# Plotting and exploring lexical semantic maps: Resources, tools, and methodological issues

Stéphane Polis
(F.R.S.-FNRS / ULiège)

17.04.2018
**National Research University Higher School of Economics - Moscow**

# Semantic maps

- Two main types
  - Connectivity maps
  - Proximity maps (= MDS maps)

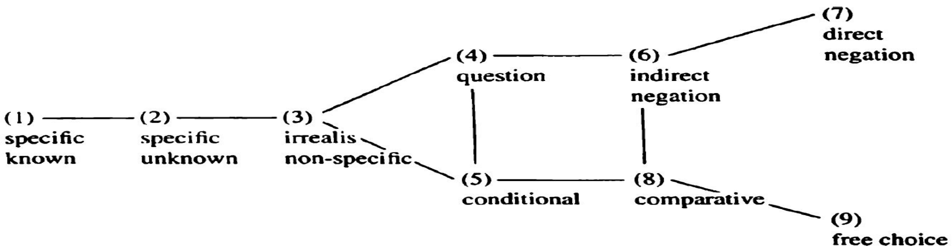



**Figure 1a**. Haspelmath's (1997: 4) original semantic map of the indefinite pronouns functions

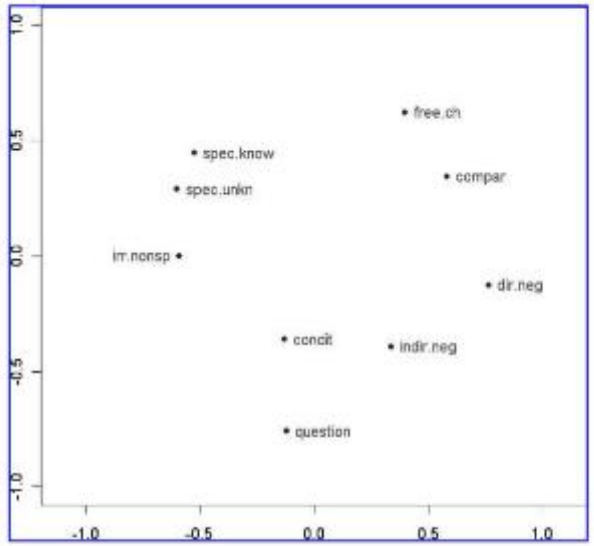



**Figure 1b**. MDS analysis of Haspelmath's (1997) data on indefinite pronouns (Croft & Poole 2008: 15)

- Graphs
  - Nodes = meanings
  - Edges = relationships between meanings

- Two-dimensional spaces
  - Points = meanings (or contexts)
  - Proximity = similarity between meanings (or contexts)

# Semantic maps

- Two main types
  - Connectivity maps
  - Proximity maps (= MDS maps)

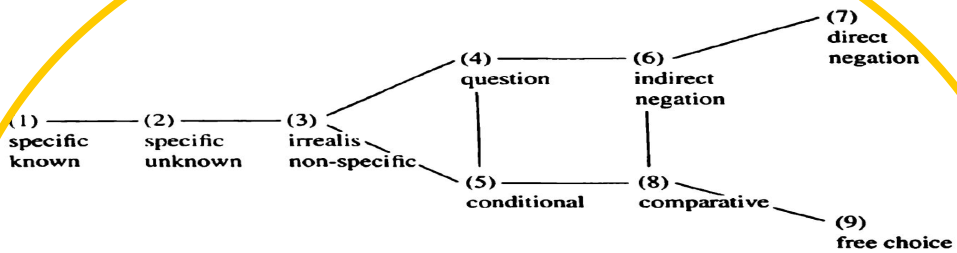



**Figure 1a**. Haspelmath's (1997: 4) original semantic map of the indefinite pronouns functions

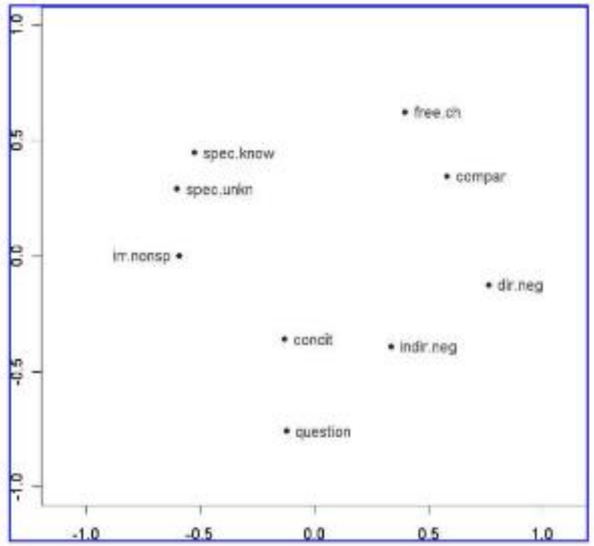
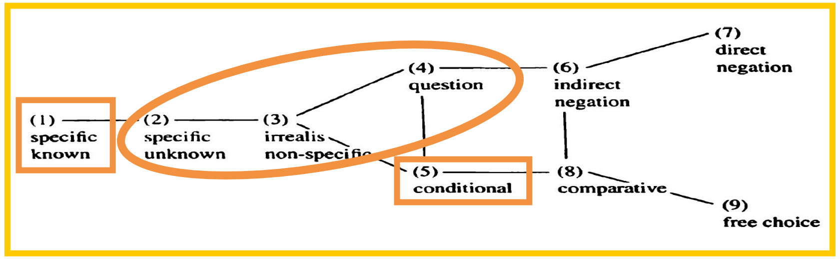



**Figure 1b**. MDS analysis of Haspelmath's (1997) data on indefinite pronouns (Croft & Poole 2008: 15)

- Graphs
  - Nodes = meanings
  - Edges = relationships between meanings

- Two-dimensional spaces
  - Points = meanings (or contexts)
  - Proximity = similarity between meanings (or contexts)

# Semantic maps



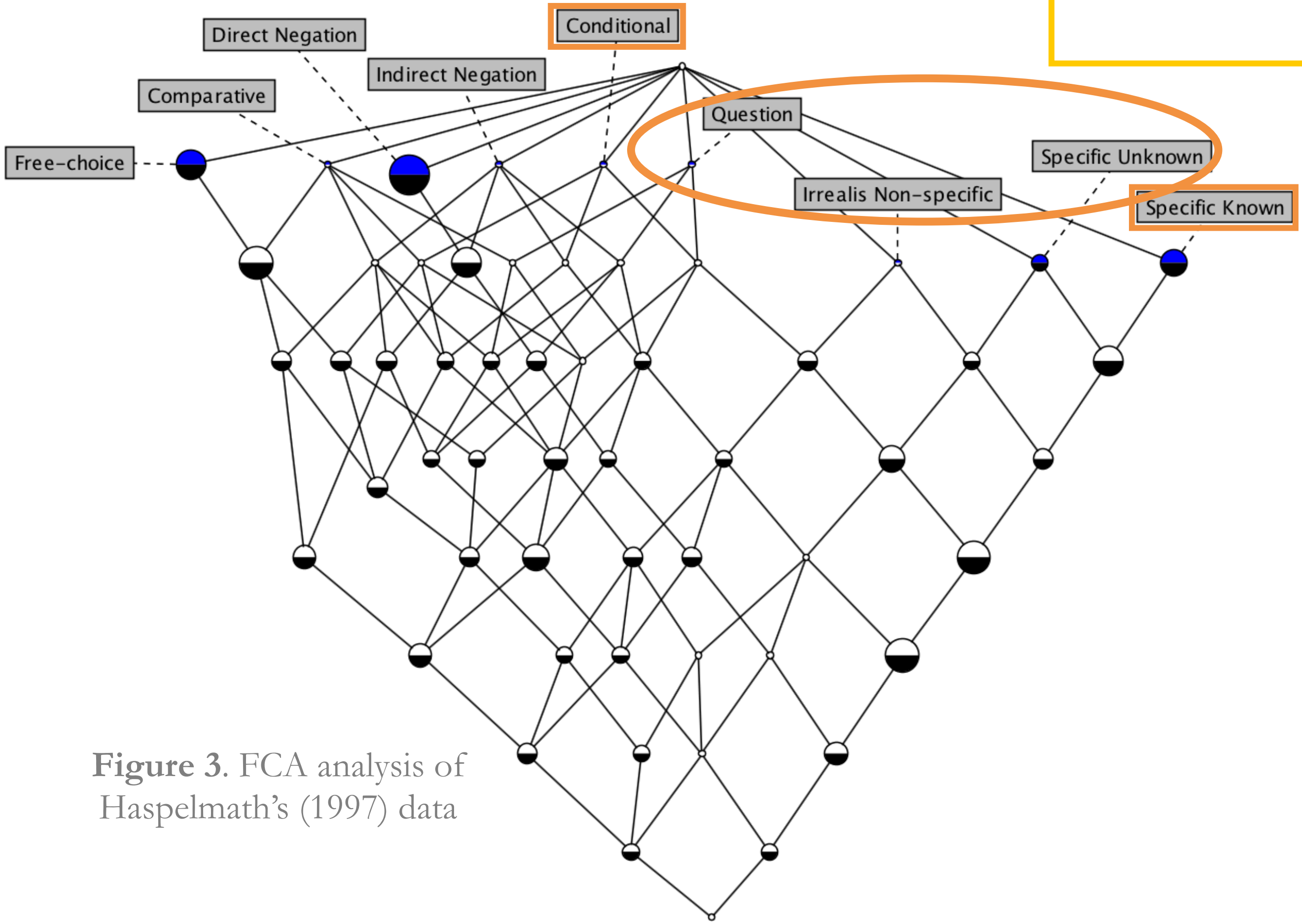




**Figure 3**. FCA analysis of Haspelmath's (1997) data

# Outline of the talk

- Different kinds of information captured by classical semantic maps
  - Semantic closeness
  - Diachrony
  - Frequency
  - Types of semantic relationships

# Outline of the talk

- Different kinds of information captured by classical semantic maps
  - Semantic closeness
  - Diachrony
  - Frequency
  - Types of semantic relationships
- Large-scale resources for lexical typology
  - CLICS and CLICS 2.0
  - Multilingual Wordnet

# Outline of the talk

- Different kinds of information captured by classical semantic maps
  - Semantic closeness
  - Diachrony
  - Frequency
  - Types of semantic relationships
- Large-scale resources for lexical typology
  - CLICS and CLICS 2.0
  - Multilingual Wordnet
- Inferring classical semantic maps
  - Regier et al. (2013)
  - Weighted semantic maps
  - Diachronic semantic maps

# Outline of the talk

- Different kinds of information captured by classical semantic maps
  - Semantic closeness
  - Diachrony
  - Frequency
  - Types of semantic relationships
- Large-scale resources for lexical typology
  - CLICS and CLICS 2.0
  - Multilingual Wordnet
- Inferring classical semantic maps
  - Regier et al. (2013)
  - Weighted semantic maps
  - Diachronic semantic maps
- Exploring automatically-plotted semantic maps
  - Gephi
  - (Cytoscape)

# Outline of the talk

- Different kinds of information captured by classical semantic maps
  - Semantic closeness
  - Diachrony
  - Frequency
  - Types of semantic relationships
- Large-scale resources for lexical typology
  - CLICS and CLICS 2.0
  - Multilingual Wordnet
- Inferring classical semantic maps
  - Regier et al. (2013)
  - Weighted semantic maps
  - Diachronic semantic maps
- Exploring automatically-plotted semantic maps
  - Gephi
  - (Cytoscape)
- Methodological issues
  - When does automatic plotting not work?
  - Alternative solutions?

# Types of information captured by classical semantic maps

# Types of information

- 'A semantic map is a geometrical representation of functions (…) that are linked by connecting lines and thus constitute a network' (Haspelmath 2003).

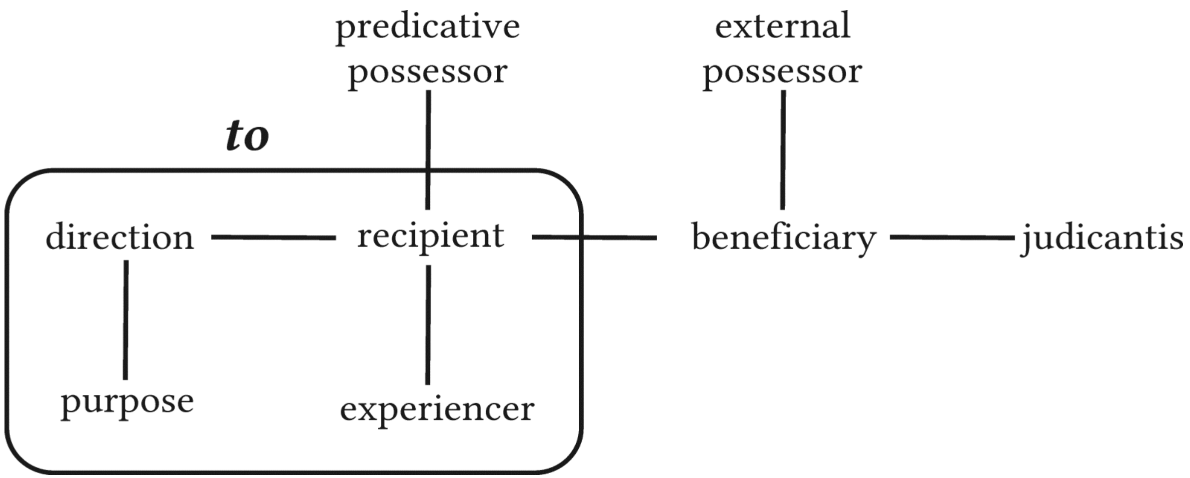



**Figure 3**. A semantic map of typical dative functions / the boundaries of English *to* (based on Haspelmath 2003: 213, 215)

# Types of information

- Diachronic semantic maps

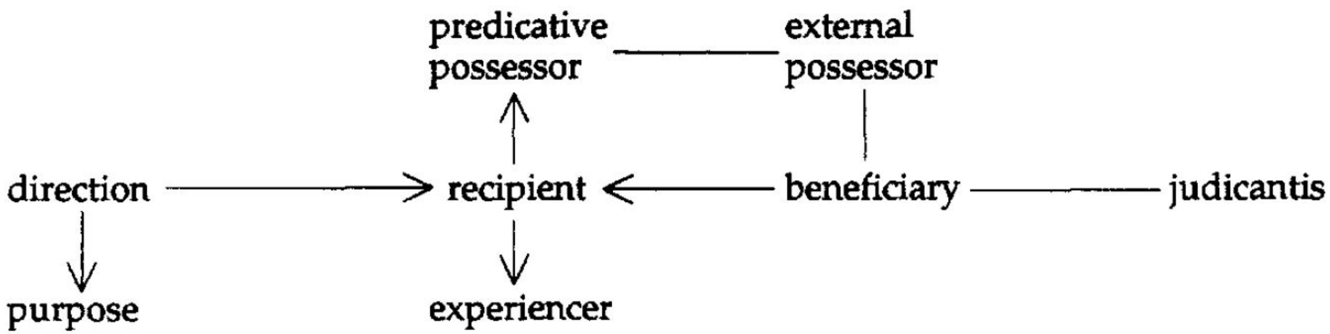



Figure 4. Dynamicized semantic map of dative functions
(Haspelmath 2003: 234)

# Types of information

- Diachronic semantic maps

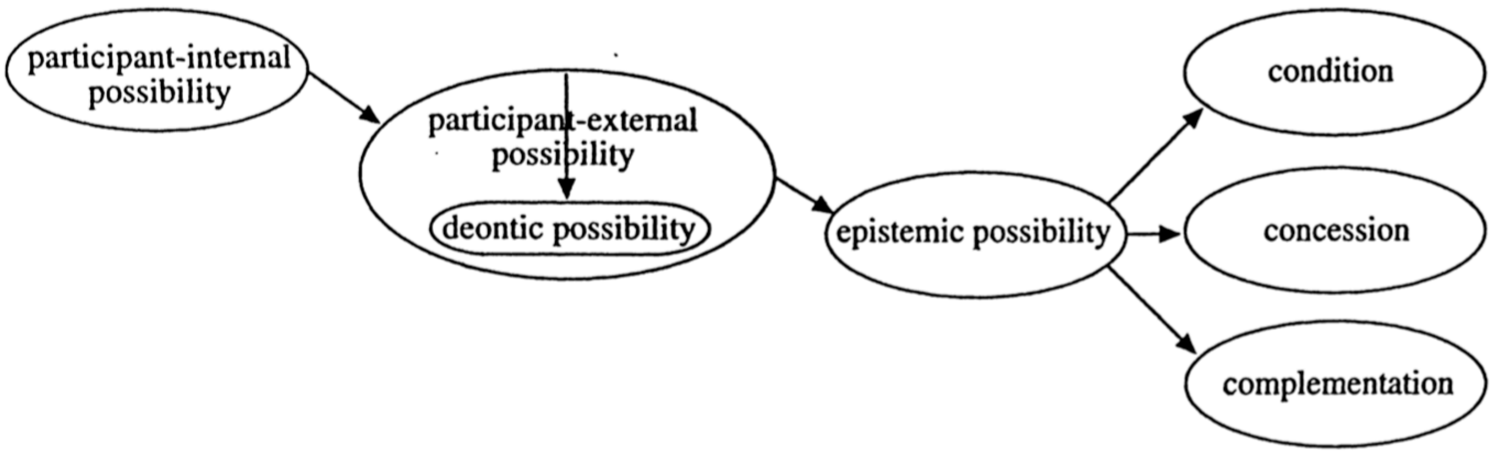



Figure 5. Dynamicized semantic map of modal possibility
(van der Auwera & Plungian 1998: Fig. 4)

**Modality's semantic map**

JOHAN VAN DER AUWERA and VLADIMIR A. PLUNGIAN

# Types of information

- Diachronic semantic maps

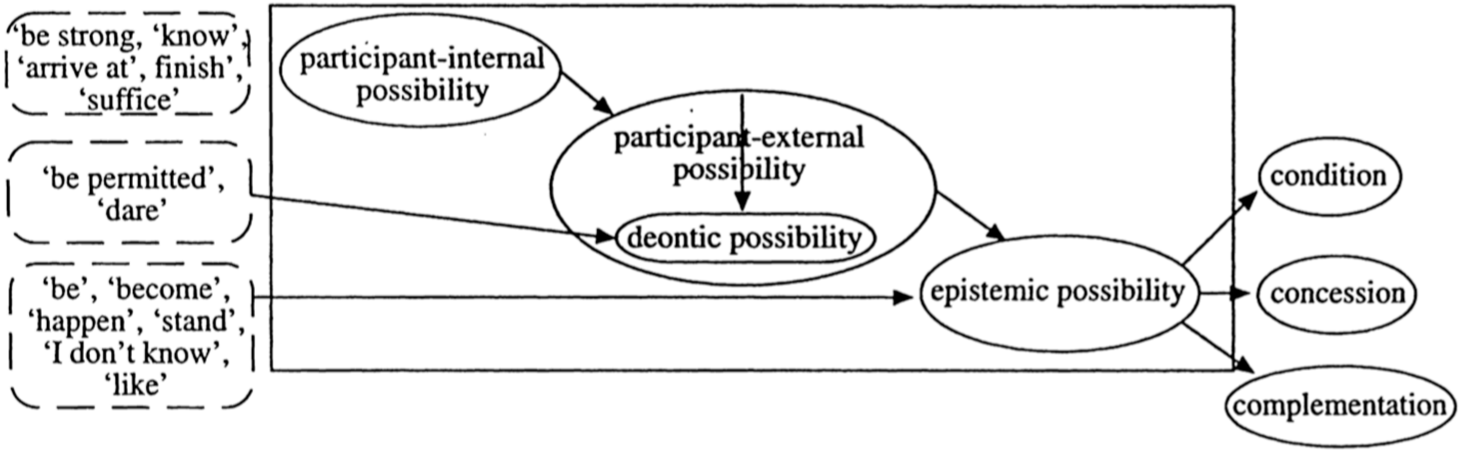



Figure 6. Dynamicized semantic map of modal possibility
(van der Auwera & Plungian 1998: Fig. 4)

**Modality's semantic map**

JOHAN VAN DER AUWERA and VLADIMIR A. PLUNGIAN

# Types of information

- Weighted semantic maps

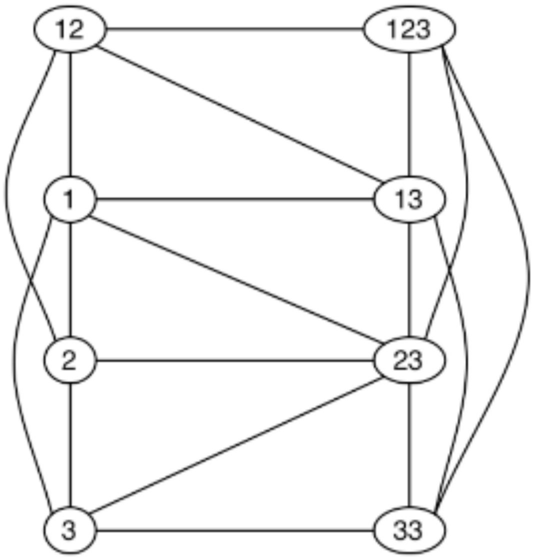



**Figure 7a**. A **simple** semantic map of person marking (Cysouw 2007: 231)

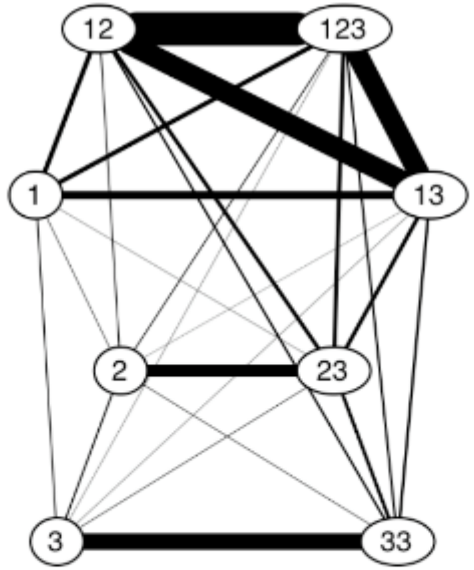



**Figure 7b**. A **weighted** semantic map of person marking (Cysouw 2007: 233)

# Types of information

- Weighted semantic maps

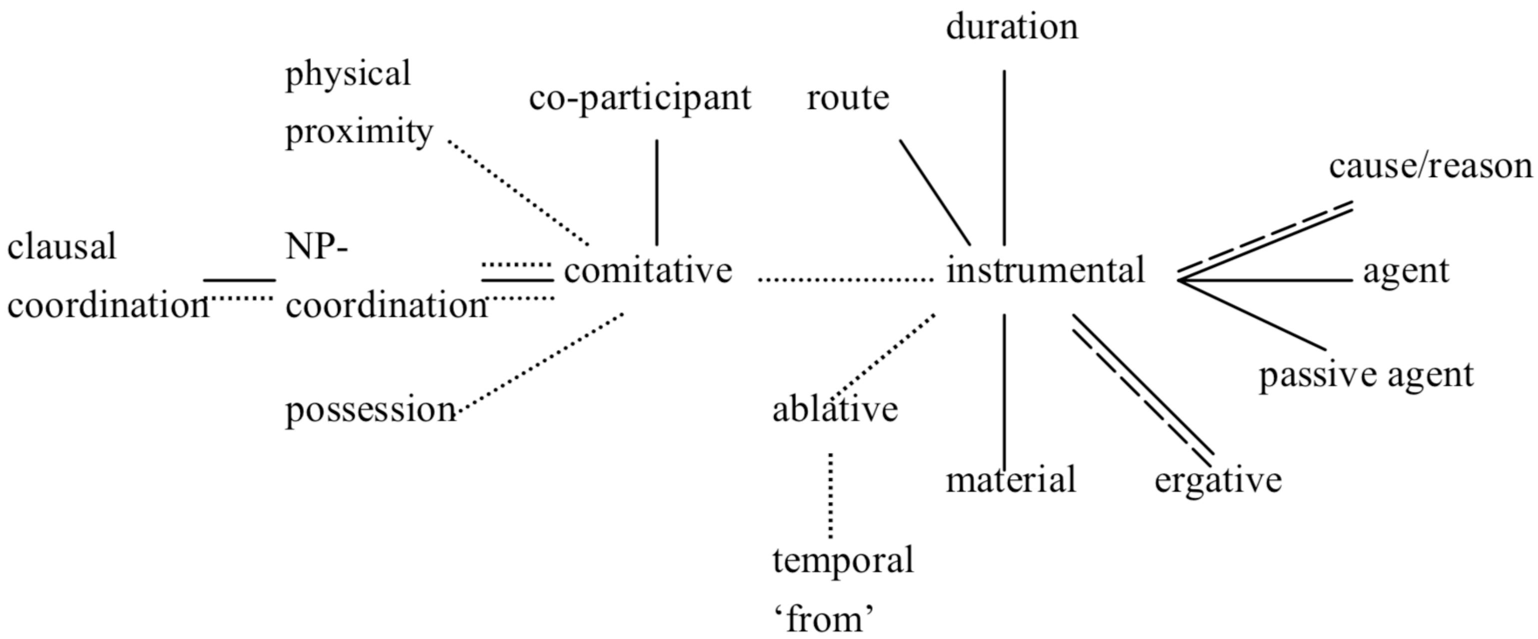



**Figure 8**. A map of comitative and instrumental functions
(Narrog & Ito 2007: 283)

# Types of information

- Semantic relationships

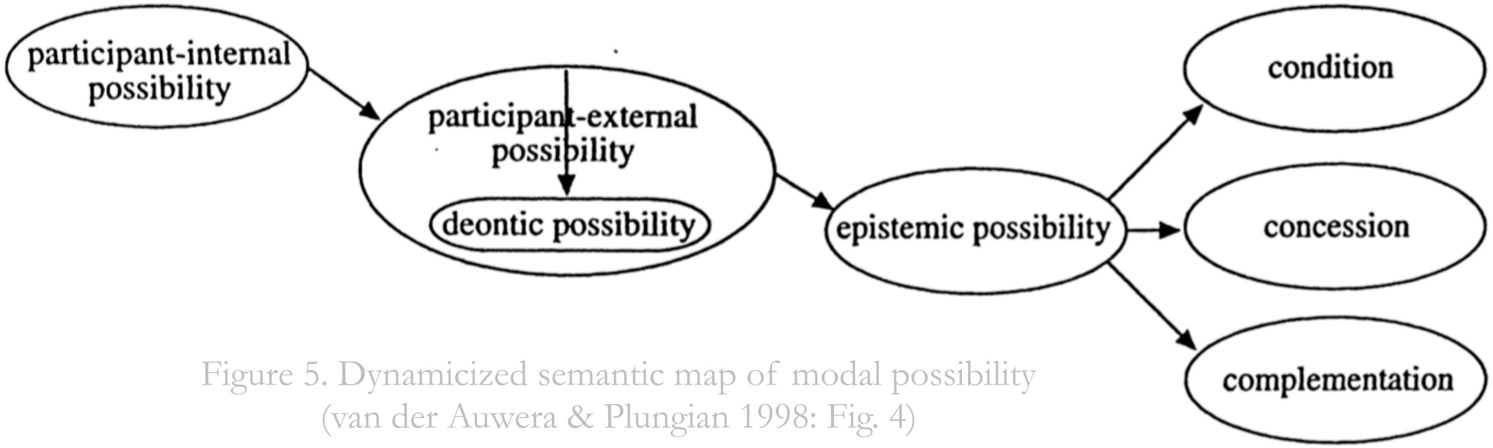



Figure 5. Dynamicized semantic map of modal possibility
(van der Auwera & Plungian 1998: Fig. 4)

✓ DEONTIC POSSIBILITY (e.g., "as far as I'm concerned, you may go to the party tonight") is defined as a subtype (hyponym) of PARTICIPANT–EXTERNAL POSSIBILITY (e.g., "you may take the bus in front of the train station")

✓ PARTICIPANT–EXTERNAL POSSIBILITY and EPISTEMIC POSSIBILITY (e.g., "he may be at the office right now") are seen as metonymically related

