On Frames and their components

Wiebke Petersen

Heinrich-Heine-Universität Düsseldorf
Collaborative Research Centre 991:
The Structure of Representations in Language, Cognition, and Science

`www.phil-fak.uni-duesseldorf.de/~petersen/`

SFB 991

17. April 2012, Tilburg

outline

outline

outline

outline

outline

outline

Frame hypothesis (Löbner 2012)

H1 The human cognitive system operates with one general format of representations.

H2 If the human cognitive system operates with one general format of representations, this format is essentially Barsalou frame.

A frame model is needed, that

- is sufficiently expressive to capture the diversity of representations
- sufficiently precise and restrictive in order to be testable

Aim: theory of concepts based on frames as concept representations

Düsseldorf frame group

- Semantics
- Syntax
- Computational Linguistics
- Psycholinguistics
- Neurolinguistics
- Neuroscience
- Cognitive Science
- Psychology
- Philosophy
- History of Science
- Philologies (German, Romanistic)

The task

Formalizing Barsalou's cognitive frame theory

- bridging the gap between cognitive linguistics and compositional semantics

Hypothesis: Frames can be defined as generalized typed feature structures (in the sense of Carpenter 1992)

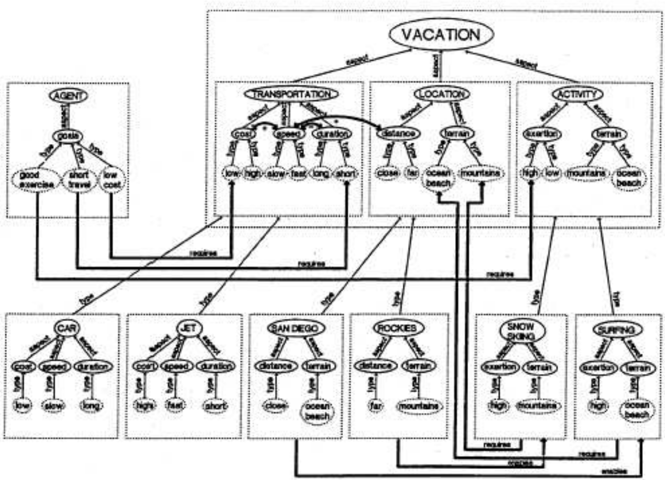
outline

frames

Barsalou (1992) *Frames, Concepts, and Conceptual Fields*

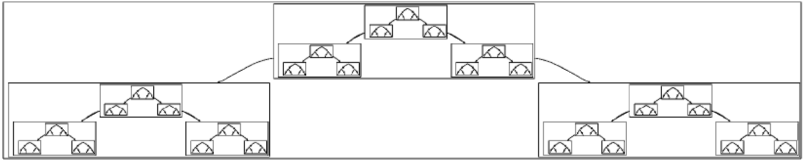
- Frames provide the fundamental representation of knowledge in human cognition.
- At their core, frames contain **attribute-value sets**.
- Frames further contain a variety of relations.
  - **Constraints**

Example: vacation frame with constraints (Barsalou 1992)

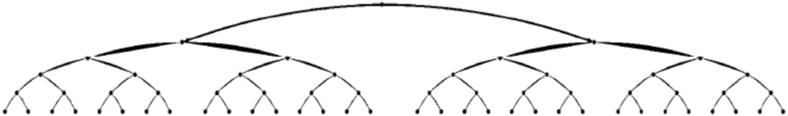


Unlimited recursion in frames

Self-similarity in Barsalou's frames (attributes are frames):



Recursion in classical feature structure theories:



feature structures

typed feature structure

$$\begin{bmatrix} \text{phrase} & \\ \text{HEAD:} & \begin{bmatrix} \text{noun} & \\ \text{AGR:} & \begin{bmatrix} \text{agr} & \\ \text{PERS:} & 3 \\ \text{NUM:} & \text{pl} \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

untyped feature structure

$$\begin{bmatrix} \text{CAT:} & \text{phrase} \\ \text{HEAD:} & \begin{bmatrix} \text{CAT:} & \text{noun} \\ \text{AGR:} & \begin{bmatrix} \text{PERS:} & 3 \\ \text{NUM:} & \text{pl} \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

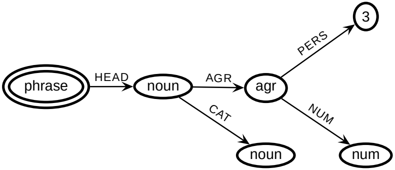
feature structures

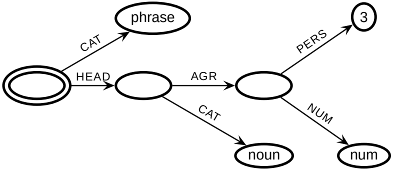
typed feature structure

$$\begin{bmatrix} \textit{phrase} & \\ \text{HEAD}: & \begin{bmatrix} \textit{noun} & \\ \text{AGR}: & \begin{bmatrix} \textit{agr} & \\ \text{PERS}: & 3 \\ \text{NUM}: & \textit{pl} \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

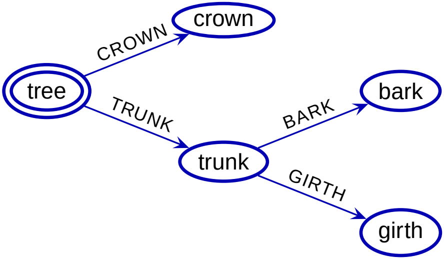
untyped feature structure

$$\begin{bmatrix} \text{CAT}: & \textit{phrase} \\ \text{HEAD}: & \begin{bmatrix} \text{CAT}: & \textit{noun} \\ \text{AGR}: & \begin{bmatrix} \text{PERS}: & 3 \\ \text{NUM}: & \textit{pl} \end{bmatrix} \end{bmatrix} \end{bmatrix}$$





frames as generalized feature structures



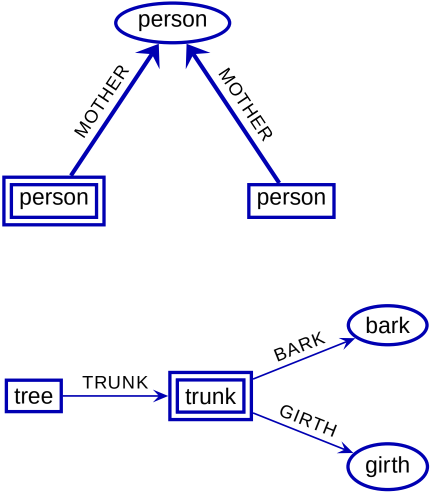
$$\begin{bmatrix} \textit{tree} & \\ \text{CROWN}: & \textit{crown} \\ \text{TRUNK}: & \begin{bmatrix} \textit{trunk} & \\ \text{BARK}: & \textit{bark} \\ \text{GIRTH}: & \textit{girth} \end{bmatrix} \end{bmatrix}$$

feature structures (Carpenter 1992)

feature structures
are connected directed graphs with

- one central node
- nodes labeled with types
- arcs labeled with attributes
- no node with two outgoing arcs with the same label
- and such that each node can be reached from the central node via directed arcs.

frames as generalized feature structures



Frames (Petersen 2007)

Frames
are connected directed graphs with

- one central node
- nodes labeled with types
- arcs labeled with attributes
- no node with two outgoing arcs with the same label

Open argument nodes are marked as rectangular nodes.

Frames are unrooted feature structures.

Formal Definitions

Definition (Frames)

Given a set TYPEof types and a finite set ATTR of attributes. A *frame* is a tuple $F = (Q, \bar{q}, \delta, \theta)$ where:

- Q is a finite set of nodes,
- $\bar{q} \in Q$ is the central node,
- δ : ATTR $\times Q \rightarrow Q$ is the partial *transition function*,
- $\theta : Q \rightarrow$ TYPE is the total *node typing function*,

such that the underlying graph (Q, E) with edge set $E = \{\{q_1, q_2\} | \exists a \in$ ATTR $: \delta(a, q_1) = q\}$ is connected.

