# Competitive and therefore defeasible?

## On deciding prosodic outcomes

Dafydd Gibbon
Universität Bielefeld

Doris Bleiching
Institut for Innovation & Transfer GmbH

Poznań Linguistic Meeting 2010, Gniezno

# Overview

The ‘Competition Metaphor’:

- three competitions
  - explanations, models, properties
- the logic of linguistic competition
  - Modelling competitions

Challenges:

1. Redundancy of feature values
2. Making Sense of German Declination (including stress-shift)
3. German Morphoprosody in Word Formation
4. Tone displacement in Kikuyu
5. Tonal lookahead in Baule

Conclusion: mutatis mutandis

# Prosodic outcomes

The main challenge:

- how to predict prosodic forms in different functional contexts

Various methods:

- derivation of output as theorems
  - rule cascading
  - constraint ranking
  - default inheritance
- all of these are methods for deductive inference

# The ‘Competition Metaphor’

Starting naively...

- Definition (CED):
  - the activity or condition of striving to gain or win something by defeating or establishing superiority over others
  - Ecology: interaction between animal or plant species, or individual organisms, that are attempting to gain a share of a limited environmental resource

- Imagine:
  - all the PEOPLE who compete, it’s not
    - theories
    - products
  - use of the term of other entities than people is
    - metonymy, strictly speaking
    - anthropomorphic metaphor, to all intents and purposes

# ‘Competitive therefore defeasible’

The term ‘defeasible’

- comes from nonmonotonic reasoning, default logics
- refers to the following kinds of choice criterion:
  - <u>preference</u>
  - non-determinism, ambiguity
  - prototype, stereotype
  - ‘typically’, ‘normally’, ‘standardly’
  - ‘mutatis mutandis’, ‘faute de mieux’, ‘other things being equal’
- is related to concepts such as
  - defaults and overrides
  - elsewhere condition (Kiparsky)
  - underspecification, markedness
  - ‘most detailed specification wins’
  - ‘longest path wins’ (DATR)

In this presentation:

- the DATR default logic formalism
- for nonmonotonic reasoning

# Competitions

# Competitions

## <u>Competition of explanations</u>

- functional explanation
- causal explanation
- *deductive-nomological explanation*

## <u>Competition of models</u>

- atomistic
- modular + interface
- *interconnected: ‘un système où tout se tient’*

## <u>Competition of properties</u>

- equipollent
- *privative / marked / statistical / ‘easier’*

# Competition of explanations

Sometimes we get bitter fights between proponents of ...

- Functional explanation
- Causal explanation
- Deductive-nomological explanation

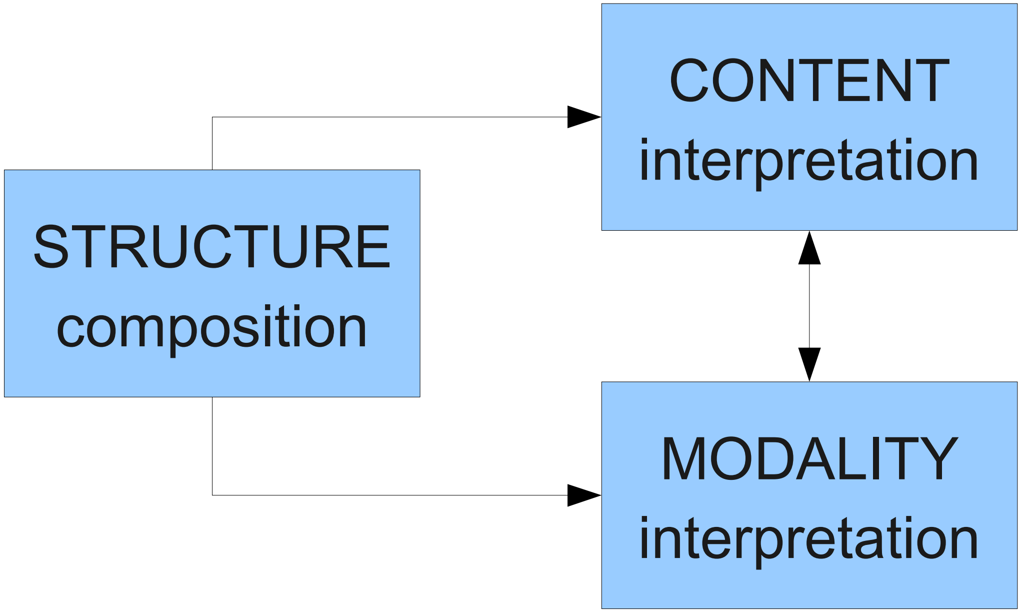
Thesis:

*In semiotic disciplines we need all of these types.*

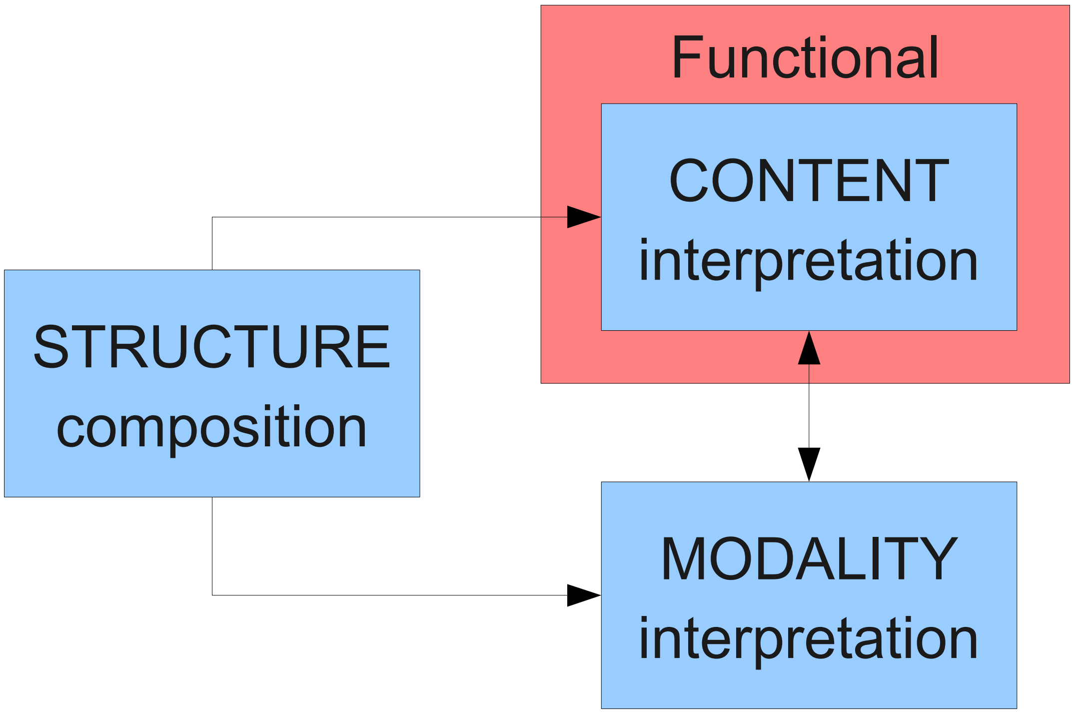
How does this apply?