# Types of information

- Semantic relationships

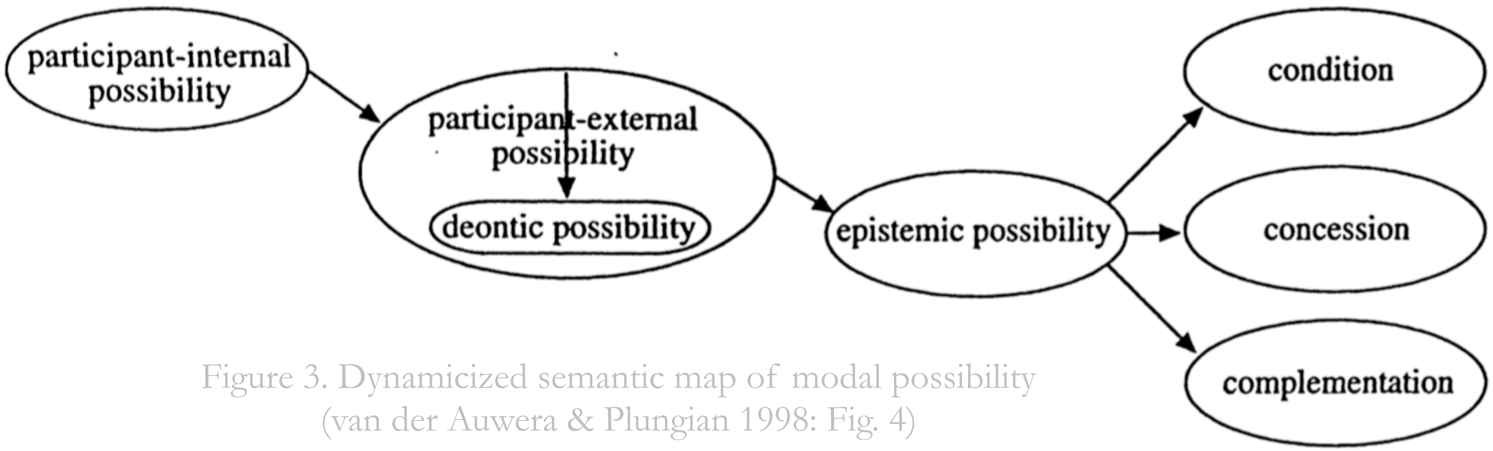
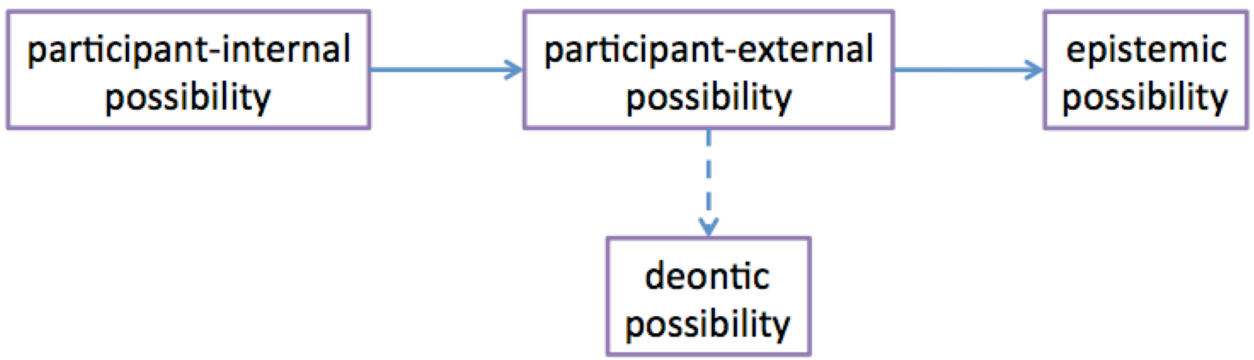



Figure 3. Dynamicized semantic map of modal possibility
(van der Auwera & Plungian 1998: Fig. 4)

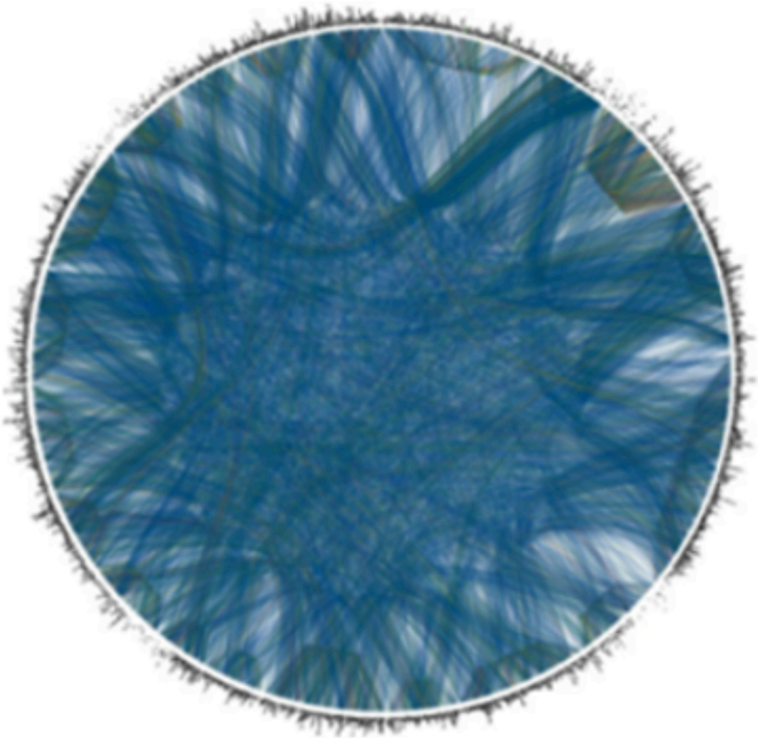


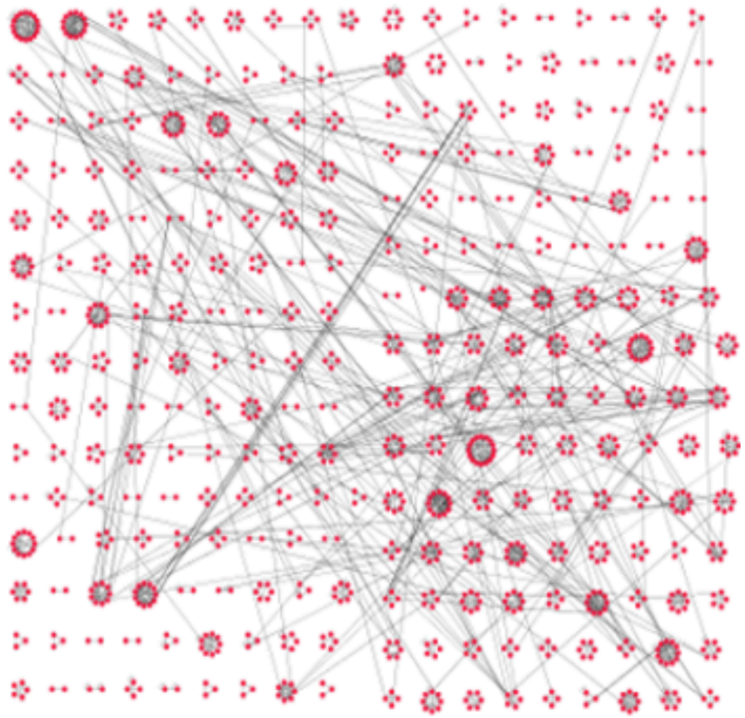
# Large-scale electronic resources for lexical typology

# Electronic resources for lexical typology

## CLICS

CLICS is an online database of *synchronic lexical associations* [READ MORE]





**Database of Cross-Linguistic Colexifications**

# Electronic resources for lexical typology

- *N* of lgs: 221
- *N* of lg families: 64
- *N* of concepts: 1280

| Meaning 1 | Meaning 2 | *N of language* | *N of forms* | language:form |
|---|---|---|---|---|
| see | know | 5 | 6 | aro_std:[ba]//ayo_std:[iˈmoʔ]//haw_std:[ʔike]//mcq_std:[ɓanahe]//mri_std:[kitea]//tel_std:[aarayu]//tel_std:[arayu] |
| see | find | 15 | 23 | agr_std:[wainat]//arn_std:[pe]//con_std:[ˈatʰeye]//cwg_std:[yow]//emp_std:[uˈnu]//kgp_std:[we]//kpv_std:[addzɯɩ]//kyh_std:[mah]//mca_std:[wen]//mri_std:[kitea]//oym_std:[ɛsa]//pbb_std:[uy]//plt_std:[mahìta]//pui_std:[duk]//ray_std:[tikeʔa]//rtm_std:[ræe]//sap_Enlhet:[neŋwetayˀ]//sei_std:[aʔo]//shb_std:[taa]//sja_std:[unu]//swh_std:[ona]//tbc_std:[le]//yag_std:[tiki] |
| see | get, obtain | 6 | 6 | kgp_std:[we]//mbc_std:[eraʔma]//pbb_std:[uy]//sap_Standard:[akwitayi]//srq_std:[tea]//udi_std:[акъсун] |

Polysemy data from CLiCs (http://clics.lingpy.org/download.php) (List et al. 2014)

# Electronic resources for lexical typology

## The Intercontinental Dictionary Series

*Founding Editor:*
†Mary Ritchie Key
*General Editor:*
Bernard Comrie

**Purpose** ⇧

The Intercontinental Dictionary Series (IDS) is a database where lexical material across the languages of the world is organized in such a way that comparisons can be made. Historical and comparative studies and theoretical linguistic research can be based on this documentation. The IDS was conceived of by Mary Ritchie Key (University of California, Irvine) in the 1980s as a long-term cooperative project that will go on for the next generation or so and will involve linguists all over the world. It is aimed towards international understanding and cooperation. This is a pioneering effort that will have global impact. The purpose also contributes to preserving information on the little-known and "non-prestigious" languages of the world, many of which are becoming extinct.

# Electronic resources for lexical typology

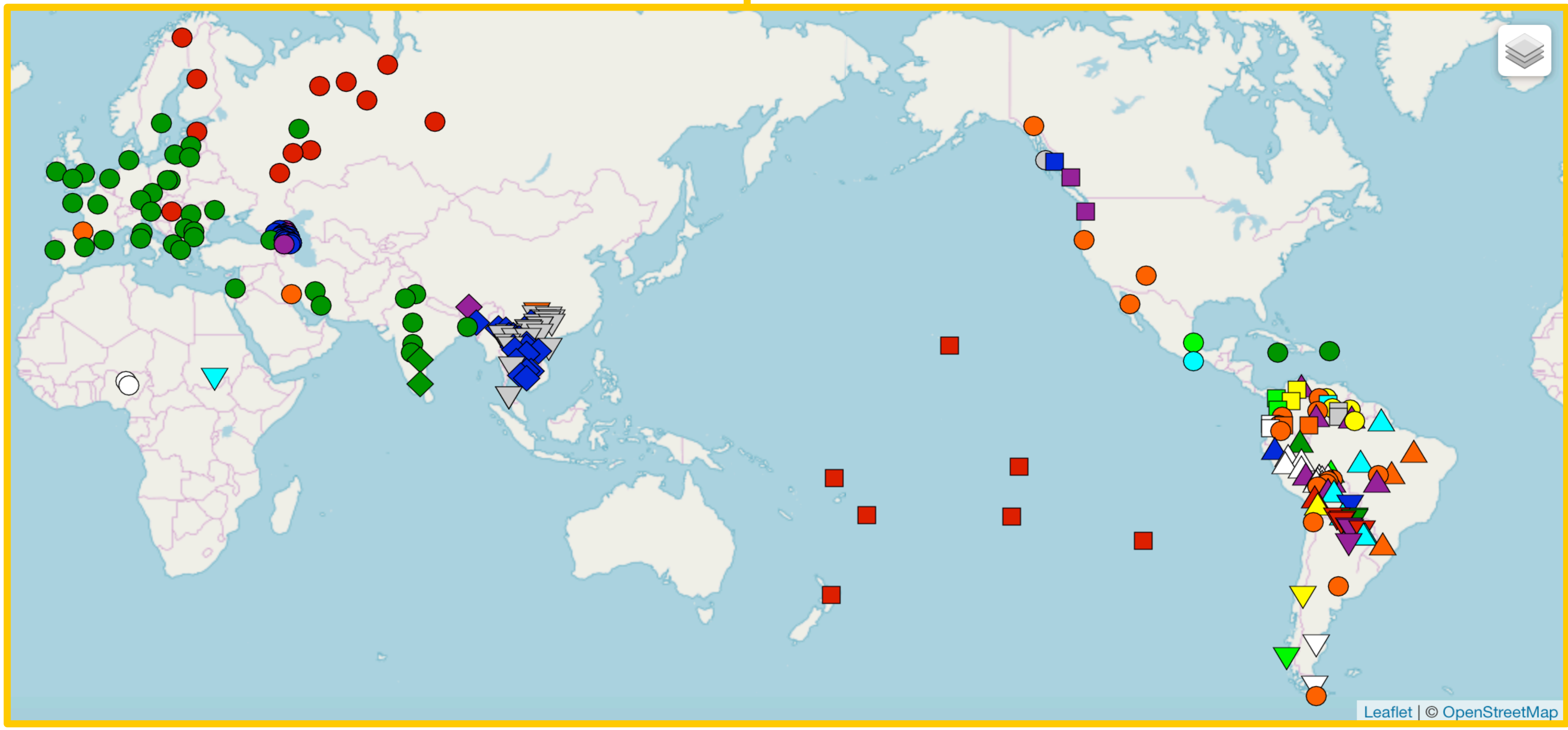
## The Intercontinental Dictionary Series

*Founding Editor:*

†Mary Ritchie Key

*General Editor:*

Bernard Comrie



# Electronic resources for lexical typology

| | A | B | C |
|---|---|---|---|
| 119 | day | afternoon | hau_std:rana//ket_std:iʔ//plj_std:piidi//rus_std:den//tli_std:yakyee |
| 120 | day | again | kha_std:sngi |
| 121 | day | age | gui_std:ˈara//yad_std:hnda |
| 122 | day | anger | tzz_std:k'ak'al |
| 123 | day | bright | tzz_std:k'ak'al |
| 124 | day | clock, timepiece | gue_std:wurlngarn//sei_std:šãʔ |
| 125 | day | cloud | haw_std:ao |
| 126 | day | country | cbr_std:niti//shp_std:niti |
| 127 | day | dawn | haw_std:ao//waw_std:enmari |
| 128 | day | doubt | haw_std:lā |
| 129 | day | earth, land | cag_std:nału//haw_std:ao//mri_std:ao//tzz_std:osil |
| 130 | day | east | tob_std:naʔaʔk |
| 131 | day | fever | tzz_std:k'ak'al |
| 132 | day | fin (dorsal) | haw_std:lā |
| 133 | day | fire | jpn_std:hi |
| 134 | day | go | ote_std:pa//oym_std:aa |
| 135 | day | go away, depart | ote_std:pa |
| 136 | day | hour | sap_Standard:aknim//shb_std:thəm |
| 137 | day | lamp, torch | ito_std:uwayo |
| 138 | day | lick | cmn_std:tian |
| 139 | day | light (in color) | mri_std:ao |
| 140 | day | light (noun) | con_std:aʔta//crt_std:xloma//haw_std:ao//hdn_Northern:ˁkatˀkáa//ito_std:uwayo//mzh |
| 141 | day | live, living, life | shp_std:niti |
| 142 | day | Monday | shp_std:niti |
| 143 | day | morning | crt_std:xloma//guq_std:kreibu |
| 144 | day | noon, midday | ind_std:siang//plj_std:piidi |

Meanings

Languages and word forms

# Electronic resources for lexical typology

```
Tmap = [Tsenses]
for t in Tclean:
        split_langWord = t[2].split('//')
        for couple in split_langWord:
                langWord = couple.split(':')
                line = [langWord[0], langWord[1]]
                for i in range (2,len(Tsenses)):
                        line.append('0')
                line[Tsenses.index(t[0])] = '1'
                line[Tsenses.index(t[1])] = '1'
                Tmap.append(line)
```

Python script $\alpha$

| | A | B | C | D | E | F |
|---|---|---|---|---|---|---|
| 1 | | | age | acid, sour | city, town | day |
| 2 | yad_std | hnda | 1 | 1 | 0 | 1 |
| 3 | vec_std | edat | 1 | 0 | 0 | 0 |
| 4 | jpn_std | toshi | 1 | 0 | 1 | 0 |
| 5 | gui_std | ˈara | 1 | 0 | 0 | 1 |
| 6 | nog_std | йуз | 1 | 0 | 0 | 0 |
| 7 | mri_std | pakeke | 1 | 0 | 0 | 0 |
| 8 | pbb_std | hiʔph | 1 | 0 | 0 | 0 |
| 9 | khv_Khvarshi | замана | 1 | 0 | 0 | 0 |

Lexical matrix

# Electronic resources for lexical typology

```
Tmap = [Tsenses]
for t in Tclean:
        split_langWord = t[2].split('//')
        for couple in split_langWord:
                langWord = couple.split(':')
                line = [langWord[0], langWord[1]]
                for i in range (2,len(Tsenses)):
                        line.append('0')
                line[Tsenses.index(t[0])] = '1'
                line[Tsenses.index(t[1])] = '1'
                Tmap.append(line)
```

Python script $\alpha$

Languages Forms

| | A | B | C | D | E | F |
|---|---|---|---|---|---|---|
| 1 | | | age | acid, sour | city, town | day |
| 2 | yad_std | hnda | 1 | 1 | 0 | 1 |
| 3 | vec_std | edat | 1 | 0 | 0 | 0 |
| 4 | jpn_std | toshi | 1 | 0 | 1 | 0 |
| 5 | gui_std | ˈara | 1 | 0 | 0 | 1 |
| 6 | nog_std | йуз | 1 | 0 | 0 | 0 |
| 7 | mri_std | pakeke | 1 | 0 | 0 | 0 |
| 8 | pbb_std | hi?ph | 1 | 0 | 0 | 0 |
| 9 | khv_Khvarshi | замана | 1 | 0 | 0 | 0 |

Lexical matrix

# Electronic resources for lexical typology

```
Tmap = [Tsenses]
for t in Tclean:
        split_langWord = t[2].split('//')
        for couple in split_langWord:
                langWord = couple.split(':')
                line = [langWord[0], langWord[1]]
                for i in range (2,len(Tsenses)):
                        line.append('0')
                line[Tsenses.index(t[0])] = '1'
                line[Tsenses.index(t[1])] = '1'
                Tmap.append(line)
```

Python script $\alpha$

Languages  Forms  Meanings

| | A | B | C | D | E | F |
|---|---|---|---|---|---|---|
| 1 | | | age | acid, sour | city, town | day |
| 2 | yad_std | hnda | 1 | 1 | 0 | 1 |
| 3 | vec_std | edat | 1 | 0 | 0 | 0 |
| 4 | jpn_std | toshi | 1 | 0 | 1 | 0 |
| 5 | gui_std | ˈara | 1 | 0 | 0 | 1 |
| 6 | nog_std | йуз | 1 | 0 | 0 | 0 |
| 7 | mri_std | pakeke | 1 | 0 | 0 | 0 |
| 8 | pbb_std | hi?ph | 1 | 0 | 0 | 0 |
| 9 | khv_Khvarshi | замана | 1 | 0 | 0 | 0 |

Lexical matrix

# Electronic resources for lexical typology

```python
Tmap = [Tsenses]
for t in Tclean:
        split_langWord = t[2].split('//')
        for couple in split_langWord:
                langWord = couple.split(':')
                line = [langWord[0], langWord[1]]
                for i in range (2,len(Tsenses)):
                        line.append('0')
                line[Tsenses.index(t[0])] = '1'
                line[Tsenses.index(t[1])] = '1'
                Tmap.append(line)
```

Python script $\alpha$

Languages Forms Meanings

| | A | B | C | D | E | F |
|---|---|---|---|---|---|---|
| 1 | | | age | acid, sour | city, town | day |
| 2 | yad_std | hnda | 1 | 1 | 0 | 1 |
| 3 | vec_std | edat | 1 | 0 | 0 | 0 |
| 4 | jpn_std | toshi | 1 | 0 | 1 | 0 |
| 5 | gui_std | ˈara | 1 | 0 | 0 | 1 |
| 6 | nog_std | йуз | 1 | 0 | 0 | 0 |
| 7 | mri_std | pakeke | 1 | 0 | 0 | 0 |
| 8 | pbb_std | hiʔph | 1 | 0 | 0 | 0 |
| 9 | khv_Khvarshi | замана | 1 | 0 | 0 | 0 |

1 when a meaning is attested for one form

Lexical matrix

# Electronic resources for lexical typology

Waiting for CLICS 2.0 …

(List et al. 2018)

**An Improved Database of Cross-Linguistic Colexifications: Assembling Lexical Data with Help of Cross-Linguistic Data Formats**

**Johann-Mattis List[1]***
**Simon Greenhill[12]**
**Cormac Anderson[1]**
**Thomas Mayer[3]**
**Tiago Tresoldi[1]**
**Robert Forkel[1]**

# Electronic resources for lexical typology

Increased quantity of data

Waiting for CLICS 2.0 …

(List et al. 2018)

1280 concepts => 2463 concepts (but 'only' 1521 colexified)
221 => 1156 language varieties (= 996 in *Glottolog*)

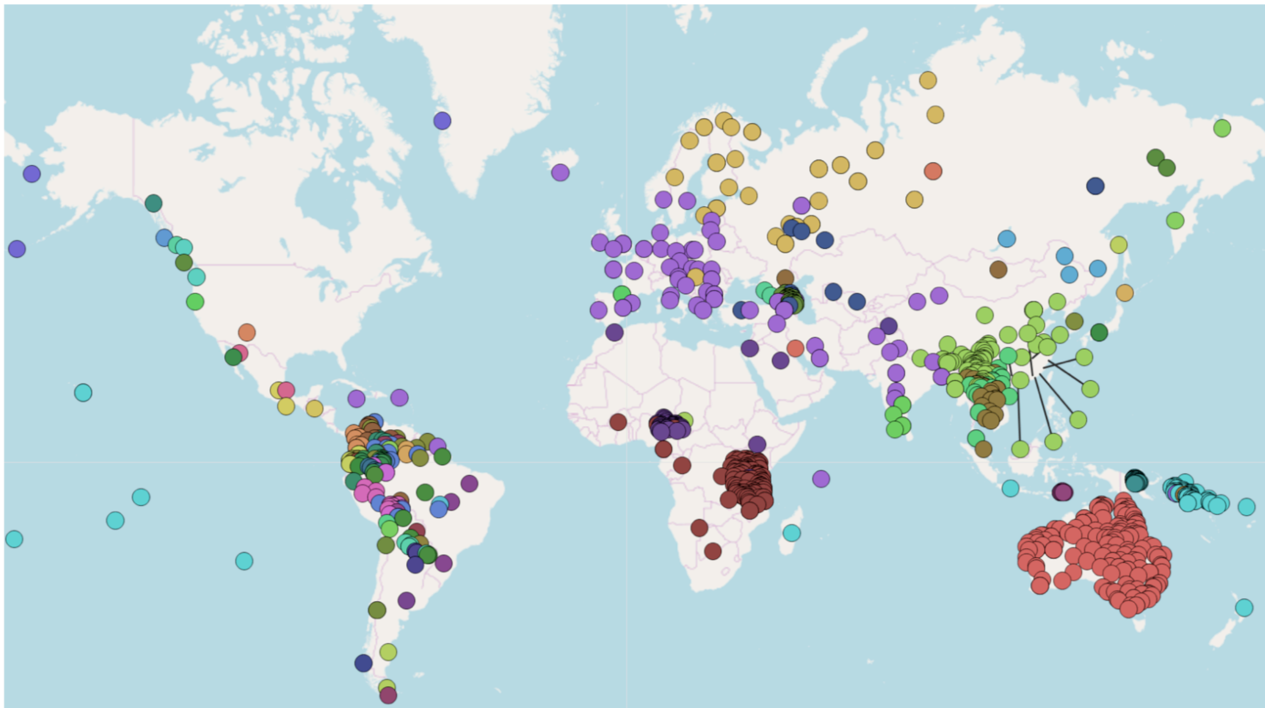
**55376 individual instances of colexification**

# Electronic resources for lexical typology

Increased quantity of data

Waiting for CLICS 2.0 …

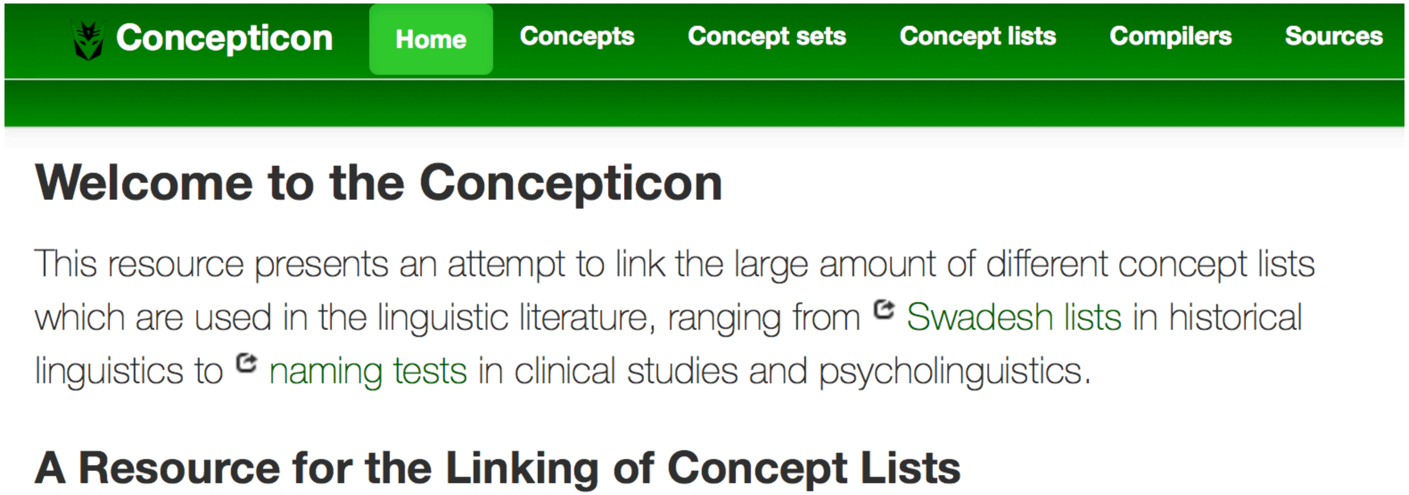
(List et al. 2018)



# Electronic resources for lexical typology

Waiting for CLICS 2.0 …
(List et al. 2018)

- Increased quantity of data
- Increased quality of data (e.g., links to the Concepticon)



# Electronic resources for lexical typology

Waiting for CLICS 2.0 …

(List et al. 2018)

- Increased quantity of data
- Increased quality of data (e.g., links to the Concepticon)
- Include partial colexifications
- Normalize the data which is analysed by CLICS

# Electronic resources for lexical typology


## Core concept

Words are grouped together as sets of synonyms (Fellbaum 1998: 72ff.)

**Synset**: A synonym set; a set of words that are roughly synonymous in a given context

# Electronic resources for lexical typology

**Verb**

- S: (v) **see** (perceive by sight or have the power to perceive by sight) *"You have to be a good observer to see all the details"; "Can you see the bird in that tree?"; "He is blind--he cannot see"*
- S: (v) understand, realize, realise, **see** (perceive (an idea or situation) mentally) *"Now I see!"; "I just can't see your point"; "Does she realize how important this decision is?"; "I don't understand the idea"*
- S: (v) witness, find, **see** (perceive or be contemporaneous with) *"We found Republicans winning the offices"; "You'll see a lot of cheating in this school"; "The 1960's saw the rebellion of the younger generation against established traditions"; "I want to see results"*
- S: (v) visualize, visualise, envision, project, fancy, **see**, figure, picture, image (imagine; conceive of; see in one's mind) *"I can't see him on horseback!"; "I can see what will happen"; "I can see a risk in this strategy"*
- S: (v) **see**, consider, reckon, view, regard (deem to be) *"She views this quite differently from me"; "I consider her to be shallow"; "I don't see the situation quite as negatively as you do"*
- S: (v) learn, hear, get word, get wind, pick up, find out, get a line, discover, **see** (get to know or become aware of, usually accidentally) *"I learned that she has two grown-up children"; "I see that you have been promoted"*
- S: (v) watch, view, **see**, catch, take in (see or watch) *"view a show on television"; "This program will be seen all over the world"; "view an exhibition"; "Catch a show on Broadway"; "see a movie"*

# Electronic resources for lexical typology


### Core concept

Words are grouped together as sets of synonyms (Fellbaum 1998: 72ff.)

**Synset**: A synonym set; a set of words that are roughly synonymous in a given context


**Open Multilingual Wordnet**

**34 languages**

# Electronic resources for lexical typology

## Open Multilingual Wordnet

**34 languages**

OMW can be queried as a corpus with the Natural Language Tool-kit (NLTK) interface in Python

↓

Possible to build lexical matrix!

