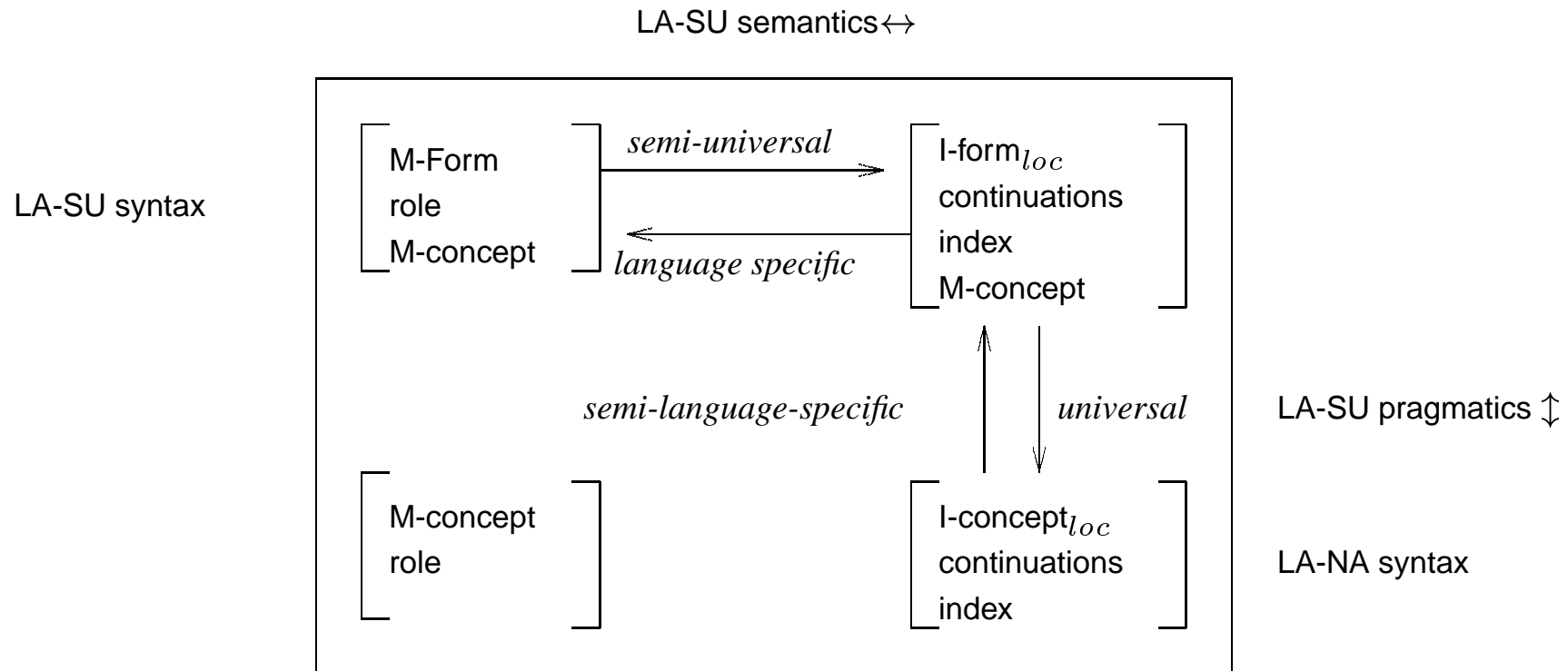
$$\left[ \begin{array}{l} \text{I-concept}_{loc}: \textit{eat} \\ \text{P: indicative} \\ \text{C: } \left[ \begin{array}{l} \text{MOD:} \\ \text{NP: Peter, salad} \end{array} \right] \\ \text{I: } \left[ \begin{array}{l} \text{prn: } 5 \\ \text{epr: } \left[ \begin{array}{l} 5 \text{ then } 6 \\ 4 \text{ then } 5 \end{array} \right] \end{array} \right] \end{array} \right] \quad \left[ \begin{array}{l} \text{I-concept}_{loc}: \textit{pay} \\ \text{P: indicative} \\ \text{C: } \left[ \begin{array}{l} \text{MOD:} \\ \text{NP: Peter, salad} \end{array} \right] \\ \text{I: } \left[ \begin{array}{l} \text{prn: } 6 \\ \text{epr: } \left[ \begin{array}{l} 6 \text{ then } 7 \\ 5 \text{ then } 6 \end{array} \right] \end{array} \right] \end{array} \right] \quad \left[ \begin{array}{l} \text{I-concept}_{loc}: \textit{pay} \\ \text{P: indicative} \\ \text{C: } \left[ \begin{array}{l} \text{MOD:} \\ \text{NP: Peter, salad} \end{array} \right] \\ \text{I: } \left[ \begin{array}{l} \text{prn: } 6 \\ \text{epr: } \left[ \begin{array}{l} 6 \text{ then } 7 \\ 5 \text{ then } 6 \end{array} \right] \end{array} \right] \end{array} \right]$$