- E.g. to a fairly standard Content-Structure-Modality (CSM) model of the sign?

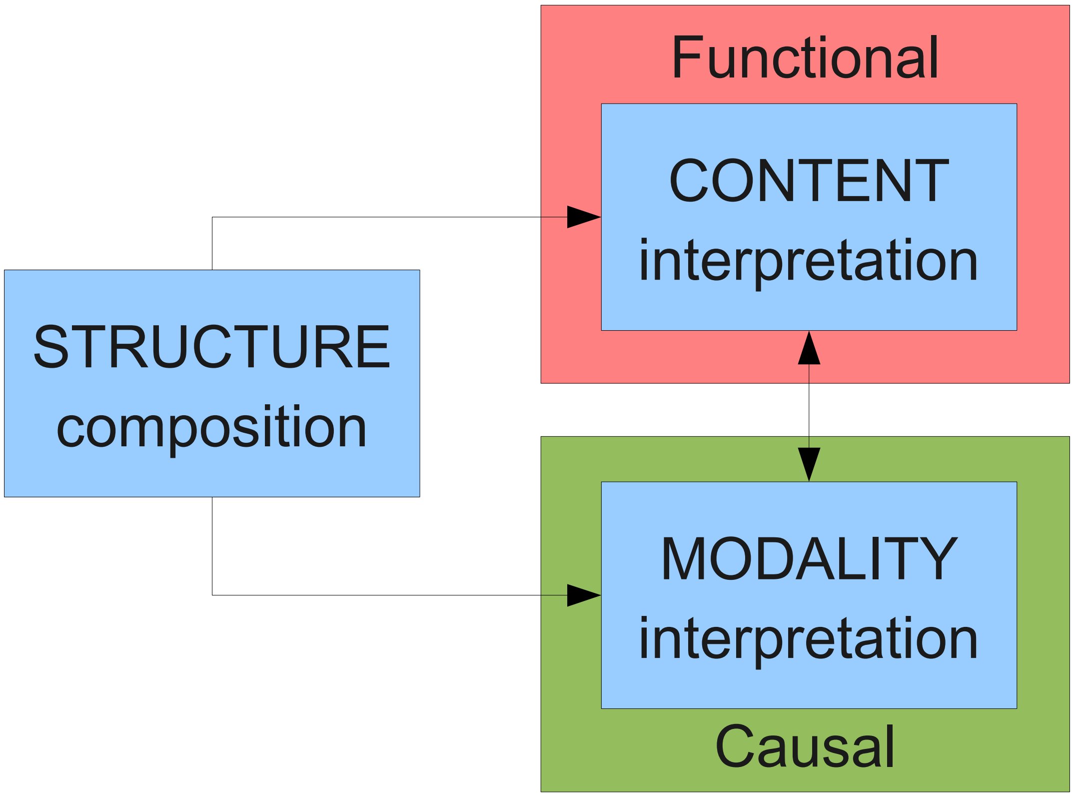
# Integrating Explanations



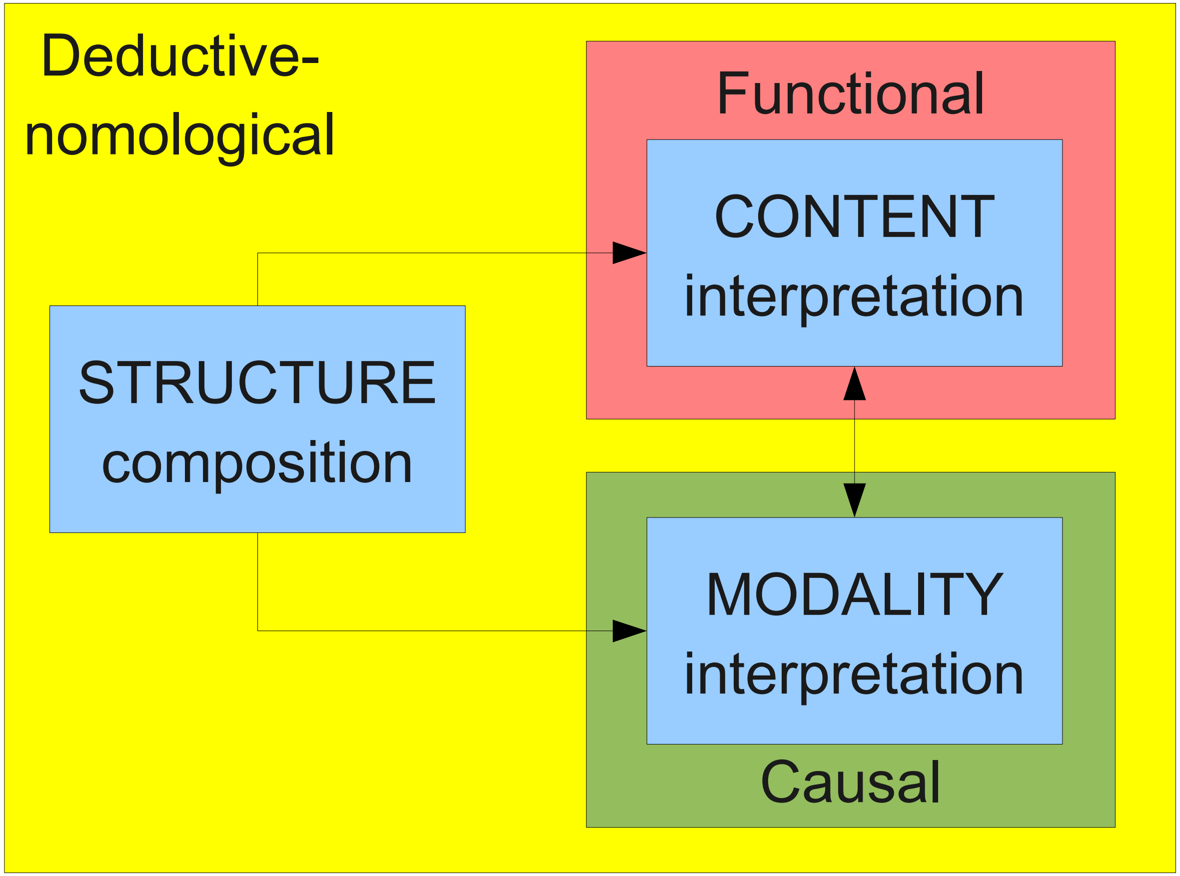
# Integrating Explanations



# Integrating Explanations



# Integrating Explanations



# Competition of models

Approaches:

- atomistic:
  - Pick a problem
  - Analyse it
  - Puzzle about how to relate it to other work
- modular: modules + interfaces
  - Phon-Morph-Syn-Text
  - Interfaces
  - Interpretations
- connected:
  - ‘un système où tout se tient’
  - complex: computational modelling needed
- and, of course, competition between
  - empirical solutions, generalisations
  - formal solutions, notation and calculi/logics

# Competition of properties

Standard Prague School model:

- equipollent properties (features; values of attributes)
- privative
  - unmarked - marked
  - widespread – rare
  - easier – harder
    - to produce
    - to perceive
  - statistical weighting

Where is there competition in language structure?

- interpretative relations of content & modality?
- syntagmatic relations of sequential and simultaneous compositionality?
- paradigmatic relations of classification by difference & similarity relations?

# The logic of linguistic competition

Anywhere there is <u>choice</u> there is competition!

So:

- competition takes place
  - in *paradigmatic relations*, i.e. in relations of classification and categorisation, taxonomic & mereonomic inventories
    - wholes, parts
    - sets, members
- competition does <u>not</u> take place
  - in *syntagmatic relations*, i.e. in relations of compositionality, with complementary functionalities:
    - sequential syntagmatic relations (linear, hierarchical)
    - simultaneous syntagmatic relations (feature structures, prosody)
  - except:
    - in structural ambiguities
    - in contradictions – sometimes we 'lose it' (e.g. double bind prosody)

# Illustrations from various domains

Cases of competition:

- e.g. vocabulary:
  - substitution of ‘left’/’right’
  - malapropisms
  - spoonerisms
  - tip-of-the-tongue phenomena
  - disfluencies
