A Set-Theoretical Approach for the Induction of Inheritance Hierarchies

Wiebke Petersen
Institute of Language and Information
Department of Computational Linguistics
University of Düsseldorf

overview

1) presentation of the approach
   1) what kind of data can be captured by the approach?
   2) brief introduction to formal concept lattices
   3) the induced hierarchies and their qualities
   4) how can attribute implications be employed for the induction of hierarchies?
2) learning type hierarchies
   1) input data: untyped feature structures
   2) assigning types to concepts
   3) constructing the type hierarchy

example data:
inflectional paradigms of german nouns

	<sing nom>	<sing gen>	<sing dat>	<sing acc>	<plur nom>	<plur gen>	<plur dat>	<plur acc>
Herr	NULL	_n	_n	_n	_n	_n	_n	_n
Friede	NULL	_n _s	_n	_n	_n	_n	_n	_n
Hemd	NULL	_s	NULL	NULL	_n	_n	_n	_n
Farbe	NULL	NULL	NULL	NULL	_n	_n	_n	_n
Ufer	NULL	_s	NULL	NULL	NULL	NULL	_n	NULL

Herr 'mister', *Friede* 'peace', *Hemd* 'shirt', *Farbe* 'colour', *Ufer* 'shore'

example data:
inflectional paradigms of german nouns

	<sing nom>	<sing gen>	<sing dat>	<sing acc>	<plur nom>	<plur gen>	<plur dat>	<plur acc>
Herr	NULL	_n	_n	_n	_n	_n	_n	_n
Friede	NULL	_n _s	_n	_n	_n	_n	_n	_n
Hemd	NULL	_s	NULL	NULL	_n	_n	_n	_n
Farbe	NULL	NULL	NULL	NULL	_n	_n	_n	_n
Ufer	NULL	_s	NULL	NULL	NULL	NULL	_n	NULL

Herr 'mister', *Friede* 'peace', *Hemd* 'shirt', *Farbe* 'colour', *Ufer* 'shore'

(Friede, <plur nom>, _n):
plural nominative form of *Friede* is *Frieden*

formal context

Def.: A **formal context** K is a tripel (G,M,I) where

• G is a set of objects

• M is a set of attributes

• I is a binary relation $I \subseteq G \times M$ where $(g,m) \in I$ is read as "object g has attribute m."

	Herr	Friede	Hemd	Farbe	Ufer
sing nom: NULL	x	x	x	x	x
sing gen: NULL				x	
sing gen: _s			x		x
sing gen: _n	x				
sing gen: _ns		x			
sing dat: NULL			x	x	x
sing dat: _n	x	x			
sing acc: NULL			x	x	x
sing acc: _n	x	x			
plur nom: NULL					x
plur nom: _n	x	x	x	x	
plur gen: NULL					x
plur gen: _n	x	x	x	x	
plur dat: _n	x	x	x	x	x
plur acc: NULL					x
plur acc: _n	x	x	x	x	

formal concept

Def.: for A⊆G and B⊆M be
A'={m∈M| ∀g∈A: (g,m)∈I}
B'={g∈G| ∀m∈B: (g,m)∈I}

Def.: (A,B) is a **formal concept** of the formal context (G,M,I) iff A⊆G, B⊆M, A'=B and B'=A.

A is called the **extent** and B the **intent** of the concept.

Note: (A'',A') is a formal concept for all A⊆G.

	Herr	Friede	Hemd	Farbe	Ufer
sing nom: NULL	x	x	x	x	x
sing gen: NULL				x	
sing gen: _s			x		x
sing gen: _n	x				
sing gen: _ns		x			
sing dat: NULL			x	x	x
sing dat: _n	x	x			
sing acc: NULL			x	x	x
sing acc: _n	x	x			
plur nom: NULL					x
plur nom: _n	x	x	x	x	
plur gen: NULL					x
plur gen: _n	x	x	x	x	
plur dat: _n	x	x	x	x	x
plur acc: NULL					x
plur acc: _n	x	x	x	x	

formal concept

Def.: for A⊆G and B⊆M be
A'={m∈M| ∀g∈A: (g,m)∈I}
B'={g∈G| ∀m∈B: (g,m)∈I}

Def.: (A,B) is a **formal concept** of the formal context (G,M,I) iff A⊆G, B⊆M, A'=B and B'=A.

A is called the **extent** and B the **intent** of the concept.

Note: (A'',A') is a formal concept for all A⊆G.

A

	Herr	Friede	Hemd	Farbe	Ufer
sing nom: NULL	x	x	x	x	x
sing gen: NULL				x	
sing gen: _s			x		x
sing gen: _n	x				
sing gen: _ns		x			
sing dat: NULL			x	x	x
sing dat: _n	x	x			
sing acc: NULL			x	x	x
sing acc: _n	x	x			
plur nom: NULL					x
plur nom: _n	x	x	x	x	
plur gen: NULL					x
plur gen: _n	x	x	x	x	
plur dat: _n	x	x	x	x	x
plur acc: NULL					x
plur acc: _n	x	x	x	x	

formal concept

Def.: for A⊆G and B⊆M be
A'={m∈M| ∀g∈A: (g,m)∈I}
B'={g∈G| ∀m∈B: (g,m)∈I}

Def.: (A,B) is a **formal concept** of the formal context (G,M,I) iff A⊆G, B⊆M, A'=B and B'=A.

A is called the **extent** and B the **intent** of the concept.

Note: (A",A') is a formal concept for all A⊆G.

A

	Herr	Friede	Hemd	Farbe	Ufer
sing nom: NULL	x	x	x	x	x
sing gen: NULL				x	
sing gen: _s			x		x
sing gen: _n	x				
sing gen: _ns		x			
sing dat: NULL			x	x	x
sing dat: _n	x	x			
sing acc: NULL			x	x	x
sing acc: _n	x	x			
plur nom: NULL					x
plur nom: _n	x	x	x	x	
plur gen: NULL					x
plur gen: _n	x	x	x	x	
plur dat: _n	x	x	x	x	x
plur acc: NULL					x
plur acc: _n	x	x	x	x	

A'

formal concept

Def.: for $A \subseteq G$ and $B \subseteq M$ be
A'$=\{m \in M \mid \forall g \in A: (g,m) \in I\}$
B'$=\{g \in G \mid \forall m \in B: (g,m) \in I\}$

Def.: (A,B) is a **formal concept** of the formal context (G,M,I) iff $A \subseteq G$, $B \subseteq M$, $A'=B$ and $B'=A$.

A is called the **extent** and B the **intent** of the concept.

Note: (A'',A') is a formal concept for all $A \subseteq G$.

A''

A

A'

	Herr	Friede	Hemd	Farbe	Ufer
sing nom: NULL	x	x	x	x	x
sing gen: NULL				x	
sing gen: _s			x		x
sing gen: _n	x				
sing gen: _ns		x			
sing dat: NULL			x	x	x
sing dat: _n	x	x			
sing acc: NULL			x	x	x
sing acc: _n	x	x			
plur nom: NULL					x
plur nom: _n	x	x	x	x	
plur gen: NULL					x
plur gen: _n	x	x	x	x	
plur dat: _n	x	x	x	x	x
plur acc: NULL					x
plur acc: _n	x	x	x	x	

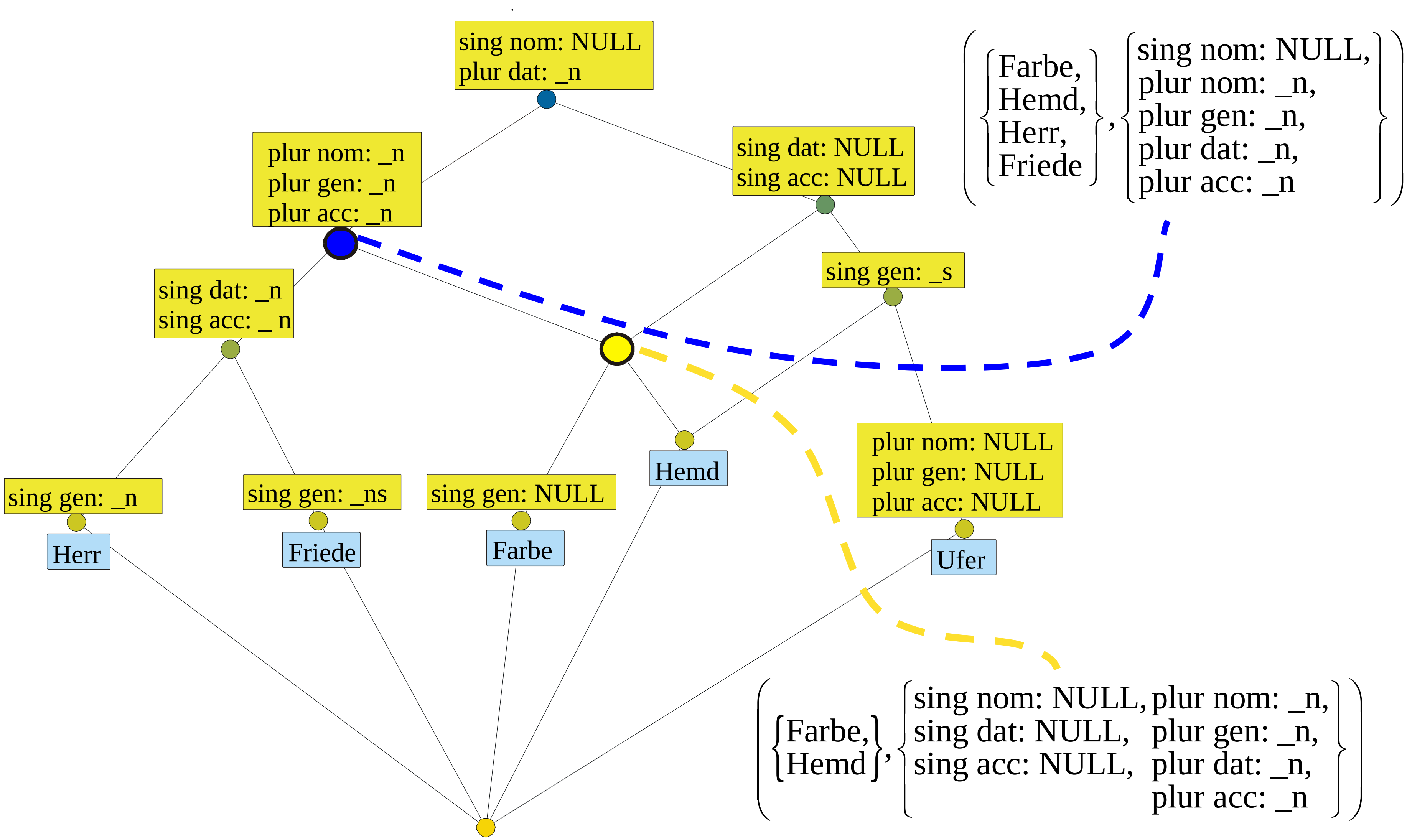
subconcept-superconcept relation

Def.: A concept is a **subconcept** of another concept, if its extent is a subset of the extent of the latter:
$(A_1,B_1) \leq (A_2,B_2) \Leftrightarrow A_1 \subseteq A_2 \Leftrightarrow B_1 \supseteq B_2$

Prop.: If $\mathcal{B}(G,M,I)$ is the set of all formal concepts of the context (G,M,I), then $(\mathcal{B}(G,M,I), \leq)$ forms a lattice.

