Syntactic Theory

Head-driven Phrase Structure Grammar (HPSG)

Yi Zhang

Department of Computational Linguistics
Saarland University

December 15th, 2009

History of HPSG and its influences

- HPSG1: Pollard and Sag (1987)
  Formalism (typed feature structures), subcategorization, LP rules, hierarchical lexicon
- HPSG2: Pollard and Sag (1994) Chapter 1-8
  The structure of signs, control theory, binding theory
- HPSG3: Pollard and Sag (1994) Chapter 9 “Reflections and Revisions”
  Valence features SUBJ, COMPS, SPR
- HPSG4, HPSG5, . . .
  Unbounded dependency constructions, linking theory, semantic representation, argument realization, . . .

History of HPSG and its influences (cont.)

The development of HPSG is influenced by contemoporary theories:

- Syntax
  - Generalized Phrase Structure Grammar (Gazdar, Klein, Pullum & Sag, 1985)
  - Categorial Grammar (McGee Wood, 1993)
  - Lexical-Functional Grammar (Kaplan & Bresnan, 1982)
  - Construction Grammar (Goldberg, 1995)
  - Government-Binding Theory (Haegeman, 1994)
- Semantics
  - Situation Semantics (Barwise & Perry, 1983)
  - Discourse Representation Theory (Kamp & Reyle, 1993)x

HPSG vs. “Classical” Phrase Structure Grammar

Similarities

- Both are **monostratal**: every analysis is represented by a single structure
- Grammar rules have **local** scope: mother phrase and its immediate daughters

HPSG vs. "Classical" Phrase Structure Grammar

Differences

- HPSG uses complex categories while classical PSG uses simple/atomic ones
- HPSG specifies **Immediate Dominance (ID)** and **Linear Precedence (LP)** separately
  - ID specifies the mother and daughters in a local tree without specifying the order of the daughters
  - LP determines the relative order of the daughters in a local tree without making reference to the mother
  - Further universal **principles** are specified in HPSG to constrain the set of local trees admitted by the ID schemata
- HPSG analyses include semantic representations in addition to syntactic representations

HPSG vs. Transformational Grammar

Similarities

- Both try to account for a similar range of data (e.g. in the development of the Binding Theory)
- Both are theories of **generative grammar**

HPSG vs. Transformational Grammar

Differences

- HPSG is **non-derivational**, TG is derivational
  - TG analyses start with a base generated tree, which is then subject to a variety of transformation (e.g., movement, deletion, reanalysis) that produce the desired surface structure
  - HPSG analyses generate only the surface structure, rule ordering is irrelevant
- HPSG constraints are **local**, TG allows non-local statements
- HPSG uses more complex categories than TG
- HPSG is more committed to precise formalization than TG
- HPSG is better suited to computational implementation than TG

Key Properties of HPSG and their consequences

- HPSG is monostratal, declarative, non-derivational
  No transformations, no rule ordering. Analyses are surface oriented, with a desire to avoid abstract structure such as traces and functional categories
- HPSG is constraint-based
  A structure is well-formed if and only if it satisfies all relevant constraints. Constraints are not violable (as in Optimality Theory, for example)
- HPSG is a lexicalist theory
  Strong lexicalism; Word-internal structures and phrase structure are handled separately
- HPSG is a unification-based linguistic framework where all linguistic objects are represented as “typed feature structures”

Psycholinguistic Evidence

- Human language processing is **incremental**:
  Partial interpretations can be generated for partial utterances
  HPSG constraints can apply to partial structures as well as complete trees
- HLP is **integrative**:
  Linguistic interpretations depend on a large amount of non-linguistic information (e.g. world knowledge)
  The signs in HPSG can incorporate both linguistic and non-linguistic information using the same formal representation

Psycholinguistic Evidence

- HLP is **order-independent**:
  There is no fixed sequence in which pieces of information are consulted and incorporated into a linguistic interpretation
  HPSG is a declarative and non-derivational model
- HLP is **reversible**:
  Utterances can be understood and generated
  HPSG is process-neutral, and can be applied for either production or comprehension