Definition (Subsumption)

A frame $F_1 = \langle Q_1, \bar{q}_1, \delta_1, \theta_1 \rangle$ subsumes a frame $F_2 = \langle Q_2, \bar{q}_2, \delta_2, \theta_2 \rangle$ ($F \sqsubseteq F'$) iff there is a total function $h : Q_1 \rightarrow Q_2$ with

- $h(\bar{q}_1) = \bar{q}_2$,
- $\forall q \in Q_1 : \theta_1(q) \sqsubseteq \theta_2(h(q))$,
- if $\delta_1(f, q)$ is defined, then $h(\delta_1(f, q)) = \delta_2(f, h(q))$.

(Carpenter 1992)

Definition (Equivalence)

Two frames F_1 and F_2 are equivalent ($F_1 \sim F_2$), if $F_1 \sqsubseteq F_2$ and $F_2 \sqsubseteq F_1$.

Definition (Subsumption)

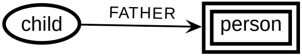
A frame $F_1 = \langle Q_1, \bar{q}_1, \delta_1, \theta_1 \rangle$ subsumes a frame $F_2 = \langle Q_2, \bar{q}_2, \delta_2, \theta_2 \rangle$ ($F \sqsubseteq F'$) iff there is a total injective function $h : Q_1 \to Q_2$ with

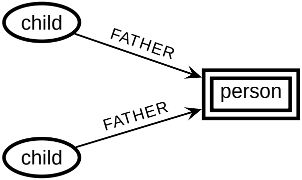
- $h(\bar{q}_1) = \bar{q}_2$,
- $\forall q \in Q_1 : \theta_1(q) \sqsubseteq \theta_2(h(q))$,
- if $\delta_1(f, q)$ is defined, then $h(\delta_1(f, q)) = \delta_2(f, h(q))$.

Definition (Equivalence)

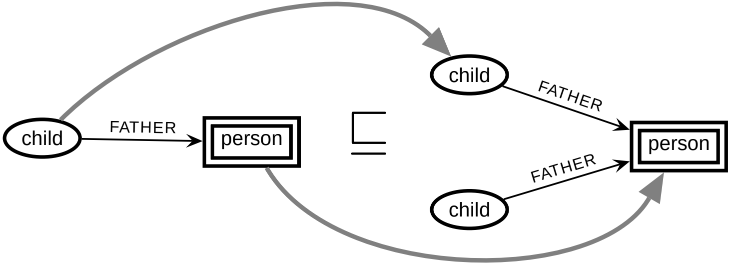
Two frames F_1 and F_2 are equivalent ($F_1 \sim F_2$), if $F_1 \sqsubseteq F_2$ and $F_2 \sqsubseteq F_1$.

Adaption of subsumption relation

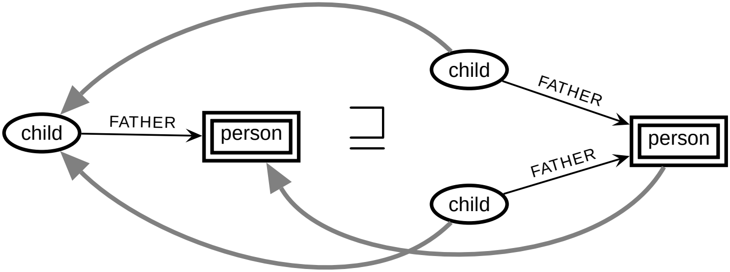




Adaption of subsumption relation



Adaption of subsumption relation

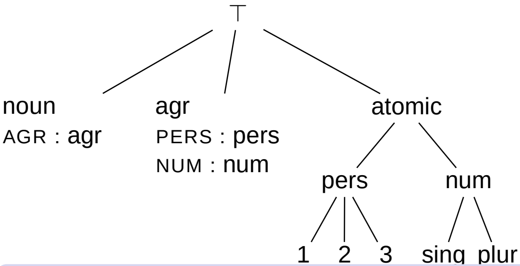


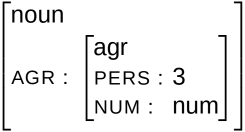
why typed frames and type signatures?

modeling convention (Carpenter 1992:34)

- The nodes of a feature structure are taken to represent objects, and we assume that every node is labeled with a type symbol which represents the most specific conceptual class to which the object is known to belong.
- An arc between two nodes indicates that the object represented by the source node has a feature, represented by a feature symbol, which has a value represented by the target node.
- We think of our types as organizing feature structures into natural classes.

why typed frames and type signatures?



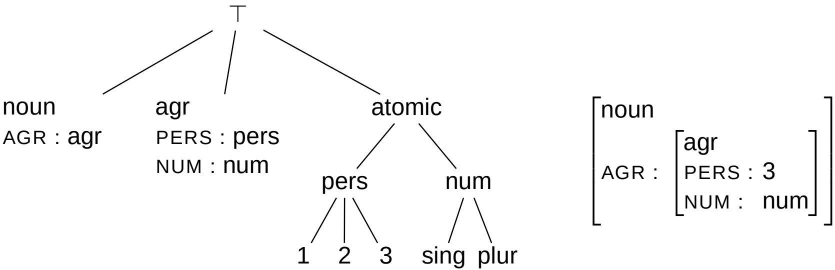


type signature

A type signature consists of a type hierarchy, $\langle T, \sqsubseteq \rangle$, a finite set of attributes ATTR and an **appropriateness specification**, i.e. a partial function, Approp : $\text{ATTR} \times T \to T$ that respects:

- attribute introduction (each attribute is introduced at a unique most general type)
- upward closure / right monotonicity (inheritance of appropriateness conditions)

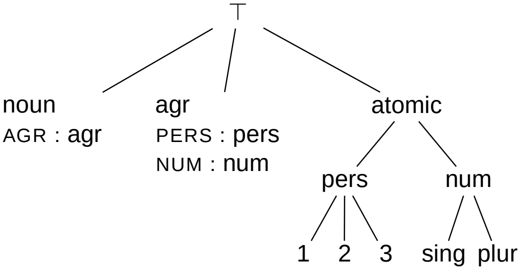
why typed frames and type signatures?



Type signatures

- capture hierarchical relations
- capture generalizations
- express constraints
- enable underspecified frames

why typed frames and type signatures?



$$\begin{bmatrix} \text{noun} & \\ \text{AGR :} & \begin{bmatrix} \text{agr} & \\ \text{PERS :} & 3 \\ \text{NUM :} & \text{num} \end{bmatrix} \end{bmatrix}$$

However,

- redundancy in attribute and type labels
- status of types is not clear (Carpenter 1992: types represent conceptual classes)
- frames and types: two means of expressing concepts?

possible solutions

1st perspective:

The distinction between the attribute set and the type set is artificial. The attribute set should be taken as a subset of the type set.

$\mathcal{ATTR} \subseteq \mathcal{TYPE}$

2nd perspective:

The distinction between the attribute set and the type set is artificial. Types are definable on the basis of attribute domain and ranges.

$\mathcal{TYPE} \rightsquigarrow \mathcal{ATTR}$

outline

attributes in frames

Barsalou, 1992

“I define an attribute as a **concept** that describes an aspect of at least some category member.”
“Values are subordinate concepts of an attribute.”

Guarino, 1992: *Concepts, attributes and arbitrary relations*

“We define attributes as **concepts** having an associate relational interpretation, allowing them to act as conceptual components as well as concepts on their own.”

excursus: interpretation of functional concepts

denotational interpretation

A functional concept denotes a set of entities:

$$\delta : \mathcal{R} \rightarrow 2^{\mathcal{U}}$$

$\delta(\text{mother}) = \{m \mid m \text{ is the mother of someone}\}$

relational interpretation

A functional concept has also a relational interpretation:

$$\varrho : \mathcal{R} \rightarrow 2^{\mathcal{U} \times \mathcal{U}}$$

$\varrho(\text{mother}) = \{(p, m) \mid m \text{ is the mother of } p\}$

consistency postulate (Guarino, 1992)

Any value of an relationally interpreted functional concept is also an instance of the denotation of that concept.