# Electronic resources for lexical typology

## Open Multilingual Wordnet

**34 languages**

OMW can be queried as a corpus with the Natural Language Tool-kit (NLTK) interface in Python

↓

Possible to build lexical matrix!

**Method**

1. Choose the basic senses belonging to the semantic field to be investigated (e.g., SEE, HEAR, LOOK, LISTEN)
2. Collect all the forms that lexicalize these 4 senses
3. Retrieve the list of all the senses of these forms (the total of the synsets in which this forms appear)
4. For each form, check whether the senses collected are among its senses
5. Generate a polysemy matrix

## Inferring classical semantic maps from lexical matrices

# Inferring semantic maps

"ideally (…) it should be possible to generate semantic maps automatically on the basis of a given set of data"
(Narrog & Ito 2007: 280)

# Inferring semantic maps

Limitation of the semantic map method: practically impossible to handle large-scale crosslinguistic datasets manually

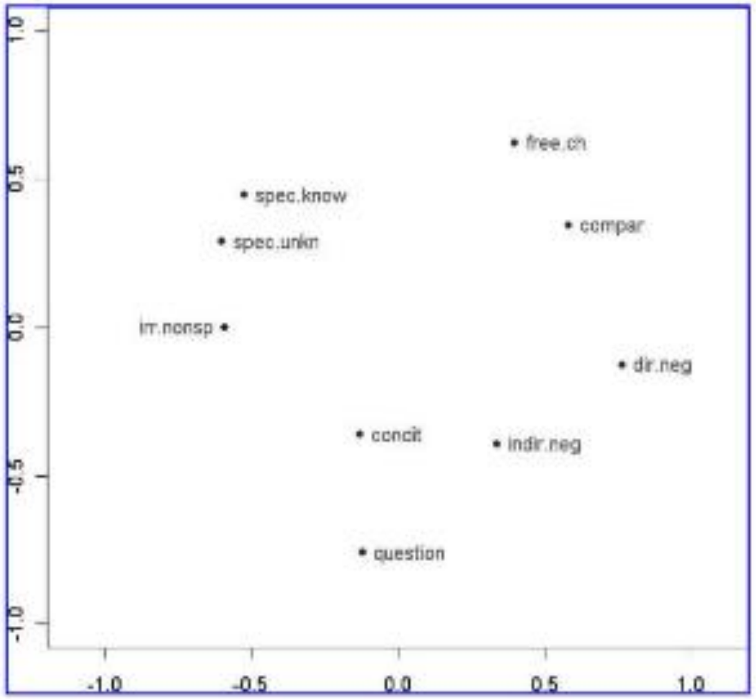
> "not mathematically well-defined or computationally tractable, making it impossible to use with large and highly variable crosslinguistic datasets"
>
> (Croft & Poole 2008: 1)

# Inferring semantic maps

Limitation of the semantic map method: practically impossible to handle large-scale crosslinguistic datasets manually

MDS

**Figure 5**. MDS analysis of Haspelmath's 1997 data on indefinite pronouns (Croft & Poole 2008: 15)

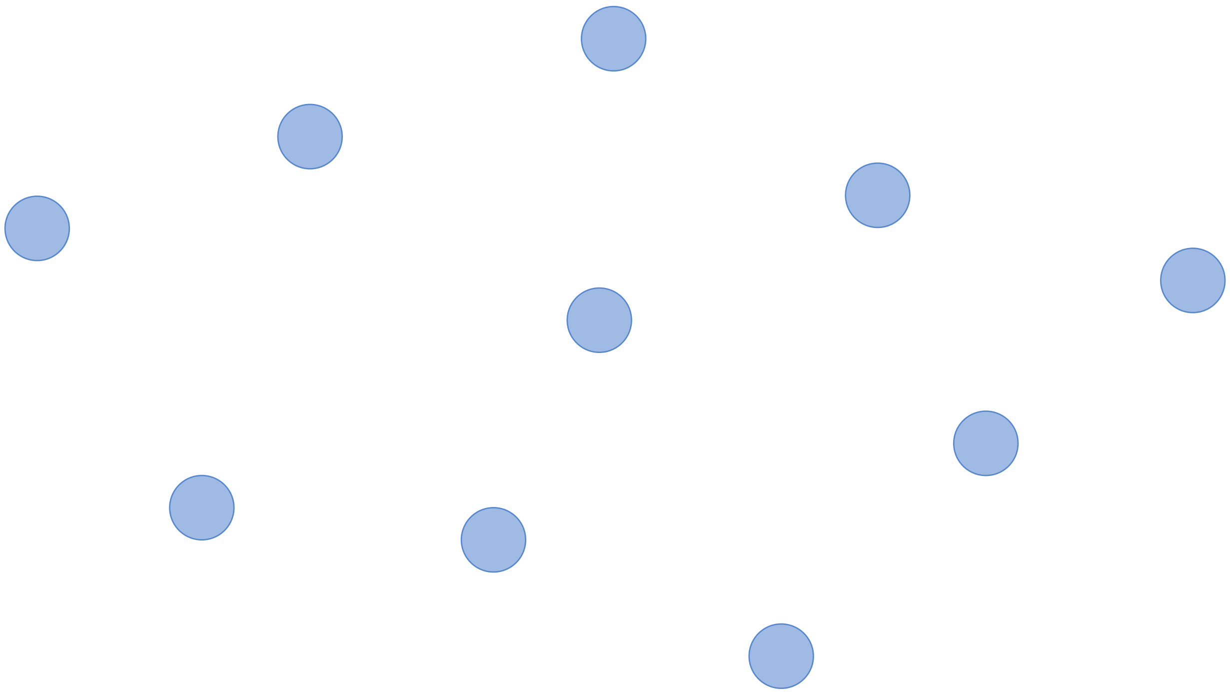


# Inferring semantic maps

Regier, Khetarpal, and Majid showed that the semantic map inference problem is "formally identical to another problem that superficially appears unrelated: inferring a social network from outbreaks of disease in a population" (Regier et al., 2013: 91)

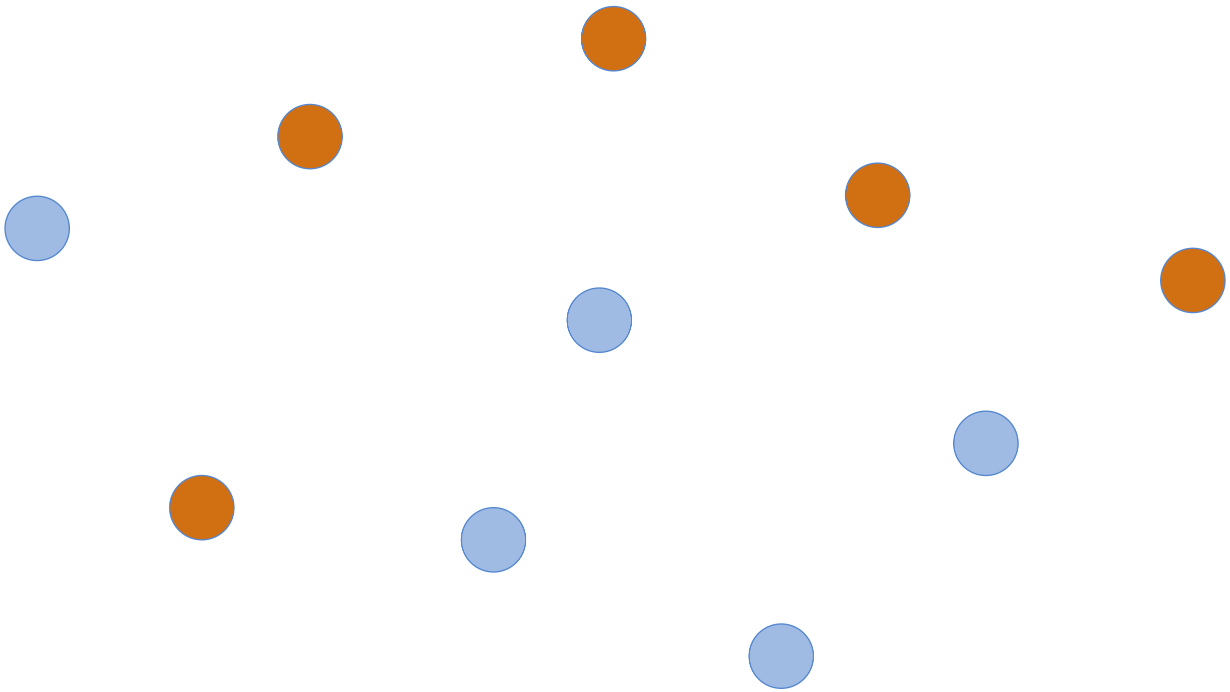
# Inferring semantic maps

- What's the idea?
  - Let's consider a group of social agents (represented by the nodes of a potential graph)



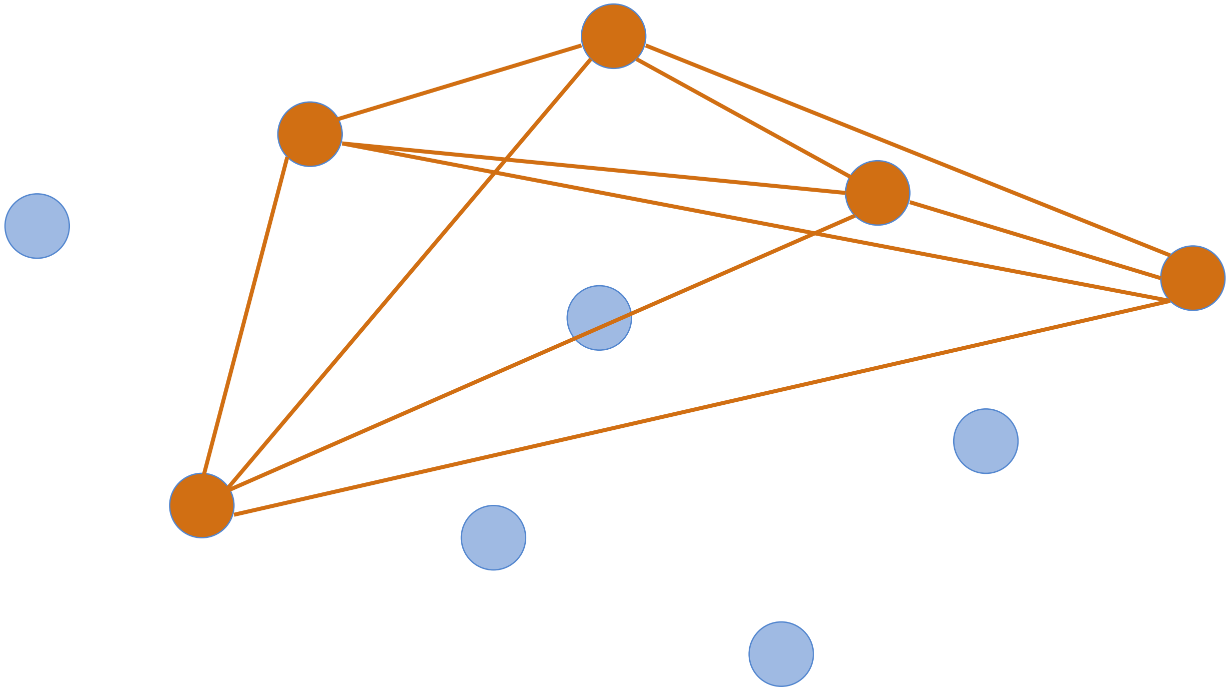
# Inferring semantic maps

- What's the idea?
  - If one observes the same disease for five of these agents (technically called a constraint on the nodes of the graph)



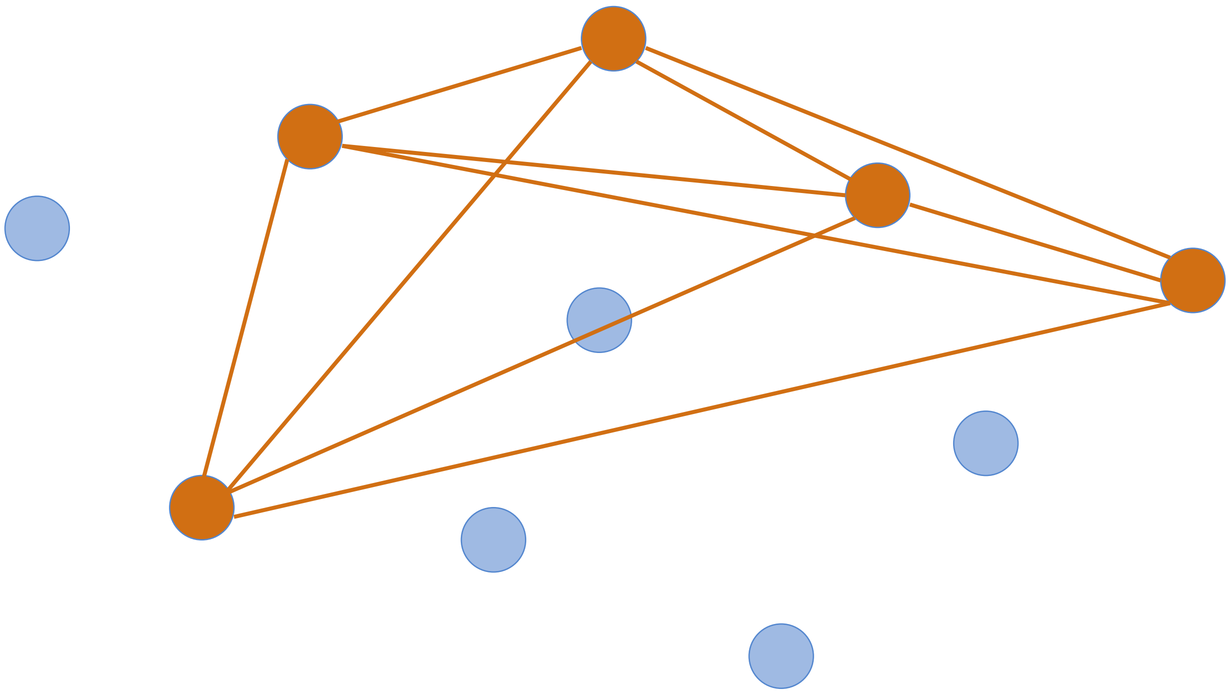
# Inferring semantic maps

- What's the idea?
  - One can postulate that all the agents met, so that all the nodes of the graph are connected (10 edges between the 5 nodes)



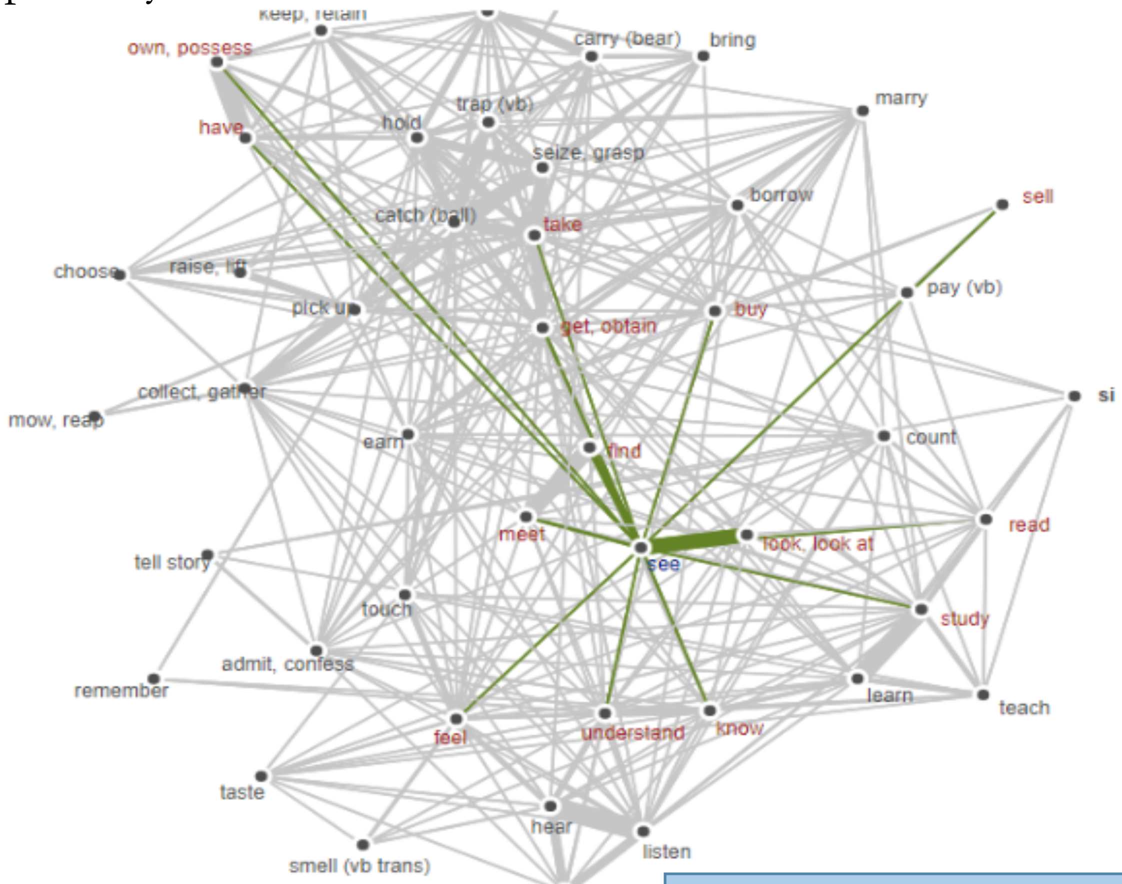
# Inferring semantic maps

- What's the idea?
  - This is neither a very likely, nor a very economic explanation



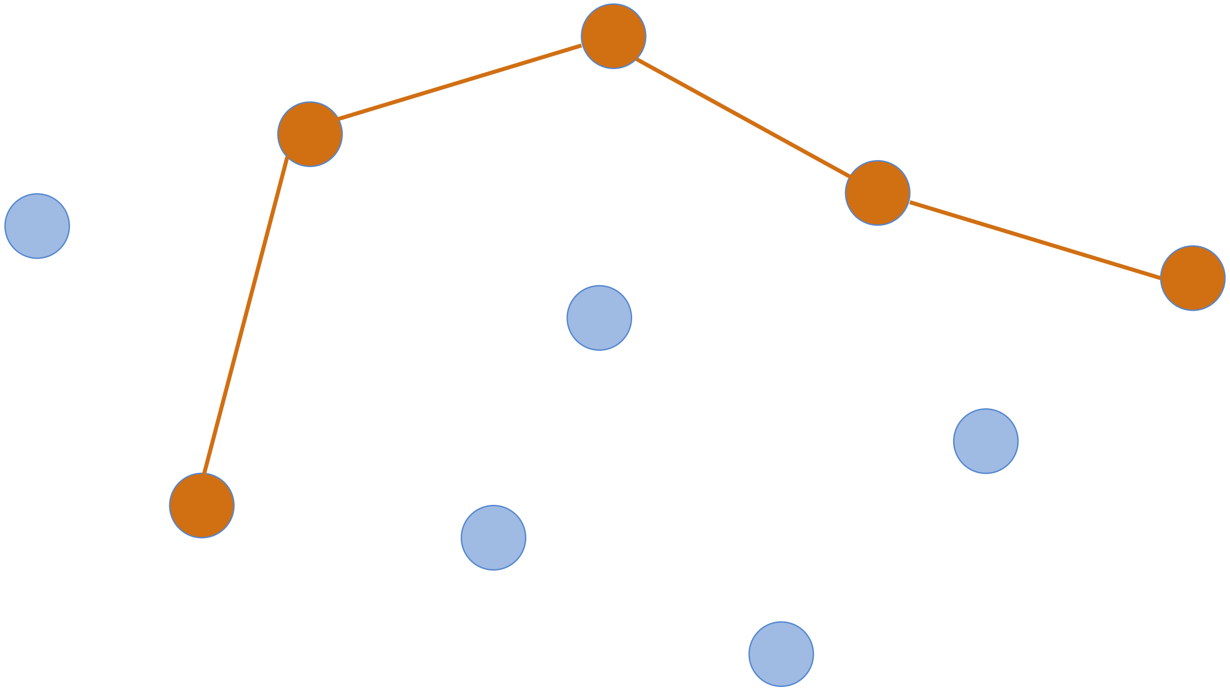
# Inferring semantic maps

- What's the idea?
  - But this is precisely what a colexification network does



# Inferring semantic maps

- What's the idea?
  - The goal would be to find a more economical solution and to have all the social agents connected with as few edges as possible



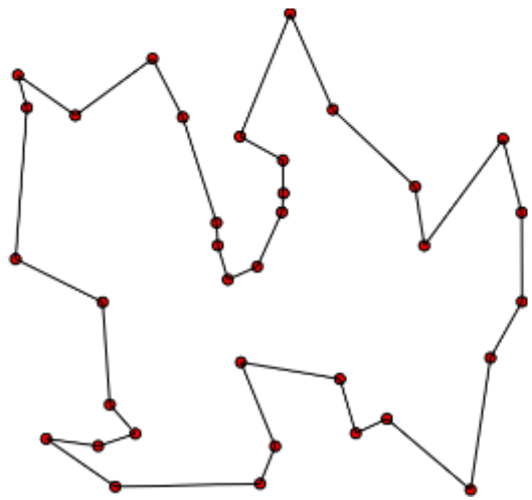
# Inferring semantic maps

- What's the idea?
  - Such a Network Inference problem looks intuitively simple, but is computationally hard to solve
    - Cf. the travelling salesman problem [TSP]: "Given a list of cities and the distance between each pair of cities, what is the shortest possible route that visits each city exactly once?"

  - Angluin et al. (2010) concluded that the problem is indeed computationally intractable, but proposed an algorithm that approximates the optimal solution nearly as well as is theoretically possible

**Inferring Social Networks from Outbreaks**

Dana Angluin[1,*], James Aspnes[1,**], and Lev Reyzin[2,***]

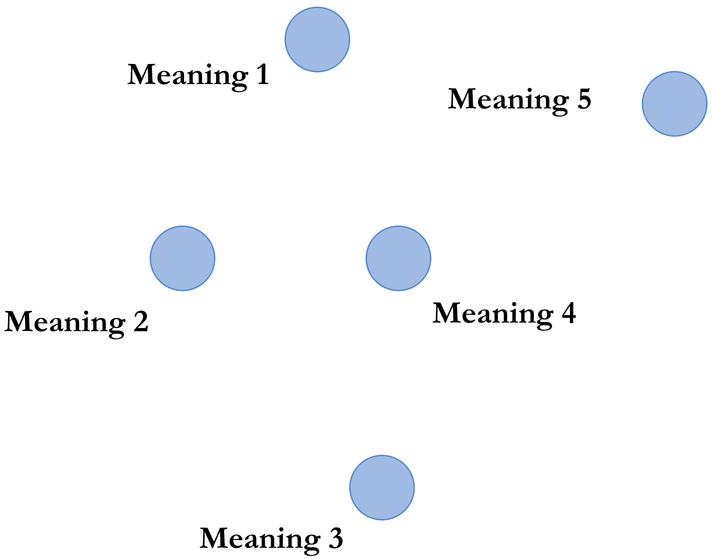


# Inferring semantic maps

- How does it transfer to semantic maps?

# Inferring semantic maps

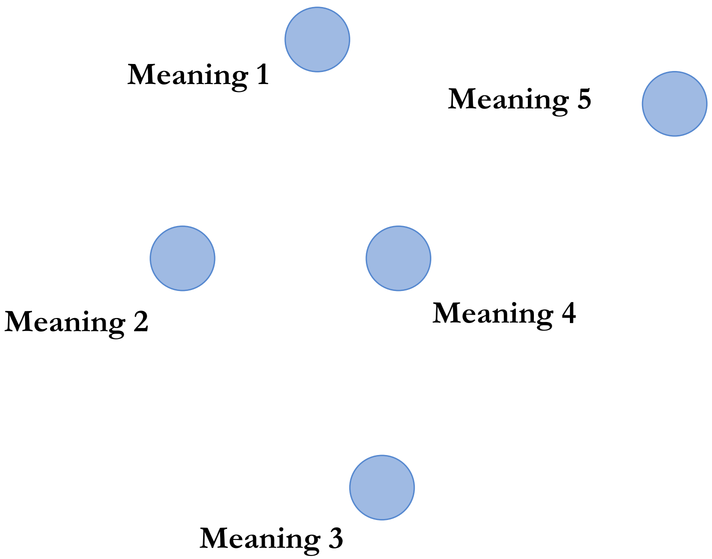
- How does it transfer to semantic maps?
  - Nodes are meanings




| Meaning | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|

# Inferring semantic maps

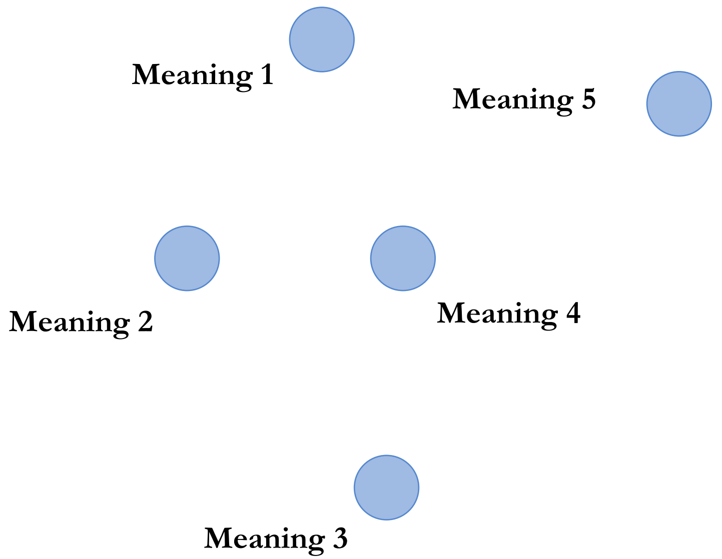
- How does it transfer to semantic maps?
  - Nodes are meanings
  - Constraints are Polysemic items




| Meaning | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| Polysemic item A | √ | √ | | | |
| Polysemic item B | | √ | √ | √ | |
| Polysemic item C | | | √ | √ | √ |

# Inferring semantic maps

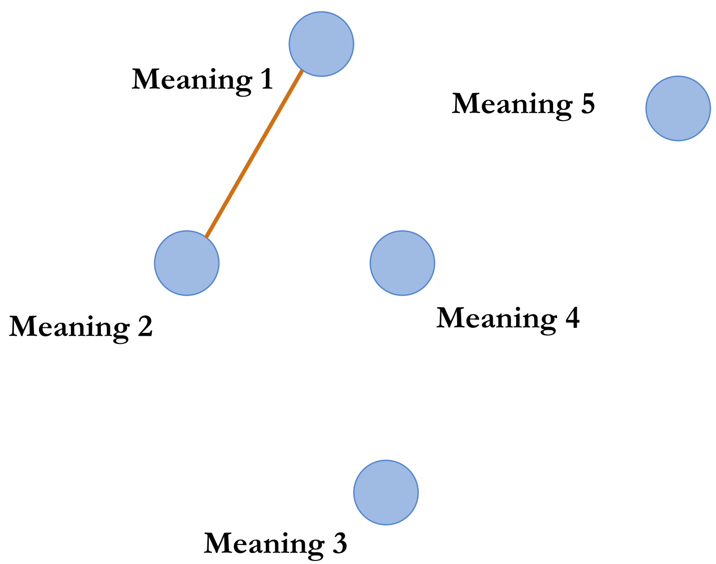
- How does it transfer to semantic maps?
  - Nodes are meanings
  - Constraints are Polysemic items
  - One connects the nodes economically based on these constraints




| Meaning | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| Polysemic item A | √ | √ | | | |
| Polysemic item B | | √ | √ | √ | |
| Polysemic item C | | | √ | √ | √ |

# Inferring semantic maps

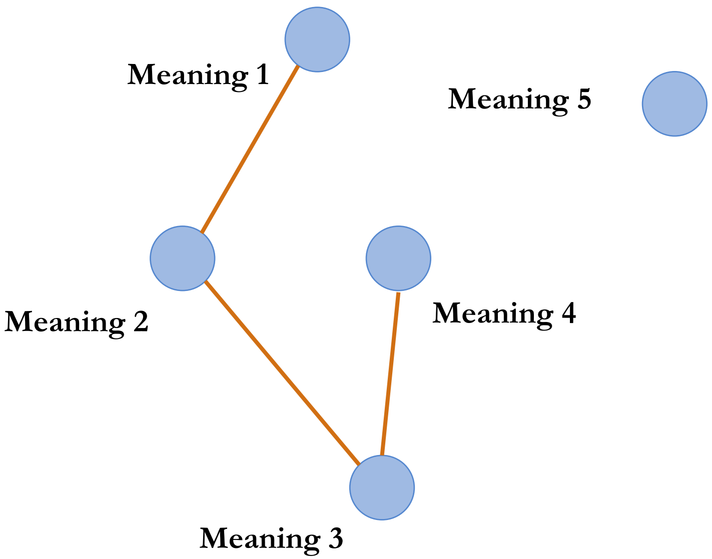
- How does it transfer to semantic maps?
  - Nodes are meanings
  - Constraints are Polysemic items
  - One connects the nodes economically based on these constraints




| Meaning | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| Polysemic item A | √ | √ | | | |

# Inferring semantic maps

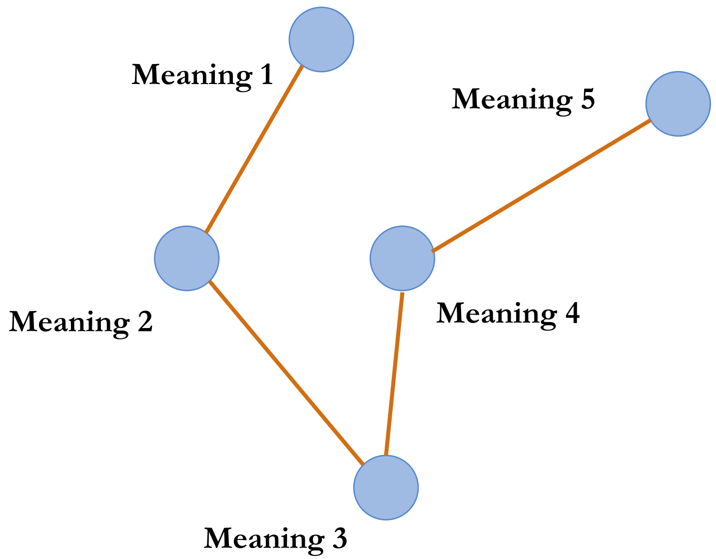
- How does it transfer to semantic maps?
  - Nodes are meanings
  - Constraints are Polysemic items
  - One connects the nodes economically based on these constraints




| Meaning | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| Polysemic item A | √ | √ | | | |
| Polysemic item B | | √ | √ | √ | |

# Inferring semantic maps

- How does it transfer to semantic maps?
  - Nodes are meanings
  - Constraints are Polysemic items
  - One connects the nodes economically based on these constraints

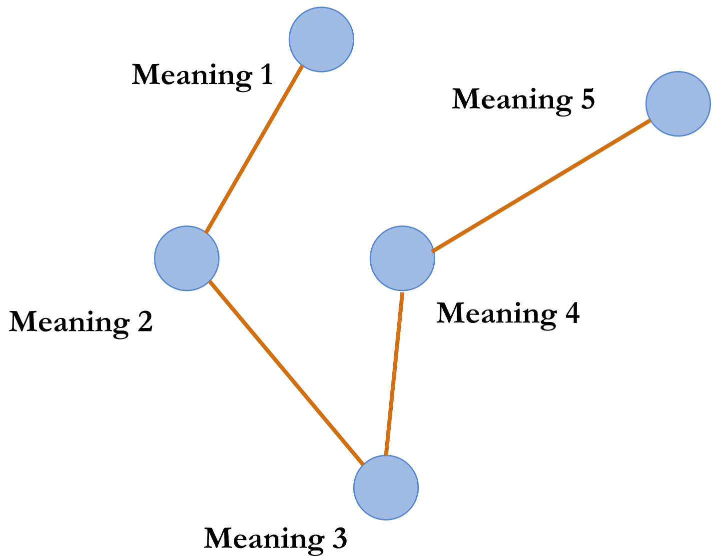



| Meaning | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| Polysemic item A | √ | √ | | | |
| Polysemic item B | | √ | √ | √ | |
| Polysemic item C | | | √ | √ | √ |

# Inferring semantic maps

- How does it transfer to semantic maps?