Signs in HPSG

Sign is the basic sort/type in HPSG used to describe lexical items (of type *word*) and phrases (of type *phrase*). All signs carry the following two features:

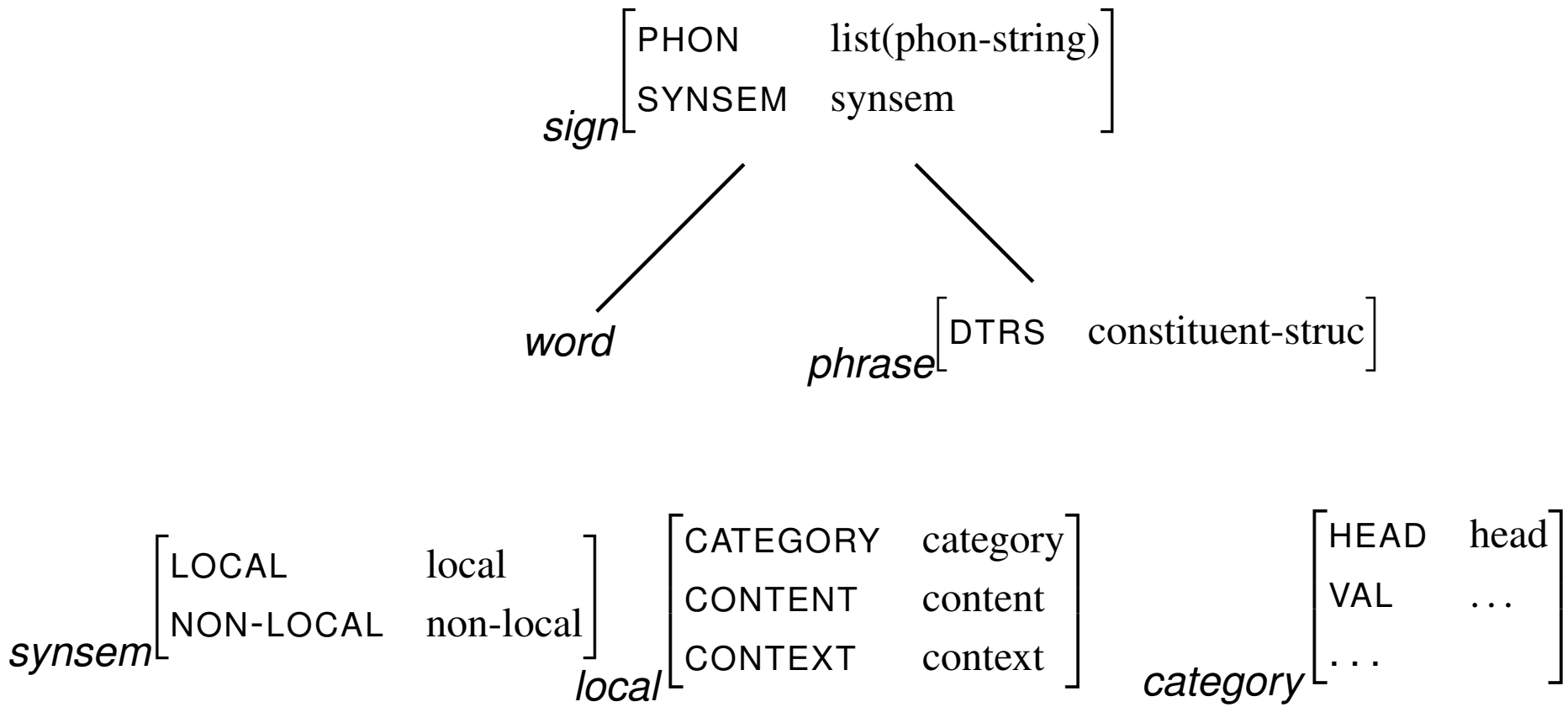
- PHON encodes the phonological representation of the sign
- SYNSEM syntax and semantics

$$\textit{sign}\begin{bmatrix}\text{PHON} & \text{list(phon-string)}\\ \text{SYNSEM} & \text{synsem}\end{bmatrix}$$