If $(p, m) \in \varrho(\text{mother})$, then $m \in \delta(\text{mother})$.

excursus: interpretation of functional concepts

denotational interpretation

A functional concept denotes a set of entities:

$$\delta : \mathcal{R} \rightarrow 2^{\mathcal{U}}$$

$\delta(\text{mother}) = \{m \mid m \text{ is the mother of someone}\}$

relational interpretation

A functional concept has also a relational interpretation:

$$\varrho : \mathcal{R} \rightarrow 2^{\mathcal{U} \times \mathcal{U}}$$

$\varrho(\text{mother}) = \{(p, m) \mid m \text{ is the mother of } p\}$

consistency postulate (Guarino, 1992)

Any value of an relationally interpreted functional concept is also an instance of the denotation of that concept.

If $(p, m) \in \varrho(\text{mother})$, then $m \in \delta(\text{mother})$.

excursus: interpretation of functional concepts

denotational interpretation

A functional concept denotes a set of entities:

$$\delta : \mathcal{R} \to 2^{\mathcal{U}}$$

$\delta(\text{mother}) = \{m \mid m \text{ is the mother of someone}\}$

relational interpretation

A functional concept has also a relational interpretation:

$$\varrho : \mathcal{R} \to 2^{\mathcal{U} \times \mathcal{U}}$$

$\varrho(\text{mother}) = \{(p, m) \mid m \text{ is the mother of } p\}$

consistency postulate (Guarino, 1992)

Any value of an relationally interpreted functional concept is also an instance of the denotation of that concept.

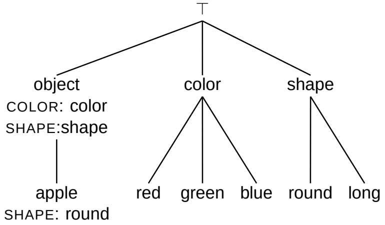
If $(p, m) \in \varrho(\text{mother})$, then $m \in \delta(\text{mother})$.

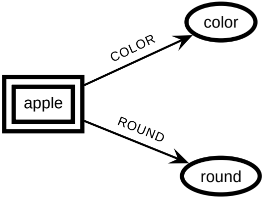
attributes in frames

Thesis:

Attributes in frames are relationally interpreted functional concepts!

- attributes are not frames themselves
- attributes are unstructured
- the possible values of an attribute are subconcepts of the denotationally interpreted functional concept





denotational and relational interpretation

denotational

type
mother

relational

attribute
MOTHER : $\delta(\text{person}) \rightarrow \delta(\text{mother})$

attributes in frames (1st perspective)

thesis:

Attributes in frames are relationally interpreted functional concepts!

consequence (1):

Frames decompose concepts into relationally interpreted functional concepts!

consequence (2):

The distinction between the attribute set and the type set is artificial. The attribute set should be taken as a subset of the type set: $\mathcal{ATTR} \subseteq \mathcal{TYPE}$.

attributes in frames (1st perspective)

thesis:

Attributes in frames are relationally interpreted functional concepts!

consequence (1):

Frames decompose concepts into relationally interpreted functional concepts!

consequence (2):

The distinction between the attribute set and the type set is artificial. The attribute set should be taken as a subset of the type set: $\mathcal{ATTR} \subseteq \mathcal{TYPE}$.

attributes in frames (1st perspective)

thesis:

Attributes in frames are relationally interpreted functional concepts!

consequence (1):

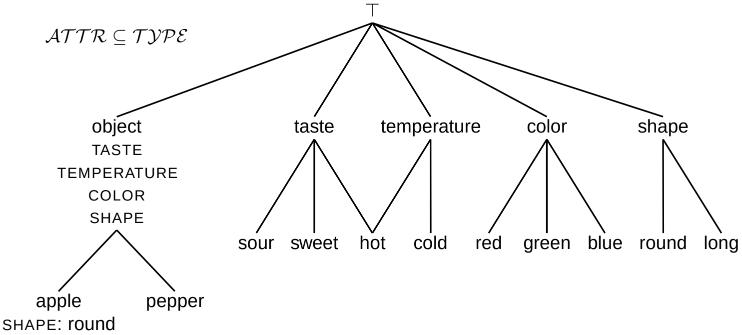
Frames decompose concepts into relationally interpreted functional concepts!

consequence (2):

The distinction between the attribute set and the type set is artificial. The attribute set should be taken as a subset of the type set: $\mathcal{ATTR} \subseteq \mathcal{TYPE}$.

type signature and minimal upper attributes (1st perspective)

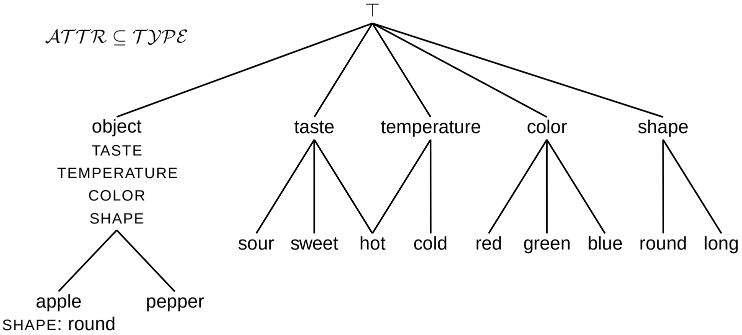
$\mathcal{ATTR} \subseteq \mathcal{TYPE}$



Barsalou, 1992: *Frames, Concepts, and Conceptual Fields*

“I define an attribute as a **concept** that describes an aspect of at least some category member.”
“Values are subordinate concepts of an attribute.”

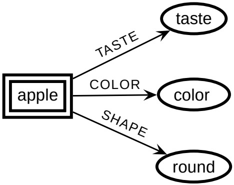
type signature and minimal upper attributes (1st perspective)



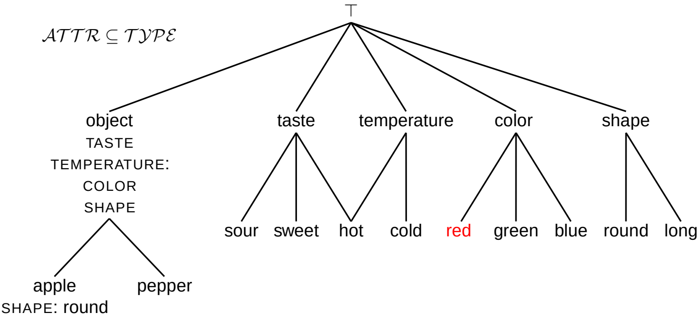
Definition

A minimal upper attribute of a type is a minimal element of the set of upper attributes of the type. Where an upper attribute of a type is an attribute which is a supertype of the type.