**The result is a map that accounts for all the polysemy patterns, while remaining as economic as possible**




| Meaning | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| Polysemic item A | √ | √ | | | |
| Polysemic item B | | √ | √ | √ | |
| Polysemic item C | | | √ | √ | √ |

# Inferring semantic maps

- Regier et al. (2013): the approximations produced by the Angluin et al. algorithm are of high quality
  - Tested on the crosslinguistic data of Haspelmath (1997) and Levinson et al. (2003)

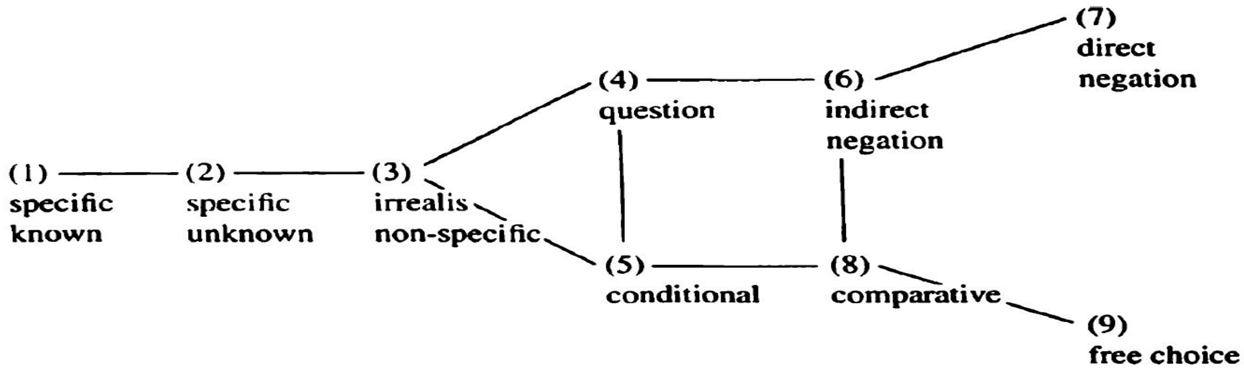



Figure. Haspelmath's (1997: 4) original semantic map of the indefinite pronouns functions

# Inferring semantic maps

INPUT
(lexical matrix)

| *Language* | *Word* | Specific Known | Specific Unknown | Irrealis Non-specific | Question | Conditional | Indirect Negation |
|---|---|---|---|---|---|---|---|
| | | **SK** | **SU** | **IR** | **QN** | **CD** | **IN** |
| German | "etwas" | 1 | 1 | 1 | 1 | 1 | 1 |
| German | "irgend" | 0 | 1 | 1 | 1 | 1 | 1 |
| German | "je" | 0 | 0 | 0 | 1 | 1 | 1 |
| German | "jeder" | 0 | 0 | 0 | 0 | 0 | 1 |
| German | "n-" | 0 | 0 | 0 | 0 | 0 | 0 |
| Dutch | "dan ook" | 0 | 0 | 1 | 1 | 1 | 1 |
| Dutch | "enig" | 0 | 0 | 0 | 1 | 1 | 1 |
| Dutch | "iets" | 1 | 1 | 1 | 1 | 1 | 1 |
| Dutch | "niets" | 0 | 0 | 0 | 0 | 0 | 0 |
| English | "any" | 0 | 0 | 0 | 1 | 1 | 1 |
| English | "ever" | 0 | 0 | 0 | 1 | 1 | 1 |
| English | "no" | 0 | 0 | 0 | 0 | 0 | 0 |
| English | "some" | 1 | 1 | 1 | 1 | 1 | 0 |

# Inferring semantic maps

INPUT
(lexical matrix)

| *Language* | *Word* | Specific Known<br>**SK** | Specific Unknown<br>**SU** | Irrealis Non-specific<br>**IR** | Question<br>**QN** | Conditional<br>**CD** | Indirect Negation<br>**IN** |
|---|---|---|---|---|---|---|---|
| German | "etwas" | 1 | 1 | 1 | 1 | 1 | 1 |
| German | "irgend" | 0 | 1 | 1 | 1 | 1 | 1 |
| German | "je" | 0 | 0 | 0 | 1 | 1 | 1 |
| German | "jeder" | 0 | 0 | 0 | 0 | 0 | 1 |
| German | "n-" | 0 | 0 | 0 | 0 | 0 | 0 |
| Dutch | "dan ook" | 0 | 0 | 1 | 1 | 1 | 1 |
| Dutch | "enig" | 0 | 0 | 0 | 1 | 1 | 1 |
| Dutch | "iets" | 1 | 1 | 1 | 1 | 1 | 1 |
| Dutch | "niets" | 0 | 0 | 0 | 0 | 0 | 0 |
| English | "any" | 0 | 0 | 0 | 1 | 1 | 1 |
| English | "ever" | 0 | 0 | 0 | 1 | 1 | 1 |
| English | "no" | 0 | 0 | 0 | 0 | 0 | 0 |
| English | "some" | 1 | 1 | 1 | 1 | 1 | 0 |

ALGORITHM
(python script)

```
# MAIN LOOP
objfn = C(G,T)
while (objfn < 0):
    print ("objective fn is currently", objfn,)
    max_score = 0
    # choose next edge greedily: the one that increases objfn the most
    for e in PossE:
        # temporarily add e to graph G
        G.add_edge(*e)
        score = C(G,T) - objfn
        G.remove_edge(*e)
        if (score > max_score):
            max_score = score
            max_edge = e
```

# Inferring semantic maps

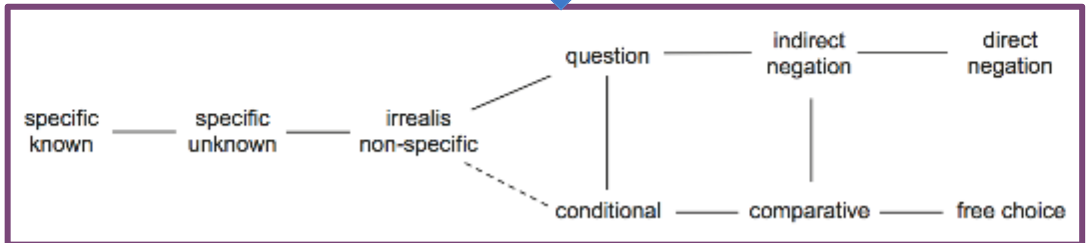
INPUT
(lexical matrix)

| *Language* | *Word* | Specific Known **SK** | Specific Unknown **SU** | Irrealis Non-specific **IR** | Question **QN** | Conditional **CD** | Indirect Negation **IN** |
|---|---|---|---|---|---|---|---|
| German | "etwas" | 1 | 1 | 1 | 1 | 1 | 1 |
| German | "irgend" | 0 | 1 | 1 | 1 | 1 | 1 |
| German | "je" | 0 | 0 | 0 | 1 | 1 | 1 |
| German | "jeder" | 0 | 0 | 0 | 0 | 0 | 1 |
| German | "n-" | 0 | 0 | 0 | 0 | 0 | 0 |
| Dutch | "dan ook" | 0 | 0 | 1 | 1 | 1 | 1 |
| Dutch | "enig" | 0 | 0 | 0 | 1 | 1 | 1 |
| Dutch | "iets" | 1 | 1 | 1 | 1 | 1 | 1 |
| Dutch | "niets" | 0 | 0 | 0 | 0 | 0 | 0 |
| English | "any" | 0 | 0 | 0 | 1 | 1 | 1 |
| English | "ever" | 0 | 0 | 0 | 1 | 1 | 1 |
| English | "no" | 0 | 0 | 0 | 0 | 0 | 0 |
| English | "some" | 1 | 1 | 1 | 1 | 1 | 0 |

ALGORITHM
(python script)

```
# MAIN LOOP
objfn = C(G,T)
while (objfn < 0):
    print ("objective fn is currently", objfn,)
    max_score = 0
    # choose next edge greedily: the one that increases objfn the most
    for e in PossE:
        # temporarily add e to graph G
        G.add_edge(*e)
        score = C(G,T) - objfn
        G.remove_edge(*e)
        if (score > max_score):
            max_score = score
            max_edge = e
```

RESULT
(semantic map)



# Automatic plotting: Two steps forward

- Weighted semantic maps are much more informative than regular semantic maps, because they visually provide information about the frequency of polysemy patterns

- Diachronic semantic maps are much more informative than regular semantic maps, because they visually provide information about possible pathways of change

> "[T]he best synchronic semantic map is a diachronic one"
> (van der Auwera 2008: 43)

# Automatic plotting: Two steps forward

## Weighted semantic maps

- Generate the map with a modified version of the algorithm of Regier et al. (2013)
  - PRINCIPLE: for each edge that is being added between two meanings of the map by the algorithm, check in the lexical matrix how many times this specific polysemy pattern is attested, and increase the weight of the edge accordingly

```
edgeWeight = 0
for sns in sensesTupleList:
        if (max_edge[0] in sns) and (max_edge[1] in sns):
                edgeWeight += 1
G.add_edge(*max_edge,weight=edgeWeight)
```

# Automatic plotting: Two steps forward

## Weighted semantic maps

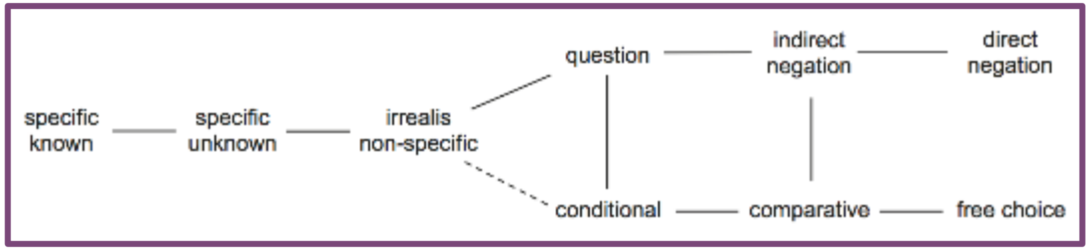
- Generate the map with a modified version of the algorithm of Regier et al. (2013)
  - PRINCIPLE: for each edge that is being added between two meanings of the map by the algorithm, check in the lexical matrix how many times this specific polysemy pattern is attested, and increase the weight of the edge accordingly

```
edgeWeight = 0
for sns in sensesTupleList:
        if (max_edge[0] in sns) and (max_edge[1] in sns):
                edgeWeight += 1
G.add_edge(*max_edge,weight=edgeWeight)
```

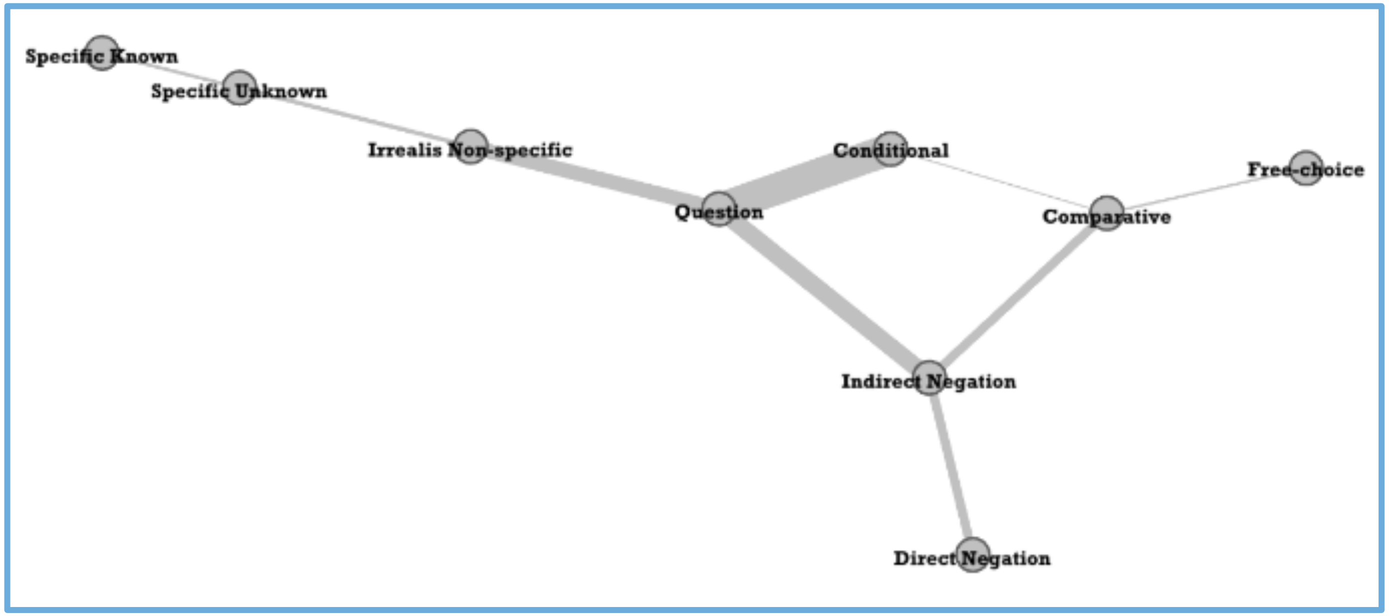
  - Based on the data of Haspelmath (1997), kindly provided by the author, the result between a non-weighted and a weighted semantic map are markedly different

# Automatic plotting: Two steps forward

## Weighted semantic maps



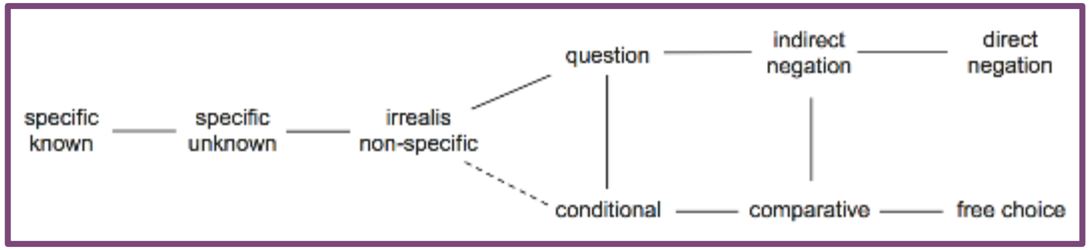

Automatically plotted semantic maps:
non-weighted vs. weighted
(data from Haspelmath 1997)



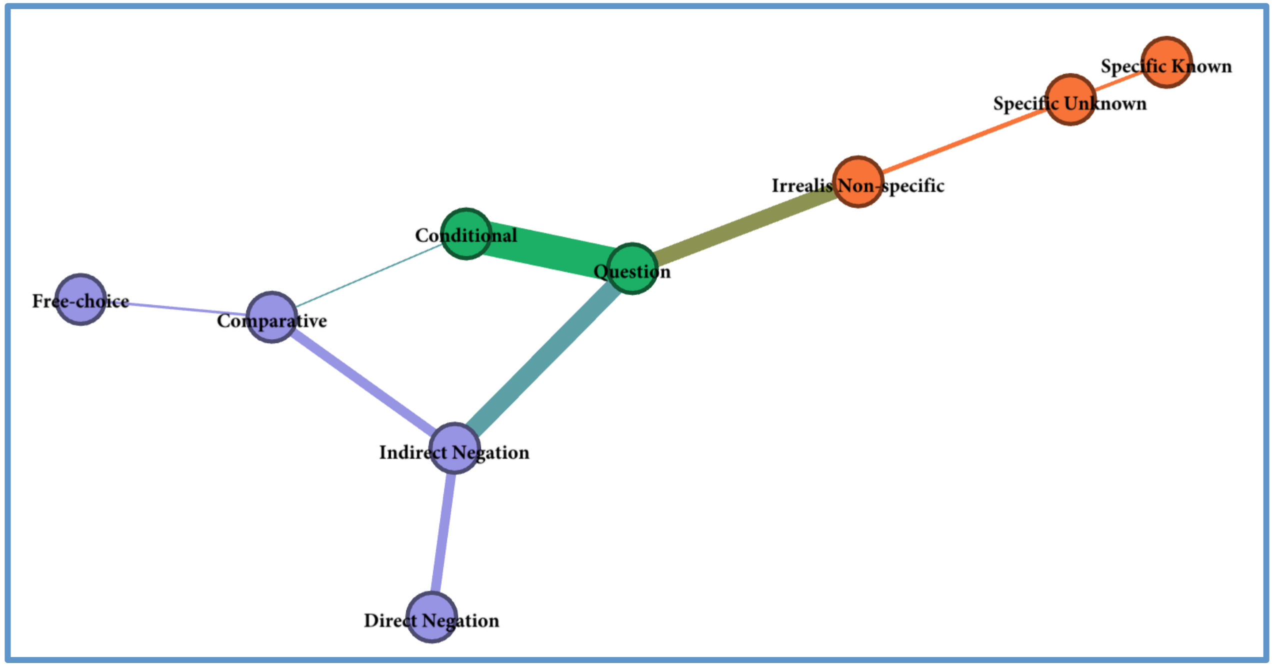

The graph is visualized in Gephi® with the *Force Atlas* algorithm

# Automatic plotting: Two steps forward

## Weighted semantic maps




Automatically plotted semantic maps:
non-weighted vs. weighted
(data from Haspelmath 1997)




The graph is visualized in Gephi® with the *Force Atlas* algorithm and modularity analysis (Lambiotte et al. 2009)

# Automatic plotting: Two steps forward

## Diachronic semantic maps

- Expand the lexical matrix so as to include information about diachrony

| *Source of constraint* | *Constraint name* | *Constraint Time* | Sense_1 Tree | Sense_2 Wood | Sense_3 Forest |
|---|---|---|---|---|---|
| Language_1 | Word_1 | 0 | 1 | 0 | 0 |
| Language_1 | Word_1 | 1 | 1 | 1 | 0 |
| Language_2 | Word_1 | 0 | 1 | 0 | 0 |
| Language_2 | Word_2 | 0 | 0 | 1 | 0 |
| Language_2 | Word_2 | 1 | 0 | 1 | 1 |
| Language_3 | Word_1 | 0 | 1 | 1 | 0 |
| Language_3 | Word_2 | 0 | 0 | 0 | 1 |

# Automatic plotting: Two steps forward

## Diachronic semantic maps

- Expand the lexical matrix so as to include information about diachrony

| *Source of constraint* | *Constraint name* | *Constraint Time* | Sense_1 Tree | Sense_2 Wood | Sense_3 Forest |
|---|---|---|---|---|---|
| Language_1 | Word_1 | 0 | 1 | 0 | 0 |
| Language_1 | Word_1 | 1 | 1 | 1 | 0 |
| Language_2 | Word_1 | 0 | 1 | 0 | 0 |
| Language_2 | Word_2 | 0 | 0 | 1 | 0 |
| Language_2 | Word_2 | 1 | 0 | 1 | 1 |
| Language_3 | Word_1 | 0 | 1 | 1 | 0 |
| Language_3 | Word_2 | 0 | 0 | 0 | 1 |

The diachronic stages are arbitrarily indexed by numbers: 0, 1, 2, etc.

# Automatic plotting: Two steps forward

## Diachronic semantic maps

- Expand the lexical matrix so as to include information about diachrony

| *Source of constraint* | *Constraint name* | *Constraint Time* | Sense_1 Tree | Sense_2 Wood | Sense_3 Forest |
|---|---|---|---|---|---|
| Language_1 | Word_1 | 0 | 1 | 0 | 0 |
| Language_1 | Word_1 | 1 | 1 | 1 | 0 |
| Language_2 | Word_1 | 0 | 1 | 0 | 0 |
| Language_2 | Word_2 | 0 | 0 | 1 | 0 |
| Language_2 | Word_2 | 1 | 0 | 1 | 1 |
| Language_3 | Word_1 | 0 | 1 | 1 | 0 |
| Language_3 | Word_2 | 0 | 0 | 0 | 1 |

The meaning of a word can change from one stage to another (e.g., Word_2 of Language_2 expresses the meaning Wood during stage 0 and Wood & Forest during stage 1)

# Automatic plotting: Two steps forward

## Diachronic semantic maps

- Generate the graph with the algorithm of Regier et al. (2013)

# Automatic plotting: Two steps forward

## Diachronic semantic maps

- Generate the graph with the algorithm of Regier et al. (2013)
- Enrich the graph with oriented edges (where relevant)
    - PRINCIPLE: (1) we convert the undirected graph into a *directed graph*
      (2) for each edge in the graph, if the meaning of node A is attested for one diachronic stage, while the meaning of node B is not, check in the lexical matrix if there is a later diachronic stage of the same language for which this specific word has both meaning A and B (or just meaning B). If this is the case, we can infer a meaning extension from A to B.

```
H = G.to_directed()     # convert the graph 'G' into a directed Graph 'H' in order to explo
                        # all the possibilities as regards the relationship between the noc
                        # (i.e., both A -> B and B -> A for all the connected nodes, crucia
                        # not only A -> B)
nx.set_edge_attributes(H, 'type', 'undirected') # set the default value to "undirected" for

for u,v,e in H.edges(data=True):        # loop over all the edges in the DiGraph 'H'
        for t in T_Full:                # look at the metadata and senses for one line in T
                if t.count(u) == 1 and t.count(v) == 0: # if the meaning of node 'u' in the
                                                        # while the meaning of node 'v' is
```

# Automatic plotting: Two steps forward

## Diachronic semantic maps

INPUT
(diachronic
lexical matrix)

| *Source of constraint* | *Constraint name* | *Constraint Time* | Sense_1 Tree | Sense_2 Wood | Sense_3 Forest |
|---|---|---|---|---|---|
| Language_1 | Word_1 | 0 | 1 | 0 | 0 |
| Language_1 | Word_1 | 1 | 1 | 1 | 0 |
| Language_2 | Word_1 | 0 | 1 | 0 | 0 |
| Language_2 | Word_2 | 0 | 0 | 1 | 0 |
| Language_2 | Word_2 | 1 | 0 | 1 | 1 |
| Language_3 | Word_1 | 0 | 1 | 1 | 0 |
| Language_3 | Word_2 | 0 | 0 | 0 | 1 |

# Automatic plotting: Two steps forward

## Diachronic semantic maps

INPUT
(diachronic
lexical matrix)

| *Source of constraint* | *Constraint name* | *Constraint Time* | Sense_1 Tree | Sense_2 Wood | Sense_3 Forest |
|---|---|---|---|---|---|
| Language_1 | Word_1 | 0 | 1 | 0 | 0 |
| Language_1 | Word_1 | 1 | 1 | 1 | 0 |
| Language_2 | Word_1 | 0 | 1 | 0 | 0 |
| Language_2 | Word_2 | 0 | 0 | 1 | 0 |
| Language_2 | Word_2 | 1 | 0 | 1 | 1 |
| Language_3 | Word_1 | 0 | 1 | 1 | 0 |
| Language_3 | Word_2 | 0 | 0 | 0 | 1 |

ALGORITHM
(python script
for inferring
oriented edges)

```
H = G.to_directed()      # convert the graph 'G' into a directed Graph 'H' in order to explo
                         # all the possibilities as regards the relationship between the noc
                         # (i.e., both A -> B and B -> A for all the connected nodes, crucic
                         # not only A -> B)
nx.set_edge_attributes(H, 'type', 'undirected') # set the default value to "undirected" for

for u,v,e in H.edges(data=True):        # loop over all the edges in the DiGraph 'H'
        for t in T_Full:                # look at the metadata and senses for one line in T
                if t.count(u) == 1 and t.count(v) == 0: # if the meaning of node 'u' in the
                                                # while the meaning of node 'v' is
```

# Automatic plotting: Two steps forward

## Diachronic semantic maps

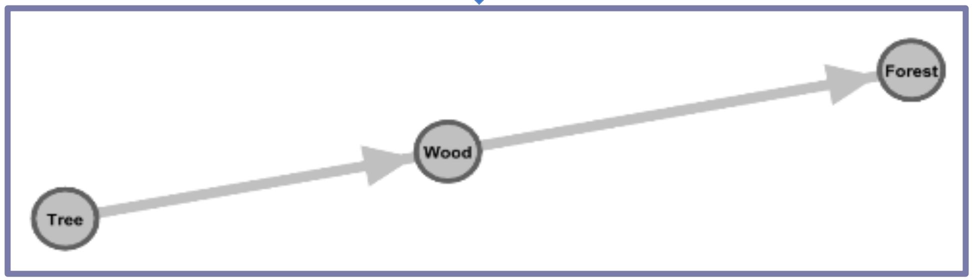
INPUT
(diachronic lexical matrix)

| *Source of constraint* | *Constraint name* | *Constraint Time* | Sense_1 Tree | Sense_2 Wood | Sense_3 Forest |
|---|---|---|---|---|---|
| Language_1 | Word_1 | 0 | 1 | 0 | 0 |
| Language_1 | Word_1 | 1 | 1 | 1 | 0 |
| Language_2 | Word_1 | 0 | 1 | 0 | 0 |
| Language_2 | Word_2 | 0 | 0 | 1 | 0 |
| Language_2 | Word_2 | 1 | 0 | 1 | 1 |
| Language_3 | Word_1 | 0 | 1 | 1 | 0 |
| Language_3 | Word_2 | 0 | 0 | 0 | 1 |