Structure of the Signs in HPSG

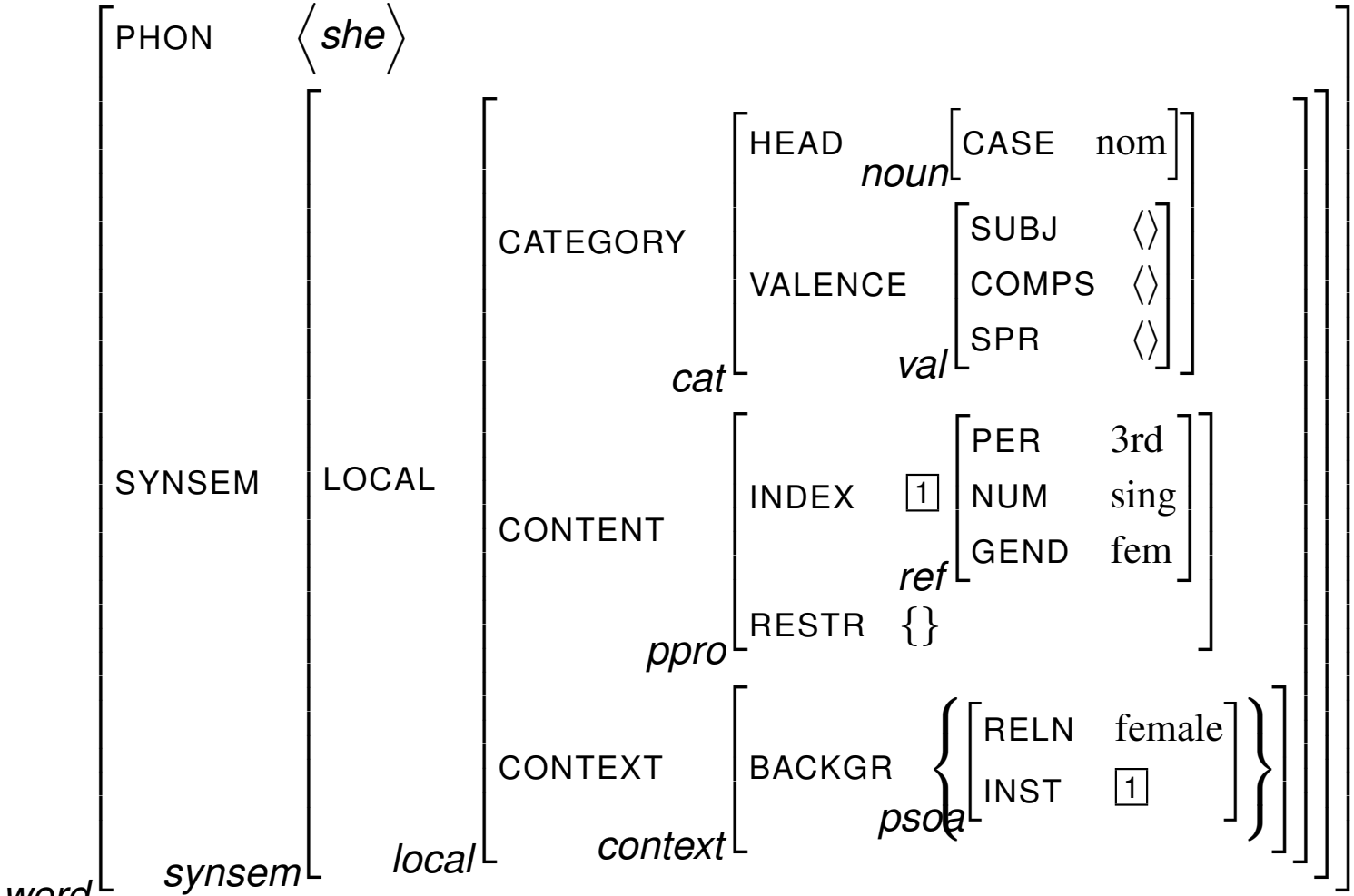
- *synsem* introduces the features LOCAL and NON-LOCAL
- *local* introduces CATEGORY (CAT) CONTENT (CONT) and CONTEXT (CONX)
- *non-local* will be discussed in connection with unbounded dependencies
- *category* includes the syntactic category and the grammatical argument of the word/phrase

An Ontology of Linguistic Objects



Structure of the Signs in HPSG (cont.)