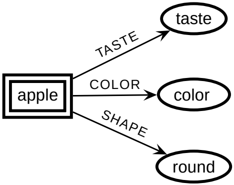
red apple

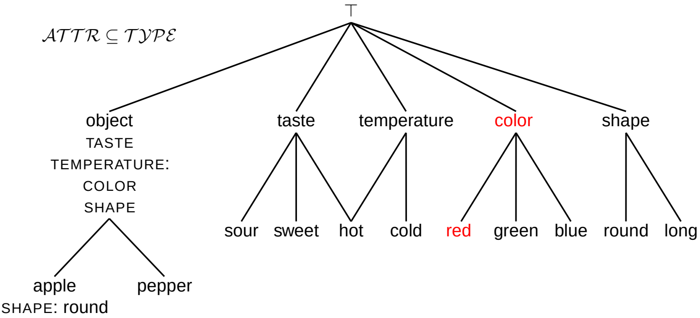


$\mathcal{ATTR} \subseteq \mathcal{TYPE}$

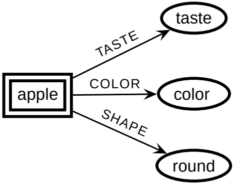


red apple

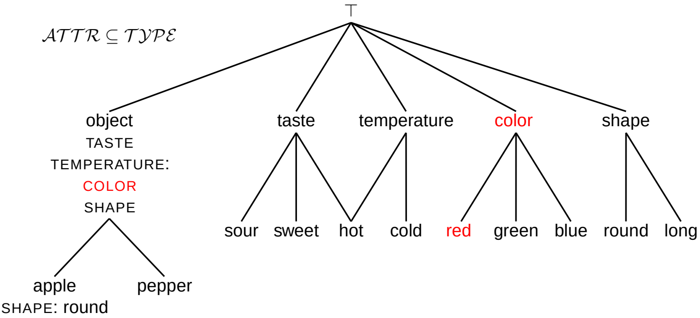



ATTR ⊆ TYPE
⊤
object
TASTE
TEMPERATURE:
COLOR
SHAPE
taste
temperature
color
shape
sour
sweet
hot
cold
red
green
blue
round
long
apple
SHAPE: round
pepper

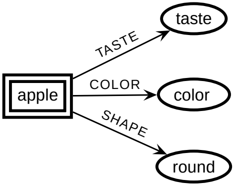
red apple


apple
TASTE
taste
COLOR
color
SHAPE
round

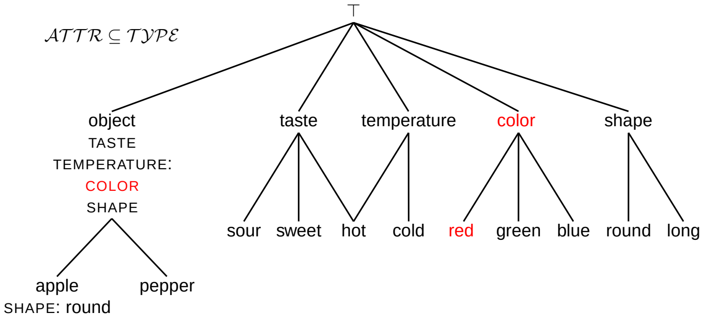
$\mathcal{ATTR} \subseteq \mathcal{TYPE}$



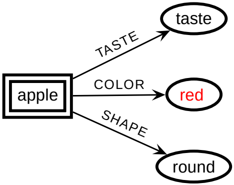
red apple



$\mathcal{ATTR} \subseteq \mathcal{TYPE}$

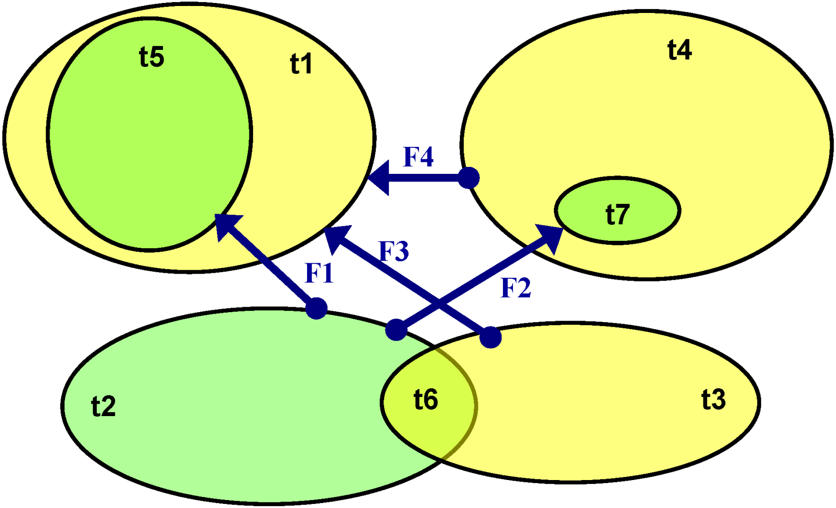


red apple

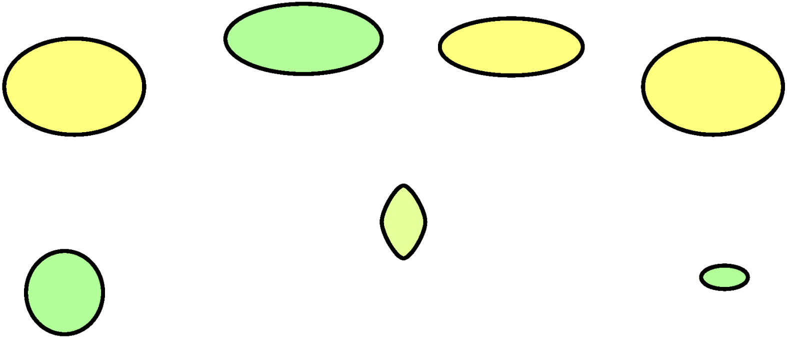


outline

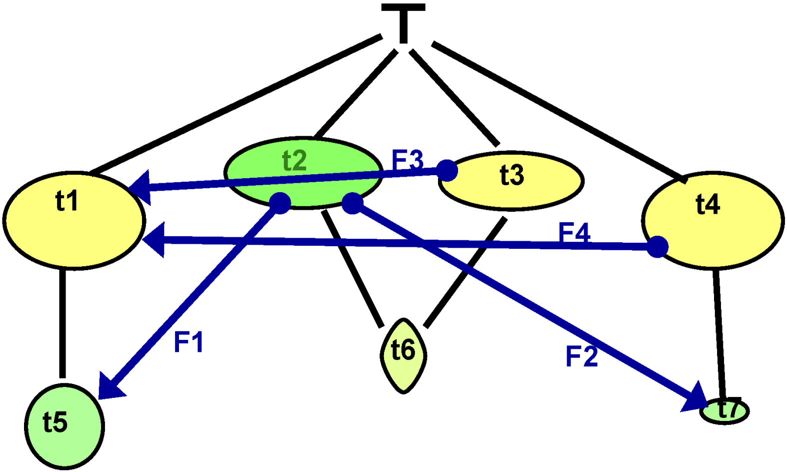
radically attribute-oriented perspective



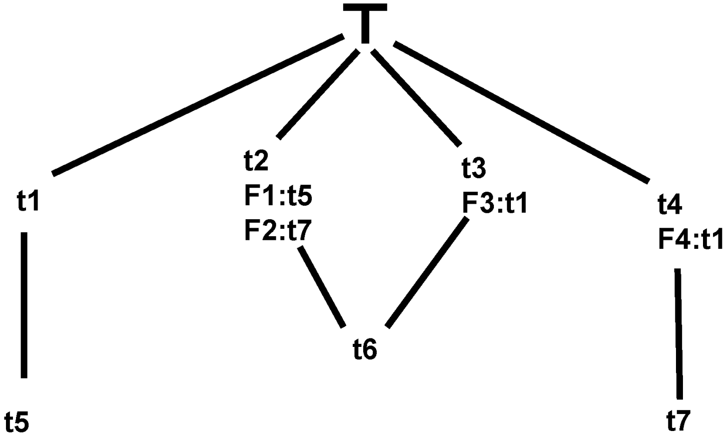
radically attribute-oriented perspective



radically attribute-oriented perspective



radically attribute-oriented perspective



attribute space

Definition

An attribute space is a tuple $(\mathcal{U}, \mathcal{A})$ consisting of a universe set $\mathcal{U}$ and a finite set of attributes $\mathcal{A} \subseteq 2^{\mathcal{U} \times \mathcal{U}}$ which are partial functions (i.e., if $(x, y), (x, z) \in \mathcal{A}$ then $y = z$).