  - stress:
    - cóntroversy / contróversy
    - kílometre / kilómetre
    - Óxford Street / Oxford Róad

Cases of non-competition, complementarity:

- parts of speech
  - nouns & conjunctions, ...
- distinctive features
  - voicing and labiality, ...

# Modelling competitions

Challenges:

- Attributes/features:
  - markedness relations between values
  - ‘typicality’, mutatis mutandis, ‘other things being equal’, ‘prototypes’
- Rules & constraints:
  - applicability:
    - yes / no / elsewhere
  - optionality
  - ordering/ranking

Generalising the competition model:

- defaults and overrides
- preferences
- rankings

# Basics:
# Redundancy and Defaults

# Redundancy of feature values

Two types of redundancy rule:

- Standard implication (‘always’):
  - [ + nasal ] → [ + voice ]
  - [ + vocalic → [ + voice ]
- Default implication (‘unmarked’, ‘typically’):
  - [ + obstruent ] → [ - voice ] (typically)
  - [ + anterior ] → [ - round] (typically)

Challenge:

- How to formulate these implication types?
- Hint: implication is a logical relation
- Hint: typicality can be overridden
  - cf. prototypes, prejudices, ...

Solution:

- default logic

# Nonmonotonic reasoning
# Default Logics

# Redundancy of feature values

Two types of redundancy rule:

- Standard implication ('always'):
  - [ + nasal ] → [ + voice ]
  - [ + vocalic → [ + voice ]
- Default implication ('unmarked', 'typically'):
  - [ + obstruent ] → [ - voice ] (typically)
  - [ + anterior ] → [ - round] (typically)

*Well, sort of 'always' ...*

Challenge:

- How to formulate these implication types?
- Hint: implication is a logical relation
- Hint: typicality can be overridden
  - cf. prototypes, prejudices, ...

Solution:

- default logic

# Nonmonotonic reasoning: Tweety triangle

Syllogism:

- Birds can fly
- Tweety is a bird
- Tweety can fly

But:

- Penguins can't fly
- Tweety is a penguin
- Tweety can't fly

Contradiction:

- Tweety can fly
- Tweety can't fly

Resolution:

- More specific overrides more general

# Nonmonotonic reasoning: Tweety triangle

Syllogism:

- Birds can fly
- Tweety is a bird
- Tweety can fly

But:

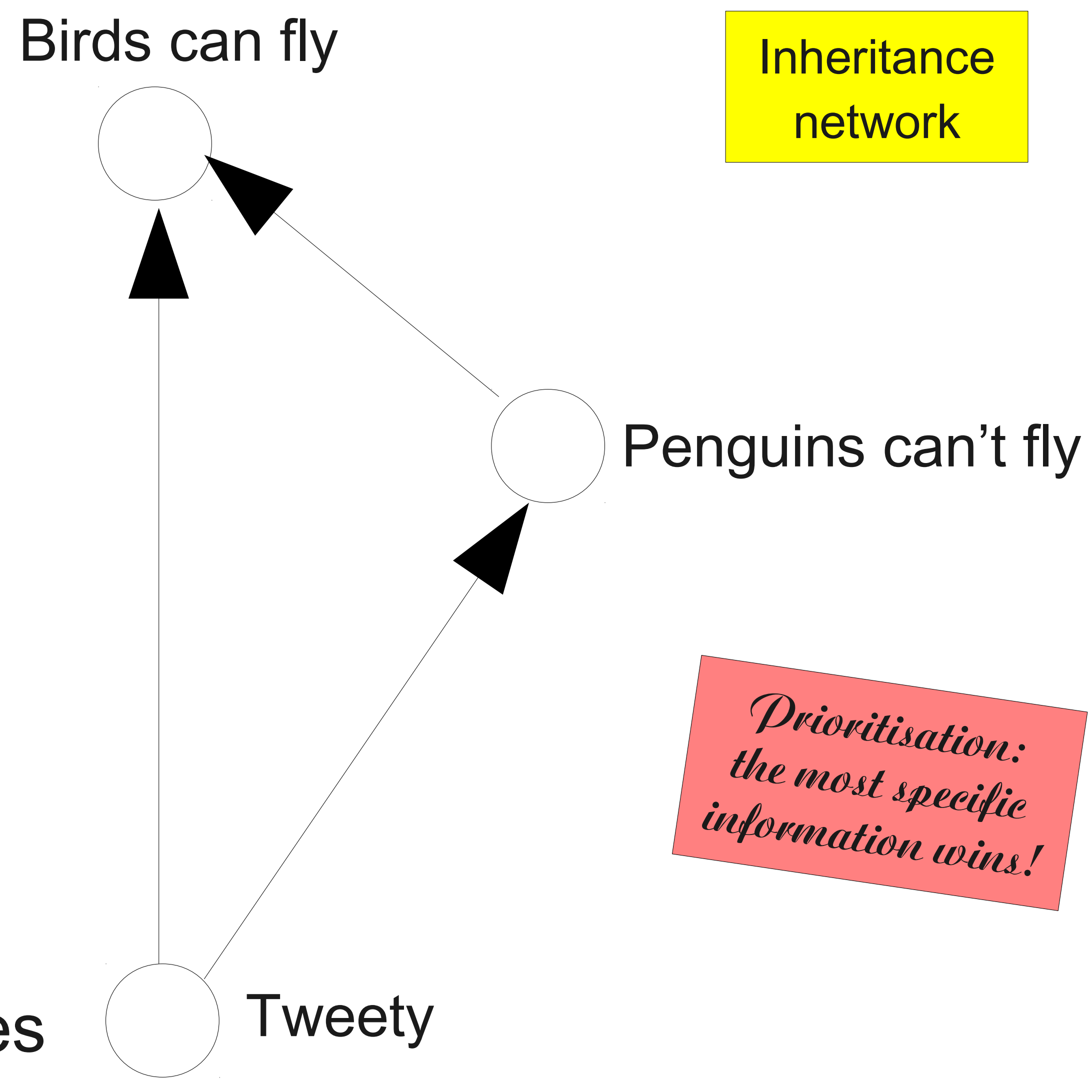
- Penguins can't fly
- Tweety is a penguin
- Tweety can't fly

Contradiction:

- Tweety can fly
- Tweety can't fly

Resolution:

- More specific overrides more general



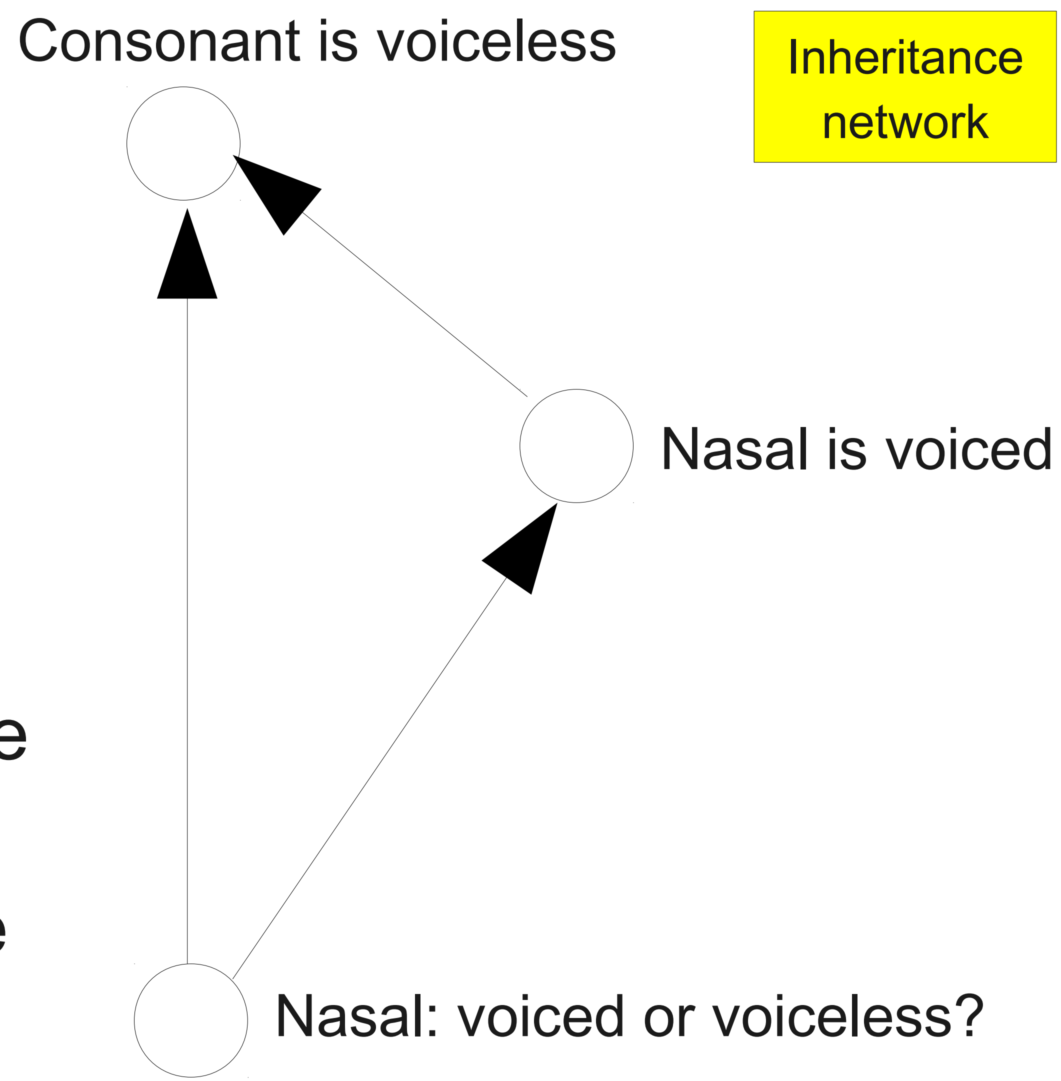
# Nonmonotonic reasoning: Consonant Triangle