Example



Syntactic Category & Valence

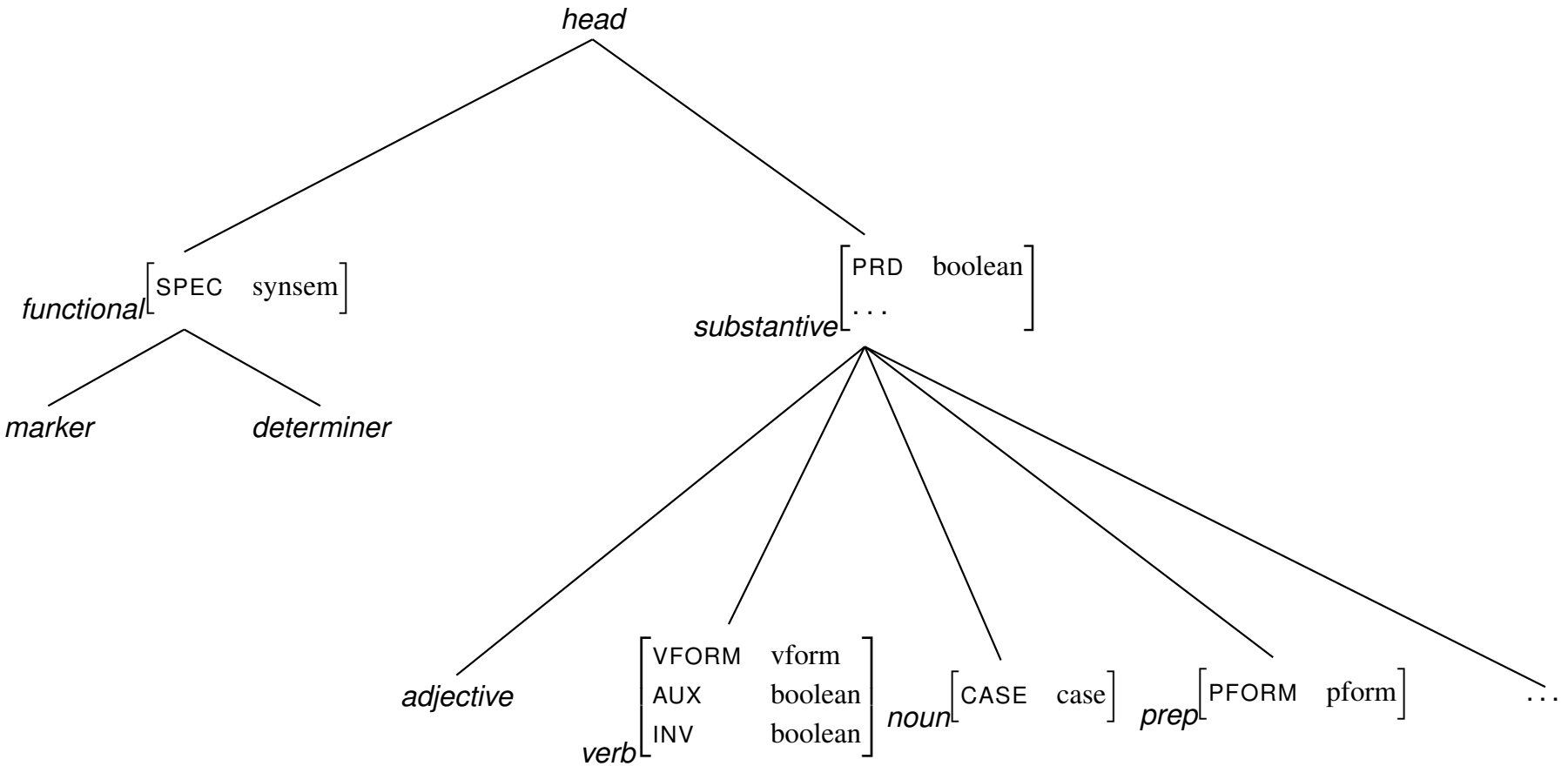
The value of CATEGORY encode information about