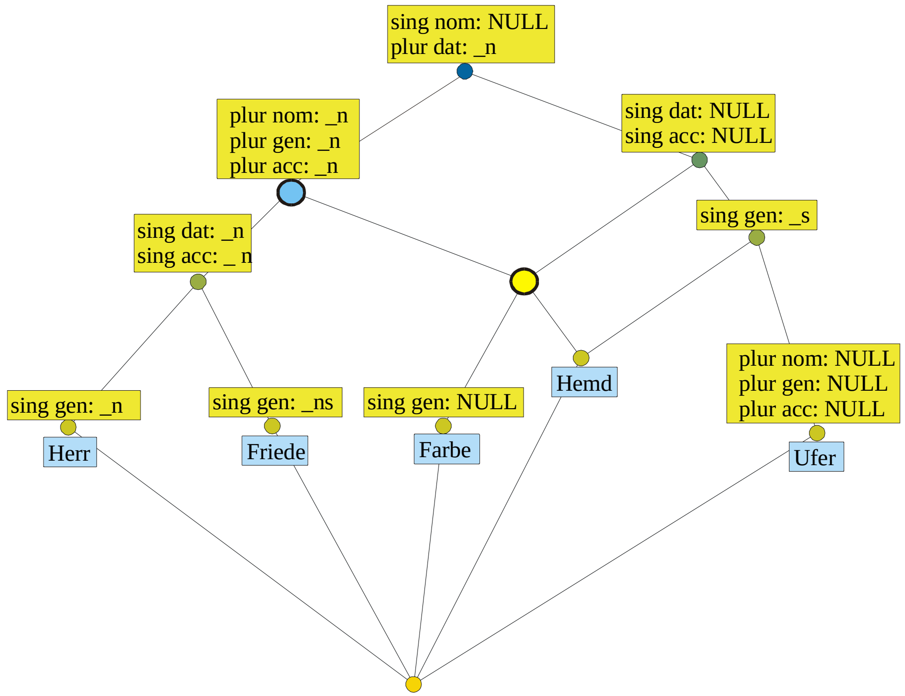
	Herr	Friede	Hemd	Farbe	Ufer
sing nom: NULL	x	x	x	x	x
sing gen: NULL				x	
sing gen: _s			x		x
sing gen: _n	x				
sing gen: _ns		x			
sing dat: NULL			x	x	x
sing dat: _n	x	x			
sing acc: NULL			x	x	x
sing acc: _n	x	x			
plur nom: NULL					x
plur nom: _n	x	x	x	x	
plur gen: NULL					x
plur gen: _n	x	x	x	x	
plur dat: _n	x	x	x	x	x
plur acc: NULL					x
plur acc: _n	x	x	x	x	

formal concept lattice



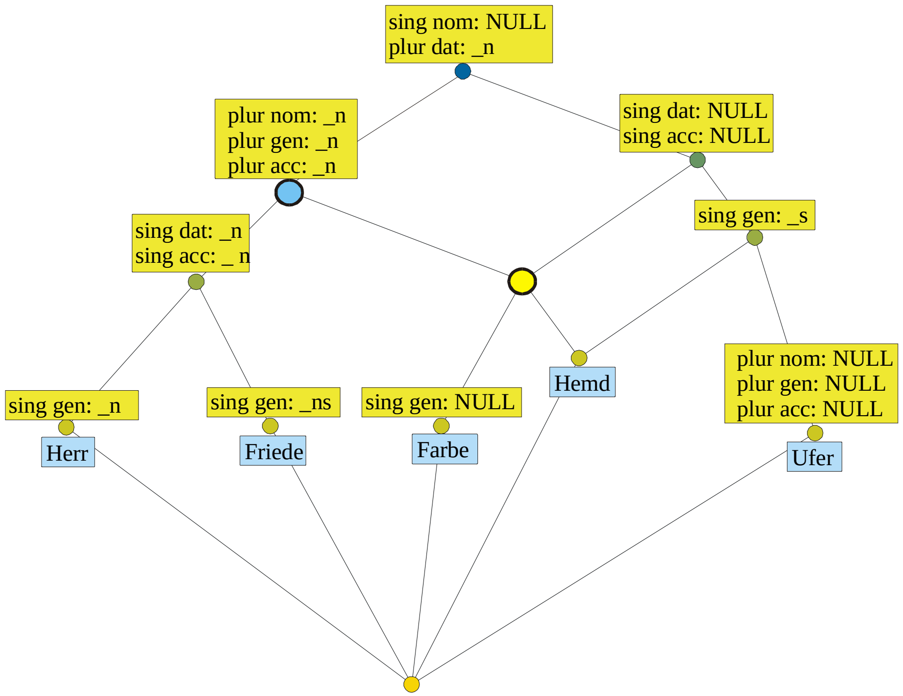
requirements for inheritance hierarchies

- minimal:
  - consistent and complete w.r.t. the data
- desirable:
  - captures generalizations (by allowing exceptions)
  - many (hierarchical) levels
  - free of redundancy: every attribute and object is stated exactly once
  - small in size



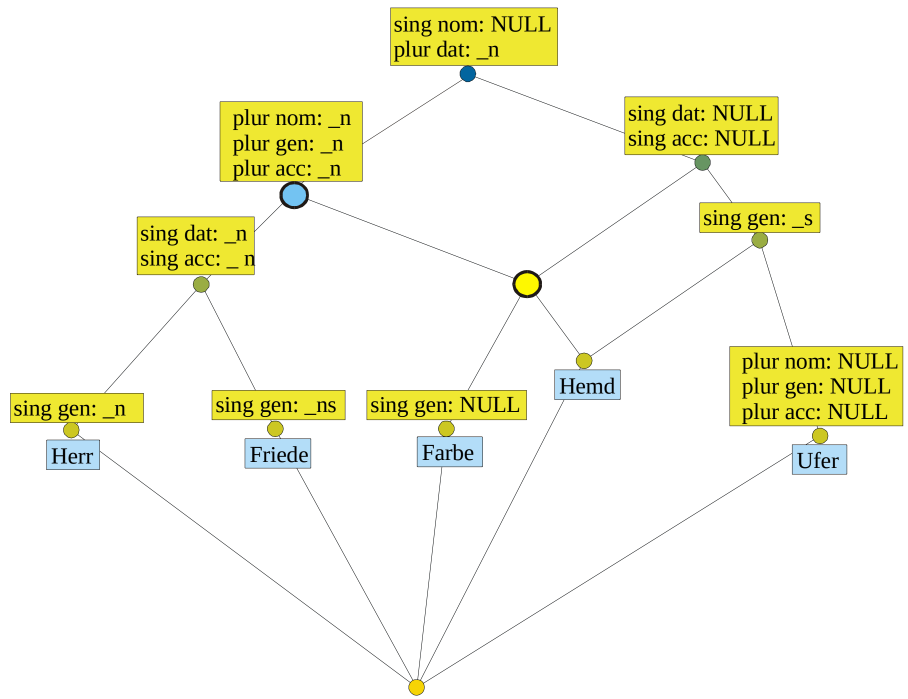
requirements for inheritance hierarchies

- minimal:
  - consistent and complete w.r.t. the data ✓
- desirable:
  - captures generalizations (by allowing exceptions)
  - many (hierarchical) levels
  - free of redundancy: every attribute and object is stated exactly once
  - small in size



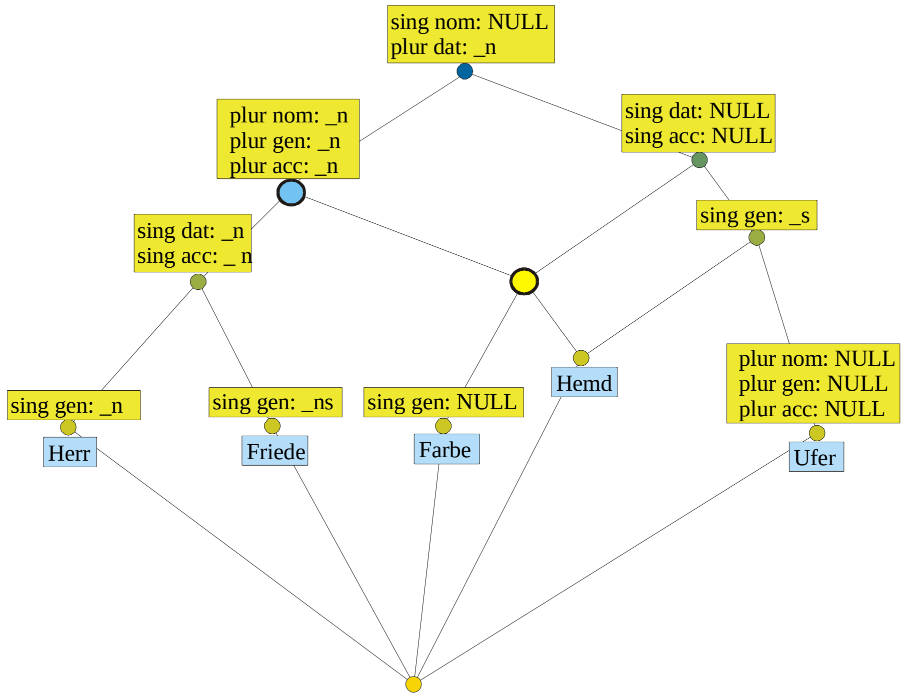
requirements for inheritance hierarchies