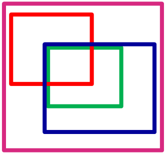
Attribute composition: the set of paths Π in $(\mathcal{U}, \mathcal{A})$ is the set of all finite attribute sequences. If $\pi = a_1 a_2 \ldots a_n$ we write $\pi(x)$ for $a_n(\ldots(a_2(a_1(x)))\ldots)$.

types

Definition

Given an attribute space $(\mathcal{U}, \mathcal{A})$ and a set S of relevant subsets of $\mathcal{U}$. The set of types $\mathcal{T}$ is $\mathcal{T} = 2^S / \sim$ with:

- $\forall \varphi, \psi \subseteq S : \varphi \sim \psi$ iff $\bigcap \varphi = \bigcap \psi$.



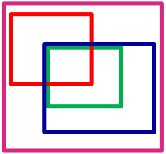
$$\{\square, \square\} \sim \{\square, \square, \square\}$$

types

Definition

The type hierarchy $(\mathcal{T}, \sqsupseteq)$ is defined by

- $[\varphi] \sqsubseteq [\psi]$ iff $\bigcup[\varphi] \subseteq \bigcup[\psi]$
- or equivalently (extensionally): $[\varphi] \sqsubseteq [\psi]$ iff $\bigcap \varphi \supseteq \bigcap \psi$



$$[\{\square, \square\}] \sqsubseteq [\{\square, \square\}]$$

The type hierarchy forms a lattice
(top element $[\emptyset]$, bottom element $[S]$).

relevant subsets (example)

$S = \mathcal{A}_d \cup \Pi_r$ with:

- $\mathcal{A}_d$ is the set of attribute domains:
  $\mathcal{A}_d = \{a_d | a \in \mathcal{A}\}$ where $a_d = \{x \in \mathcal{U} | \exists u \in \mathcal{U} : a(x) = u\}$
- Π_r is the set of path ranges:
  $\Pi_r = \{\pi_r | \pi \in \Pi\}$ where $\pi_r = \{x \in \mathcal{U} | \exists u \in \mathcal{U} : \pi(u) = x\}$

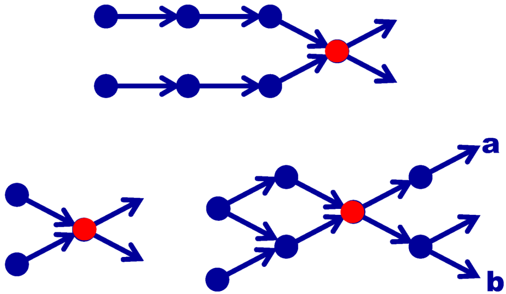
Definition

The type signature on $(\mathcal{U}, \mathcal{A})$ with relevant subset set $\mathcal{A}_d \cup \Pi_r$ is $(\mathcal{T}, \sqsupseteq, \text{Approp})$ where $\text{Approp} : \mathcal{A} \times \mathcal{T} \rightarrow \mathcal{T}$ is the appropriateness condition defined by:

- $\text{Approp}(a, [\varphi]) = [\{(\pi a)_r | \pi_r \in \bigcup[\varphi]\}]$

adjusting the granularity of the type hierarchy

The granularity of the type hierarchy can be easily adjusted by adapting the set of relevant subsets *S*



examples of attribute-defined relevant subsets

FCAType

- FCAType is a system for the automatic induction of type signatures from sets of untyped feature structures (i.e., sortal frames).
- It uses methods of Formal Concept Analysis (Ganter & Wille 1998).
- key idea: decomposition of feature structures into paths, path equations and path-value-pairs (note: attribute-based components).
- It can be straightforwardly adapted to general frames.

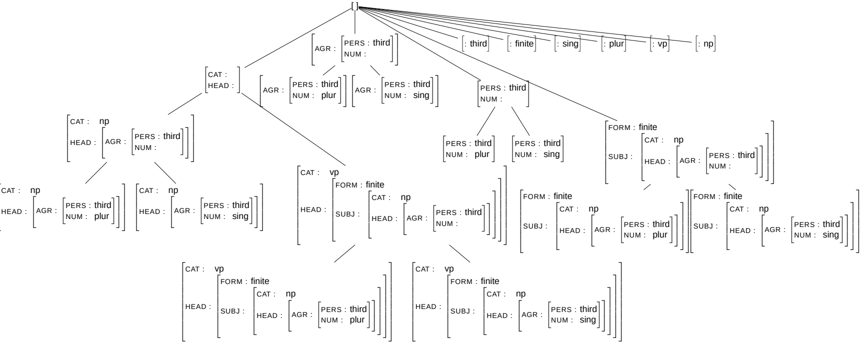
example input frames

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

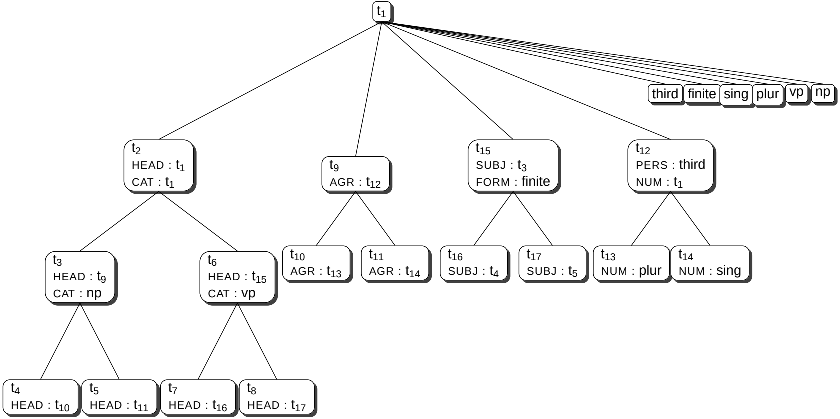
$$\text{knights} = \begin{bmatrix} \text{CAT:} & \text{np} \\ \text{HEAD:} & \begin{bmatrix} \text{AGR:} & \begin{bmatrix} \text{PERS:} & \text{third} \\ \text{NUM:} & \text{plur} \end{bmatrix} \end{bmatrix} \end{bmatrix} \qquad \text{sleep} = \begin{bmatrix} \text{CAT:} & \text{vp} \\ \text{HEAD:} &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

(taken from Shieber 1986)

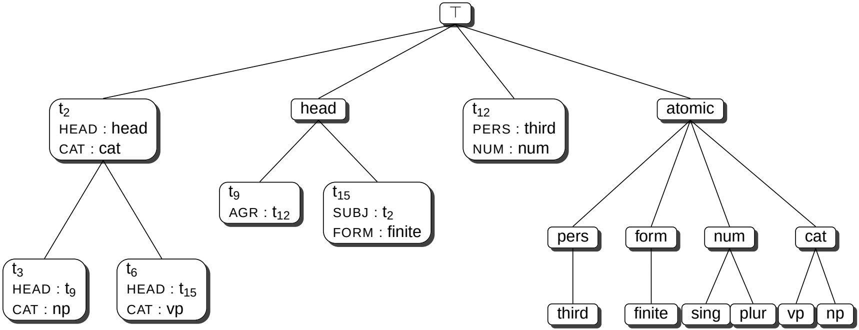
frame decomposition lattice



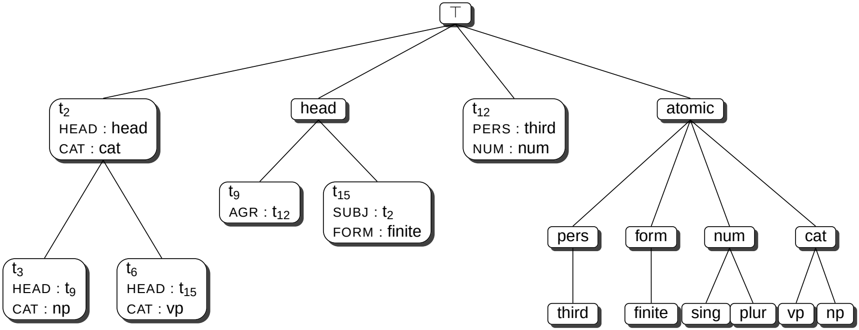
unfolded type signature



folded type signature



folded type signature



Carpenter 1992: "We think of our types as organizing feature structures into natural classes."