- The sign's syntactic category ("part-of-speech")
  - Given via the feature $\left[\text{HEAD} \quad \text{head}\right]$, where *head* is the supertype for *noun, verb, adjective, preposition, determiner, marker*; each of these types selects a particular set of **head features**
- The sign's subcategorzation frame/valence, i.e. its potential to combine with other signs to form larger phrases
  - Three list-valued features

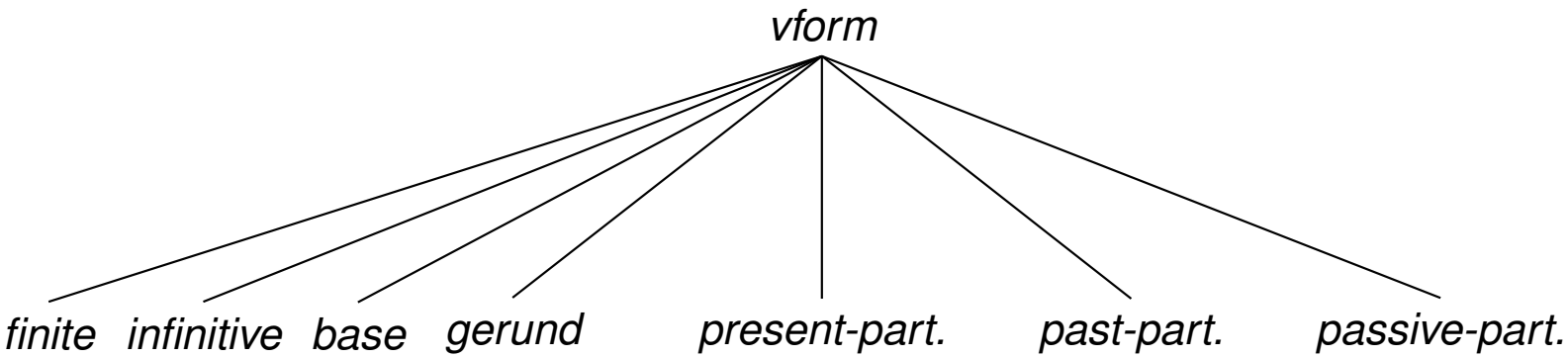
$$\left[\text{SYNSEM}|\text{LOC}|\text{CAT}|\text{VALENCE} \quad \textit{valence}\left[\begin{array}{ll}\text{SUBJECT} & \text{list(synsem)}\\ \text{SPECIFIER} & \text{list(synsem)}\\ \text{COMPLEMENTS} & \text{list(synsem)}\end{array}\right]\right]$$

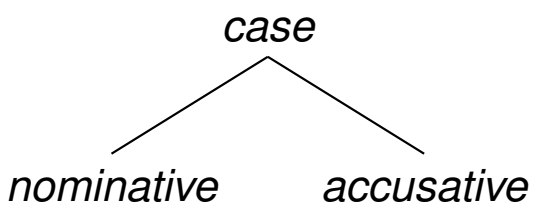
  - If any of these lists are non-empty (*"unsaturated"*), the sign has the potential to combine with another sign

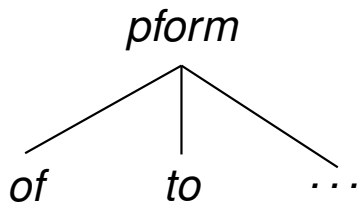
Head Information



Features of *head* Types







Valence Features

- The VALENCE lists take as values the list of *synsem*s instead of *sign*s
- This means that word does not have access to the DTRS list of items on its *valence* lists
- More discussion on different valence lists will follow when we introduce the **valence principle** and **ID schemata**