Traditional model:
- Redundancy rules
- Not explicitly connected

Logically speaking:
- Implicational hierarchies

Operationalisation:
- Feature inheritance

Resolution:
- Default inheritance



# Nonmonotonic reasoning: Liquid Triangle

Traditional model:

- Redundancy rules
- Not explicitly connected

Logically speaking:

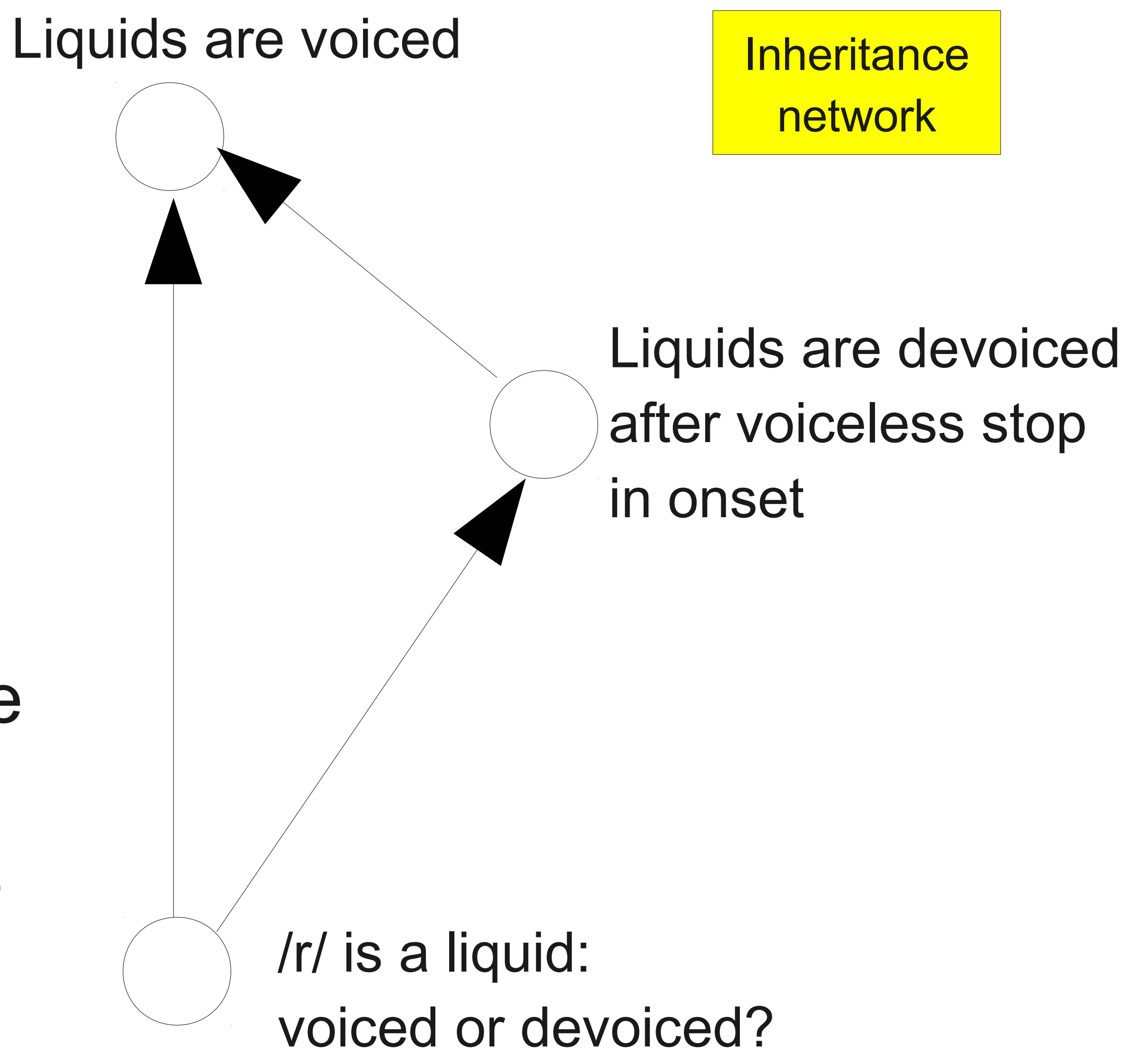
- Implicational hierarchies

Operationalisation:

- Feature inheritance

Resolution:

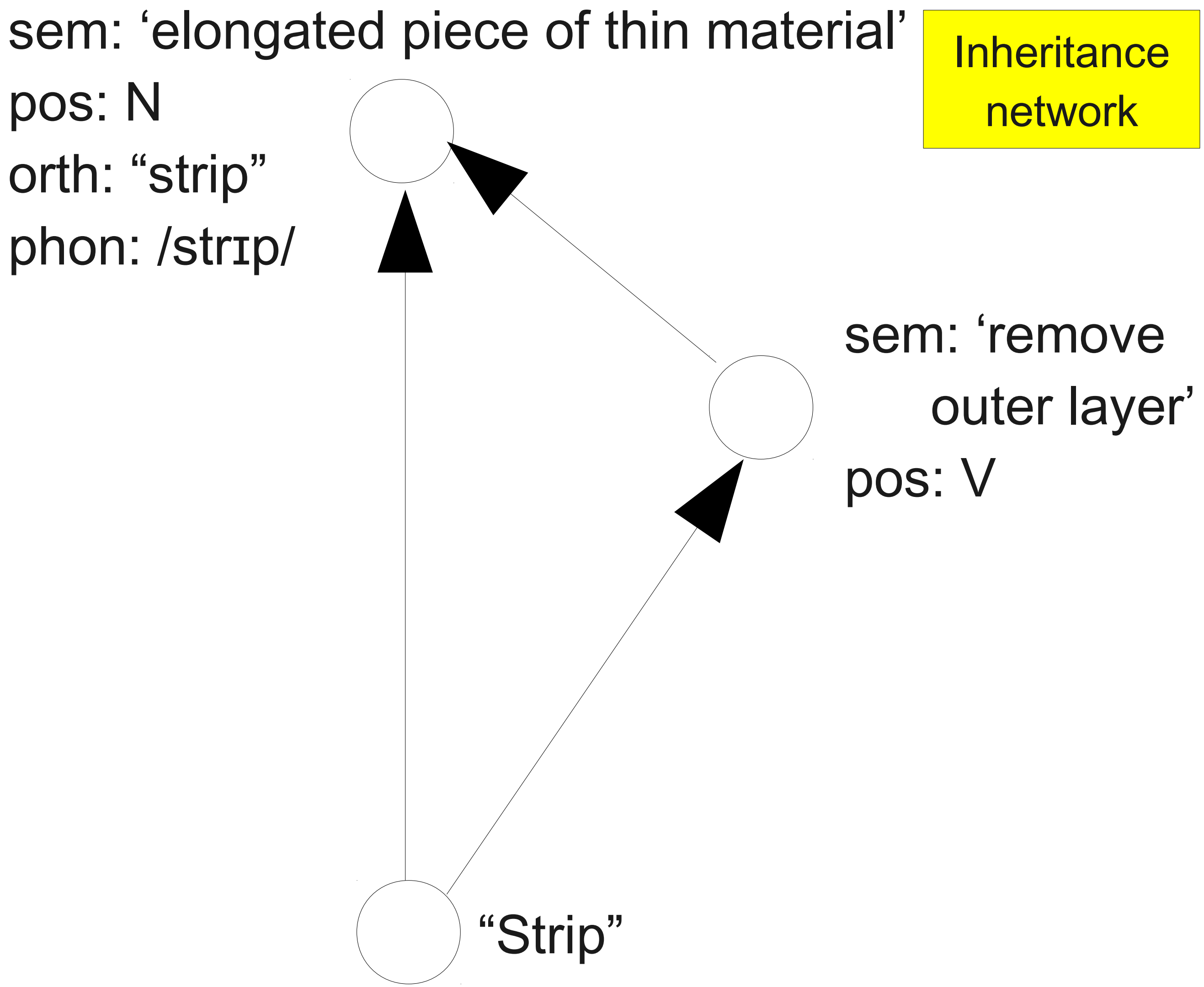
- Default inheritance



# Nonmonotonic reasoning: Homonyms

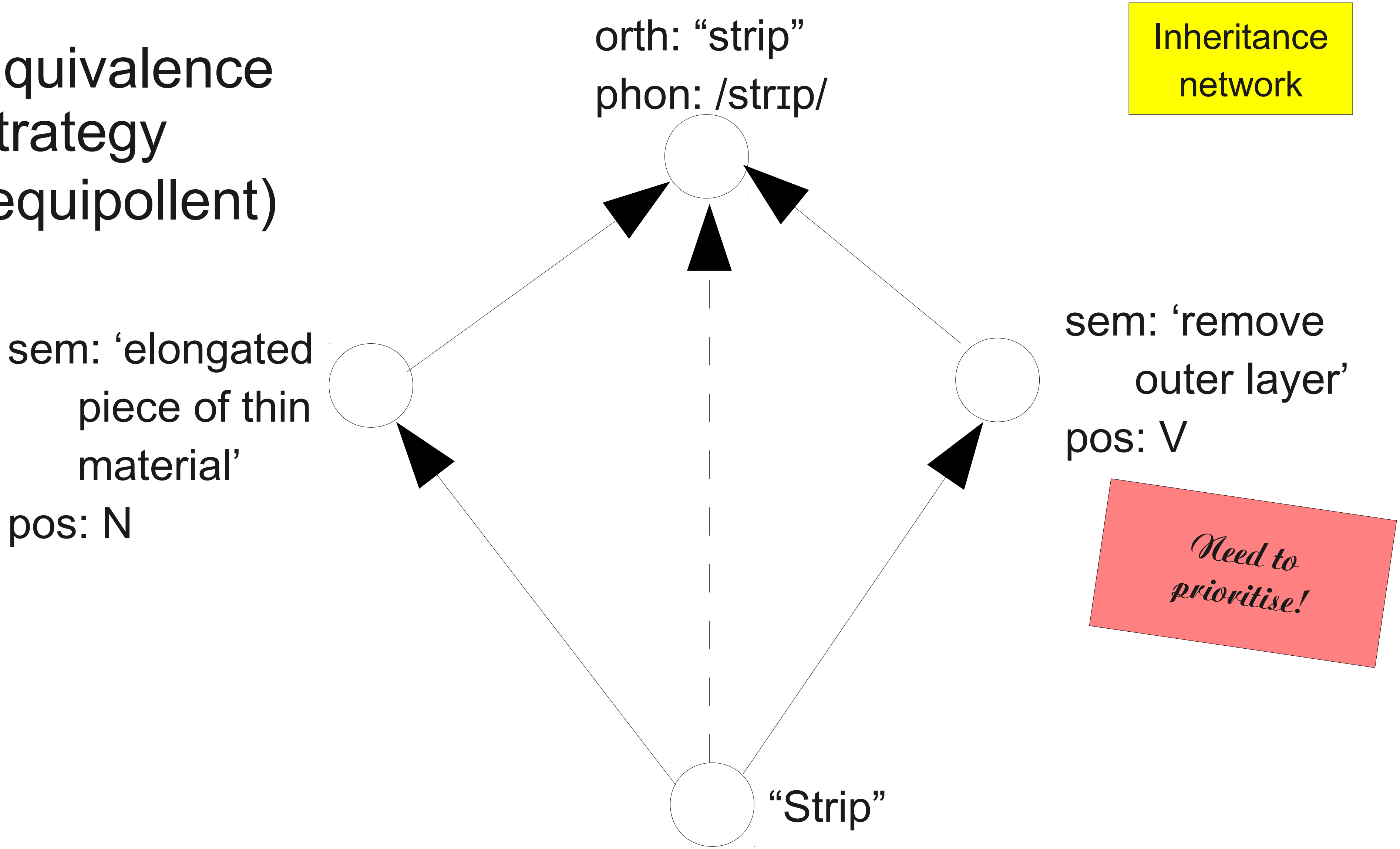
Dependency strategy (privative)