capturing constraints

The granularity of the type hierarchy can be adjusted to capture constraints:

- inverse images of (some) values (e.g. $\textsc{color}^{-1}$(red); "if a tomato is ripe, it is red)
- inverse images of value ranges (e.g. $\textsc{age}^{-1}(\leq 18)$; "a human under 18 is a child").
- attribute domains (e.g., SHAPE,SIZE; "if something has a shape, it has a size")
- path ranges (e.g., HAIR COLOR; "hair colors are restricted")
- path equations
- monotonic constraints ("the older a stamp is, the more expensive it is")
- …

outline

Düsseldorf frame model

We aim at a frame model that is

- powerful because of unlimited recursiveness (compare e.g. Fillmore frames)
- expressive because of type specifications
- precise because of restriction to functional attributes (compare e.g. semantic nets)
- formalized (mathematical definition and model-theoretic interpretation)
- empirical founded (evidence from cognitive science and psycholinguistics)

Why not traditional PL1 with truth-valued model theory?

Advantages of frame approach

- **transparent and preserving composition in frames**: the information of the parts is preserved, accumulated, and configurated
  (in truth-functional logic the meaning of the parts is not recoverable from the meaning of the whole)
- **variable freeness**: cognitive more adequate, information elements are related by attributes not by shared variables
- **no fixed arity of predicates**
- **no fixed argument order**

My project: Formal modeling of frames

subjects

1. frames in isolation
   - ontological status of frame elements
   - dynamic attributes
   - focus: space of attributes
2. frames in interaction
   - operations on frames (composition)
   - relations between frames (type shifts)
   - focus: space of frames
3. frame models of dynamic concepts
   - changes of attribute values in time
   - focus: linking object frames with the temporal domain

Aim

Frame-based cognitive semantics explaining both decompositional and compositional phenomena in a unified way

literature

- Barsalou, L. (1992): Frames, Concepts, and Conceptual Fields. In Lehrer and Kittay (eds.): Frames, Fields, and Contrasts.
- Carpenter, B. (1992): *The Logic of Typed Feature Structures*. Cambridge: CUP.
- Ganter, B. and Wille, R. (1998): Formal Concept Analysis: Mathematical Foundations. Berlin: Springer.
- Guarino, N. (1992): Concepts, attributes and arbitrary relations — some linguistic and ontological criteria for structuring knowledge bases. Data Knowl. Eng. 8, 249-261
- Petersen, W. (2007): Representation of Concepts as Frames. In: *The Baltic International Yearbook of Cognition, Logic and Communication*, Vol. 2, p. 151-170.
- Petersen, W. (2008): Type Signature Induction with FCAType. In: S. B. Yahia, E. Mephu Nguifo, R. Belohlavek (eds.): *Concept Lattices and Their Applications*. LNAI 4923, p. 276-281, Springer.
- Shieber, S. M. (1986): *An Introduction to Unification-Based Approaches to Grammar*. Stanford: CSLI Publications.

outline

concept classification

person, pope, house, verb, sun, Mary, wood, brother, mother, meaning, distance, spouse, argument, entrance

concept classification: relationality

non-relational	person, pope, house, verb, sun, Mary, wood
relational	brother, mother, meaning, distance, spouse, argument, entrance

Löbner

concept classification: uniqueness of reference

	non-unique reference	unique reference
non-relational	person, house, verb, wood	Mary, pope, sun
relational	brother, argument, entrance	mother, meaning, distance, spouse

Löbner

concept classification

	non-unique reference	unique reference
non-relational	**sortal concept**	**individual concept**
relational	**proper relational concept**	**functional concept**

Löbner

concept classification

	non-unique reference	unique reference
non-relational	**sortal concept**	**individual concept**
relational	**proper relational concept**	**functional concept**

Löbner

terminology

Definition

A node is a root of a frame if all other nodes can be reached from it by a path of directed arcs.

terminology

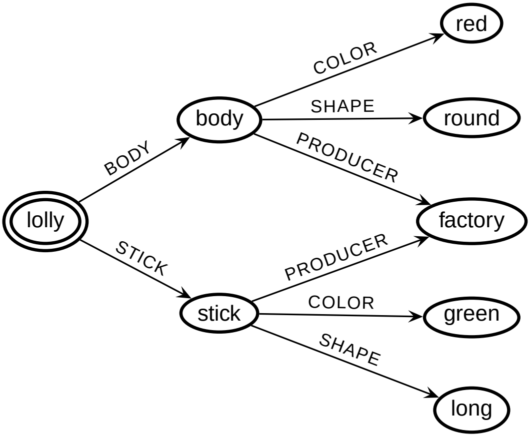
Definition

A node is a root of a frame if all other nodes can be reached from it by a path of directed arcs.

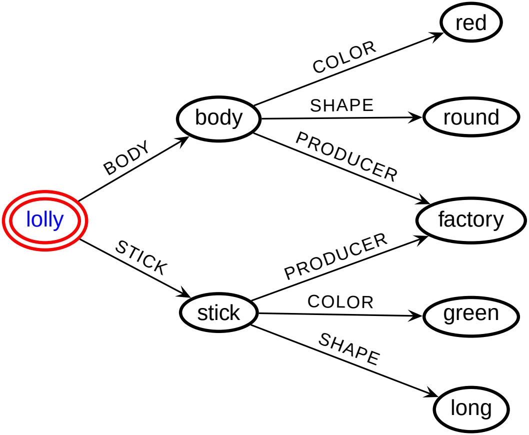
Definition

A node is a source if it has no incoming arc.

lolly-frame (sortal concept)

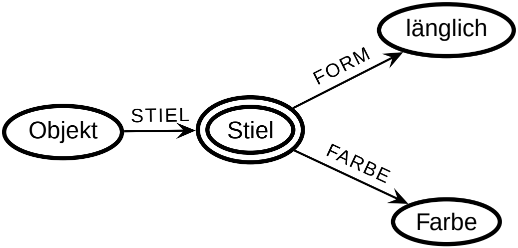


lolly-frame (sortal concept)

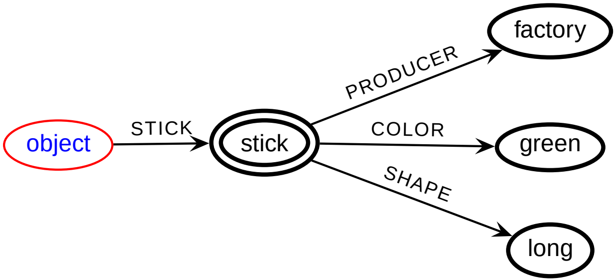


central node = root = source

stick-frame (functional concept)

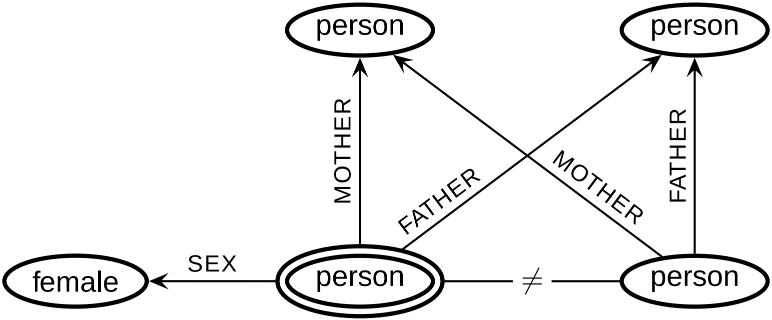


stick-frame (functional concept)