Semantic Representation

Semantic interpretation of the sign is given as the value to CONTENT

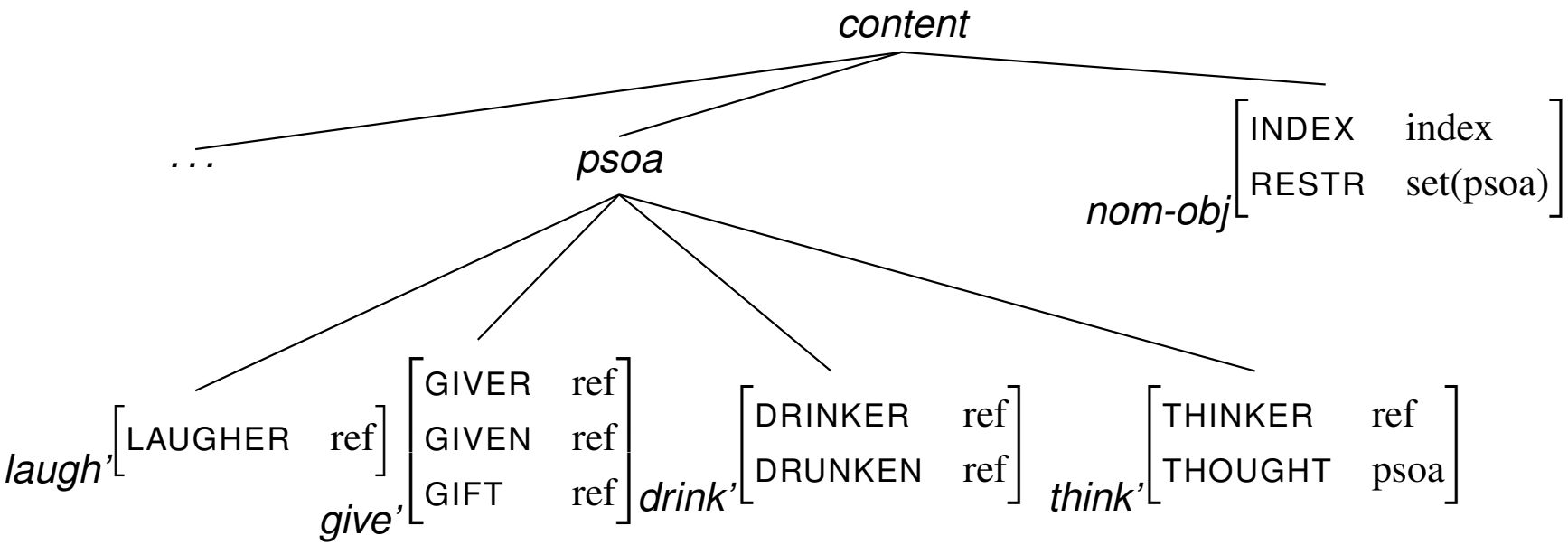
- *nominal-object*: an individual/entity (or a set of them), associated with a referring index, bearing agreement features
- *parameterized-state-of-affairs*: a partial situate; an event relation along with role names for identifying the participants of the event
- *quantifier*: *some, all, every, a, the, . . .*
- Note: many of these have been reformulated by "Minimal Recursion Semantics" which allows underspecification of quantifier scopes, though a in-depth discussion of MRS is beyond the scope of this class

Semantic Representation

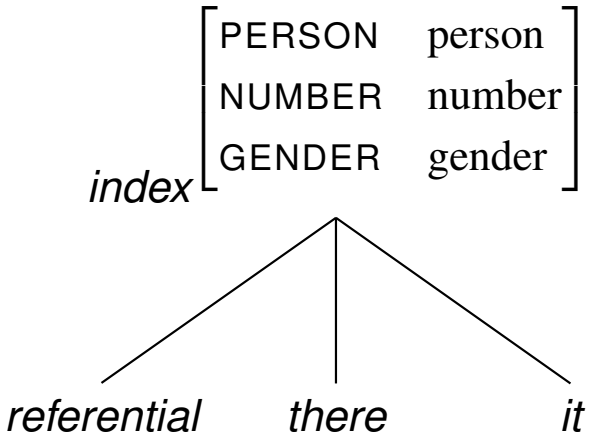
Semantic interpretation of the sign is given as the value to CONTENT