ALGORITHM
(python script for inferring oriented edges)

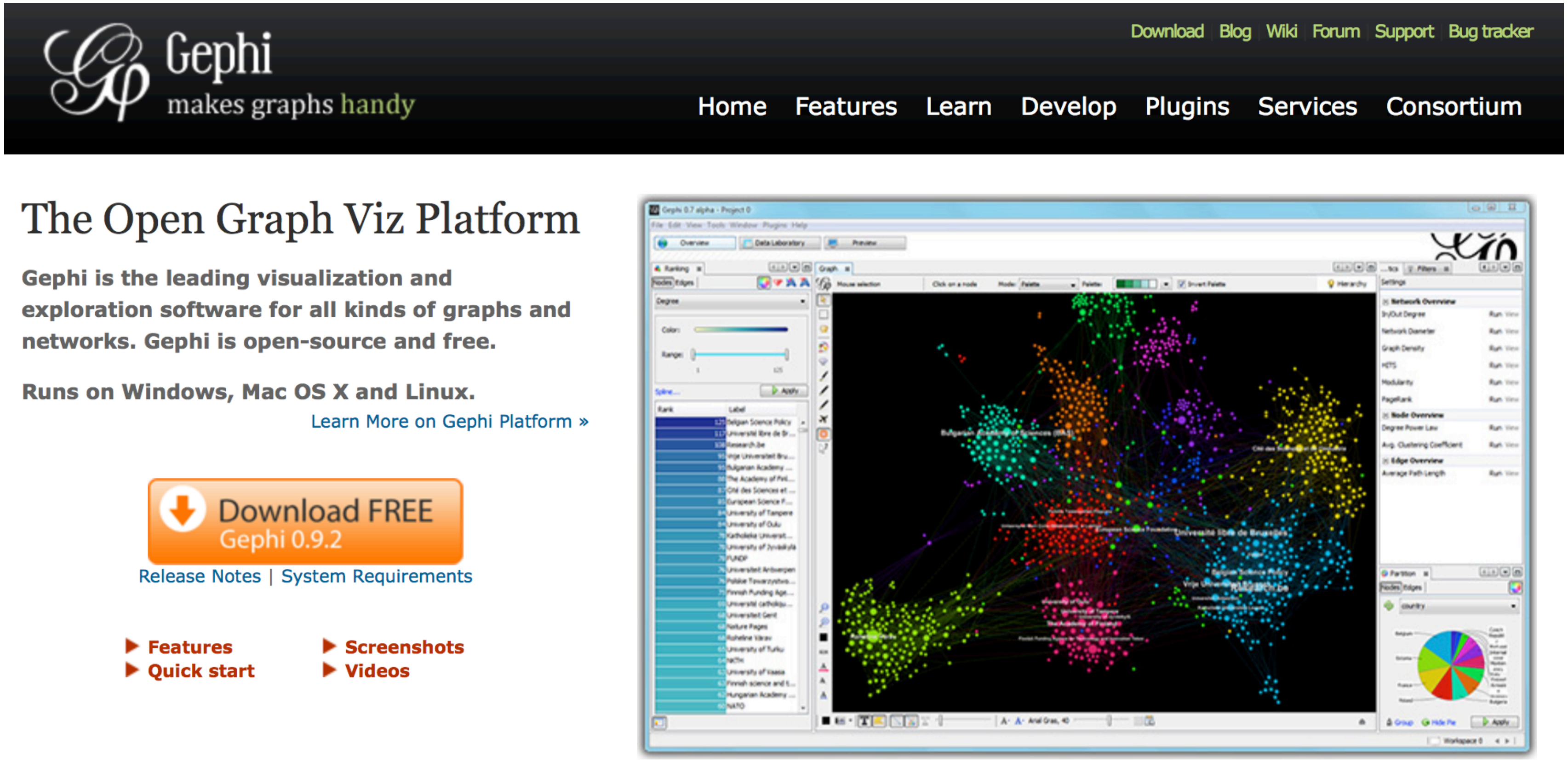
```
H = G.to_directed()      # convert the graph 'G' into a directed Graph 'H' in order to explo
                         # all the possibilities as regards the relationship between the noc
                         # (i.e., both A -> B and B -> A for all the connected nodes, crucic
                         # not only A -> B)
nx.set_edge_attributes(H, 'type', 'undirected') # set the default value to "undirected" for

for u,v,e in H.edges(data=True):        # loop over all the edges in the DiGraph 'H'
        for t in T_Full:                # look at the metadata and senses for one line in T
                if t.count(u) == 1 and t.count(v) == 0: # if the meaning of node 'u' in the
                                                         # while the meaning of node 'v' is
```

RESULT
(dynamic semantic map)

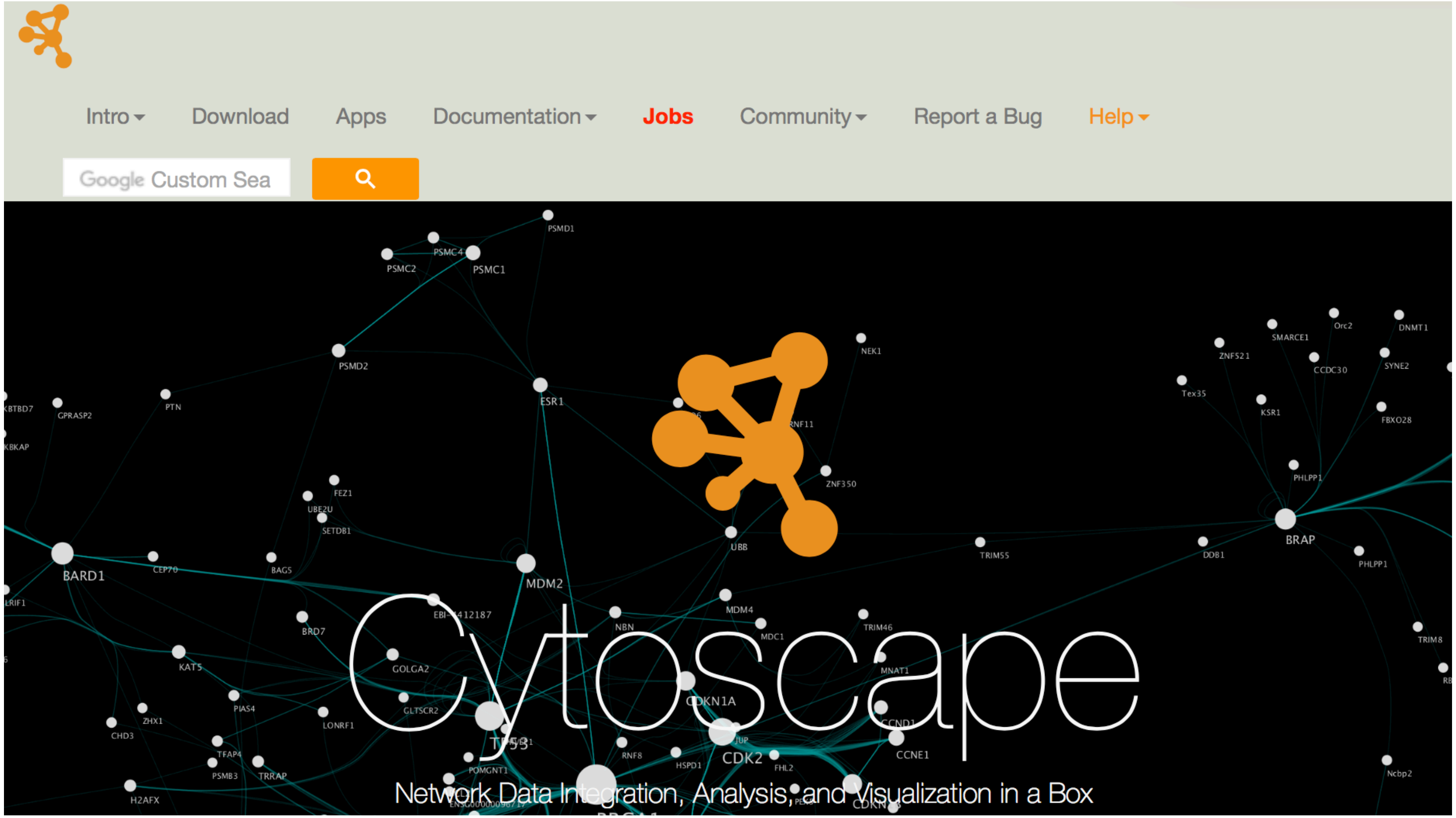


# Exploring automatically-plotted semantic maps

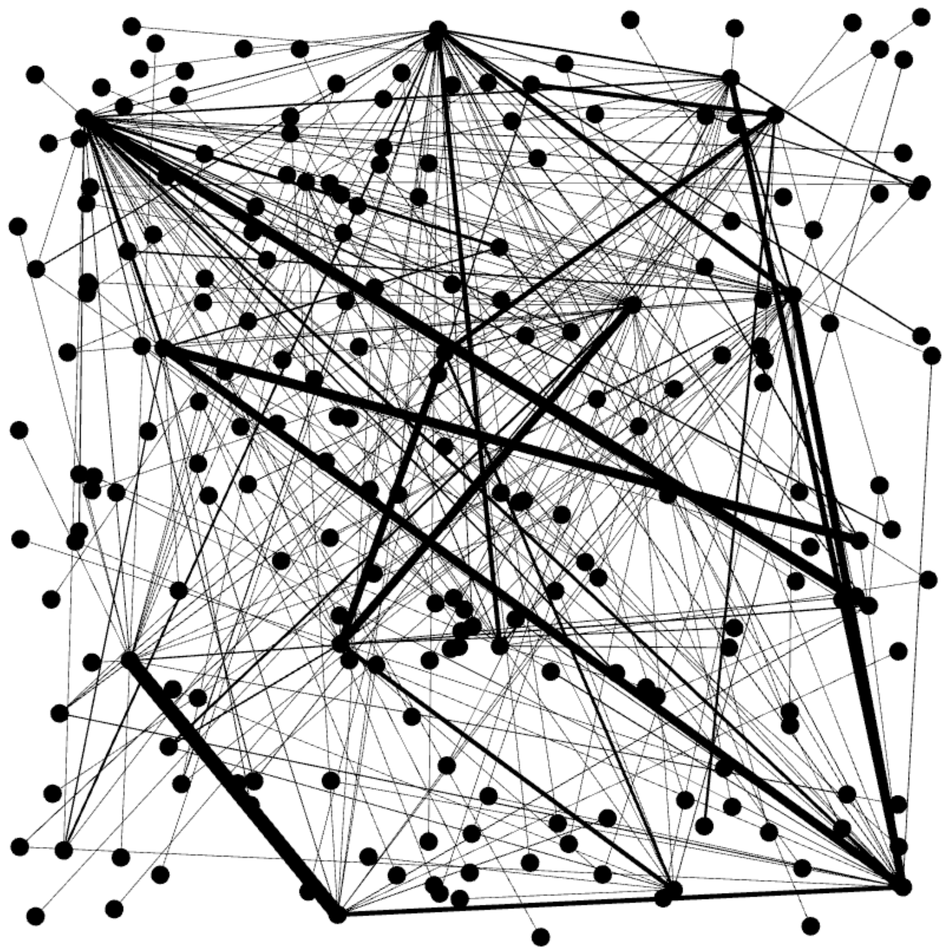
# Layout, weights, modularity



# Layout, weights, modularity

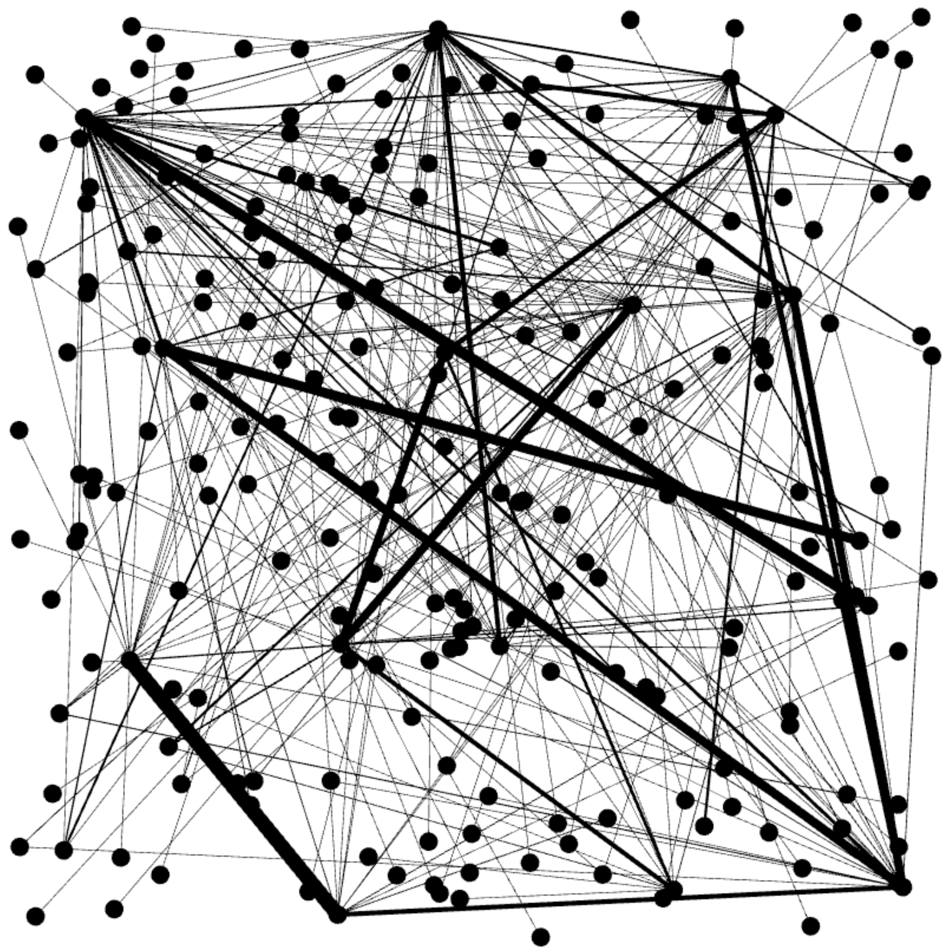


# Layout, weights, modularity

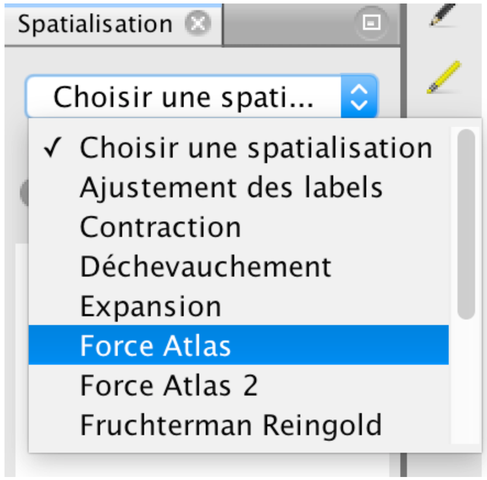


TIME-RELATED MEANINGS
(data from CLICS)

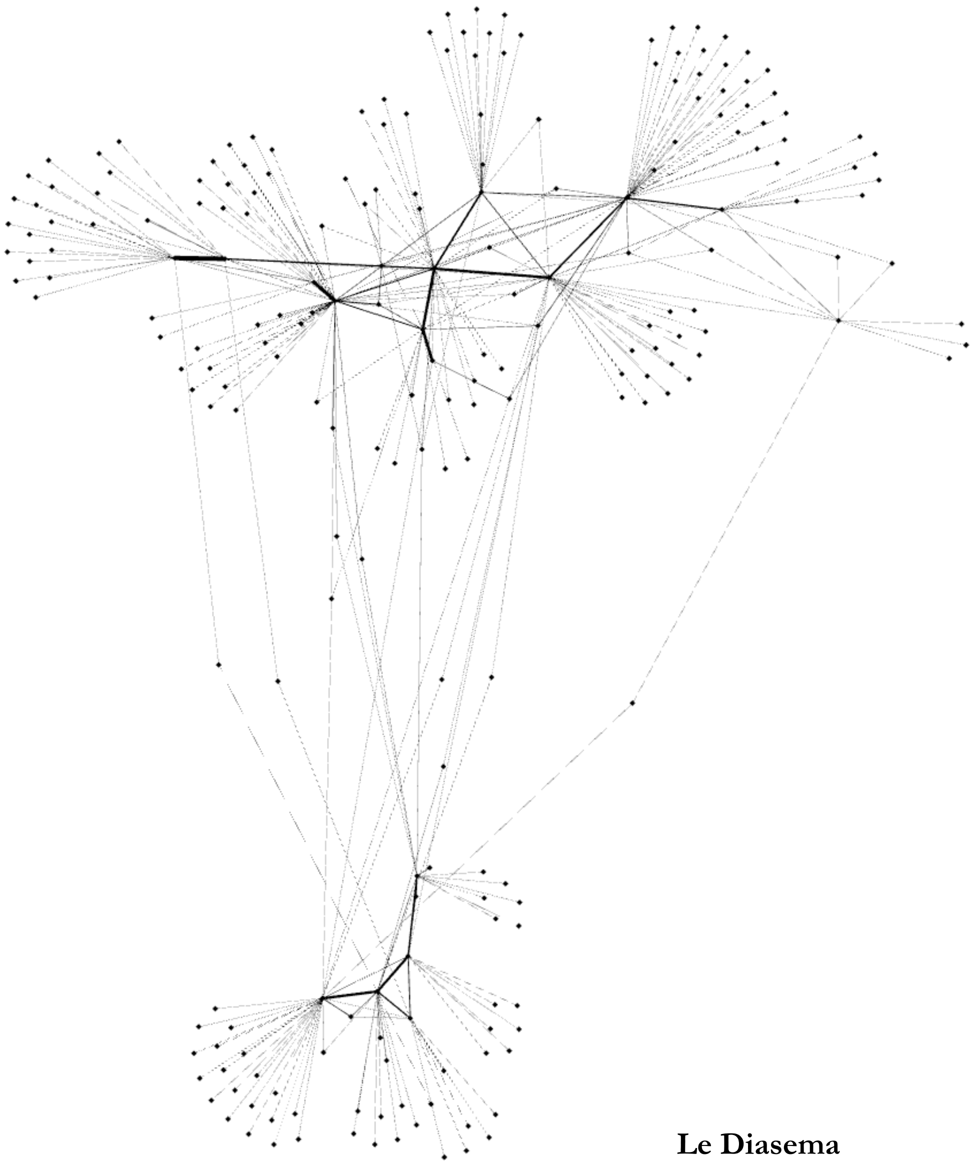
# Layout, weights, modularity



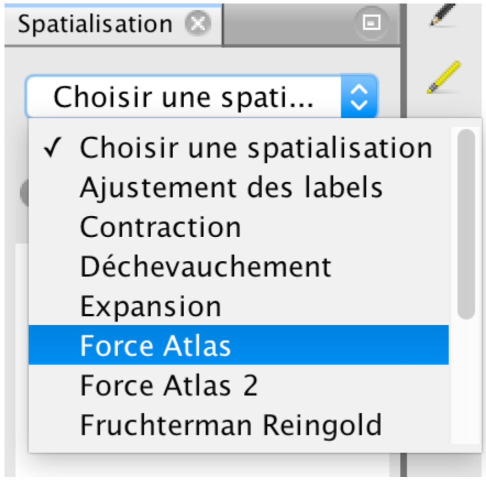
TIME-RELATED MEANINGS
(data from CLICS)



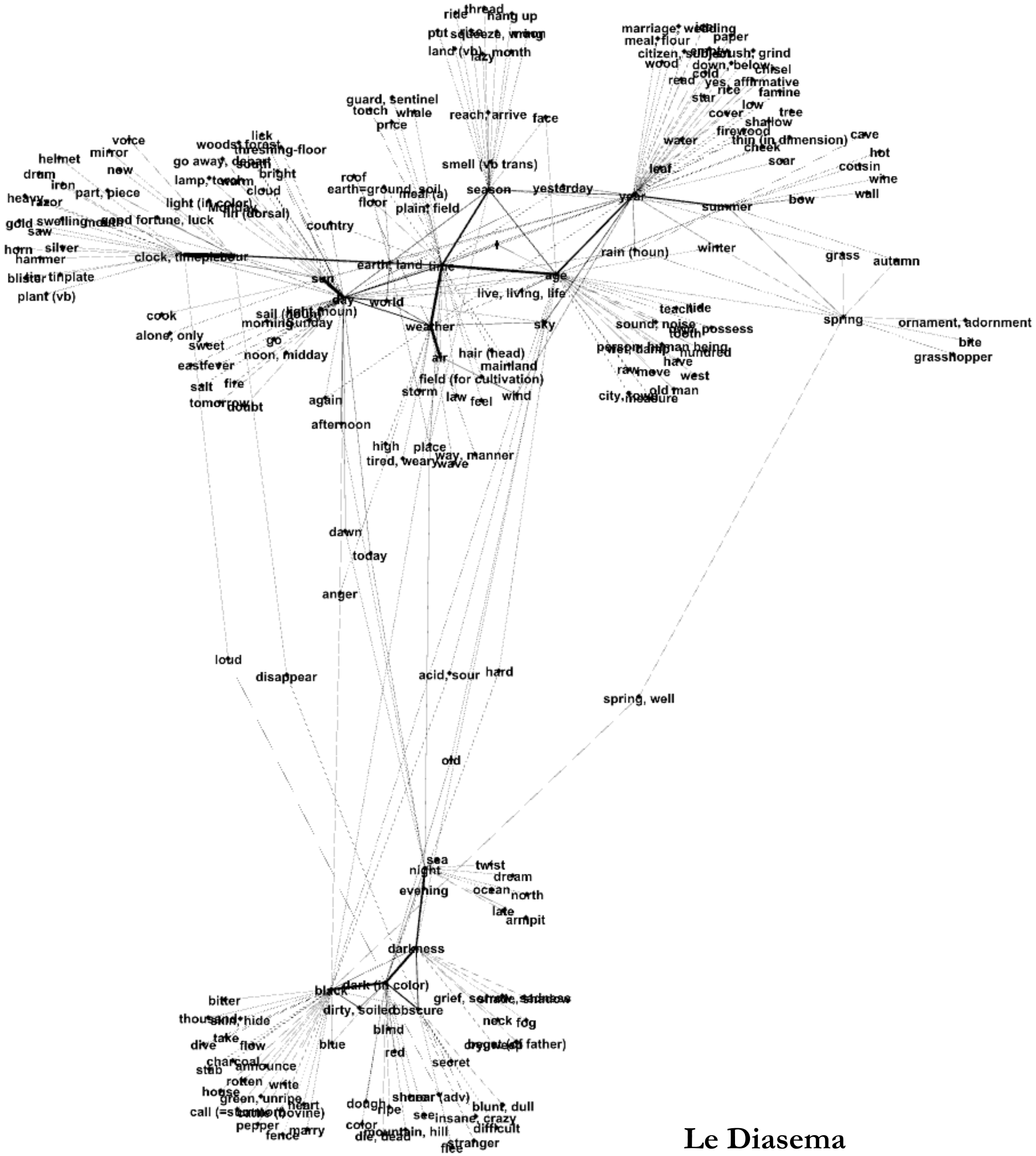
# Layout, weights, modularity



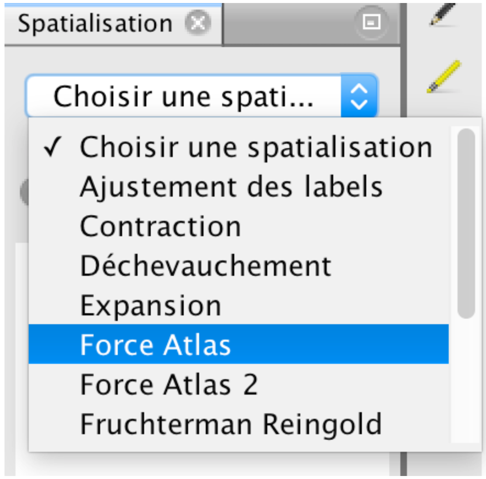
TIME-RELATED MEANINGS
(data from CLICS)



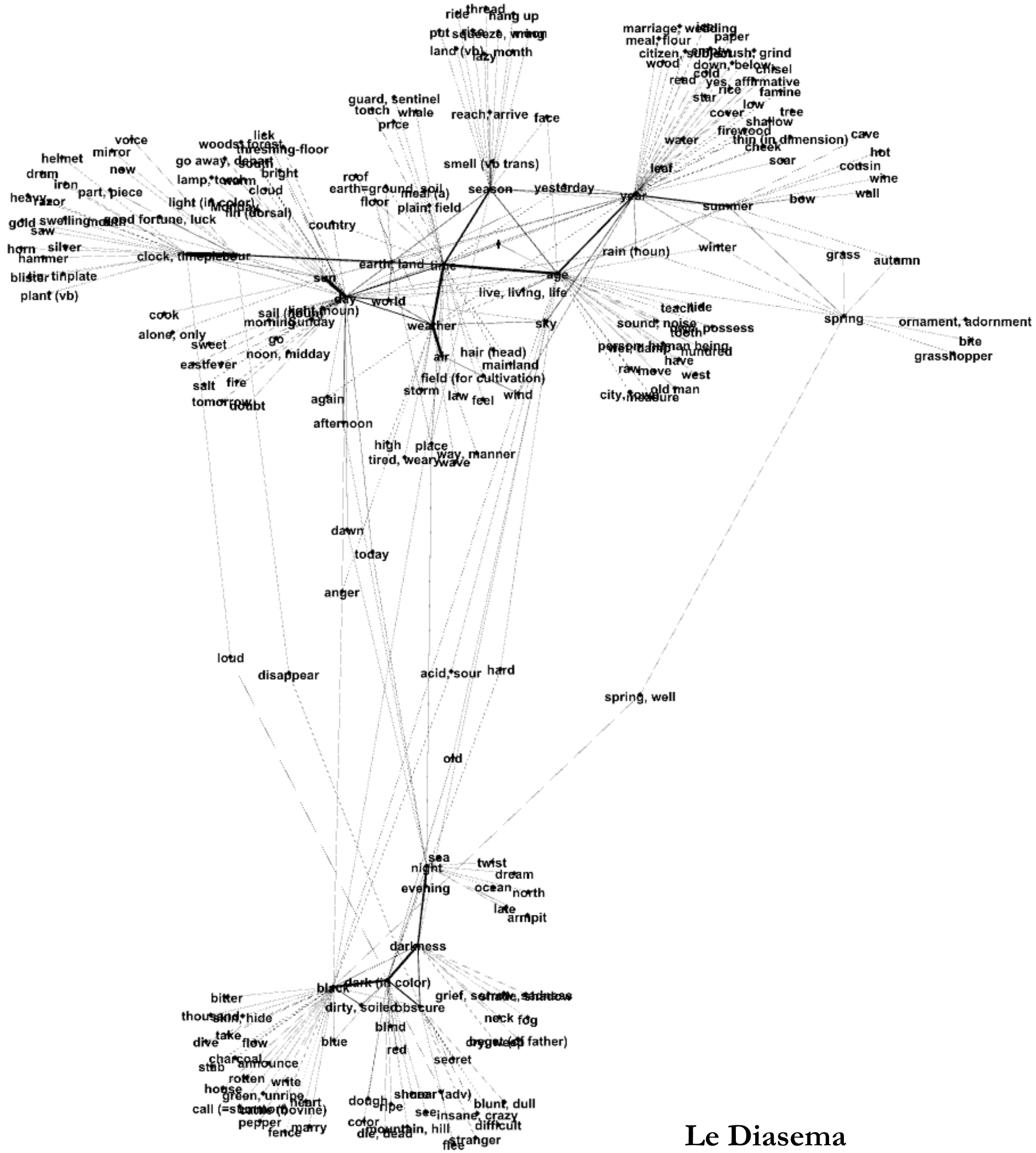
# Layout, weights, modularity



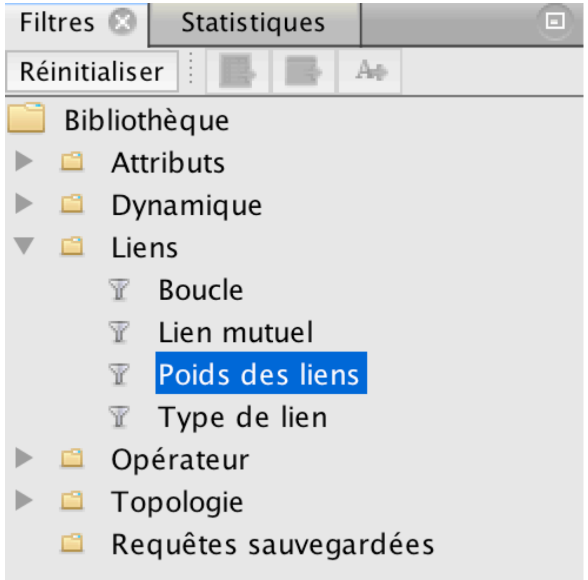
TIME-RELATED MEANINGS
(data from CLICS)