- minimal:
  - consistent and complete w.r.t. the data ✓
- desirable:
  - captures generalizations ✓ (by allowing exceptions)
  - many (hierarchical) levels
  - free of redundancy: every attribute and object is stated exactly once
  - small in size



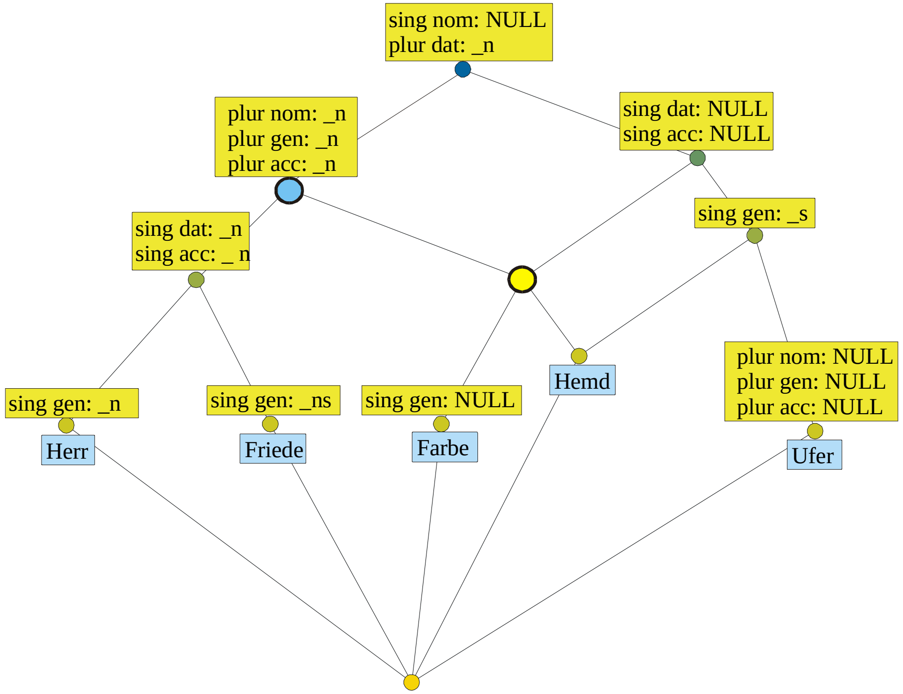
requirements for inheritance hierarchies

- minimal:
  - consistent and complete w.r.t. the data ✓
- desirable:
  - captures generalizations ✓ (by allowing exceptions)
  - many (hierarchical) levels ✓
  - free of redundancy: every attribute and object is stated exactly once
  - small in size

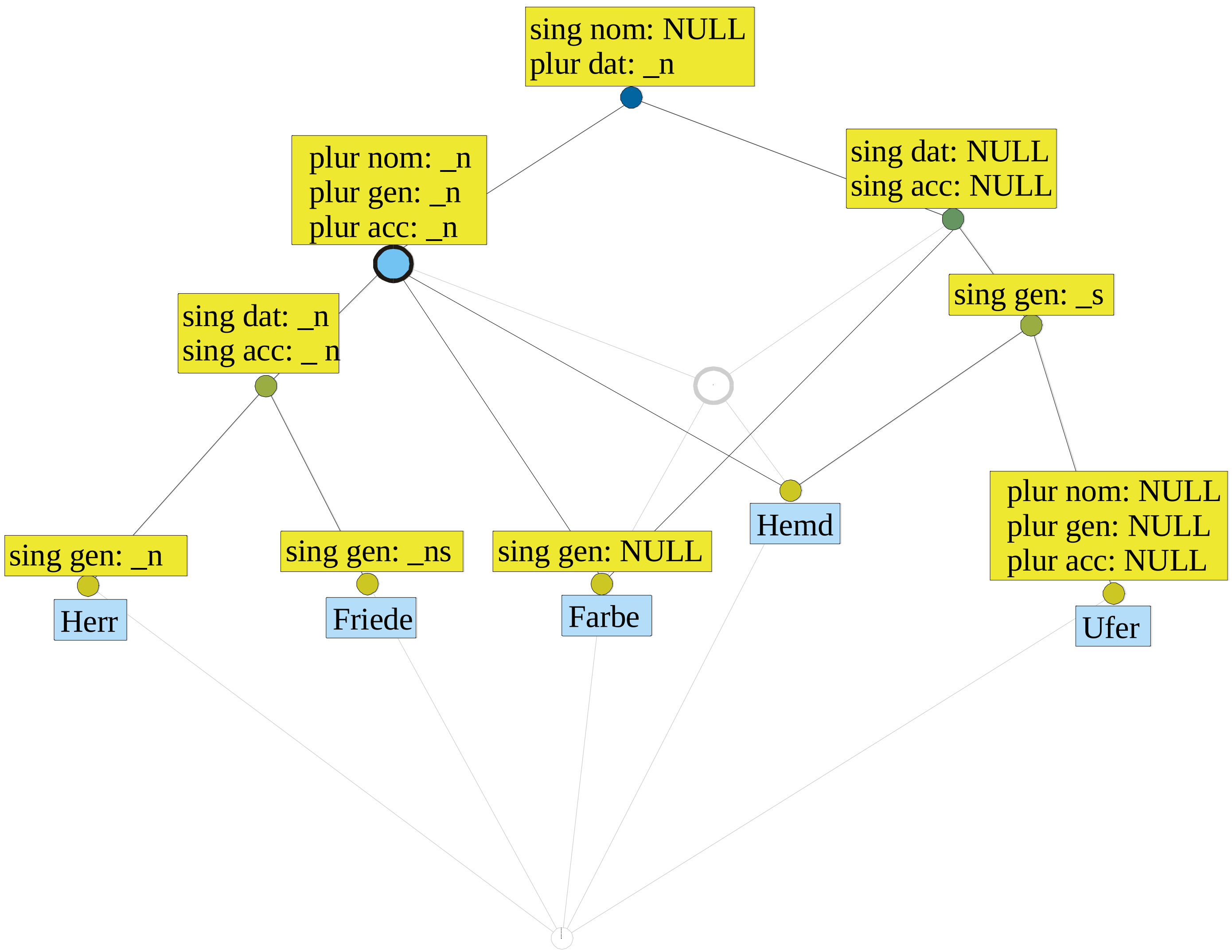


requirements for inheritance hierarchies

- minimal:
  - consistent and complete w.r.t. the data ✓
- desirable:
  - captures generalizations ✓ (by allowing exceptions)
  - many (hierarchical) levels ✓
  - free of redundancy: every attribute and object is stated exactly once ✓
  - small in size



AOC-poset


sing nom: NULL
plur dat: _n
plur nom: _n
plur gen: _n
plur acc: _n
sing dat: NULL
sing acc: NULL
sing gen: _s
sing dat: _n
sing acc: _ n
sing gen: _n
sing gen: _ns
sing gen: NULL
Hemd
plur nom: NULL
plur gen: NULL
plur acc: NULL
Herr
Friede
Farbe
Ufer

overview

1) presentation of the approach
   1) what kind of data can be captured by the approach?
   2) brief introduction to formal concept lattices
   3) the induced hierarchies and their qualities
   4) how can attribute implications be employed for the induction of hierarchies?
2) learning type hierarchies
   1) input data: untyped feature structures
   2) assigning types to concepts
   3) constructing the type hierarchy