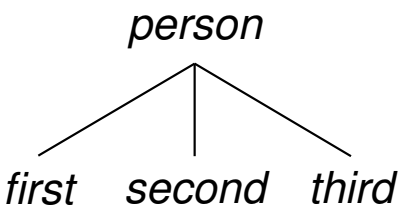
- *nominal-object*: an individual/entity (or a set of them), associated with a referring index, bearing agreement features
- *parameterized-state-of-affairs*: a partial situate; an event relation along with role names for identifying the participants of the event
- *quantifier*: *some, all, every, a, the, . . .*
- Note: many of these have been reformulated by "Minimal Recursion Semantics" which allows underspecification of quantifier scopes, though a in-depth discussion of MRS is beyond the scope of this class

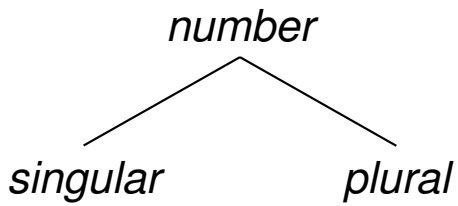
Semantic Representation

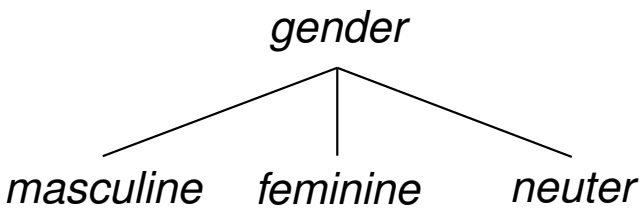


Indices

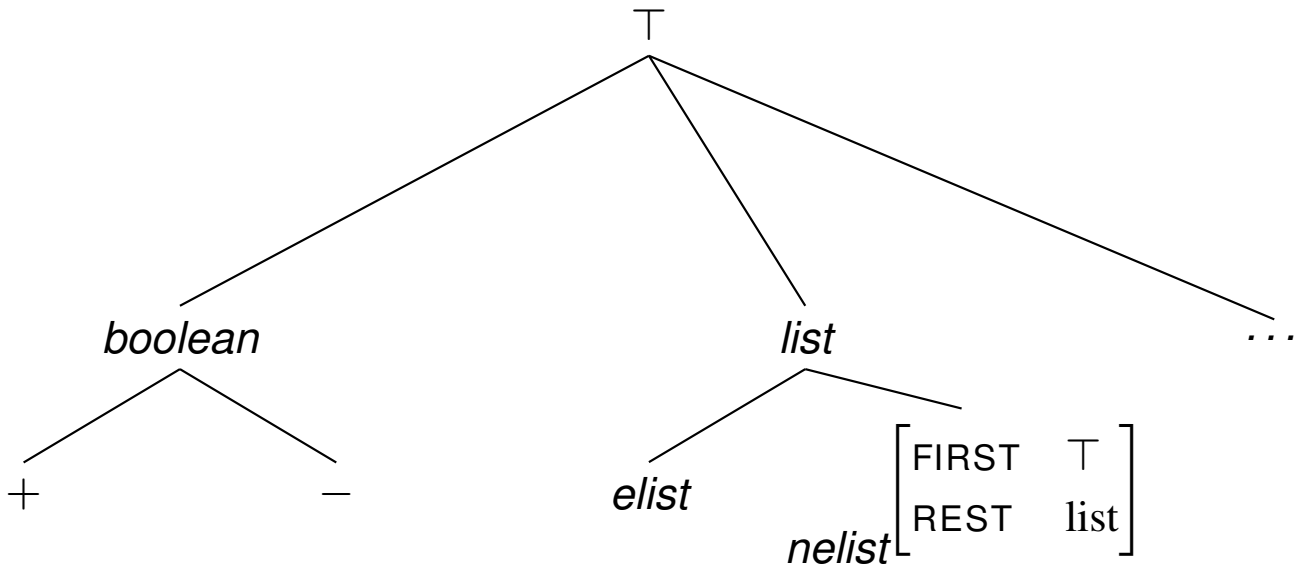

index
PERSON person
NUMBER number
GENDER gender
referential
there
it


person
first
second
third


number
singular
plural


gender
masculine
feminine
neuter

Auxiliary Data Structures



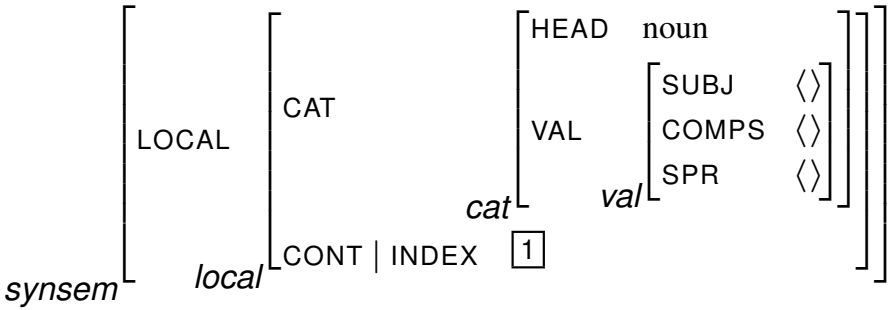
Some List Abbreviations

- Empty list (*elist*) is abbreviated as $\langle\,\rangle$