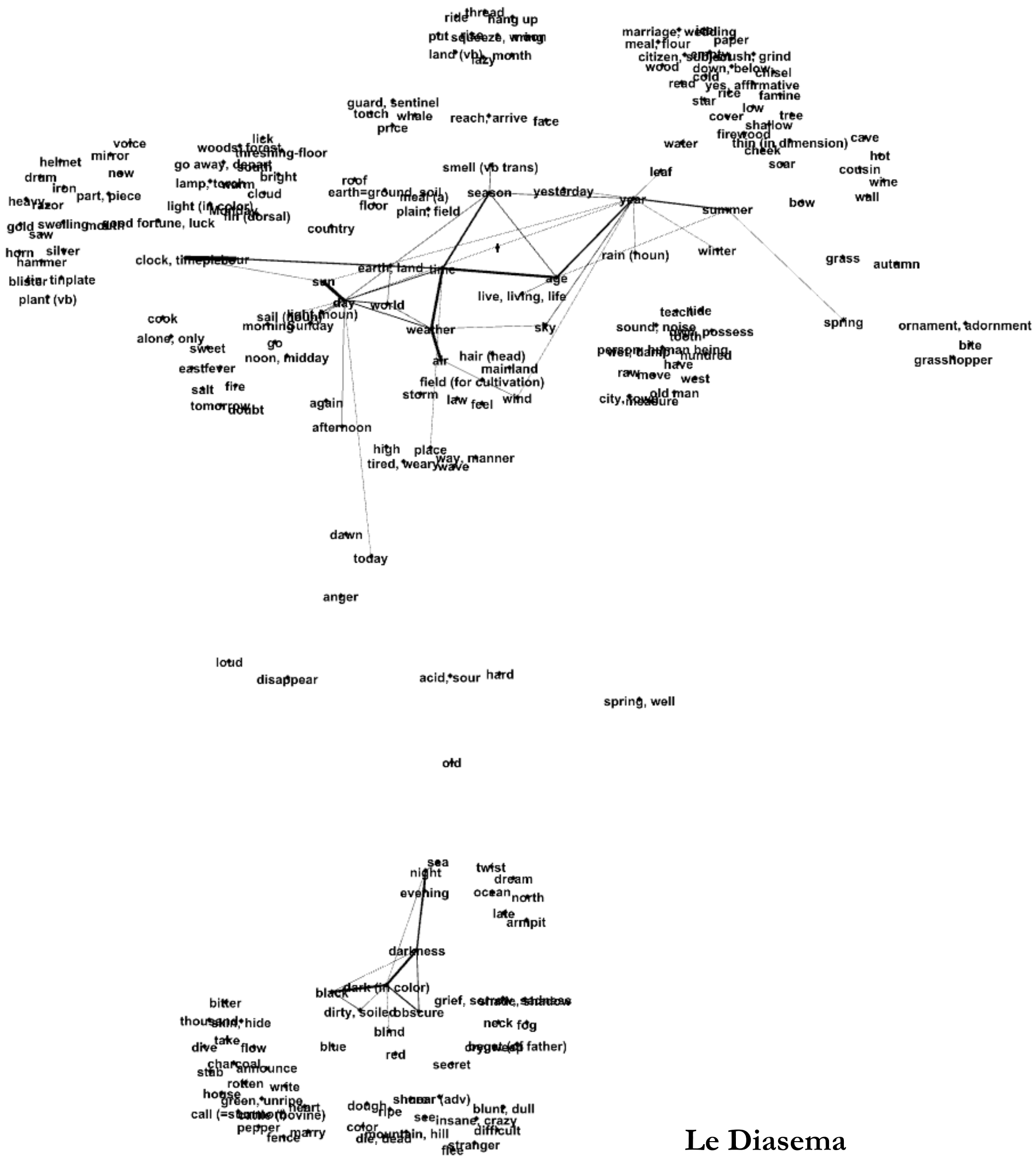
# Layout, weights, modularity



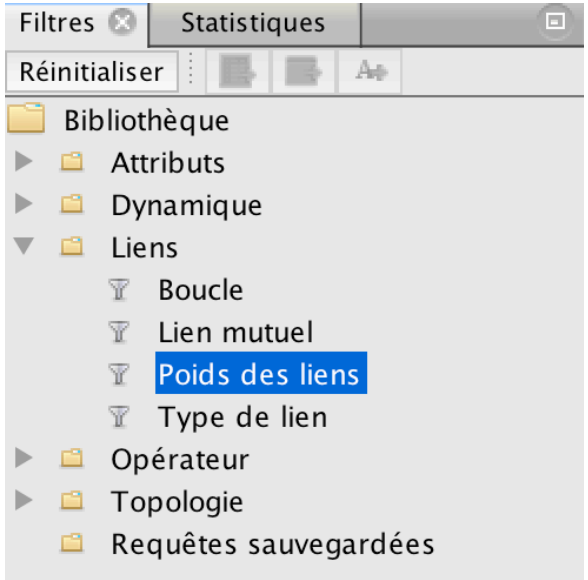

TIME-RELATED MEANINGS
(data from CLICS)



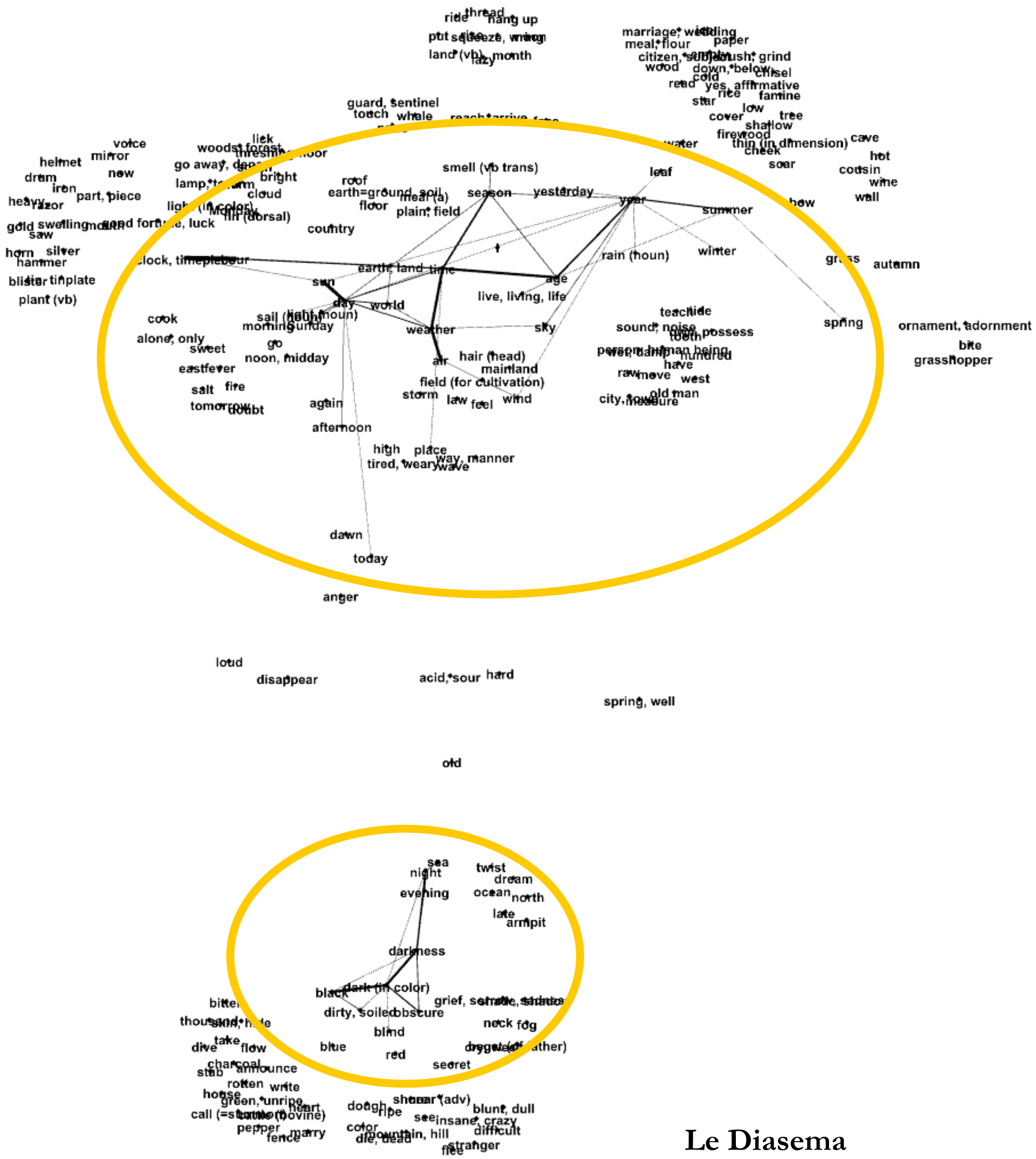
# Layout, weights, modularity



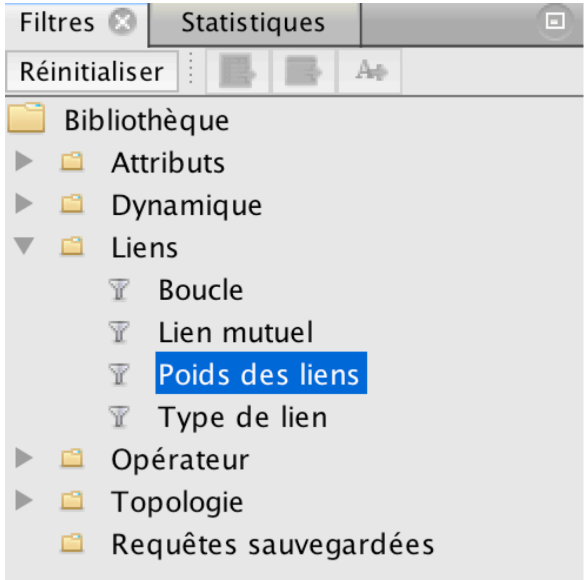

Time-related meanings
(data from CLICS)



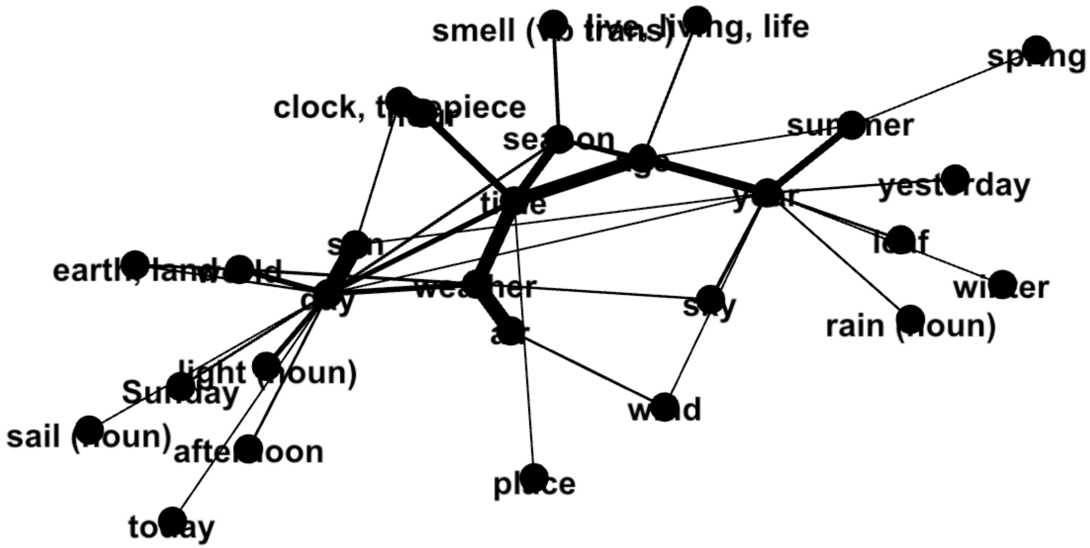
# Layout, weights, modularity



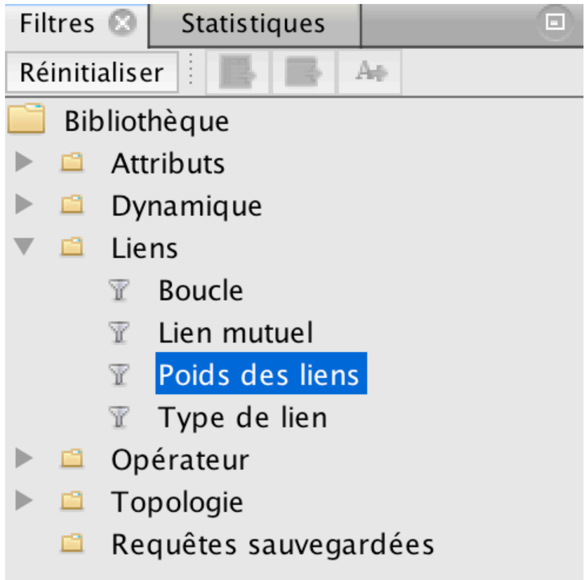

TIME-RELATED MEANINGS
(data from CLICS)



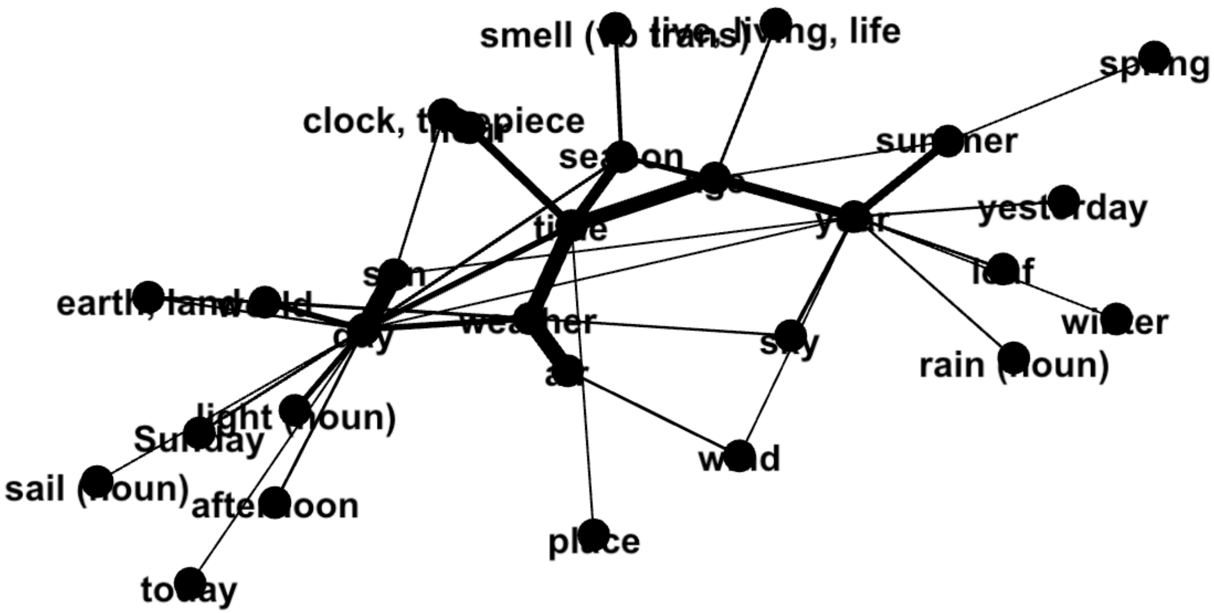
# Layout, weights, modularity

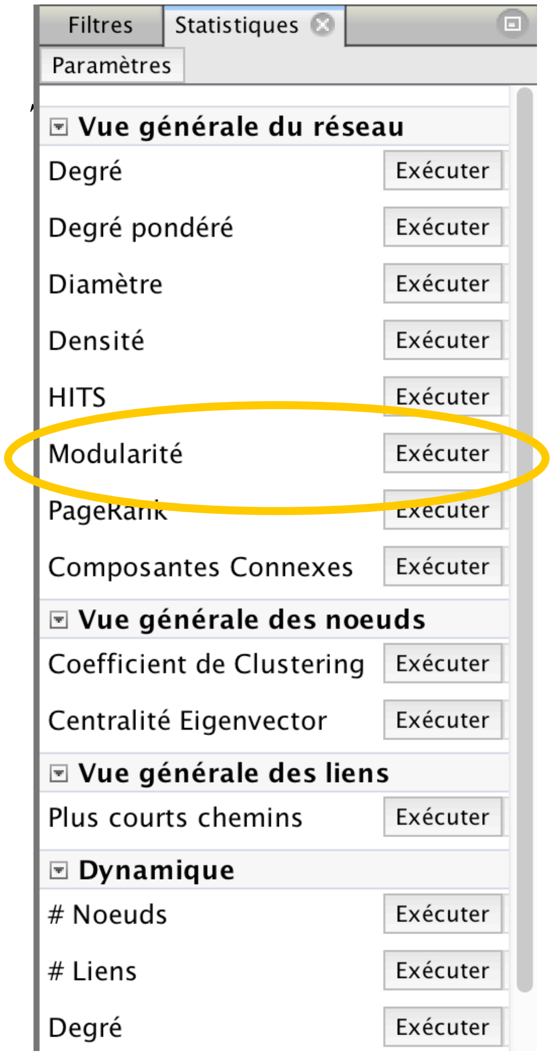


TIME-RELATED MEANINGS
(data from CLICS)

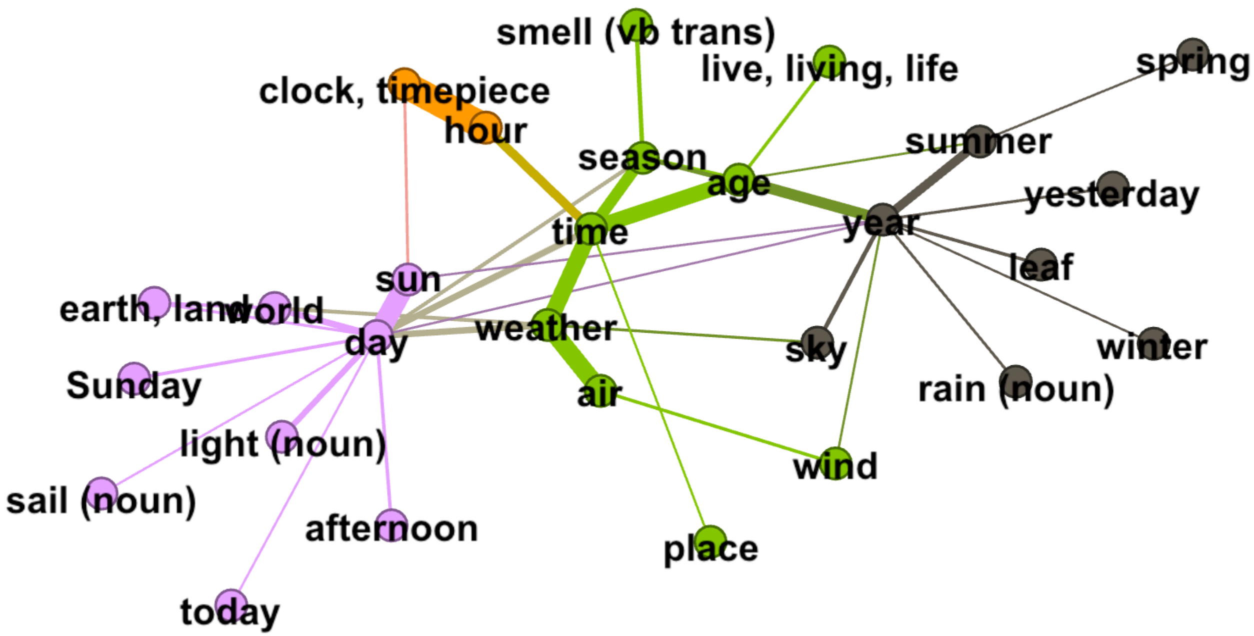


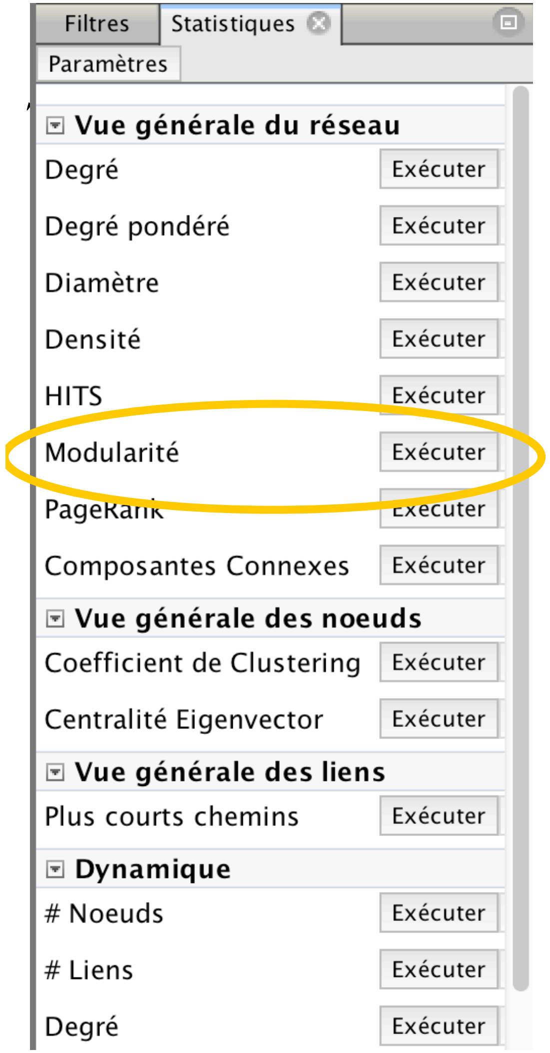
# Layout, weights, modularity



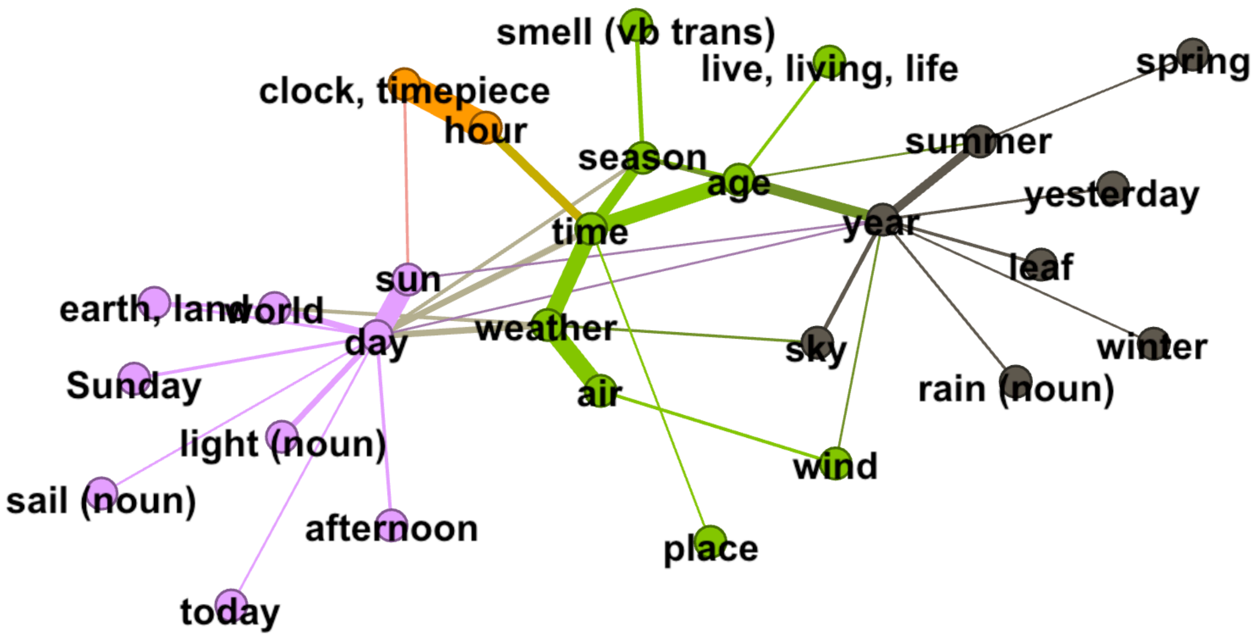


# Layout, weights, modularity



# Layout, weights, modularity




- ✓ Easy to read
- ✓ Generates interesting hypotheses and avenues for research in lexical typology

BUT

- ◆ The mapping of forms is hard to achieve; cf. Cysouw (2007) 'it overgenerates constellations of meaning'
- ◆ Hence, one cannot tell which patterns are precisely attested

# Methodological issues

# Methodological issues

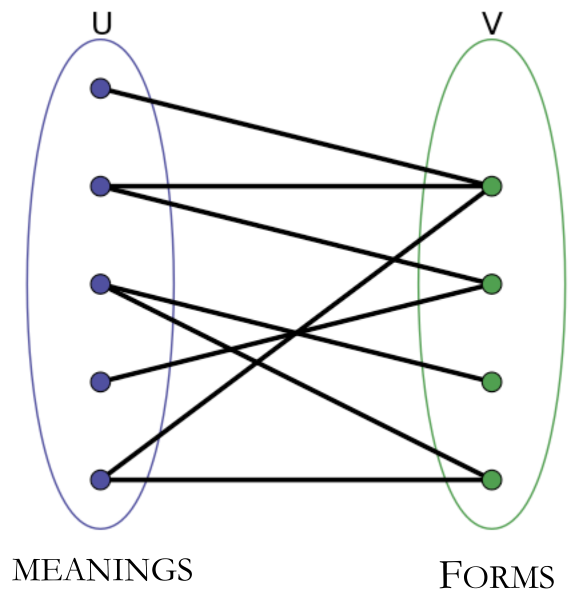
1. How can we visualize the types of polysemy patterns attested?

# Methodological issues

## 1. How can we visualize the types of polysemy patterns attested?

- Hill & List (2017): Bipartite networks

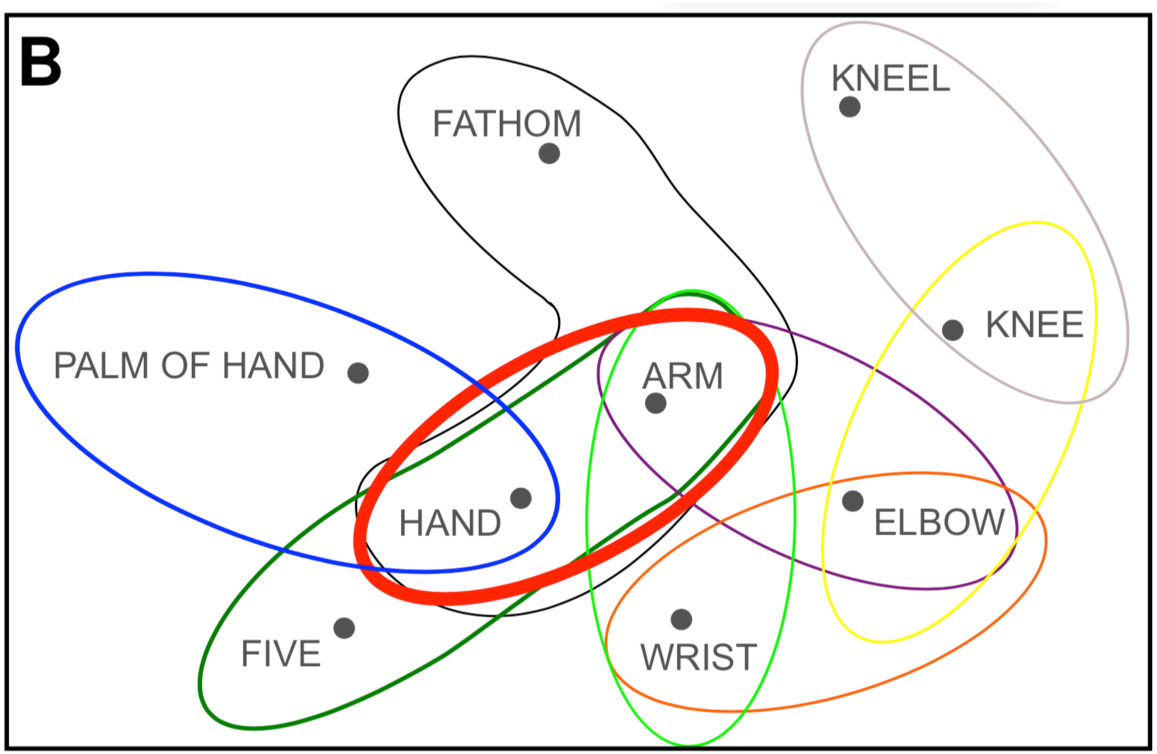
"Bipartite networks are networks consisting of two types of nodes. Edges in these networks are only allowed to be drawn from nodes of one type to nodes of another type. In our case **the first node type are the concepts** in the concept list and the second node type are **the word forms in a given language**. We create our network by linking all individual morphemes in our data to the concepts denoted by the words in which they occur."