Inheritance network



# Nonmonotonic reasoning: Homonyms

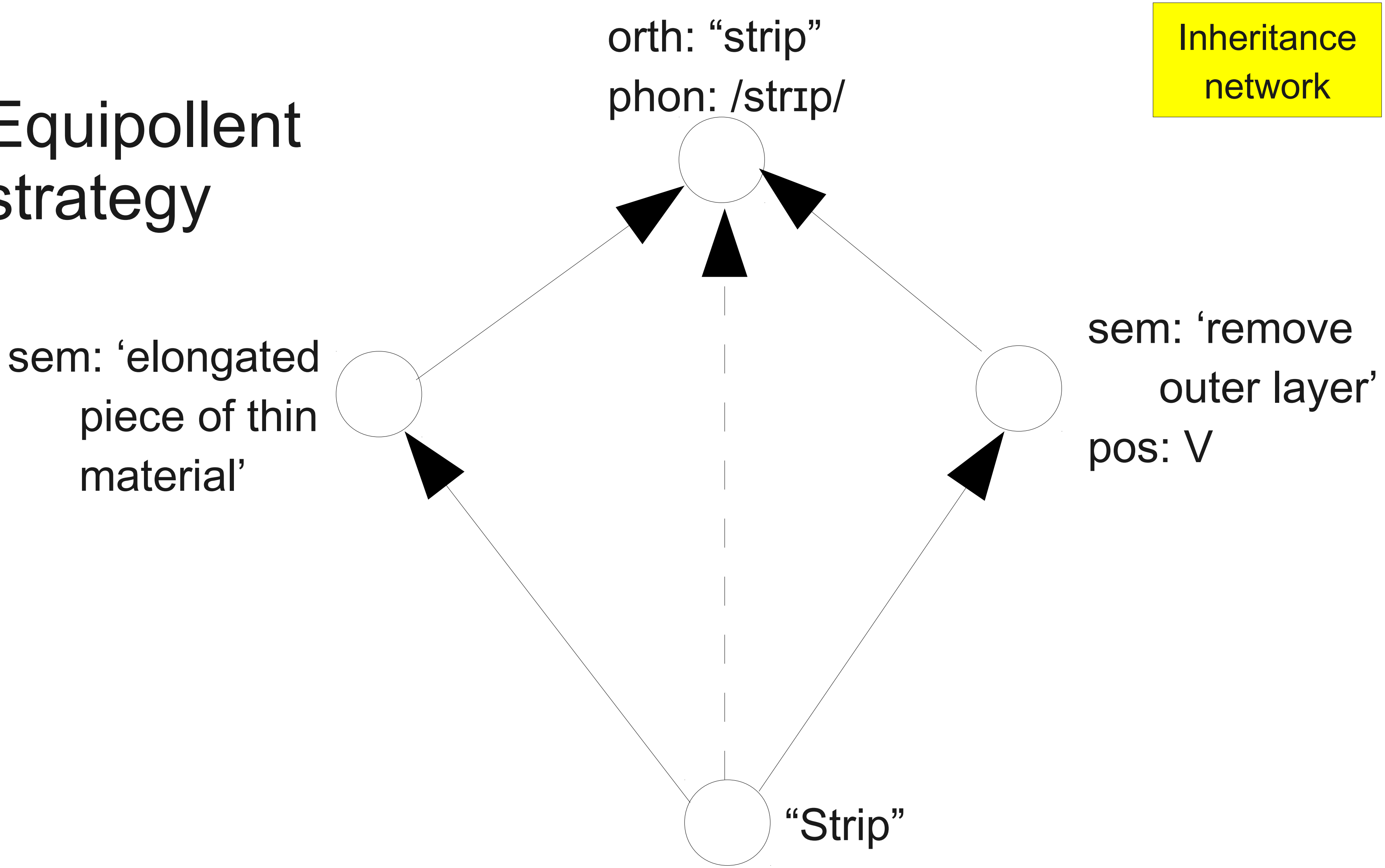
Equivalence strategy (equipollent)