- $\mathit{nelist}\left[\begin{array}{ll}\textsc{first} & \boxed{1}\\ \textsc{rest} & \boxed{2}\end{array}\right]$ is abbreviated as $\langle \boxed{1} \mid \boxed{2} \rangle$
- $\langle \ldots \boxed{1} \mid \langle\rangle \rangle$ is equivalent to $\langle \ldots \boxed{1} \rangle$
- $\mathit{nelist}\left[\begin{array}{ll}\textsc{first} & \boxed{1}\\ \textsc{rest} & \mathit{nelist}\left[\begin{array}{ll}\textsc{first} & \boxed{2}\\ \textsc{rest} & \boxed{3}\end{array}\right]\end{array}\right]$ is equivalent to $\langle \boxed{1}, \boxed{2} \mid \boxed{3} \rangle$
- $\langle \top \rangle$ and $\langle \boxed{1} \rangle$ describe all lists of length one

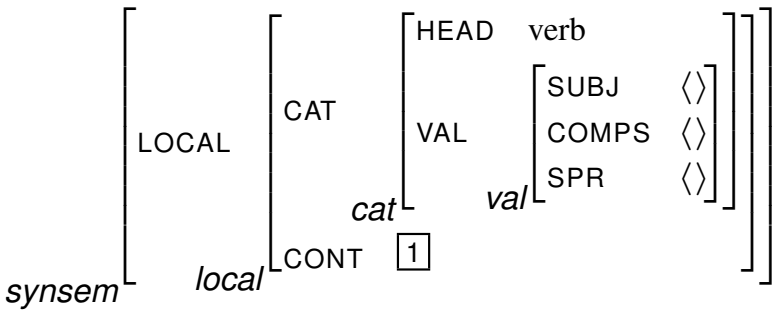
Abbreviations of Common AVMs

The following abbreviations are used to describe *synsem* objects:

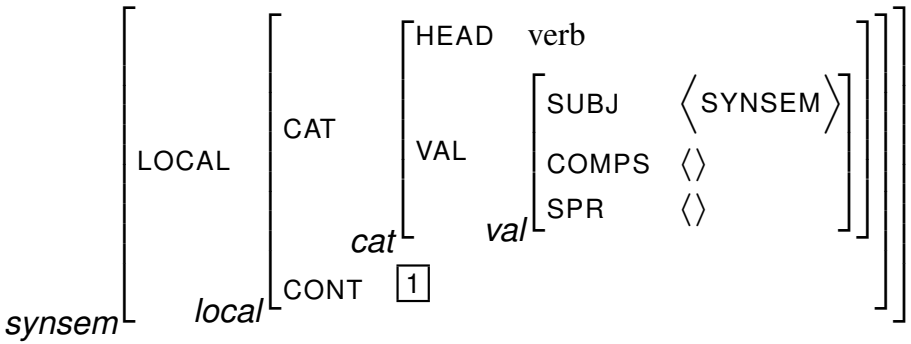
- NP[1]



- S:[1]



- VP:[1]



References I

Pollard, C. J. and Sag, I. A. (1994).
Head-Driven Phrase Structure Grammar.
University of Chicago Press, Chicago, USA.