## 24.3 Interpreting autonomous LA-navigation with language

### 24.3.1 The finite state back bone of LA-NA



### 24.3.2 Universality and language specificity in a SLIM machine



### 24.3.3 Realization principles of the basic word orders

*VSO languages*

V+NP  
 realization  
 ↑  
 buffer: [Verb] + [NP<sub>1</sub>]  $\implies$  [Verb] [NP<sub>1</sub>]

V+NP  
 realization  
 ↑  
 buffer: \*[Verb] [NP<sub>1</sub>] + [NP<sub>2</sub>]  $\implies$  \*[Verb] [NP<sub>1</sub>] [NP<sub>2</sub>]

V+NP  
 realization  
 1↑ 2↑  
 buffer: \*[Verb] \*[NP<sub>1</sub>] [NP<sub>2</sub>] + [NP<sub>3</sub>]  $\implies$  \*[Verb] \*[NP<sub>1</sub>] [NP<sub>2</sub>] [NP<sub>3</sub>]

*SVO languages*

realization

↑

V+NP  
buffer: [Verb] + [NP<sub>1</sub>]  $\implies$  [Verb] [NP<sub>1</sub>]

realization

↑

V+NP  
buffer: \*[NP<sub>1</sub>] [Verb] + [NP<sub>2</sub>]  $\implies$  \*[NP<sub>1</sub>] [Verb] [NP<sub>2</sub>]

realization

1↑    2↑

V+NP  
buffer: \*[NP<sub>1</sub>] \*[Verb] [NP<sub>2</sub>] + [NP<sub>3</sub>]  $\implies$  \*[NP<sub>1</sub>] \*[Verb] [NP<sub>2</sub>] [NP<sub>3</sub>]



## 24.4 Subordinating navigation

### 24.4.1 epr-concatenation

Peter leaves the house. Then he crosses the street.

Peter crosses the street. Before that he leaves the house.

### 24.4.2 id-concatenation

Peter orders a salad. The salad is eaten by Peter.

### 24.4.3 epr-subordination (adverbial clauses)

Before Peter crosses the street, he leaves the house.

Peter, before he crosses the street, leaves the house.

Peter leaves, before he crosses the street, the house.

Peter leaves the house, before he crosses the street.

After Peter leaves the house, he crosses the street.

Peter, after he leaves the house, crosses the street.

Peter crosses, after he leaves the house, the street.

Peter crosses the street, after he leaves the house.



### 24.4.4 id-subordination (relative clause)

Peter, who leaves the house, crosses the street.

### 24.4.5 Applying NP+id in the word bank 24.1.2

$$\text{NP+id: } \begin{bmatrix} \text{M-np: } a \\ \text{VERB: } b \\ \text{prn: } k \\ \text{id: } m \end{bmatrix} \quad \begin{bmatrix} \text{M-np: } a \\ \text{VERB: } c \\ \text{prn: } l \\ \text{id: } m \end{bmatrix} \implies \begin{bmatrix} \text{M-verb: } c \\ \text{NP: } x a y \\ \text{prn: } l \\ \text{epr:} \end{bmatrix} \{9 \text{ V+NP1, } 10 \text{ V+NP2}\}$$

$$\begin{bmatrix} \text{I-concept}_{loc}: Peter \\ \text{P: Nom} \\ \text{C: } \begin{bmatrix} \text{MOD:} \\ \text{VERB: cross} \end{bmatrix} \\ \text{I: } \begin{bmatrix} \text{prn: } 2 \\ \text{id: } 1 \end{bmatrix} \end{bmatrix}$$

$$\begin{bmatrix} \text{I-concept}_{loc}: Peter \\ \text{P: Nom} \\ \text{C: } \begin{bmatrix} \text{MOD:} \\ \text{VERB: leave} \end{bmatrix} \\ \text{I: } \begin{bmatrix} \text{prn: } 1 \\ \text{id: } 1 \end{bmatrix} \end{bmatrix}$$

$$\begin{bmatrix} \text{I-concept}_{loc}: leave \\ \text{P: indicative} \\ \text{C: } \begin{bmatrix} \text{MOD:} \\ \text{NP: Peter, house} \end{bmatrix} \\ \text{I: } \begin{bmatrix} \text{prn: } 1 \\ \text{epr: } \begin{bmatrix} \phantom{x} \\ \phantom{x} \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

### 24.4.6 Adnominal embedding navigation (preverbal)

Peter, who leaves the house, crosses the street.