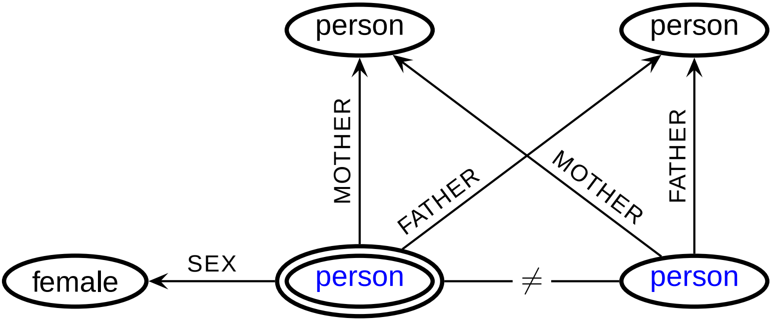


central node $\neq$ root $=$ source

sister-frame (proper relational concept)



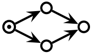
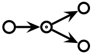
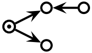
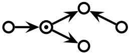
sister-frame (proper relational concept)



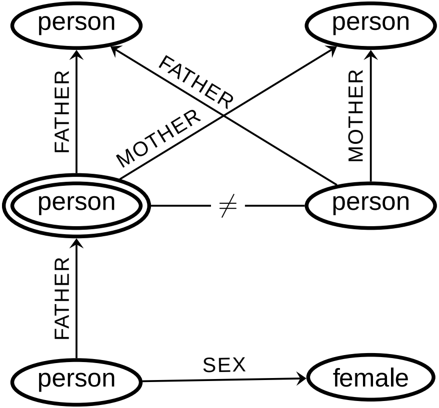
no root & central node = source

classification of acyclic frame graphs

C: central node, R: root, S: source

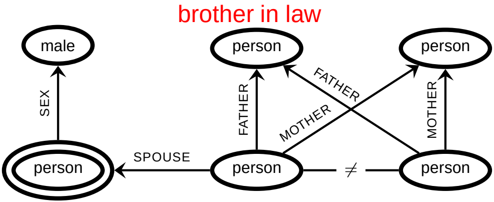
$C = R$	$C = S$	$\exists R$	$\exists S$	typical graph	frame class
$+$	$+$	$+$	$+$		sortal
$-$	$-$	$+$	$+$		functional
$-$	$+$	$-$	$+$		proper relational
$-$	$-$	$-$	$+$		???

4th frame class: not lexicalized?

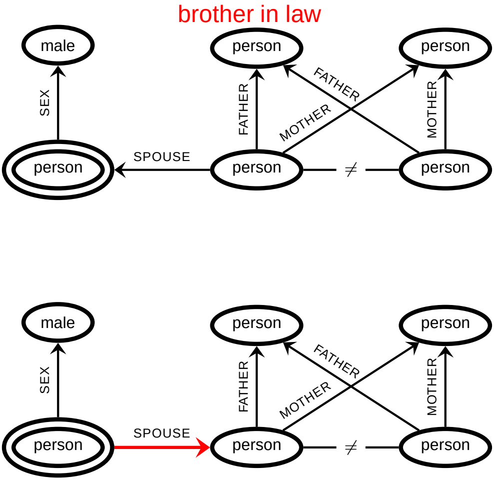


relational concept: father of a niece

4th frame class: not lexicalized?

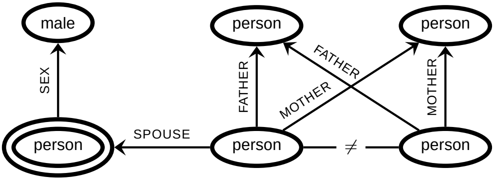


4th frame class: not lexicalized?

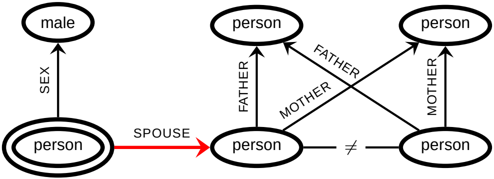


4th frame class: not lexicalized?

brother in law



“male person who is the spouse of someone who has a sibling”



“male person whose spouse has a sibling”

concept classification and frame graphs

relationality

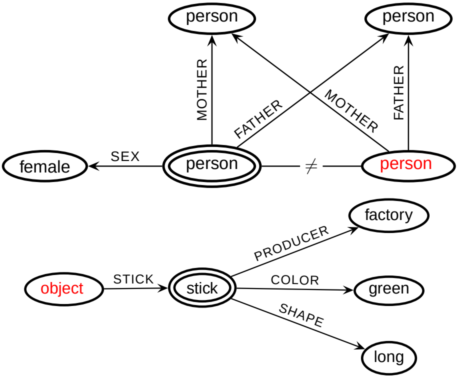
The arguments of relational concepts are modeled in frames as sources that are not identical to the central node.

functionality

The functionality of functional concepts is modeled by an incoming arc at the central node.

conclusion

The concept classification is reflected by the properties of the frame graphs.



concept classification and frame graphs

relationality

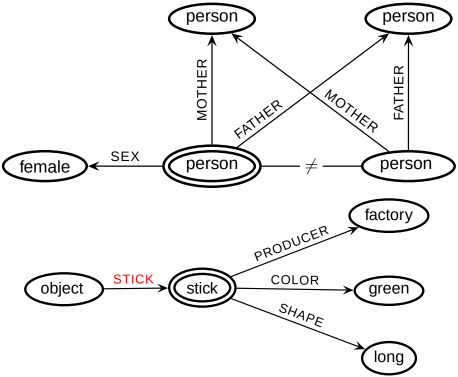
The arguments of relational concepts are modeled in frames as sources that are not identical to the central node.

functionality

The functionality of functional concepts is modeled by an incoming arc at the central node.

conclusion

The concept classification is reflected by the properties of the frame graphs.



concept classification and frame graphs

relationality

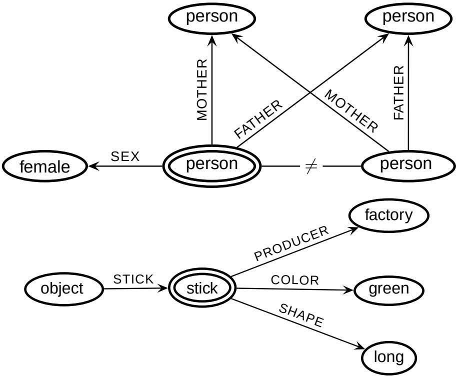
The arguments of relational concepts are modeled in frames as sources that are not identical to the central node.

functionality

The functionality of functional concepts is modeled by an incoming arc at the central node.

conclusion

The concept classification is reflected by the properties of the frame graphs.