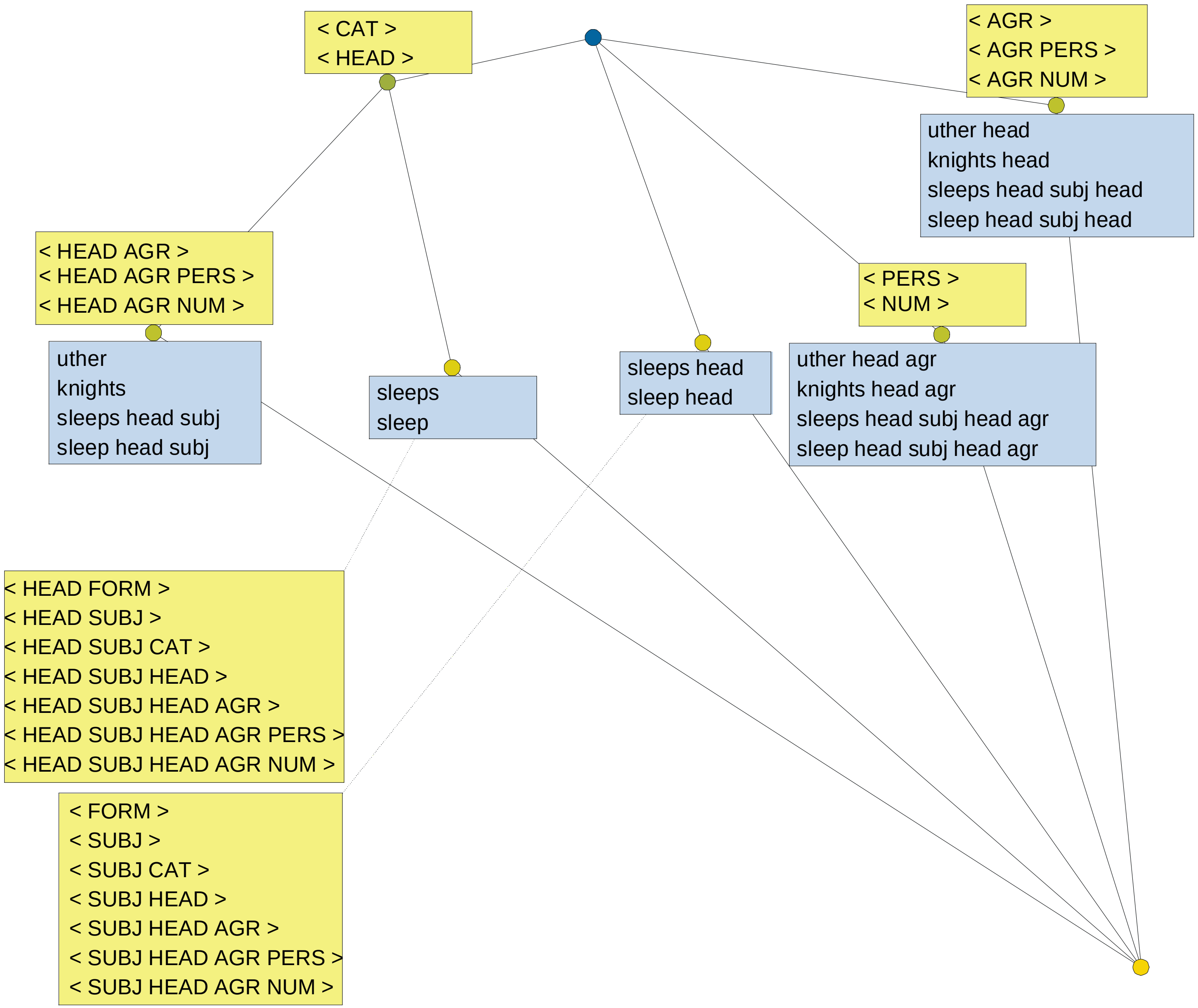
input: untyped feature structures

$$\text{Uther} \rightarrow \begin{bmatrix} \text{CAT: np} \\ \text{HEAD:} \begin{bmatrix} \text{AGR:} \begin{bmatrix} \text{PERS: third} \\ \text{NUM: sing} \end{bmatrix} \end{bmatrix} \end{bmatrix} \qquad \text{knights} \rightarrow \begin{bmatrix} \text{CAT: np} \\ \text{HEAD:} \begin{bmatrix} \text{AGR:} \begin{bmatrix} \text{PERS: third} \\ \text{NUM: plur} \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

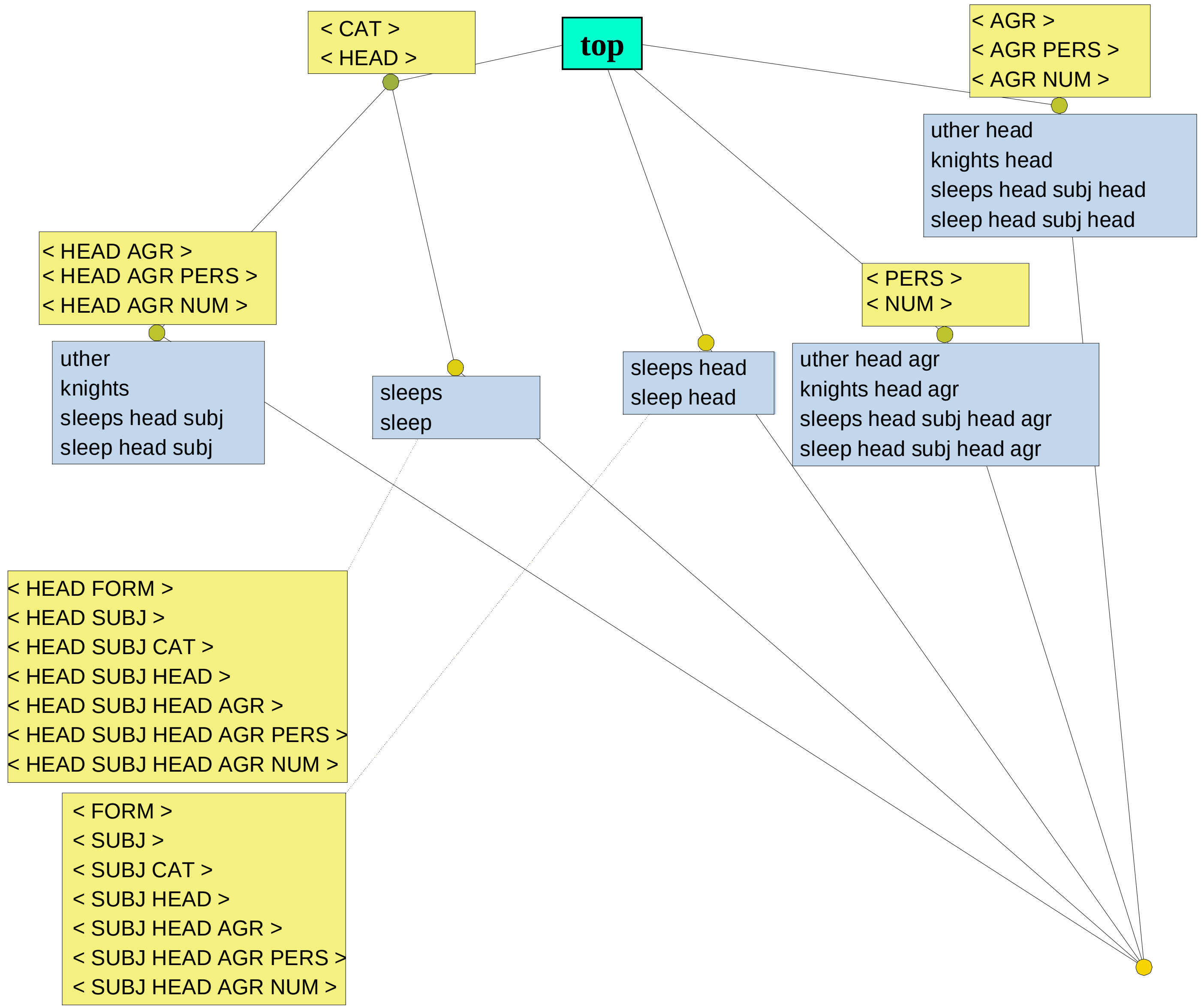
$$\text{sleeps} \rightarrow \begin{bmatrix} \text{CAT: vp} \\ \text{HEAD:} \begin{bmatrix} \text{FORM: finite} \\ \text{SUBJ:} \begin{bmatrix} \text{CAT: np} \\ \text{HEAD:} \begin{bmatrix} \text{AGR:} \begin{bmatrix} \text{PERS: third} \\ \text{NUM: sing} \end{bmatrix} \end{bmatrix} \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

$$\text{sleep} \rightarrow \begin{bmatrix} \text{CAT: vp} \\ \text{HEAD:} \begin{bmatrix} \text{FORM: finite} \\ \text{SUBJ:} \begin{bmatrix} \text{CAT: np} \\ \text{HEAD:} \begin{bmatrix} \text{AGR:} \begin{bmatrix} \text{PERS: third} \\ \text{NUM: plur} \end{bmatrix} \end{bmatrix} \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