▼ <i>cross</i>	<i>Peter</i>	NP+id:		<i>street</i>
prn:2	prn:2	<i>leave</i>	<i>house</i> ▲	prn: 2
	id: 1	prn:1	prn:1	id: 2
			id:3	

### 24.4.7 Word order of adnominal embedding in German

Peter, der das Haus verlassen hat, überquert die Straße.

*Peter, who the house left-has, crosses the street.*

realization

V+NP2  
 buffer: [Verb] + [NP<sub>1</sub>]  $\implies$  [Verb]▼ [NP<sub>1</sub>]

↑

realization

NP+id  
 buffer: \*[NP<sub>1</sub>] [Verb] ▼ + [NP1']  $\implies$  \*[NP<sub>1</sub>] PRO [Verb]▼ [Verb']

↑

realization

V+NP1  
 buffer: \*[NP<sub>1</sub>] \*[PRO] [Verb]▼ [Verb'] + [NP2']  $\implies$  \*[NP<sub>1</sub>] \*[PRO] [Verb]▲ [Verb'] [NP2']

2↑ 1↑

realization

V+NP1  
 buffer: \*[NP<sub>1</sub>] \*[PRO] \*[NP2'] \*[Verb'] [Verb]▲ + [NP<sub>2</sub>]  
 $\implies$  \*[NP<sub>1</sub>] \*[PRO] \*[NP2'] \*[Verb'] [Verb]▲ [NP<sub>2</sub>]

1↑ 2↑

### 24.4.8 Application of V+epr in the word bank 24.1.2

$$\text{V+epr: } \left[ \begin{array}{l} \text{M-verb: } a \\ \text{NP: } x \\ \text{prn: } n \\ \text{epr: } m \ C \ n \end{array} \right] \quad \left[ \begin{array}{l} \text{M-verb: } b \\ \text{NP: } y \\ \text{prn: } m \\ \text{epr: } m \ C \ n \end{array} \right] \Rightarrow \left[ \begin{array}{l} \text{M-verb: } b \\ \\ \\ \end{array} \right] \{7 \text{ V+NP1, } 8 \text{ V+NP2}\}$$



$$\left[ \begin{array}{l} \text{I-concept}_{loc}: \textit{cross} \\ \text{P: indicative} \\ \text{C: } \left[ \begin{array}{l} \text{MOD:} \\ \text{NP: Peter, street} \end{array} \right] \\ \text{I: } \left[ \begin{array}{l} \text{prn: } 2 \\ \text{epr: } \left[ \begin{array}{l} 1 \text{ then } 2 \\ 2 \text{ then } 3 \end{array} \right] \end{array} \right] \end{array} \right] \quad \left[ \begin{array}{l} \text{I-concept}_{loc}: \textit{leave} \\ \text{P: indicative} \\ \text{C: } \left[ \begin{array}{l} \text{MOD:} \\ \text{NP: Peter, house} \end{array} \right] \\ \text{I: } \left[ \begin{array}{l} \text{prn: } 1 \\ \text{epr: } \left[ \begin{array}{l} 1 \text{ then } 2 \end{array} \right] \end{array} \right] \end{array} \right] \quad \left[ \begin{array}{l} \text{I-concept}_{loc}: \textit{leave} \\ \text{P: indicative} \\ \text{C: } \left[ \begin{array}{l} \text{MOD:} \\ \text{NP: Peter, house} \end{array} \right] \\ \text{I: } \left[ \begin{array}{l} \text{prn: } 1 \\ \text{epr: } \left[ \begin{array}{l} 1 \text{ then } 2 \end{array} \right] \end{array} \right] \end{array} \right]$$

### 24.4.9 Different realizations of conjunctions

	temporal	causal	modal
coordinating forward:	P1. Then P2.	P1. Therefore P2.	P1. Thus P2.
coordinating backward:	P2. Earlier P1.		
subordinating forward:	p1, before P2, p1.	p1, for which reason P2, p1.	p1, as P2, p1
subordinating backward:	p2, after P1, p2.	p2, because P1, p2.	

### 24.4.10 Adverbial embedding navigation

Peter crossed, after he left the house, the street.

