A Formal Interpretation of Concept Types and Type Shifts

Wiebke Petersen & Tanja Osswald

Heinrich-Heine-Universität Düsseldorf
Research group on “Functional Concepts and Frames”

CogLang 2010
outline

1. Concept Types (Löbner)
2. Concept Frames
3. Type Shifts
1. Concept Types (Löbner)
2. Concept Frames
3. Type Shifts
Concept classification

person, pope, house, verb, sun, Mary, wood, brother, mother, meaning, distance, spouse, argument, entrance
## Concept classification: inherent relationality

<table>
<thead>
<tr>
<th>Type</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>non-relational</td>
<td>person, pope, house, verb, sun, Mary, wood</td>
</tr>
<tr>
<td>relational</td>
<td>brother, mother, meaning, distance, spouse, argument, entrance</td>
</tr>
</tbody>
</table>
Concept classification: inherent uniqueness of reference

<table>
<thead>
<tr>
<th></th>
<th>non-unique reference</th>
<th>unique reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>non-relational</td>
<td>person, house, verb, wood</td>
<td>Mary, pope, sun</td>
</tr>
<tr>
<td>relational</td>
<td>brother, argument, entrance</td>
<td>mother, meaning, distance, spouse</td>
</tr>
</tbody>
</table>

Löbner
## Concept classification

<table>
<thead>
<tr>
<th>Concept Type</th>
<th>Non-unique Reference</th>
<th>Unique Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Non-relational</td>
<td><strong>Sortal concept</strong></td>
<td><strong>Individual concept</strong></td>
</tr>
<tr>
<td></td>
<td>indefinite, demonstrative, plural,</td>
<td></td>
</tr>
<tr>
<td></td>
<td>quantificational, absolute</td>
<td>singular definite, absolute</td>
</tr>
<tr>
<td>Relational</td>
<td><strong>Proper relational concept</strong></td>
<td><strong>Functional concept</strong></td>
</tr>
<tr>
<td></td>
<td>indefinite, demonstrative, plural,</td>
<td></td>
</tr>
<tr>
<td></td>
<td>quantificational, relational, pos-</td>
<td></td>
</tr>
<tr>
<td></td>
<td>sessive</td>
<td>singular definite, relational, poss-</td>
</tr>
<tr>
<td></td>
<td></td>
<td>sessive</td>
</tr>
</tbody>
</table>

Löbner
Outline

1. Concept Types (Löbner)
2. Concept Frames
3. Type Shifts
Frames


- Frames provide the fundamental representation of knowledge in human cognition.
- At their core, frames contain **attribute-value sets**.
Frames as generalized feature structures

Frames can be represented by directed connected graphs with:
- one central node (double border)
- 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
- uniquely referring nodes are marked with a definiteness marker
Frames and functional concepts

- attributes represent functions
- ⇒ attributes correspond to functional concepts
- ⇒ frames decompose concepts into functional concepts
- ⇒ functional concepts embody the concept type on which categorization is based
Sortal concepts

tree-frame

![Diagram of tree-frame concept]

trunk-frame

![Diagram of trunk-frame concept]
Individual concepts

**Mary-frame**

predicate constant ‘Mary’:

![Diagram of Mary-frame]

**pope-frame**

predicate constant ‘pope’:

![Diagram of pope-frame]
Non-relational concepts

**sortal concepts**

*most simple frame:*

- one open argument (=central node)
- no path from a definite node to the central node

**individual concepts**

*most simple frame:*

- one open argument (=central node)
- there is a direct path from a definite node to the central node
Proper relational concepts

**brother-frame**

**co-parent-frame**

**child-frame**
Functional concepts

*head-frame*
predicate constant ‘head’:

*haircolor-frame*
predicate constant ‘haircolor’:
Relational concepts

**proper relational concepts**

most simple frame:

- two open arguments
- no direct path from the other open argument to the central node

**functional concepts**

most simple frame:

- two open arguments
- there is a direct path from the other open argument to the central node
Summary: concept types and their frames

<table>
<thead>
<tr>
<th>Concept Type</th>
<th>Most Simple Frame</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>sortal concepts</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>most simple frame:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Examples: stone, teenager, tree</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>individual concepts</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>most simple frame:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Examples: pope, Mary</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>proper relational concepts</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>most simple frame:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Examples: sister, son, finger</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>functional concepts</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>most simple frame:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Examples: mother, trunk, color</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
1. Concept Types (Löbner)
2. Concept Frames
3. Type Shifts
Type shifts: example

Concept *mother* (cf. Gerland and Horn 2010)

(1) Maria is Peter’s mother.
(2) Maria is a mother.
(3) Maria is the mother.
(4) Maria is a mother of Peter.
Mother as a functional concept

(1) Maria is Peter’s mother.

- Lexicalized concept type.
Mother as a sortal concept

(2) Maria is a mother.

- Type shifts can close or open arguments.
**Mother as an individual concept**

(3) Maria is the mother.

- Context can introduce definiteness.
Mother as a proper relational concept

(4) Maria is a mother of Peter.

- Type shifts can transform the frame structure of a concept - but they need a strong context for that.
Type shifts: example

Concept *flat*

(5) Many flats are offered in the newspaper.
(6) This flat is a flat of John, he owns more than five.
(7) The flat of Mary is huge and the rent is reasonable.
(5) Many flats are offered in the newspaper.
Flat as a proper relational concept

(6) This flat is a flat of John, he owns more than five.
Flat as a functional concept

(7) The flat of Mary is huge and the rent is reasonable.
Summary and outlook

Summary
- Type shifts occur in language.
- Type shifts can be modeled in frames.
- Types can be shifted arbitrarily.
- Unusual uses need a strong context for the shift.

Outlook
- metaphoric shifts
- metonymic shifts