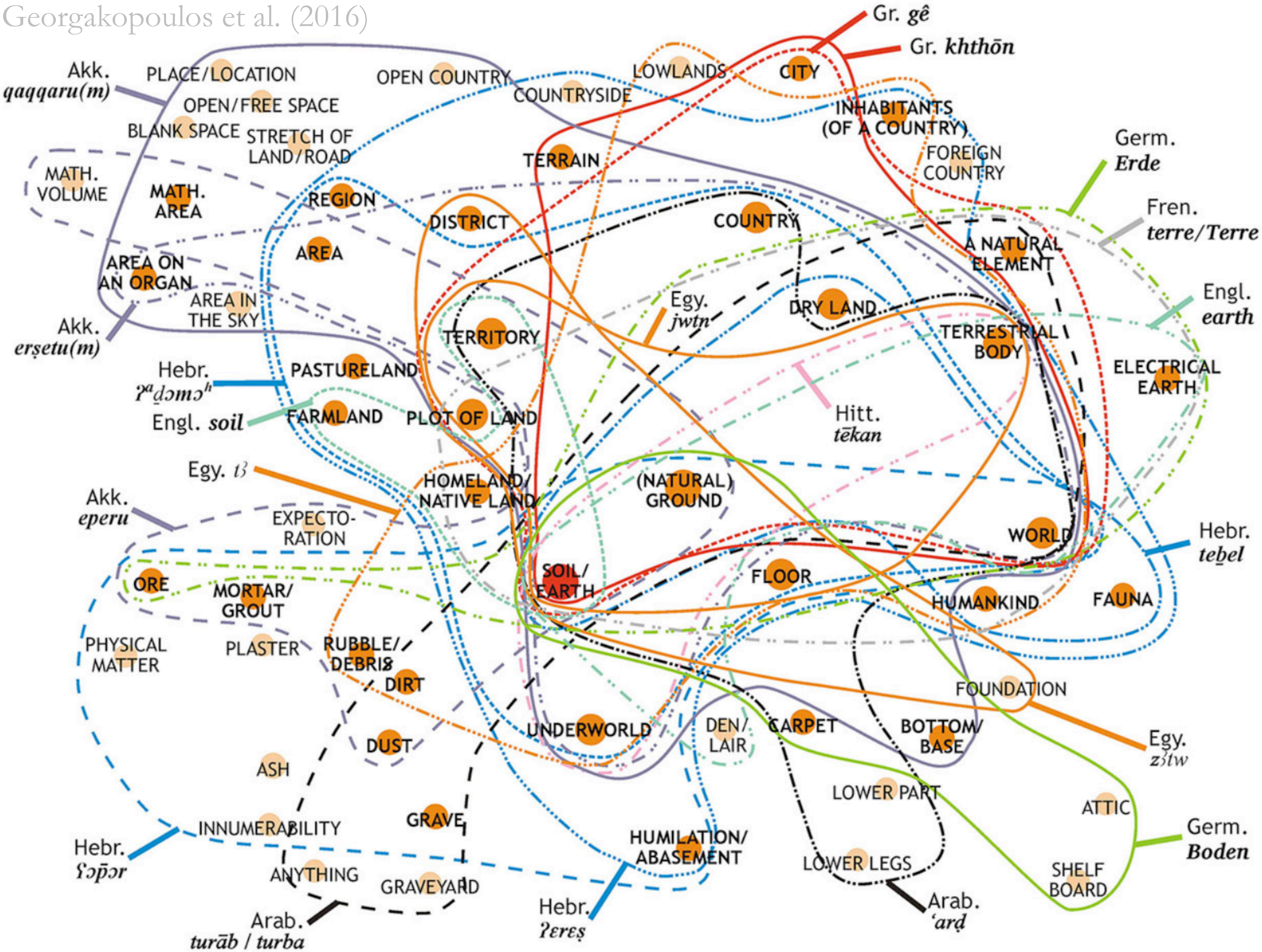
# Methodological issues

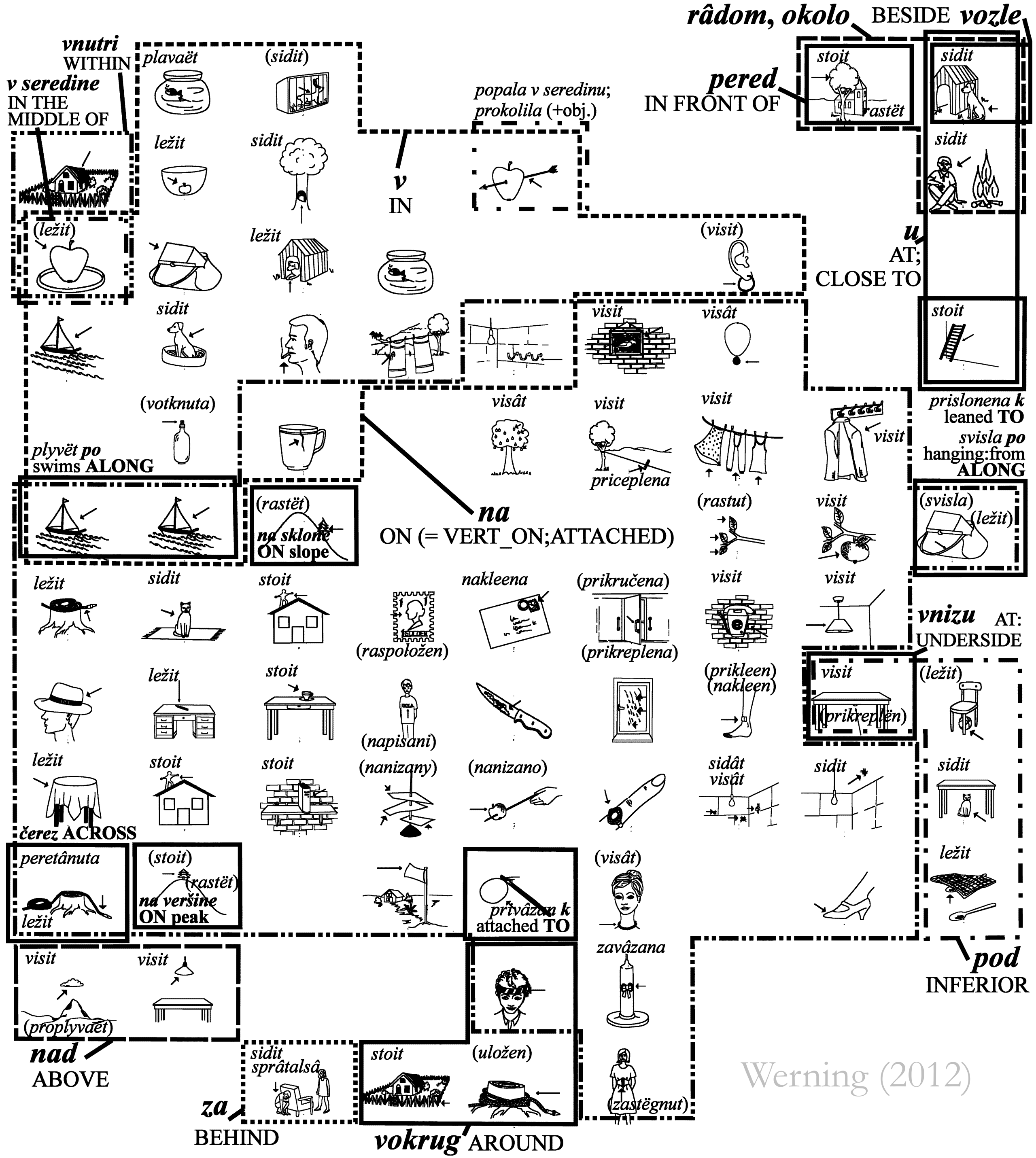
## 1. How can we visualize the types of polysemy patterns attested?

- Hill & List (2017): Bipartite networks
- List et al. (2018): Hypergraph



Georgakopoulos et al. (2016)






Werning (2012)

# Methodological issues

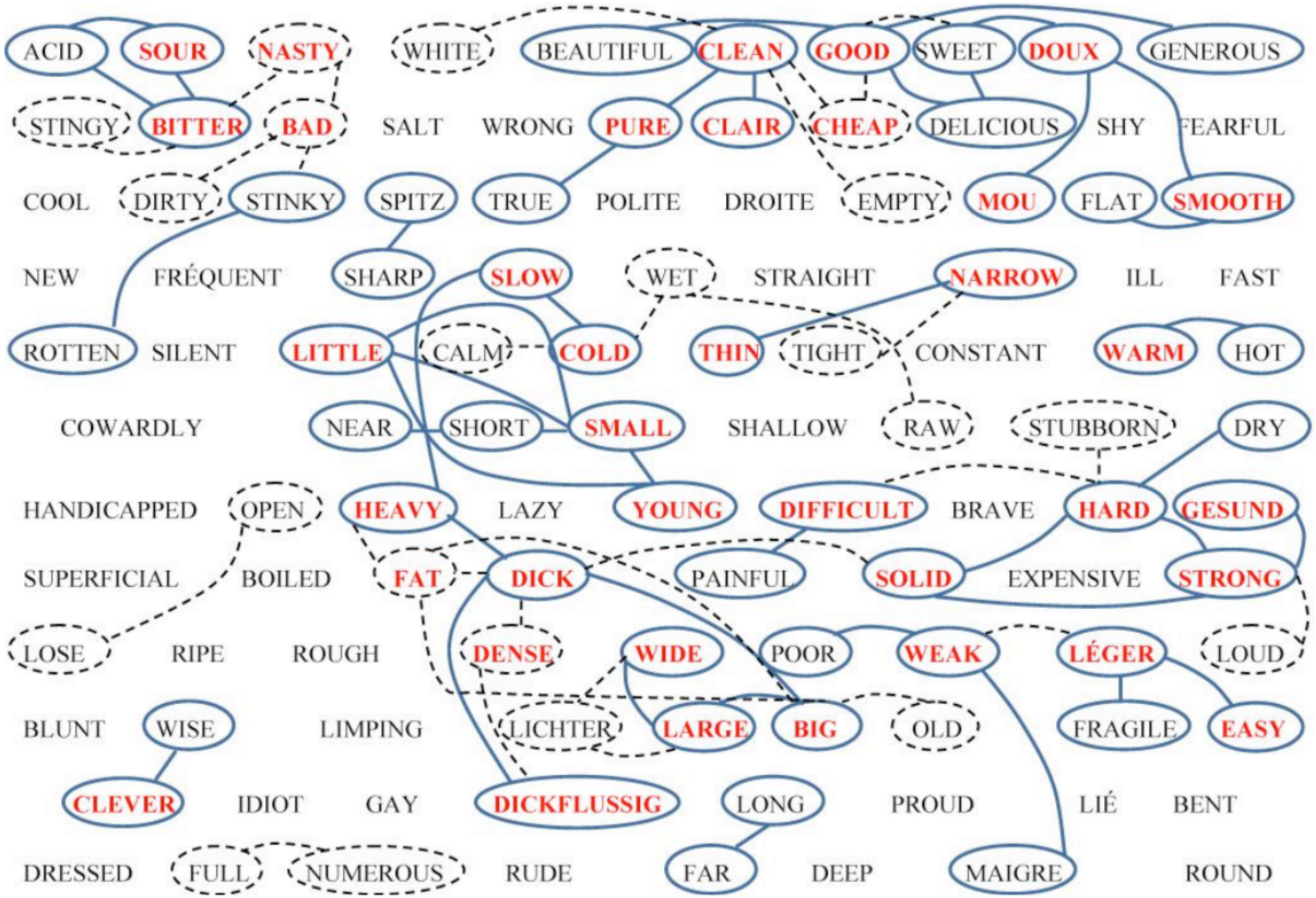



Figure 6: Conceptual map

Perrin (2012)

# Methodological issues

## How can we visualize the types of polysemy patterns attested?

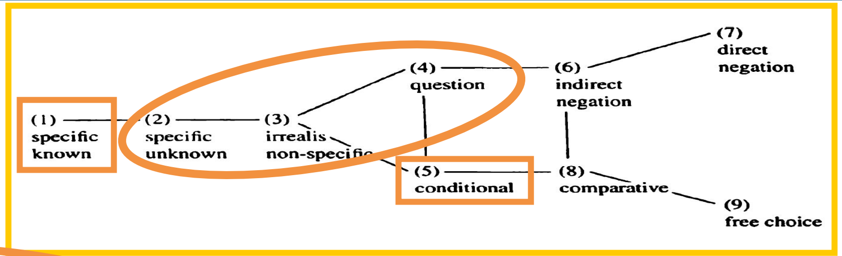
- Hill & List (2017): Bipartite networks
- List et al. (2018): Hypergraph
- Ryzhova & Obiedkov (2017): Formal concept analysis

**Formal Concept Lattices as Semantic Maps**

Daria Ryzhova and Sergei Obiedkov

National Research University Higher School of Economics,
Moscow, Russia
daria.ryzhova@mail.ru sergei.obj@gmail.com

# Methodological issues



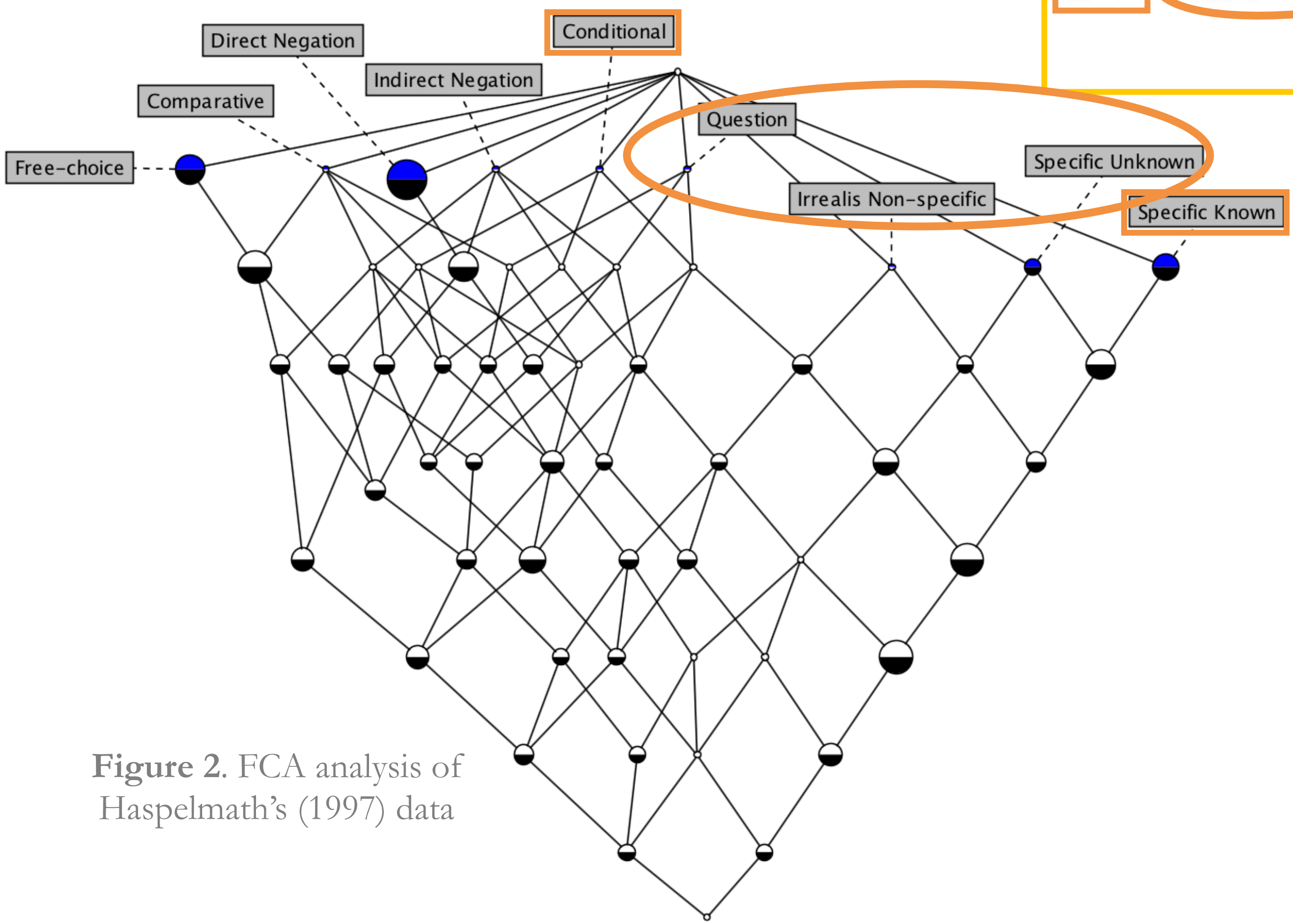



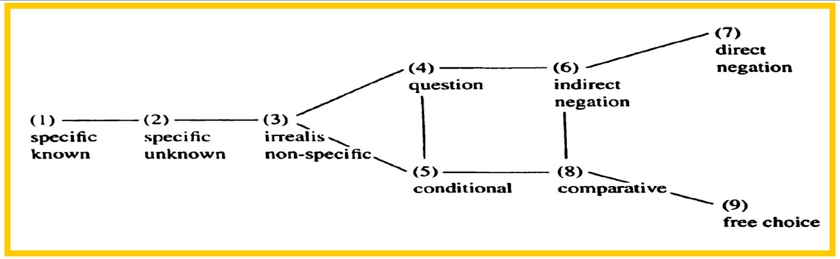

**Figure 2**. FCA analysis of Haspelmath's (1997) data

FCA solves the problem of form/ meaning mapping, since it shows:

- ✓ How forms maps onto meanings
- ✓ Which concepts are lexicalized and which are not
- ✓ Implication sets can be computed automatically

- ◆ But, less 'reader-friendly' (especially with many meanings = attributes)
- ◆ Complementarity between the two approaches

# Methodological issues



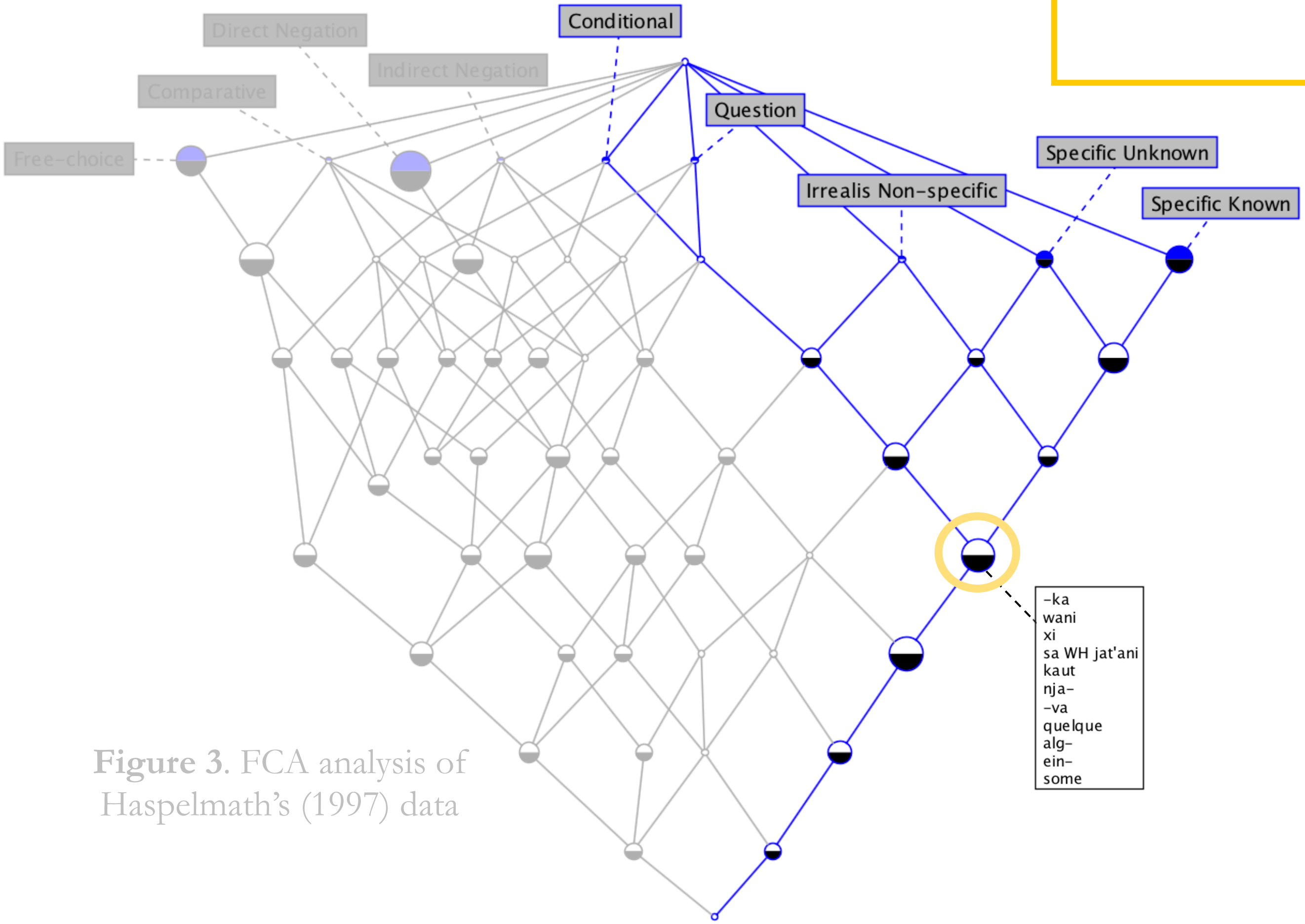
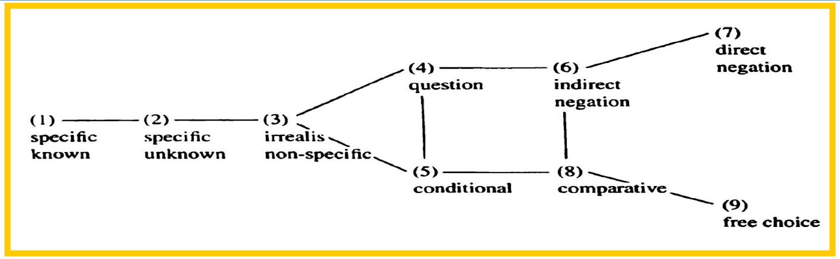




Figure 3. FCA analysis of Haspelmath's (1997) data

FCA solves the problem of form/ meaning mapping, since it shows:

- ✓ How forms maps onto meanings

# Methodological issues



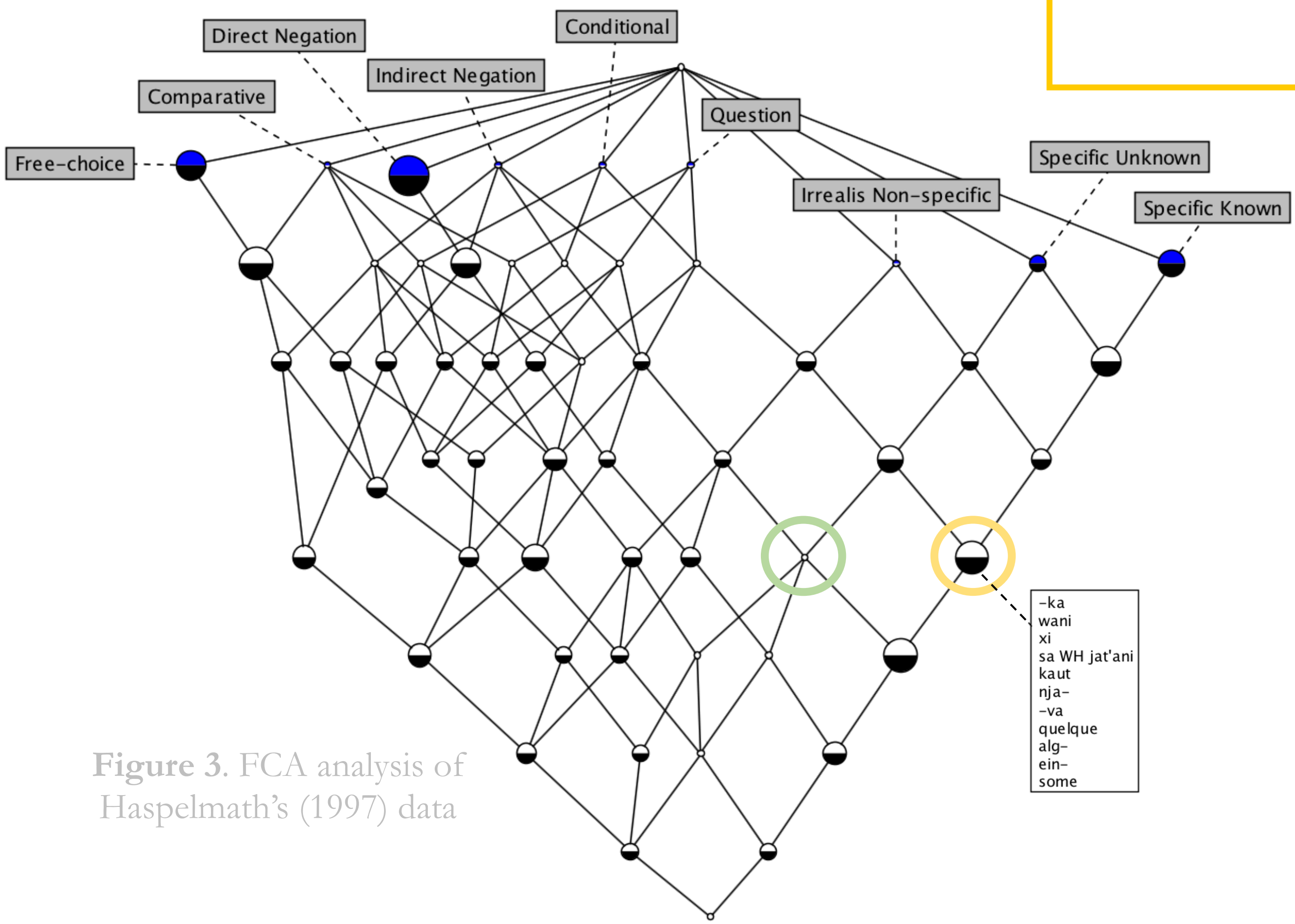
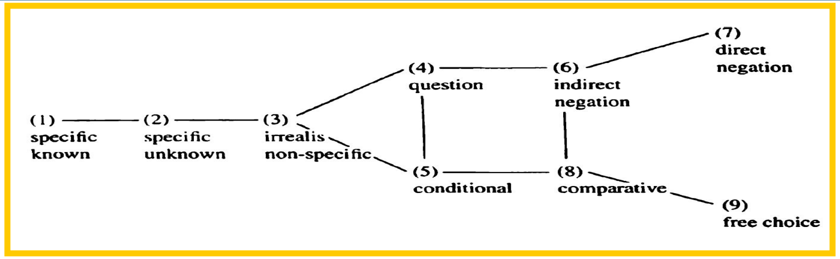




**Figure 3**. FCA analysis of Haspelmath's (1997) data

FCA solves the problem of form/ meaning mapping, since it shows:

- ✓ How forms maps onto meanings
- ✓ Which concepts are lexicalized and which are not