▼ <i>cross</i>	<i>Peter</i>	V+epr	<i>street</i>
prn:2	prn:2	<i>leave</i>	<i>Peter house</i> ▲
(2 then 3)	id: 1	prn: 1	prn:1 prn:1
(1 then 2)		(1 then 2)	id:1 id:2
			id: 3

### 24.4.11 Word order of adverbial embedding in German

Peter überquert, nachdem er das Haus verlassen hat, die Straße.

(Peter crosses, after he the house left-has, the street.)

realization

V+NP1

↑

buffer: [Verb] + [NP<sub>1</sub>]  $\implies$  [Verb] [NP<sub>1</sub>]

realization

V+epr

1↑ 2↑

buffer: \*[NP<sub>1</sub>] [Verb] + [Verb']  $\implies$  \*[NP<sub>1</sub>] [Verb]▼ [CNJ] [Verb']

realization

V+NP1

1↑

buffer: \*[NP<sub>1</sub>] \*[Verb]▼ \*[CNJ] [Verb'] + [NP1']  $\implies$  \*[NP<sub>1</sub>] \*[Verb]▼ \*[CNJ] [Verb'] [NP1']

V+NP1

realization

buffer: \*[NP<sub>1</sub>] \*[Verb]▼ \*[CNJ] \*[NP1'] [Verb'] + [NP2']

2↑ 1↑

$\implies$  \*[NP<sub>1</sub>] \*[Verb]▲ \*[CNJ] \*[NP1'] [Verb'] [NP2']

V+NP1

realization

buffer: \*[NP<sub>1</sub>] \*[Verb]▲ \*[CNJ] \*[NP1'] \*[NP2'] [Verb'] + [NP<sub>2</sub>]

1↑

$\implies$  \*[NP<sub>1</sub>] \*[Verb]▲ \*[CNJ] \*[NP1'] \*[NP2'] [Verb'] [NP<sub>2</sub>]

### 24.4.12 Multiple center embeddings in German

Peter, der den Salat, den er gegessen hatte, bezahlt hatte, verließ das Restaurant.  
 (*Peter, who the salad, which he paid-had, eaten-had, left the restaurant.*)

▼ <i>leave</i>	<i>Peter</i>	NP+id:				<i>restaurant</i>
prn:7	prn:7	▼ <i>pay</i>	<i>salad</i> ▲			▲ prn: 7
	id: 1	prn:6	prn:6	▼ <i>eat</i>	<i>Peter</i> ▲	id: 4
			id:5	prn: 5	id:1	

## 24.5 LA-search and LA-inference

### 24.5.1 Basic types of questions in natural language

#### Wh-question

Who entered the restaurant?

#### Yes/no-question

Did Peter enter the restaurant?

### 24.5.2 Search coplets of the two basic types of queries

#### Wh-question

$$\left[ \begin{array}{l} \text{I-concept}_{loc}: \textit{enter} \\ \text{E:} \\ \text{F: } \left[ \begin{array}{l} \text{MOD:} \\ \text{NP: } \sigma\text{-1, restaurant} \end{array} \right] \\ \text{I: } \left[ \text{prn: } \sigma\text{-2} \right] \end{array} \right]$$

#### Yes/no-question

$$\left[ \begin{array}{l} \text{I-concept}_{loc}: \textit{enter} \\ \text{E:} \\ \text{F: } \left[ \begin{array}{l} \text{MOD:} \\ \text{NP: Peter, restaurant} \end{array} \right] \\ \text{I: } \left[ \text{prn: } \sigma\text{-2} \right] \end{array} \right]$$



### 24.5.3 LA-Q1 (WH-questions)

$ST_S: \{([a]\{1 r_1, 2 r_2\})\}$

$$r_1: \begin{bmatrix} \text{M-verb: } a \\ \neg\text{NP: } y \sigma z \\ \text{prn: } m \end{bmatrix} \begin{bmatrix} \text{M-verb: } a \\ \neg\text{NP: } y \sigma z \\ \text{prn: } m - 1 \end{bmatrix} \Longrightarrow \begin{bmatrix} \text{M-verb: } a \\ \neg\text{NP: } y \sigma z \\ \text{prn: } m - 1 \end{bmatrix} \{3 r_1 4 r_2\}$$