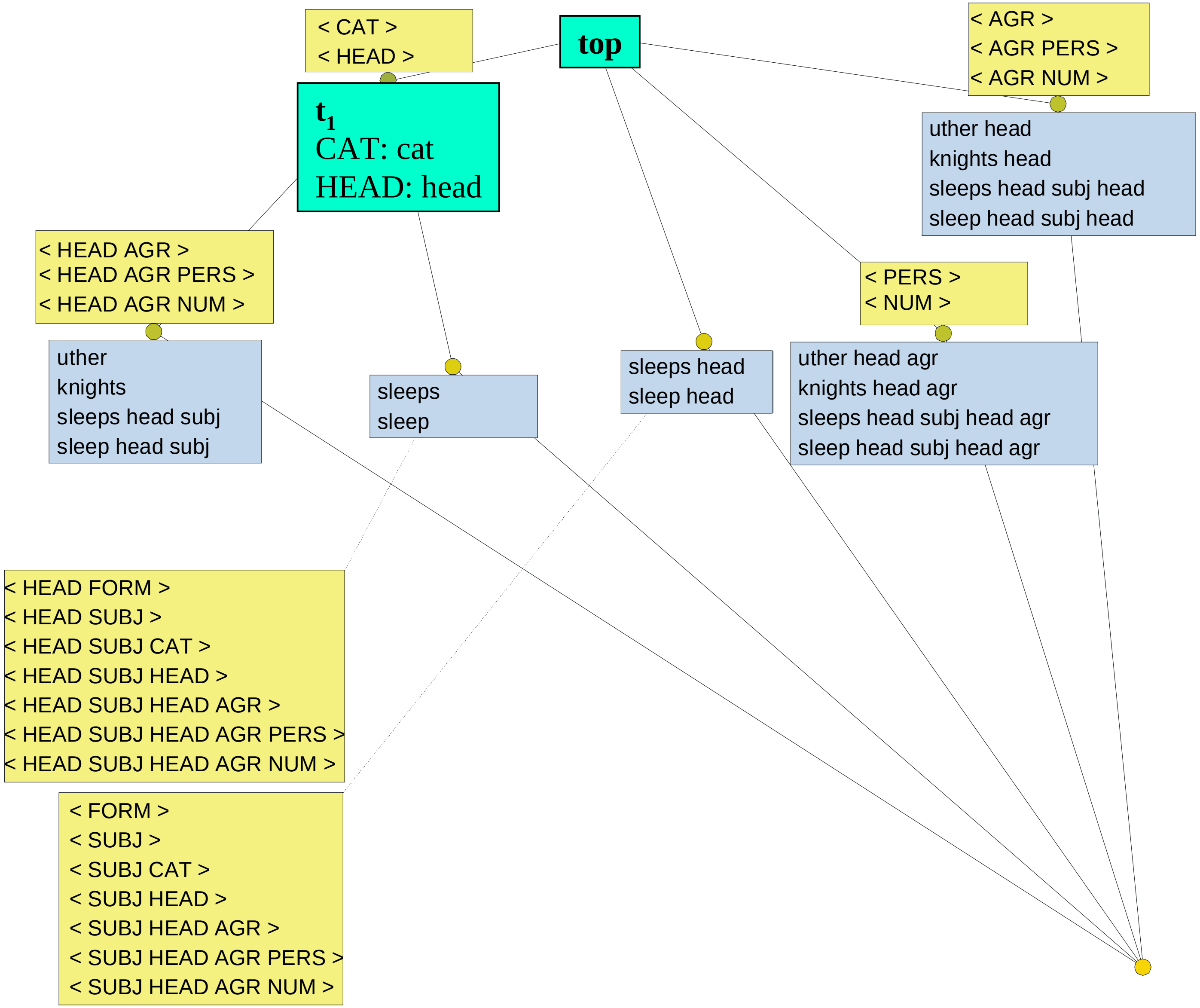
assigning types to concept nodes



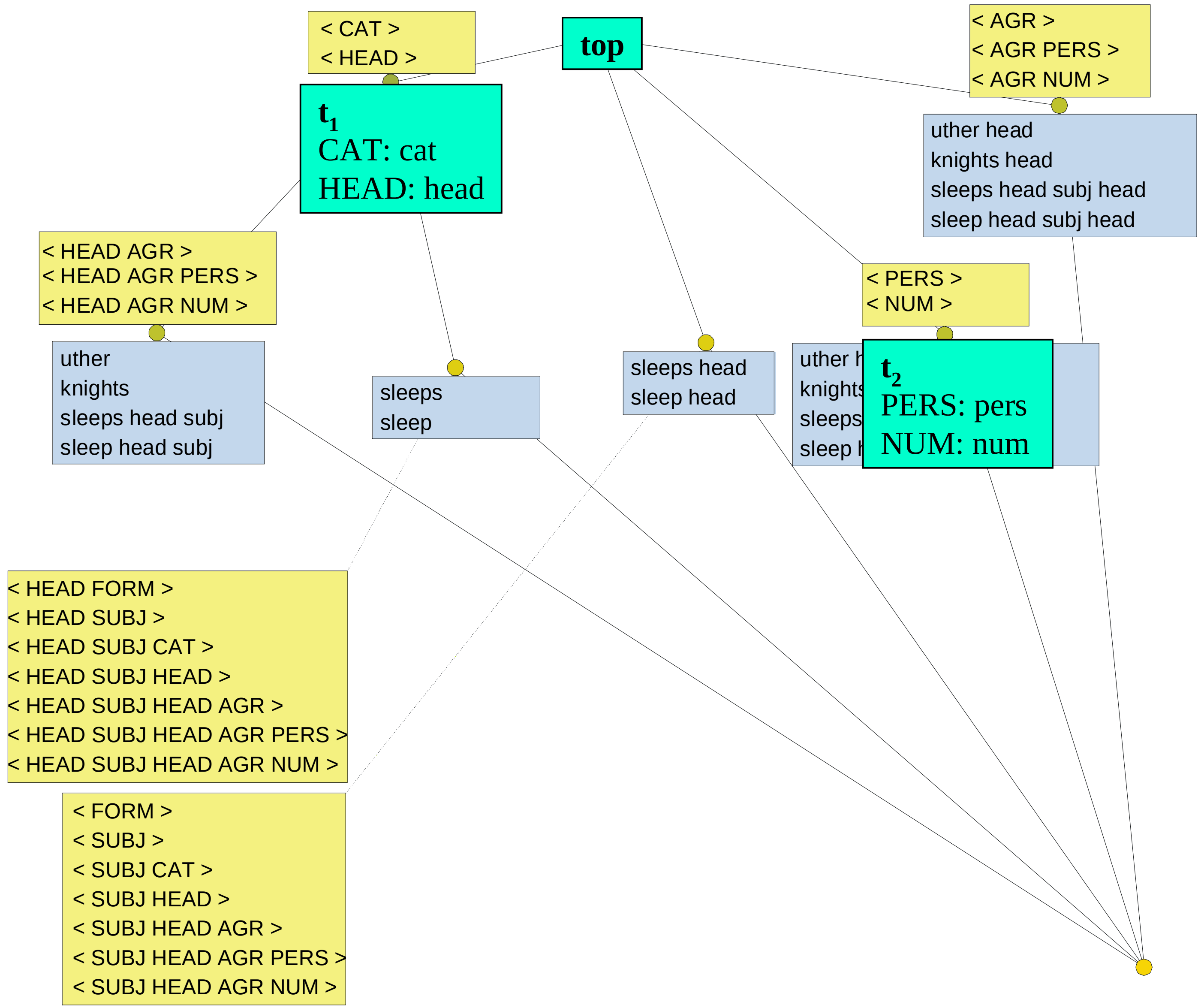
assigning types to concept nodes



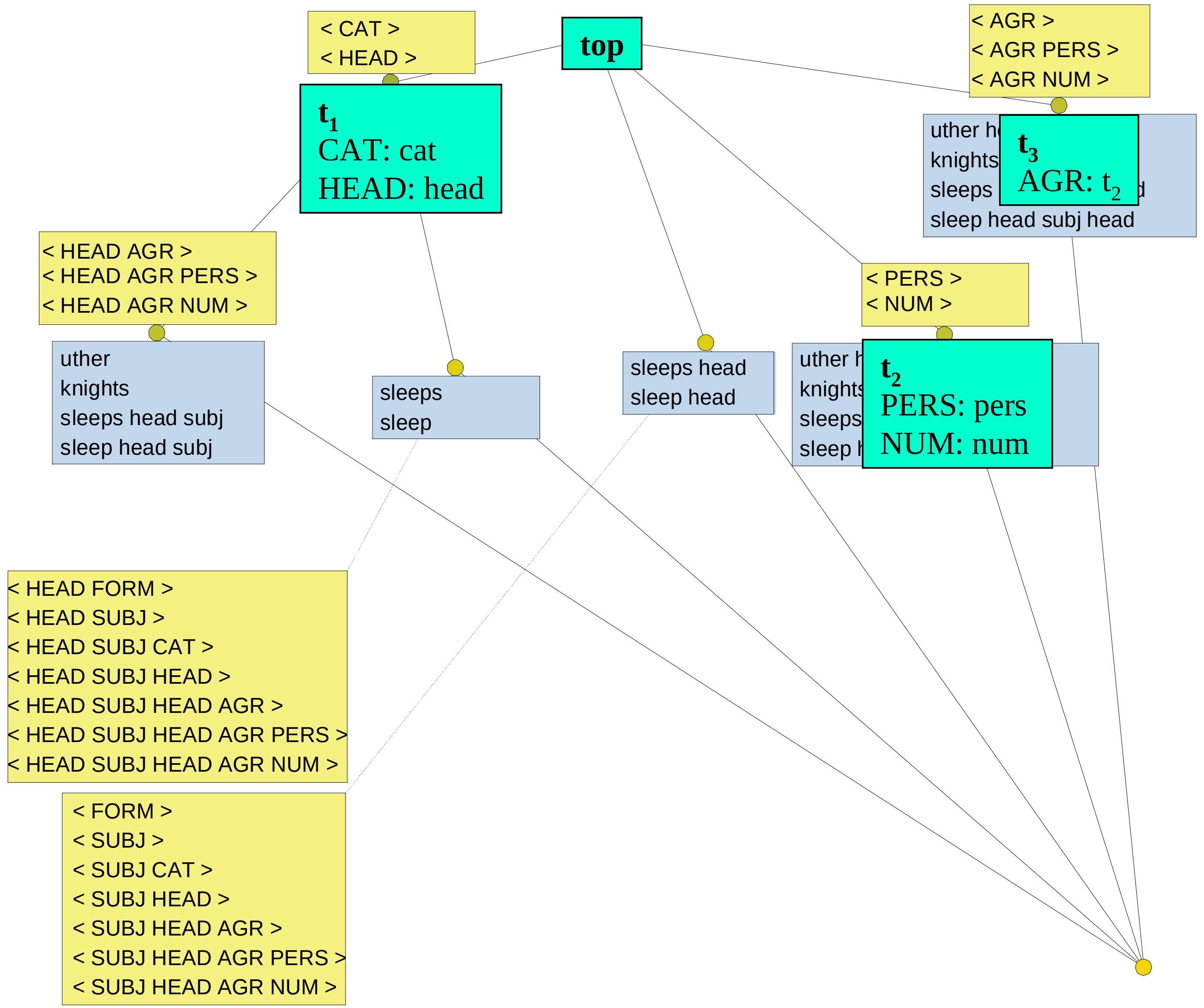
assigning types to concept nodes



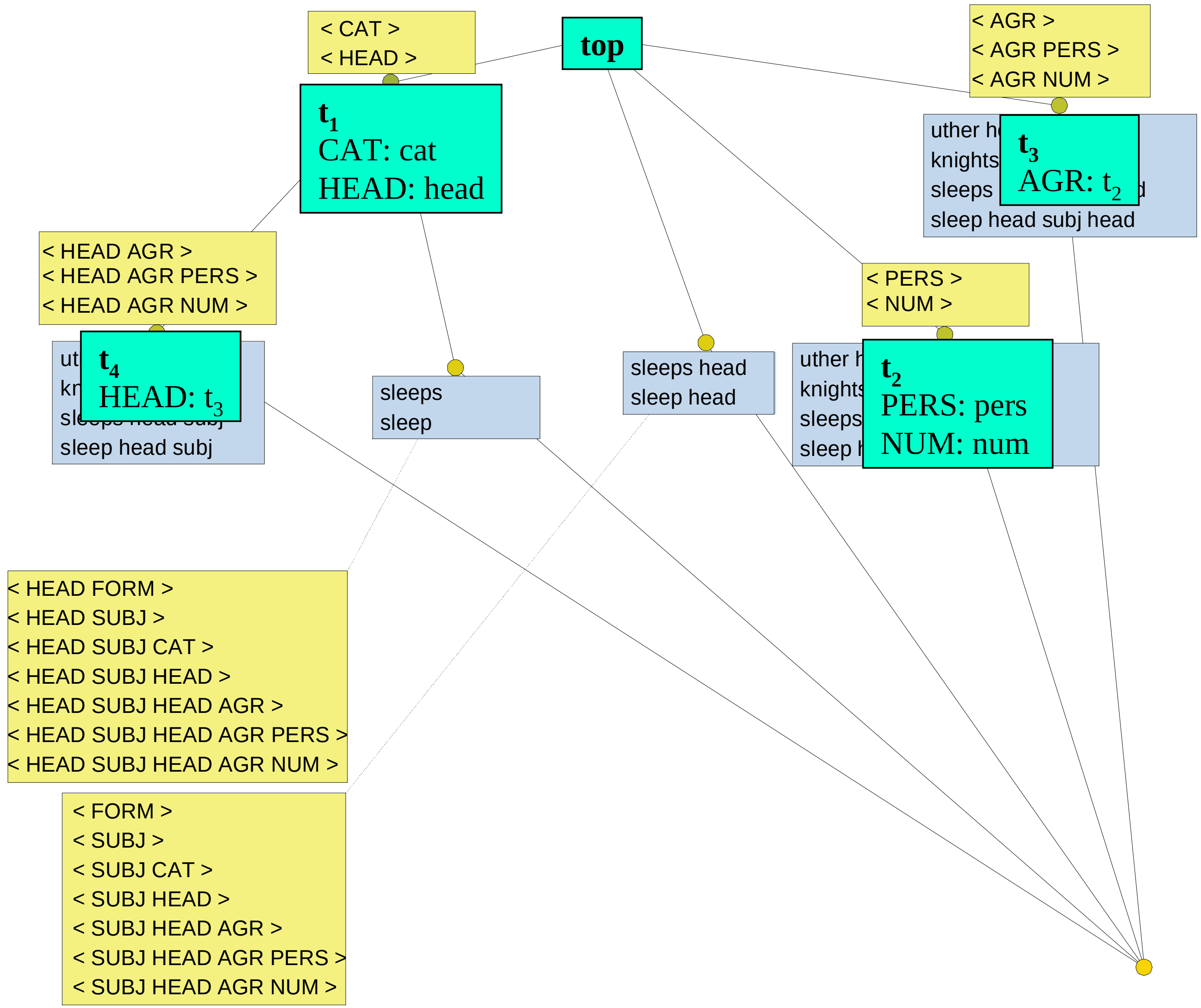
assigning types to concept nodes



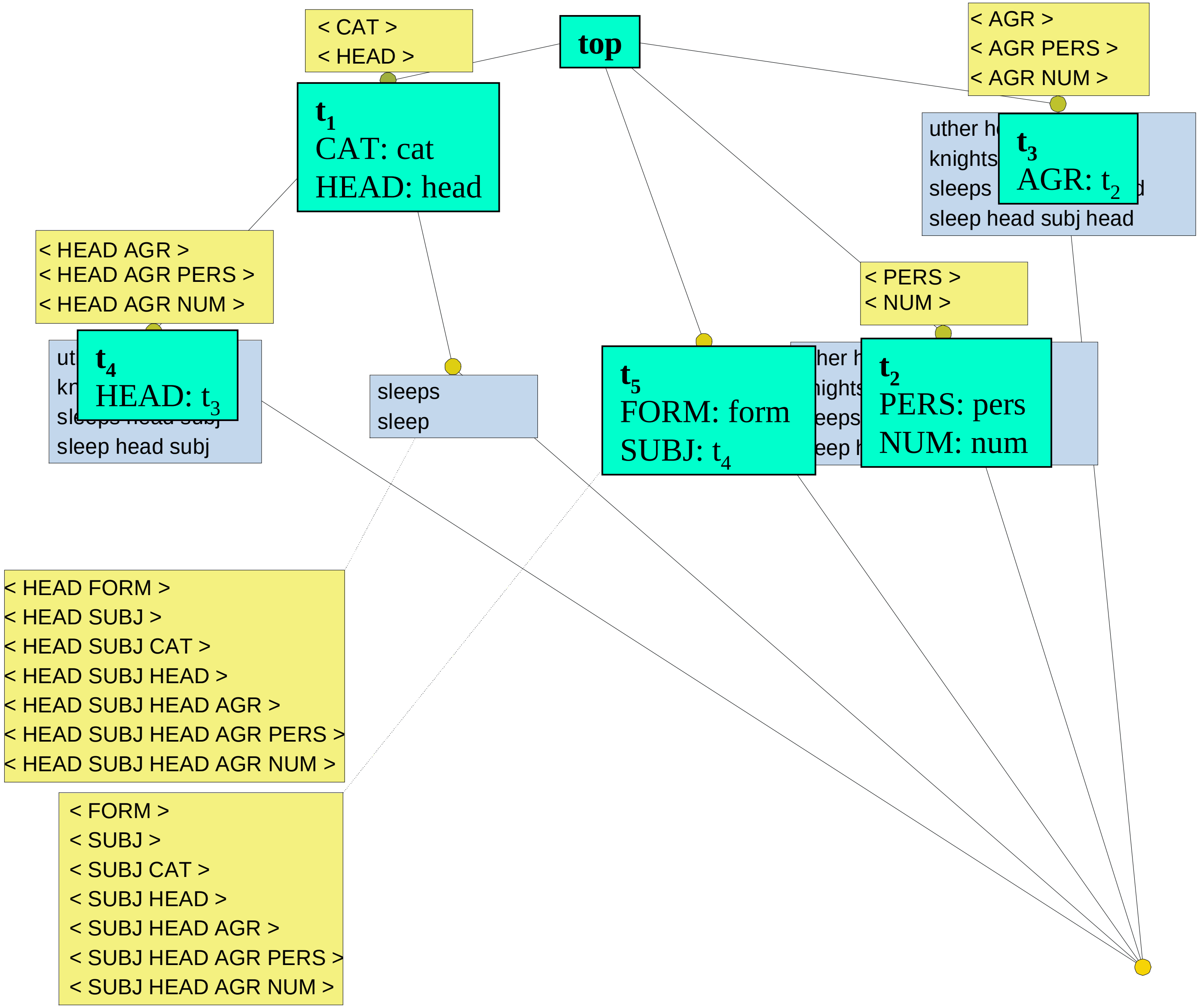
assigning types to concept nodes