# Methodological issues



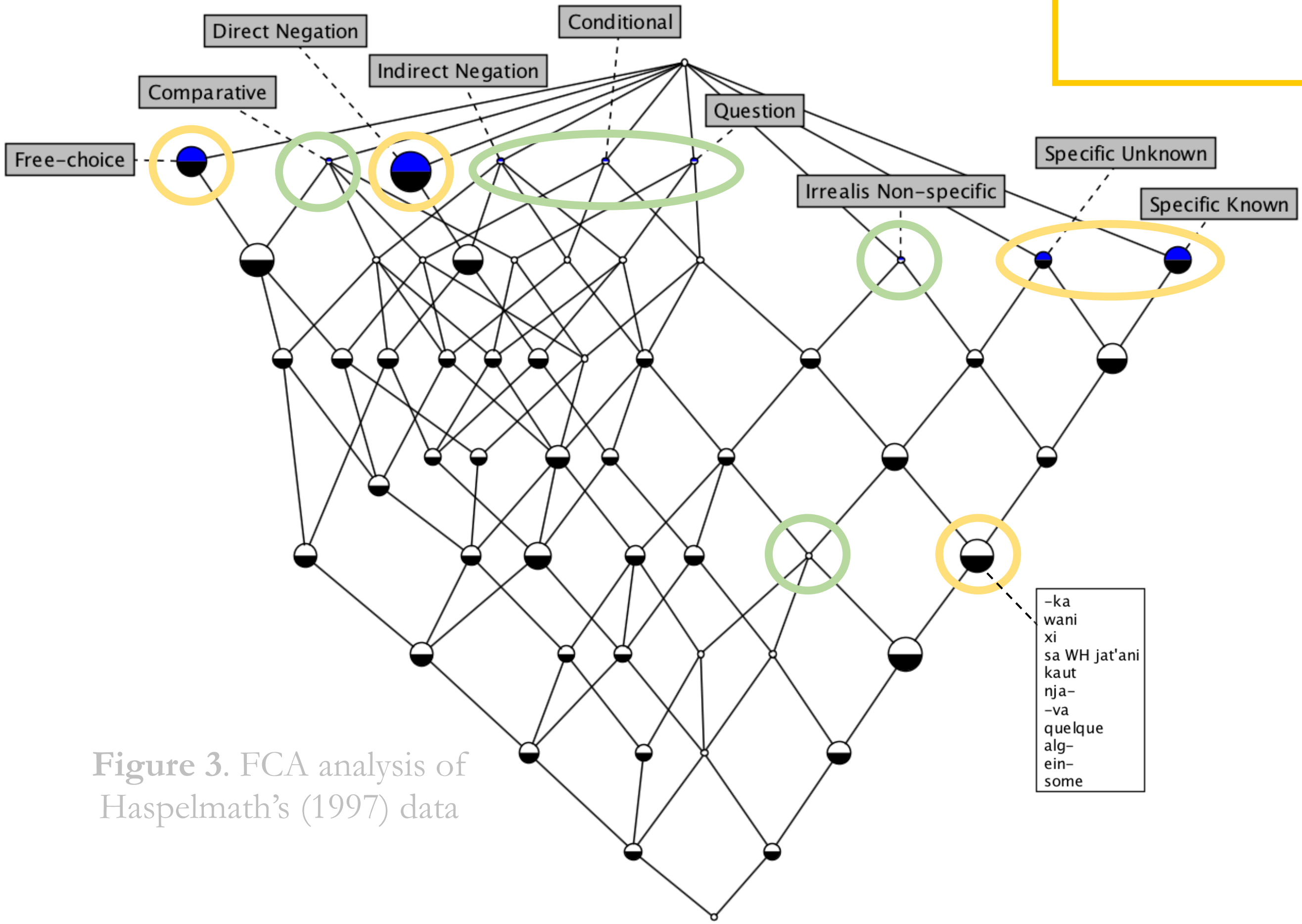
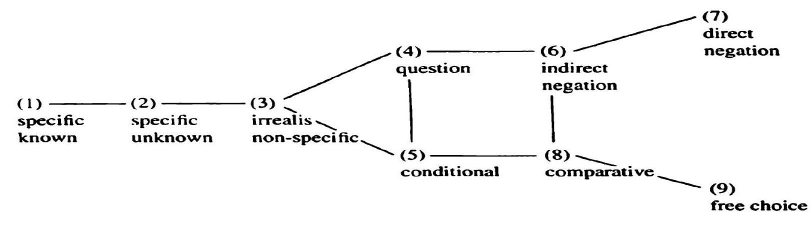



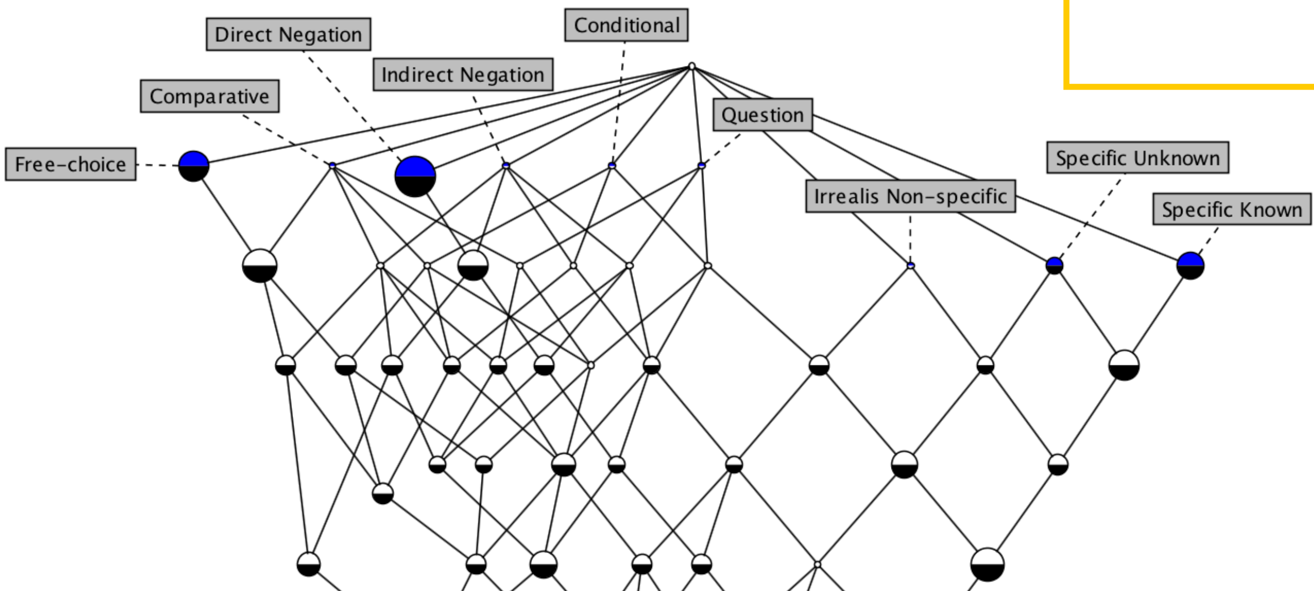

**Figure 3**. FCA analysis of Haspelmath's (1997) data

FCA solves the problem of form/ meaning mapping, since it shows:

- ✓ How forms maps onto meanings
- ✓ Which concepts are lexicalized and which are not

# Methodological issues




1 < 27 > Specific Known Irrealis Non-specific ==> Specific Unknown;
2 < 25 > Specific Known Question ==> Specific Unknown Irrealis Non-specific Conditional;
3 < 32 > Specific Unknown Question ==> Irrealis Non-specific Conditional;
4 < 46 > Irrealis Non-specific Question ==> Conditional;
5 < 25 > Specific Known Conditional ==> Specific Unknown Irrealis Non-specific Question;
6 < 32 > Specific Unknown Conditional ==> Irrealis Non-specific Question;
7 < 46 > Irrealis Non-specific Conditional ==> Question;
8 < 14 > Specific Known Indirect Negation ==> Specific Unknown Irrealis Non-specific Question Conditional;
9 < 16 > Specific Unknown Indirect Negation ==> Irrealis Non-specific Question Conditional;
10 < 27 > Irrealis Non-specific Indirect Negation ==> Question Conditional;
11 < 1 > Specific Known Comparative ==> Specific Unknown Irrealis Non-specific Question Conditional Indirect Negation Direct Negation;
12 < 3 > Specific Unknown Comparative ==> Irrealis Non-specific Question Conditional Indirect Negation;
13 < 11 > Irrealis Non-specific Comparative ==> Question Conditional Indirect Negation;
14 < 5 > Specific Known Direct Negation ==> Specific Unknown Irrealis Non-specific Question Conditional Indirect Negation;
15 < 6 > Specific Unknown Direct Negation ==> Irrealis Non-specific Question Conditional Indirect Negation;
16 < 12 > Irrealis Non-specific Direct Negation ==> Question Conditional Indirect Negation;
17 < 26 > Question Direct Negation ==> Indirect Negation;

FCA solves the problem of form/meaning mapping, since it shows:

- ✓ How forms maps onto meanings
- ✓ Which concepts are lexicalized and which are not
- ✓ (Implication sets can be computed automatically)

# Methodological issues

1. How can we visualize the types of polysemy patterns attested?
2. How can we deal with studies that take a single meaning as point of departure?

# Methodological issues

Grossman, Polis & Winand (eds.), *Lexical Semantics in Ancient Egyptian*, 175-225

## Navigating polyfunctionality in the lexicon*

Semantic maps and Ancient Egyptian lexical semantics

Eitan Grossman & Stéphane Polis, Jerusalem – Liège

# Methodological issues

Grossman, Polis & Winand (eds.), *Lexical Semantics in Ancient Egyptian*, 175-225

Navigating polyfunctionality in the lexicon*

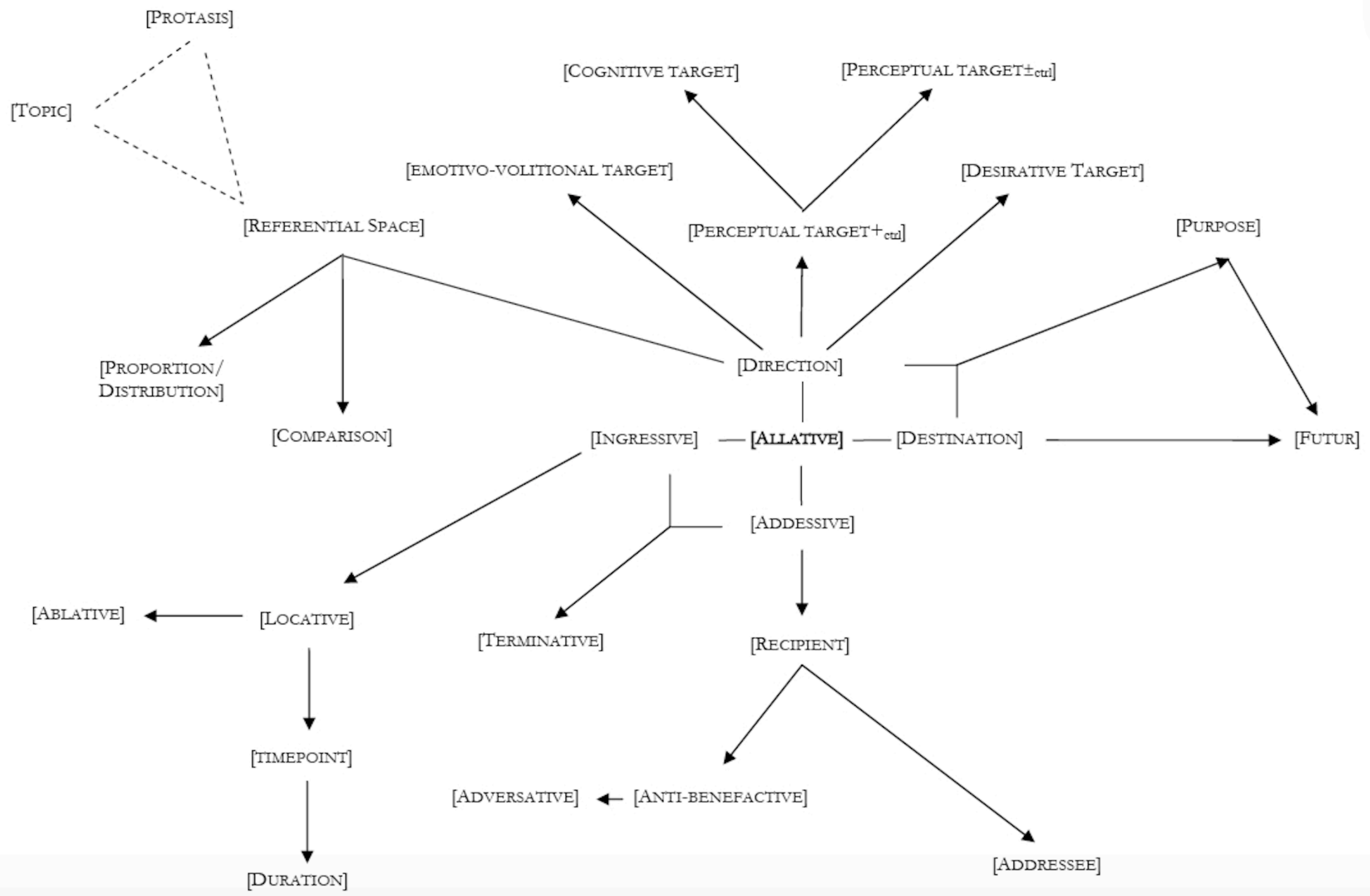
Semantic maps and Ancient Egyptian lexical semantics

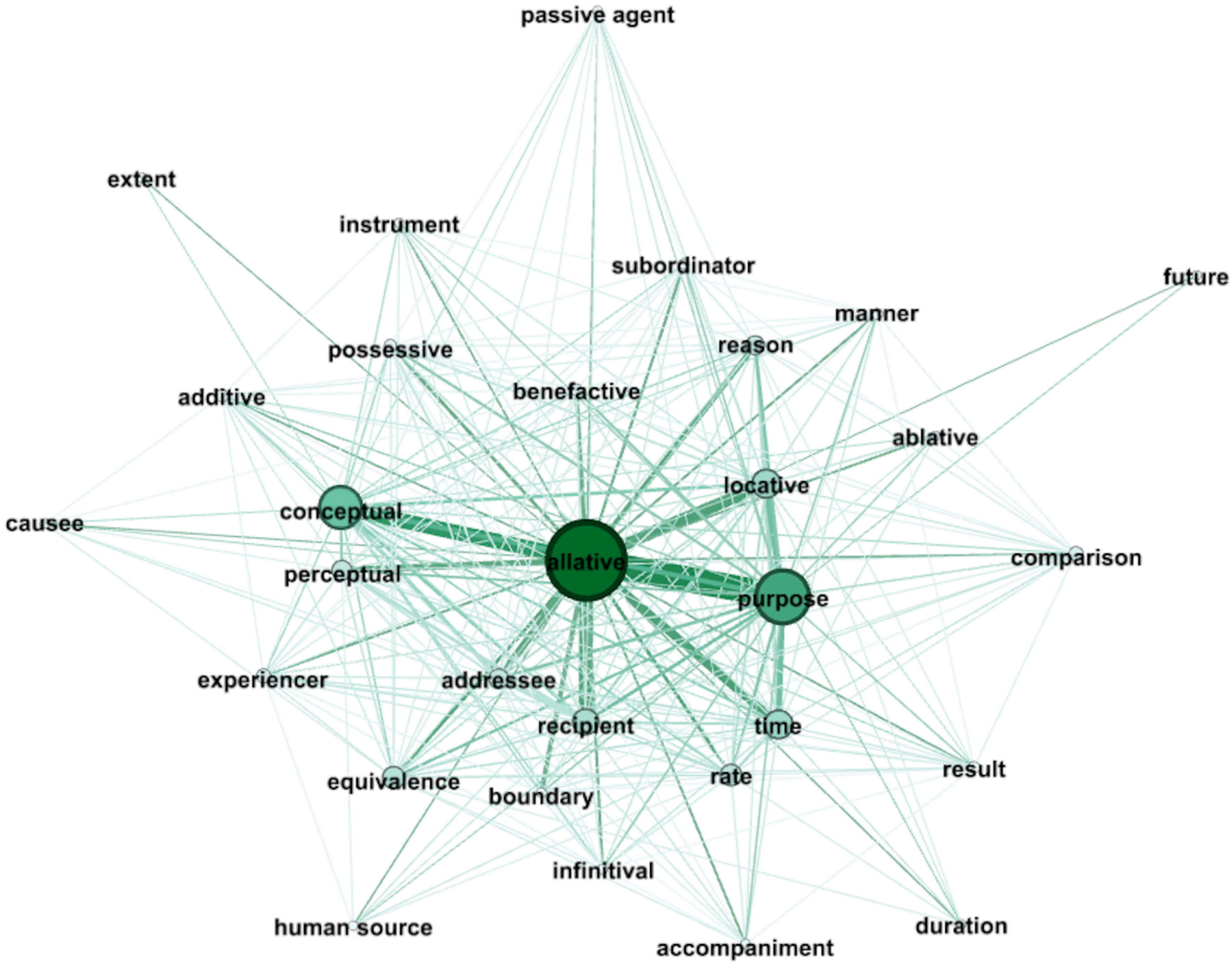
Eitan Grossman & Stéphane Polis, Jerusalem – Liège

**Crosslinguistic grammaticalization patterns of the ALLATIVE**

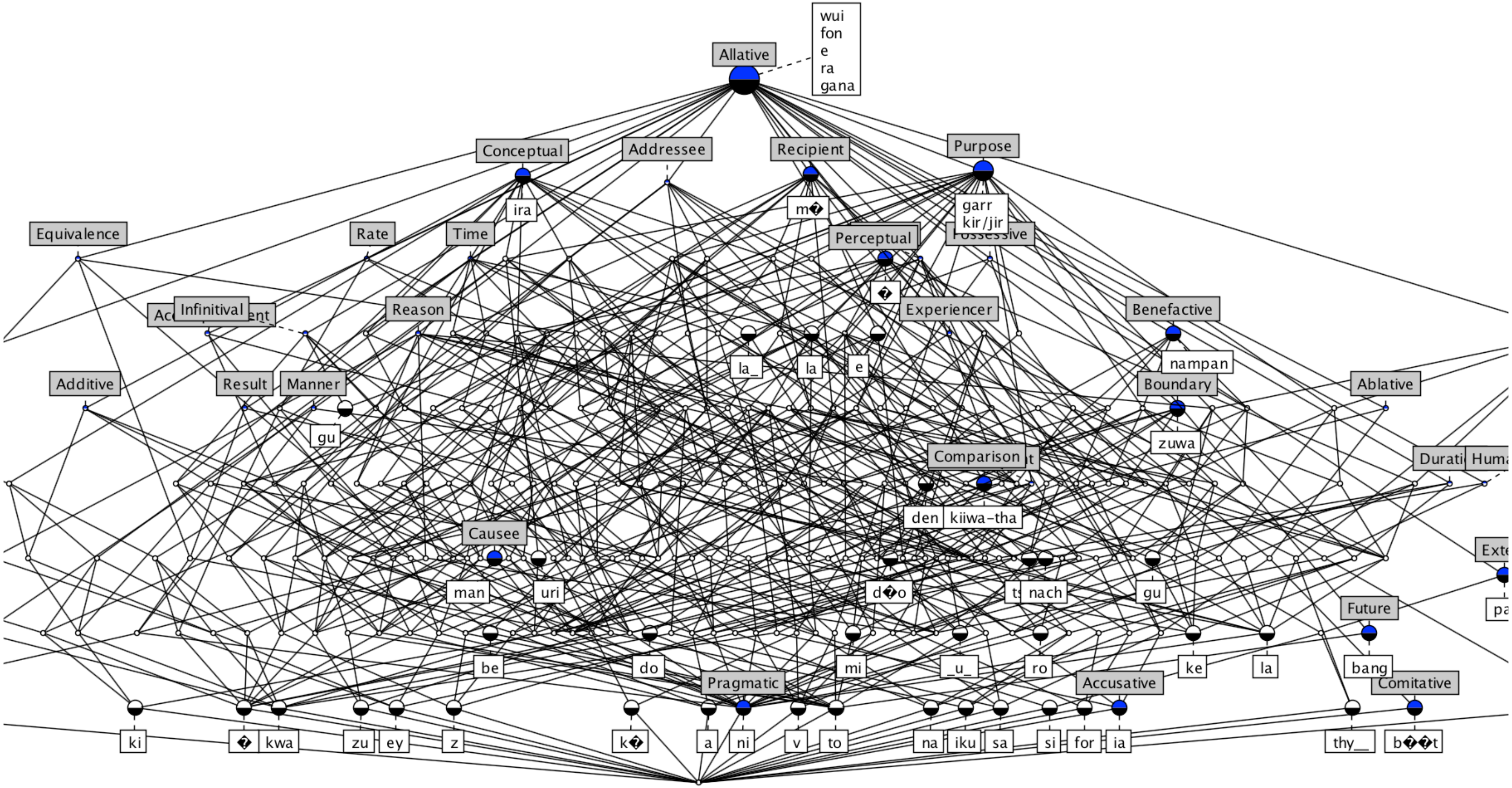
SALLY RICE and KAORI KABATA

# Methodological issues




passive agent
extent
instrument
subordinator
future
manner
reason
possessive
benefactive
additive
ablative
locative
conceptual
causee
allative
comparison
perceptual
purpose
experiencer
addressee
recipient
time
result
equivalence
rate
boundary
infinitival
human source
duration
accompaniment

# Methodological issues



# Methodological issues

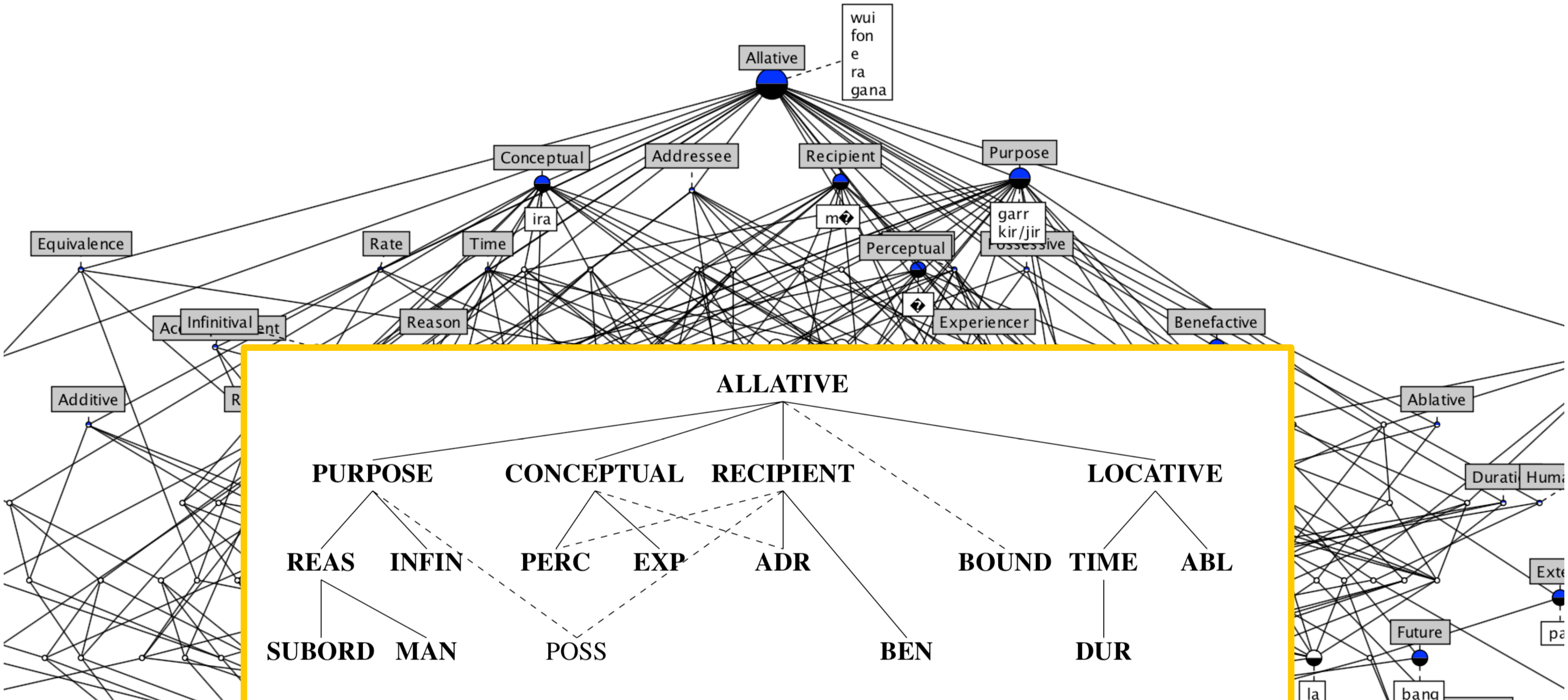
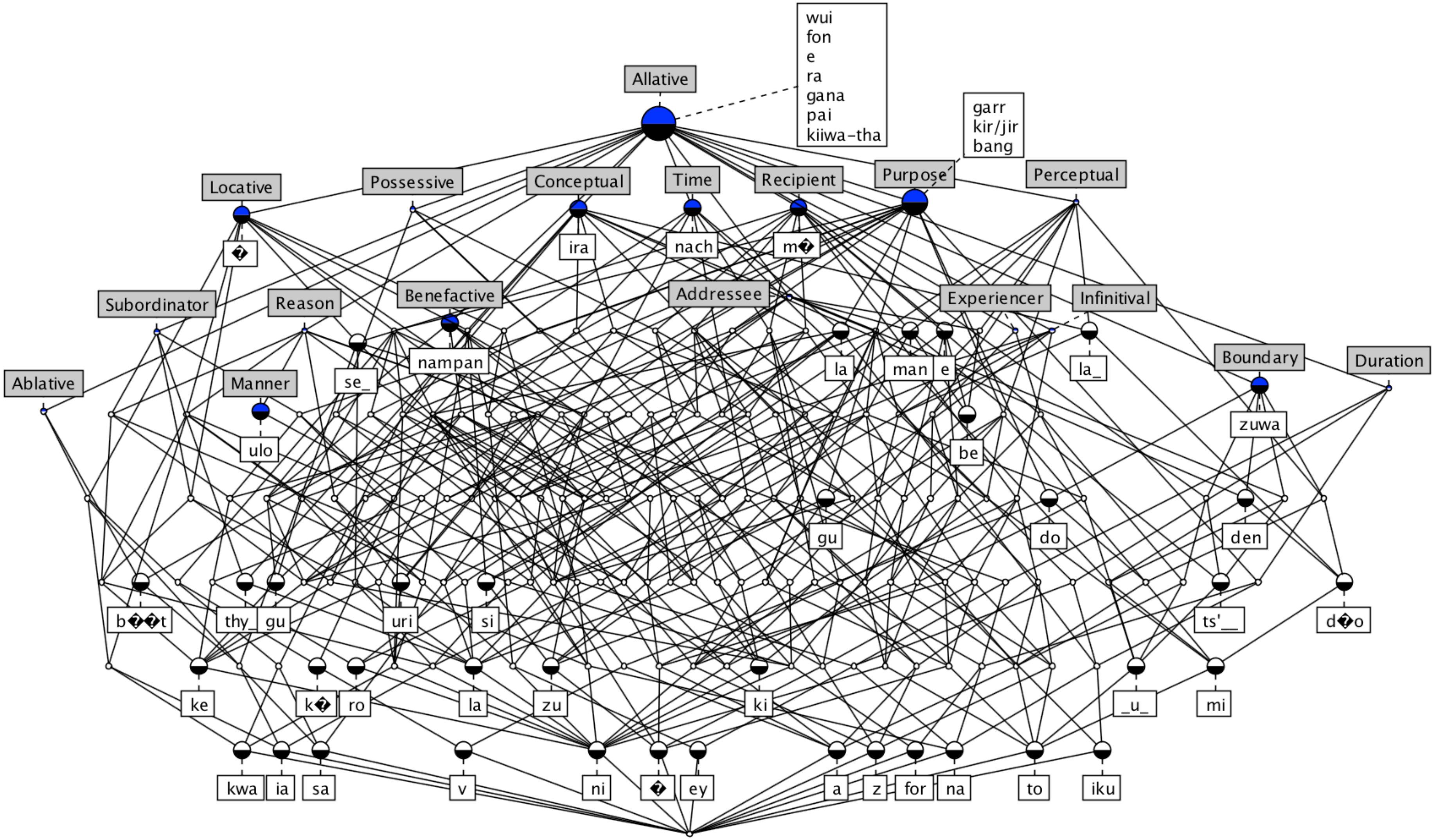



Figure 19. *A hierarchical mapping of the major relationships among the major* LOGICAL/TEXTUAL, MENTAL, SOCIAL, TEMPORAL, *and* SPATIAL *domain senses of the* ALLATIVES *in our database*

# Methodological issues



# Conclusions