$$r_2: \begin{bmatrix} \text{M-verb: } a \\ \neg\text{NP: } y \sigma z \\ \text{prn: } m \end{bmatrix} \begin{bmatrix} \text{M-verb: } a \\ \text{NP: } y \sigma z \\ \text{prn: } m - 1 \end{bmatrix} \Longrightarrow \begin{bmatrix} \text{M-verb: } a \\ \text{NP: } y \sigma z \\ \text{prn: } m - 1 \end{bmatrix} \{5 r_3\}$$

$$r_3: \begin{bmatrix} \text{M-verb: } a \\ \text{NP: } y \sigma z \\ \text{prn: } n \end{bmatrix} \begin{bmatrix} \text{M-np: } \sigma \\ \text{VERB: } a \\ \text{prn: } n \end{bmatrix} \Longrightarrow \begin{bmatrix} \text{M-np: } \sigma \\ \text{VERB: } a \\ \text{prn: } n \end{bmatrix} \{ \}$$

$ST_F: \{([M-np: \sigma] rp_3)\}$

### 24.5.4 LA-Q2 (yes/no-questions)

$ST_S: \{([a]\{1 r_1, 2 r_2\})\}$

$$r_1: \begin{bmatrix} \text{M-verb: } a \\ \neg\text{NP: } x \\ \text{prn: } m \end{bmatrix} \begin{bmatrix} \text{M-verb: } a \\ \neg\text{NP: } x \\ \text{prn: } m - 1 \end{bmatrix} \implies \begin{bmatrix} \text{M-verb: } a \\ \neg\text{NP: } x \\ \text{prn: } m - 1 \end{bmatrix} \{3 r_1 4 r_2\}$$

$$r_2: \begin{bmatrix} \text{M-verb: } a \\ \neg\text{NP: } x \\ \text{prn: } m \end{bmatrix} \begin{bmatrix} \text{M-verb: } a \\ \text{NP: } x \\ \text{prn: } m - 1 \end{bmatrix} \implies \begin{bmatrix} \text{M-verb: } a \\ \text{NP: } x \\ \text{prn: } m - 1 \end{bmatrix} \{ \}$$

$ST_F: \{([\text{verb: } a] rp_1) ([\text{verb: } a] rp_2)\}$

### 24.5.5 Inference schemata of propositional calculus

1.  $\frac{A, B}{\vdash A \& B}$
2.  $\frac{A \vee B, \neg A}{\vdash B}$
3.  $\frac{A \rightarrow B, A}{\vdash B}$
4.  $\frac{A \rightarrow B, \neg B}{\vdash \neg A}$
5.  $\frac{A \& B}{\vdash A}$
6.  $\frac{A}{\vdash A \vee B}$
7.  $\frac{\neg A}{\vdash A \rightarrow B}$
8.  $\frac{\neg \neg A}{\vdash A}$

### 24.5.6 LA-rule for the propositional inference of conjunction

$$\text{inf1: } \begin{bmatrix} \text{M-verb: } a \\ \text{prn: } m \end{bmatrix} \begin{bmatrix} \text{M-verb: } b \\ \text{prn: } n \end{bmatrix} \implies \begin{bmatrix} \text{M-verb: } a \\ \text{prn: } m \\ \text{epr: } m \text{ and } n \end{bmatrix} \begin{bmatrix} \text{M-verb: } b \\ \text{prn: } n \\ \text{epr: } m \text{ and } n \end{bmatrix}$$

### 24.5.7 Coplets of an absolute proposition

$\left[ \begin{array}{l} \text{I-concept}_{loc}: be \\ \text{NP: dog, animal} \\ \text{prn: abs327} \end{array} \right]$	$\left[ \begin{array}{l} \text{I-concept}_{loc}: dog \\ \text{VERB: be} \\ \text{prn: abs327} \end{array} \right]$	$\left[ \begin{array}{l} \text{I-concept}_{loc}: animal \\ \text{VERB: be} \\ \text{prn: abs327} \end{array} \right]$
--	--	---

### 24.5.8 Coplet of an episodic proposition

$$\left[ \begin{array}{l} \text{I-concept}_{loc}: see \\ \text{NP: Peter, dog} \\ \text{prn: 969} \end{array} \right]$$

### 24.5.9 Inference rule inf2 for absolute propositions

$$\text{inf2: } \left[ \begin{array}{l} \text{M-verb: } a \\ \text{NP: } x b y \\ \text{prn: } n \end{array} \right] \left[ \begin{array}{l} \text{M-verb: } be \\ \text{NP: } b c \\ \text{prn: } abs \end{array} \right] \implies \left[ \begin{array}{l} \text{M-verb: } a \\ \text{NP: } x c y \\ \text{prn: } n \end{array} \right]$$