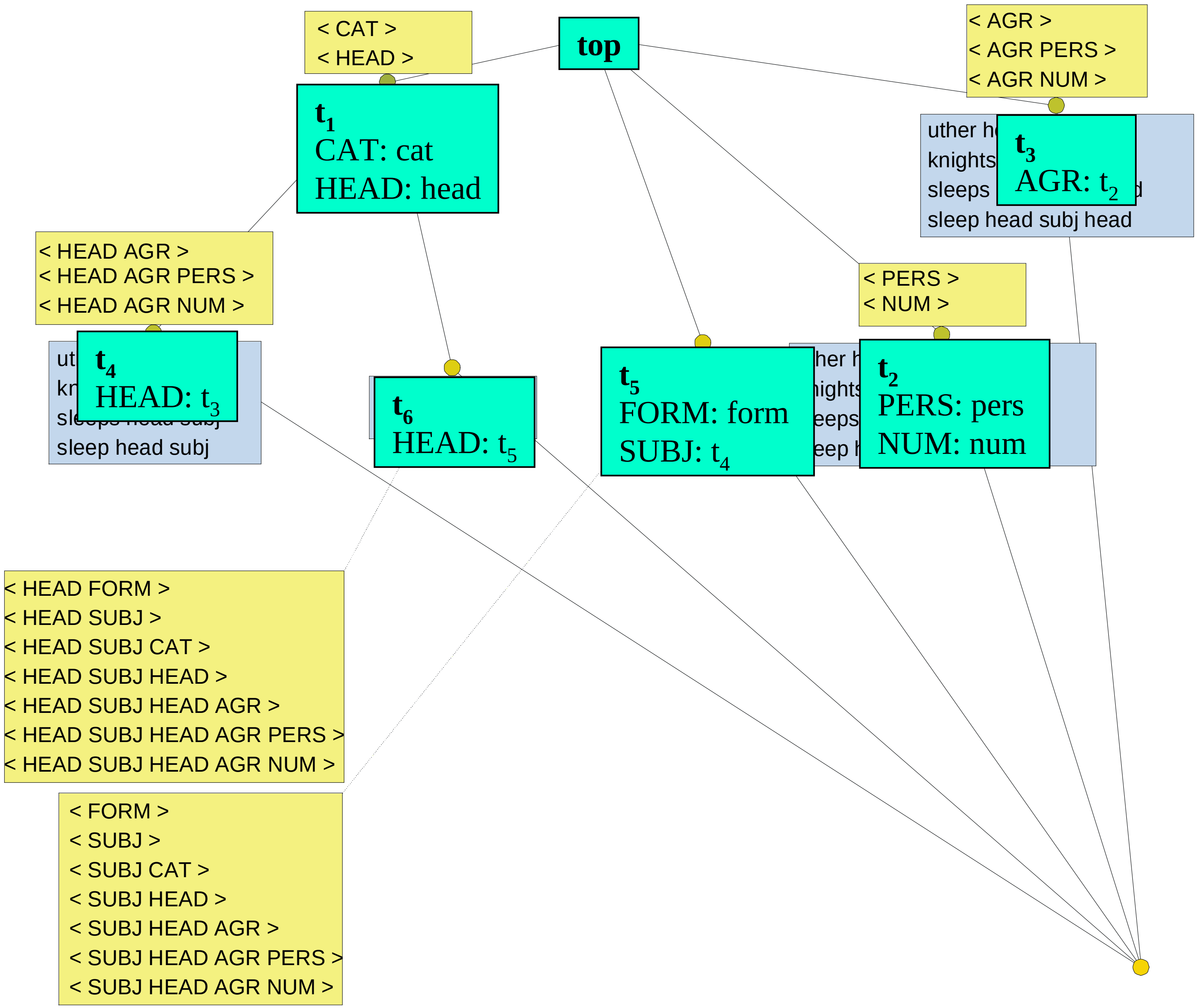
assigning types to concept nodes



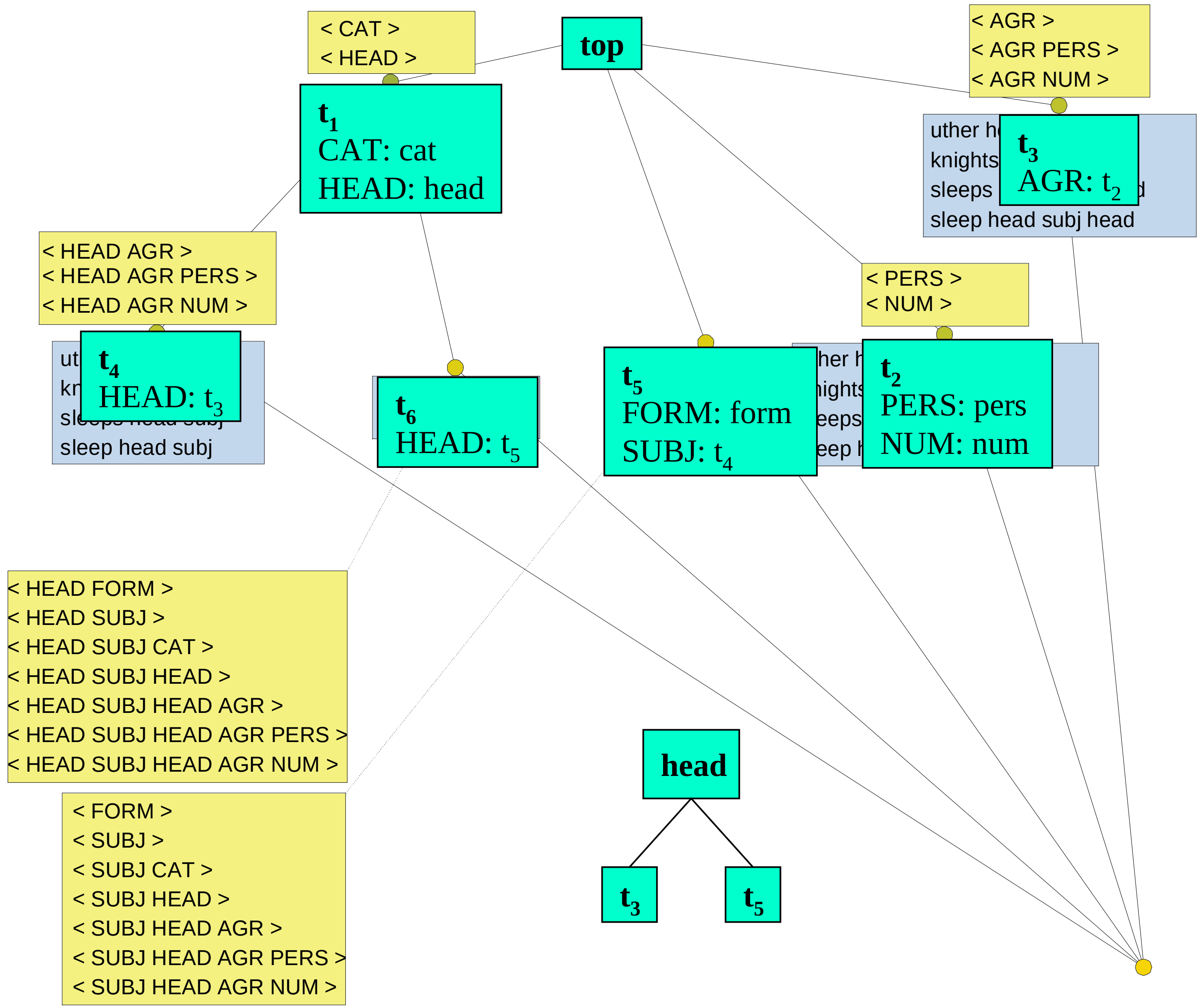
assigning types to concept nodes



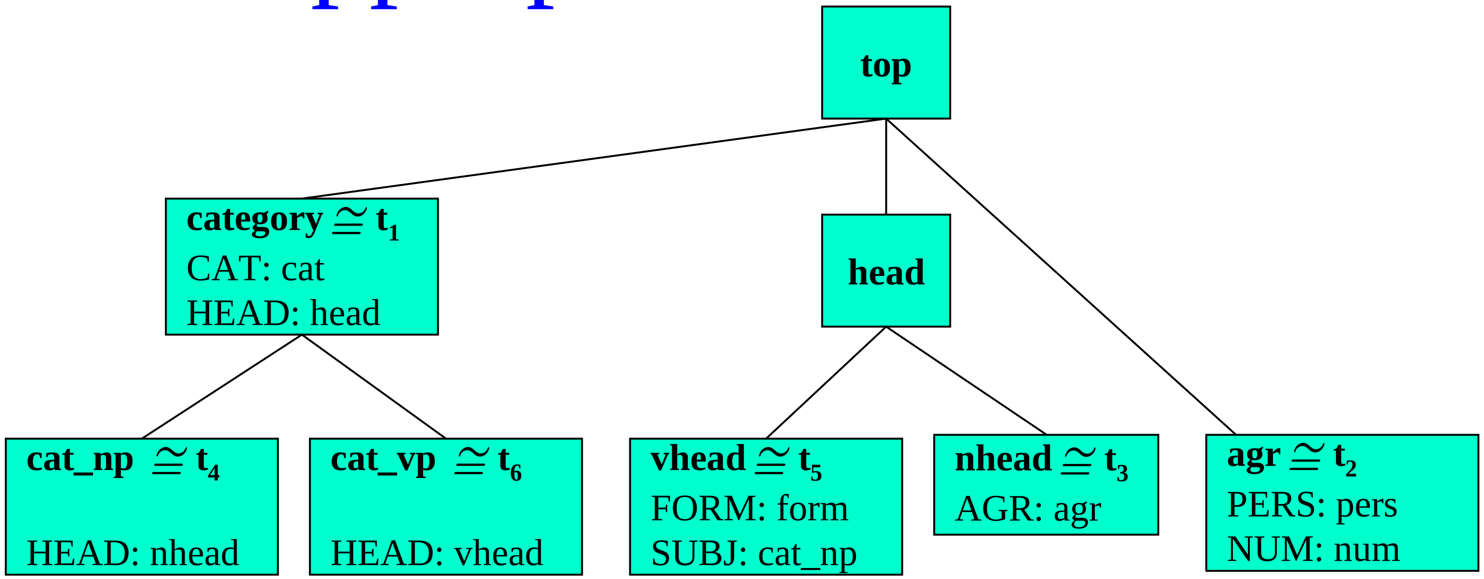
assigning types to concept nodes



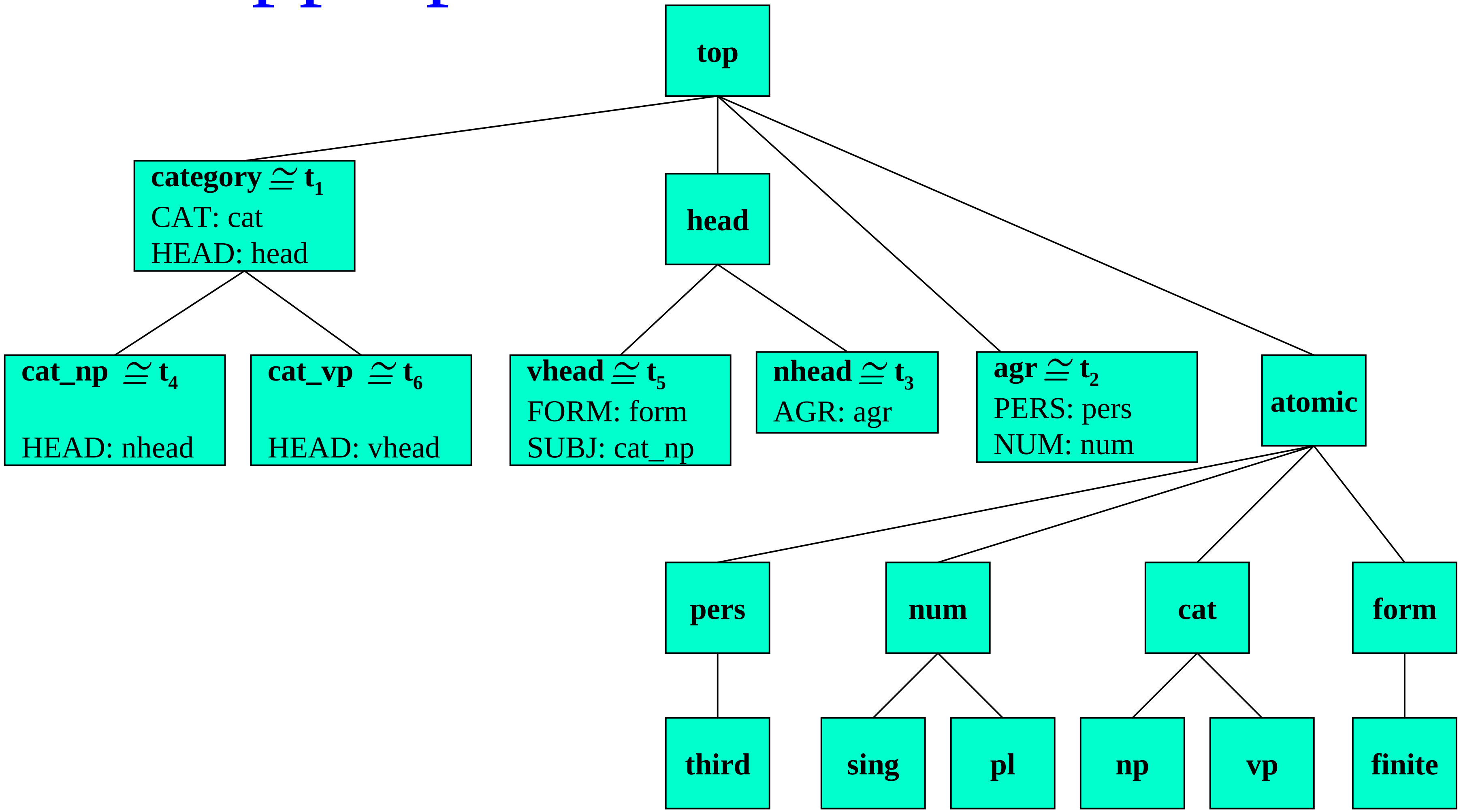
assigning types to concept nodes



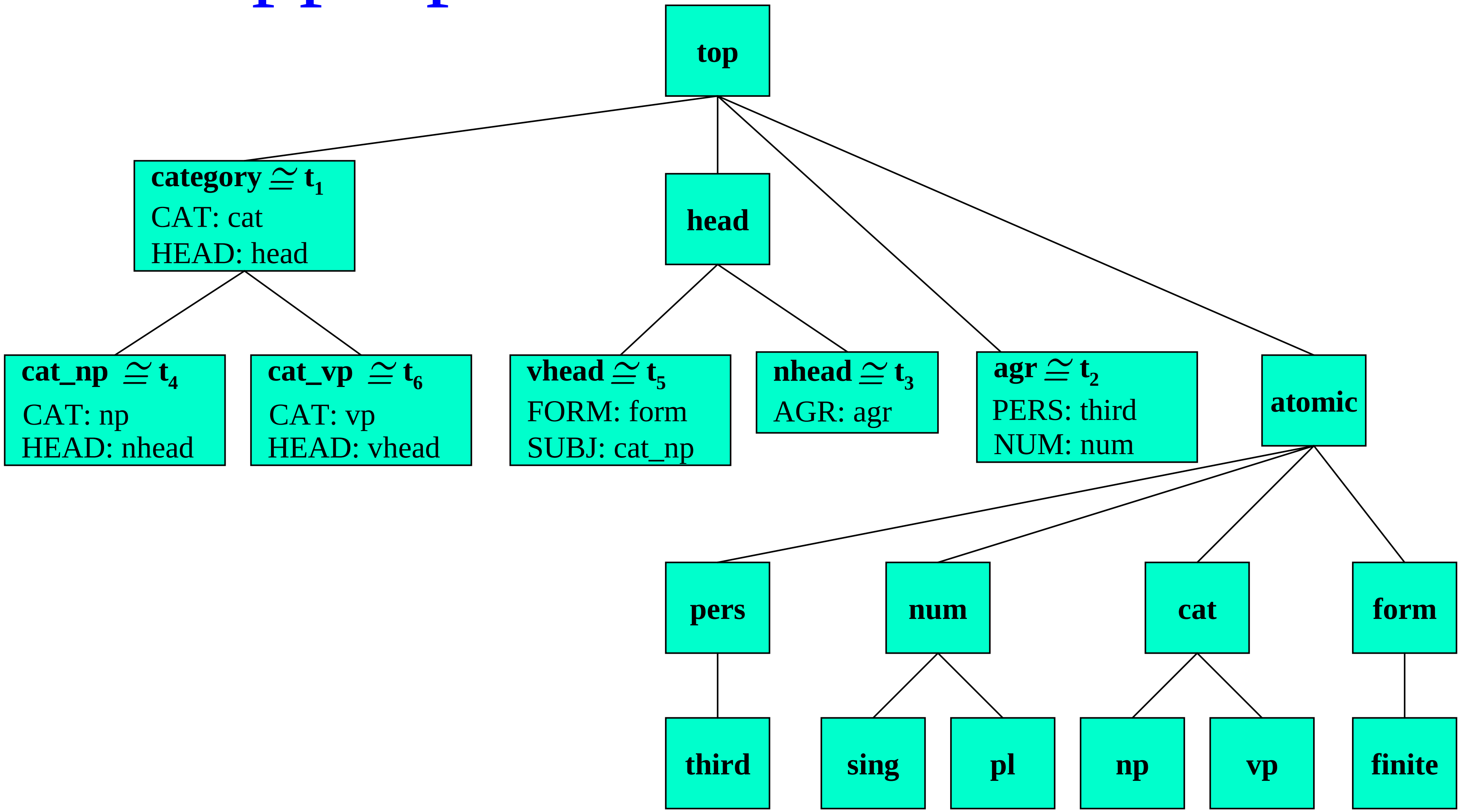
output: type hierarchy with appropriateness conditions