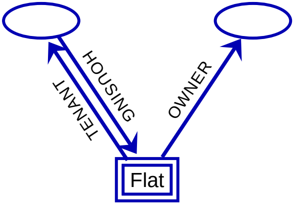


outline

type shifts: non-relational → relational

sortal	individual
proper relational	functional

↓

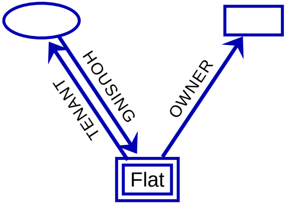


sortal concept *flat:*
“Many flats are offered in the newspaper.”

type shifts: non-relational → relational

sortal	individual
proper relational	functional

↓

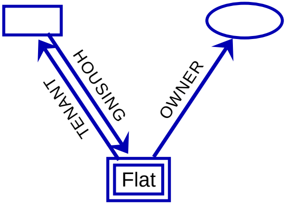


proper relational concept *flat:*
"This flat is a flat of John, he owns more than five."

type shifts: non-relational → relational

sortal	**individual**
proper relational	**functional**

↓

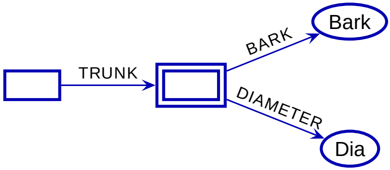


functional concept *flat:*
"The flat of Mary is huge and the rent is reasonable."

type shifts: relational → non-relational

sortal	**individual**
proper relational	**functional**

↑

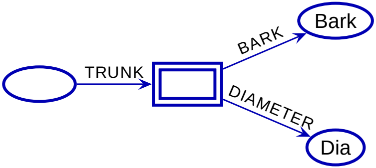


functional concept *trunk:*
“She sat with her back against the trunk of an oak.”

type shifts: relational → non-relational

sortal	**individual**
proper relational	**functional**

↑



sortal concept *trunk:*
"They rested and sat on a trunk."

outline

hypothesis: composition works uniformly with respect to concept types

RC	$\overset{OF}{\sqcup}$	SC	$\mapsto$	SC	finger OF woman
RC	$\overset{OF}{\sqcup}$	IC	$\mapsto$	SC	finger OF Mary
RC	$\overset{OF}{\sqcup}$	RC	$\mapsto$	RC	finger OF friend
RC	$\overset{OF}{\sqcup}$	FC	$\mapsto$	RC	finger OF spouse
FC	$\overset{OF}{\sqcup}$	SC	$\mapsto$	SC	head OF woman
FC	$\overset{OF}{\sqcup}$	IC	$\mapsto$	IC	head OF Mary
FC	$\overset{OF}{\sqcup}$	RC	$\mapsto$	RC	head OF friend
FC	$\overset{OF}{\sqcup}$	FC	$\mapsto$	FC	head OF spouse

proper relational concepts:

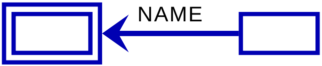
type of composed concept = relational type of possessor concept

functional concepts:

type of composed concept = referential + relational type of possessor concept

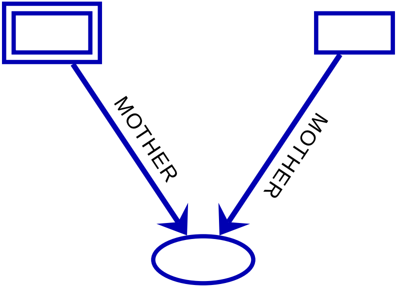
Löbner

FC $\overset{\text{OF}}{\sqcup}$ RC $\mapsto$ RC: name OF sibling



name

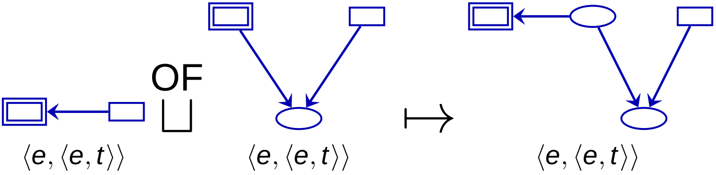
$\lambda y \lambda x.\ x = \text{NAME}(y)$



sibling

$\lambda y \lambda x.\ \text{MOTHER}(x) = \text{MOTHER}(y)$

FC $\overset{\text{OF}}{\sqcup}$ RC $\mapsto$ RC: name OF sibling

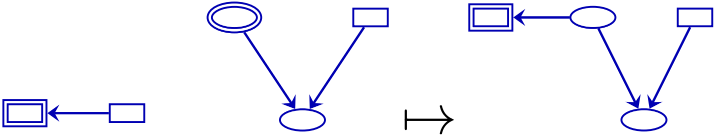


$\lambda y' \lambda x'.\, x' = f(y') \overset{\text{OF}}{\sqcup} \lambda y' \lambda x'.\, S(x', y') \mapsto \lambda y' \lambda x.\, x = f(\varepsilon u.\, S(u, y'))$

FC $\circ(\varepsilon\circ$ RC)

$\langle e, \langle e, t\rangle\rangle \circ (\langle\langle e, t\rangle, e\rangle \circ \langle e, \langle e, t\rangle\rangle) \mapsto \langle e, \langle e, t\rangle\rangle \circ \langle e, e\rangle \mapsto \langle e, \langle e, t\rangle\rangle$

FC $\overset{\text{OF}}{\sqcup}$ RC $\mapsto$ RC: name OF sibling



$\langle e, \langle e, t\rangle\rangle$ $\quad$ $\langle e, e\rangle$ $\quad\mapsto\quad$ $\langle e, \langle e, t\rangle\rangle$

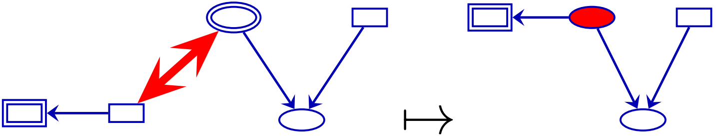
$\lambda y' \lambda x'.\, x' = f(y') \overset{\text{OF}}{\sqcup} \lambda y' \lambda x'.\, S(x', y') \mapsto \lambda y' \lambda x.\, x = f(\varepsilon u.\, S(u, y'))$

FC $\circ(\varepsilon\circ$ RC)

$\langle e, \langle e, t\rangle\rangle \circ (\langle\langle e, t\rangle, e\rangle \circ \langle e, \langle e, t\rangle\rangle) \mapsto \langle e, \langle e, t\rangle\rangle \circ \langle e, e\rangle \mapsto \langle e, \langle e, t\rangle\rangle$

1. $\varepsilon\circ$ RC: $\lambda y'(\lambda Q.\, \varepsilon u.\, Q(u)(\lambda x'.\, S(x', y'))) \to_\beta \lambda y'(\varepsilon u.\, \lambda x'.\, S(x', y')(u)) \to_\beta \lambda y'.\, \varepsilon u.\, S(u, y')$
2. FC $\circ(\varepsilon\circ$ RC): $(\lambda y \lambda x.\, x = f(y)) \circ (\lambda y'.\varepsilon u.\, S(u, y')) \to \lambda y'(\lambda y \lambda x.\, x = f(y)(\varepsilon u.\, S(u, y'))) \to_\beta \lambda y' \lambda x.\, x = f(\varepsilon u.\, S(u, y'))$

FC $\overset{\text{OF}}{\sqcup}$ RC $\mapsto$ RC: name OF sibling



$\langle e, \langle e, t\rangle\rangle$ $\langle e, e\rangle$ $\langle e, \langle e, t\rangle\rangle$

$\lambda y' \lambda x'. \; x' = f(y') \overset{\text{OF}}{\sqcup} \lambda y' \lambda x'. \; S(x', y') \mapsto \lambda y' \lambda x. \; x = f(\varepsilon u. \; S(u, y'))$

FC $\circ(\varepsilon\circ$ RC)

$\langle e, \langle e, t\rangle\rangle \circ (\langle\langle e, t\rangle, e\rangle \circ \langle e, \langle e, t\rangle\rangle) \mapsto \langle e, \langle e, t\rangle\rangle \circ \langle e, e\rangle \mapsto \langle e, \langle e, t\rangle\rangle$

1. $\varepsilon\circ$ RC: $\lambda y'(\lambda Q. \; \varepsilon u. \; Q(u)(\lambda x'. \; S(x', y'))) \rightarrow_\beta \lambda y'(\varepsilon u. \; \lambda x'. \; S(x', y')(u)) \rightarrow_\beta \lambda y'. \; \varepsilon u. \; S(u, y')$
2. FC $\circ(\varepsilon\circ$ RC): $(\lambda y \lambda x. \; x = f(y)) \circ (\lambda y'. \varepsilon u. \; S(u, y')) \rightarrow \lambda y'(\lambda y \lambda x. \; x = f(y)(\varepsilon u. \; S(u, y'))) \rightarrow_\beta \lambda y' \lambda x. \; x = f(\varepsilon u. \; S(u, y'))$