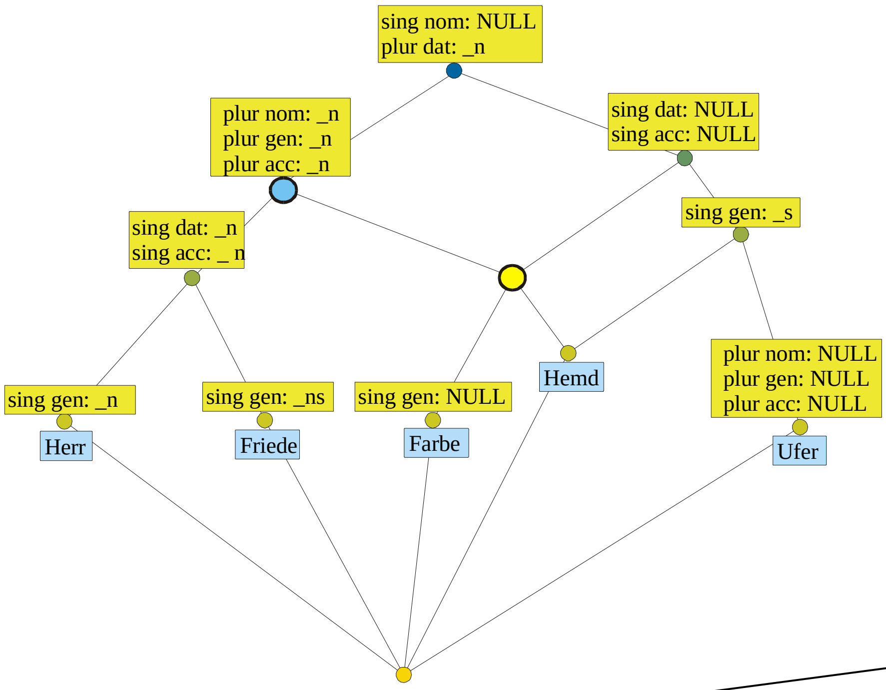
output: type hierarchy with appropriateness conditions



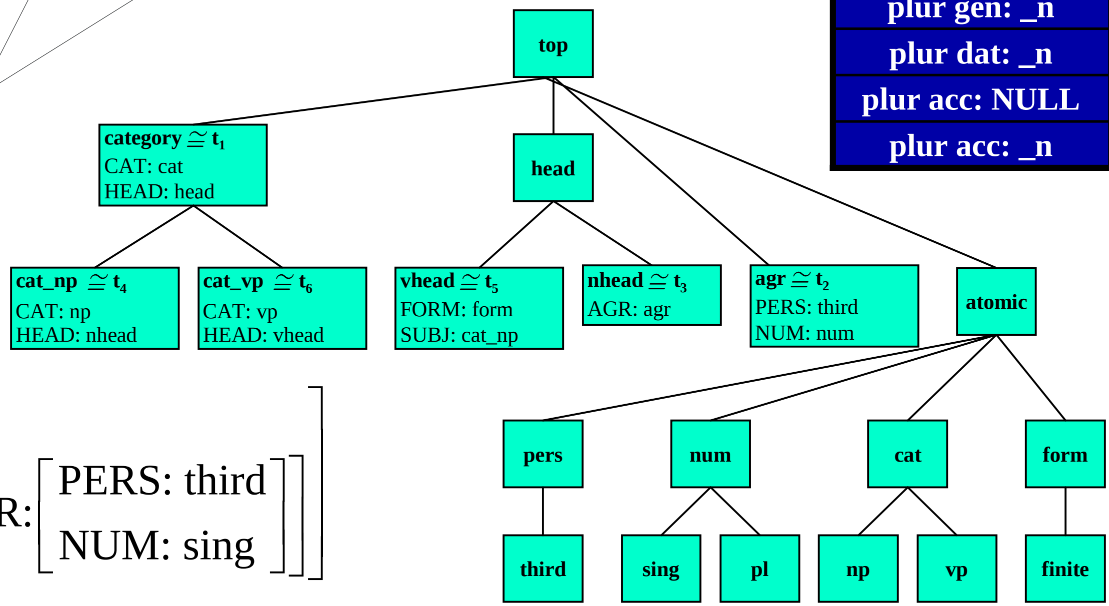
output: type hierarchy with appropriateness conditions



summary



	Herr	Friede	Hemd	Farbe	Ufer
sing nom: NULL	x	x	x	x	x
sing gen: NULL				x	
sing gen: _s			x		x
sing gen: _n	x				
sing gen: _ns		x			
sing dat: NULL			x	x	x
sing dat: _n	x	x			
sing acc: NULL			x	x	x
sing acc: _n	x	x			
plur nom: NULL					x
plur nom: _n	x	x	x	x	
plur gen: NULL					x
plur gen: _n	x	x	x	x	
plur dat: _n	x	x	x	x	x
plur acc: NULL					x
plur acc: _n	x	x	x	x	



$$\text{Uther} \rightarrow \begin{bmatrix} \text{CAT: np} \\ \text{HEAD:} \begin{bmatrix} \text{AGR:} \begin{bmatrix} \text{PERS: third} \\ \text{NUM: sing} \end{bmatrix} \end{bmatrix} \end{bmatrix}$$