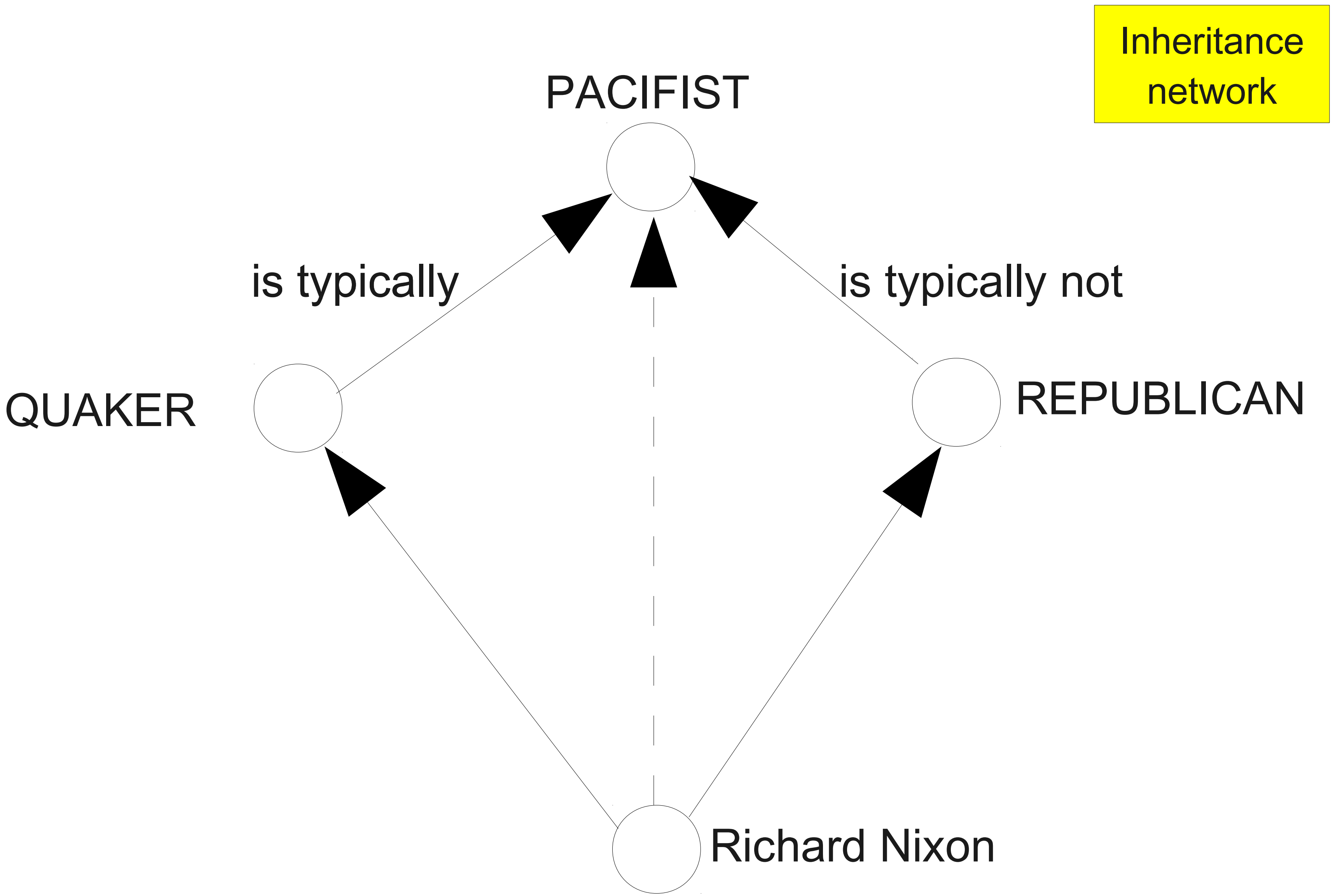
# Nonmonotonic reasoning: Homonyms

Inheritance network

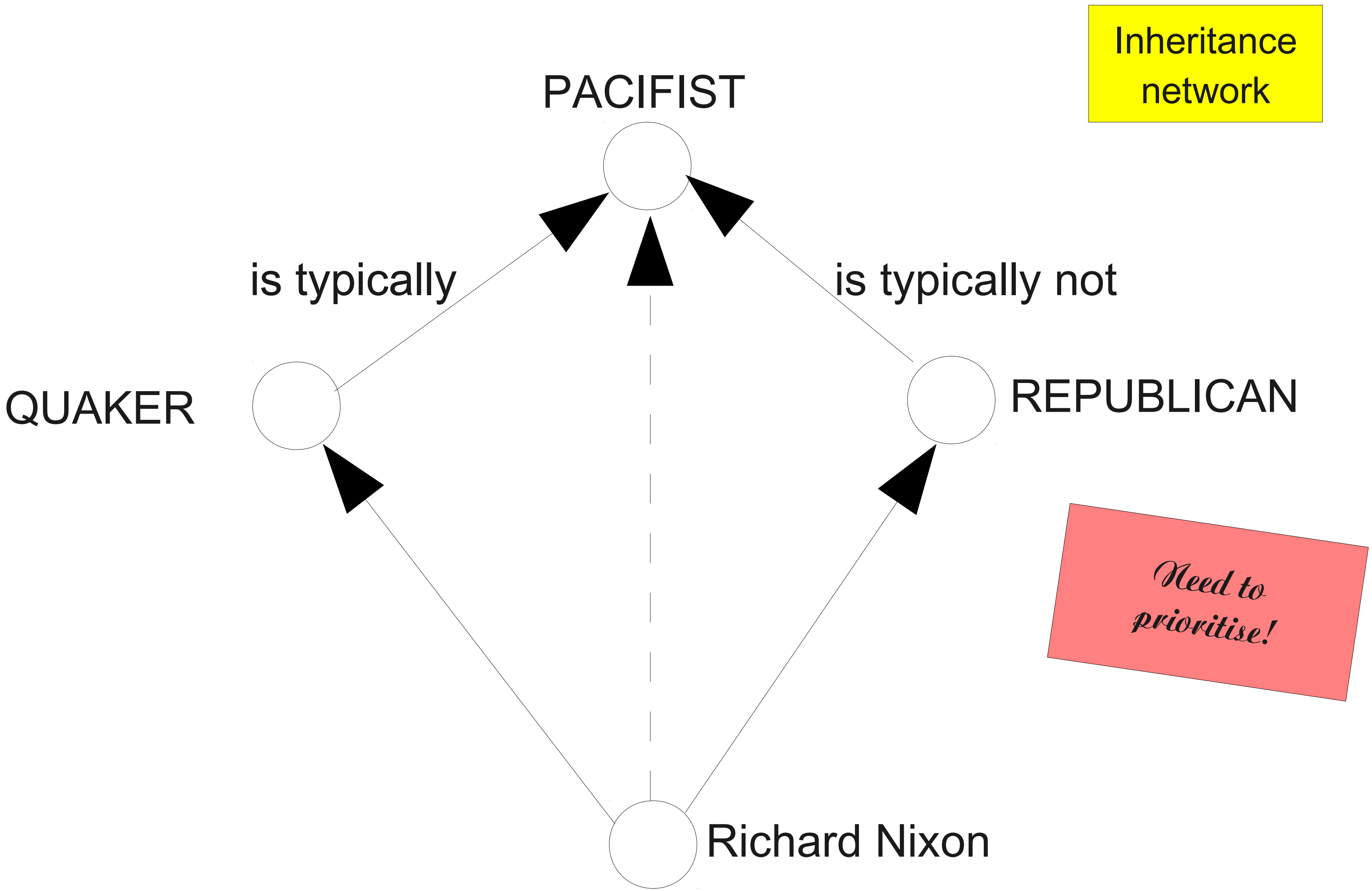
Equipollent strategy



# Nonmonotonic reasoning: Nixon Diamond



# Default logic: the 'Nixon Diamond'



# Challenge 1:

# Conservative & Radical Underspecification

# #1: Underspecification

Strip1:

<semantics> ==

'An elongated rectangular piece of thin material, e.g. paper'

<syntax> == Noun:<common>

<morphology> == Stem:<germanic monosyllable>

<spelling> == 'strip'

<phonology> == "Phon_strip:<>".

Strip2:

<semantics> ==

'To remove thin outer layer(s), e.g. paint or clothing'

<syntax> == Verb:<optional transitive>

<> == Strip1.

# #1: Underspecification

```
Strip1:
    <semantics> ==
    'An elongated rectangular piece of thin material, e.g. paper'
    <syntax> == Noun:<common>
    <morphology> == Stem:<germanic monosyllable>
    <spelling> == 'strip'
    <phonology> == "Phon_strip:<>".

Strip2:
    <semantics> ==
    'To remove thin outer layer(s), e.g. paint or clothing'
    <syntax> == Verb:<optional transitive>
    <> == Strip1.
```

Nodes (lemmata)

Attributes (features)

Values (specifications)

# #1: Underspecification

Strip1:

<semantics> ==

'An elongated rectangular piece of thin material, e.g. paper'

<syntax> == Noun:<common>

<morphology> == Stem:<germanic monosyllable>

<spelling> == 'strip'

<phonology> == "Phon_strip:<>".

Strip2:

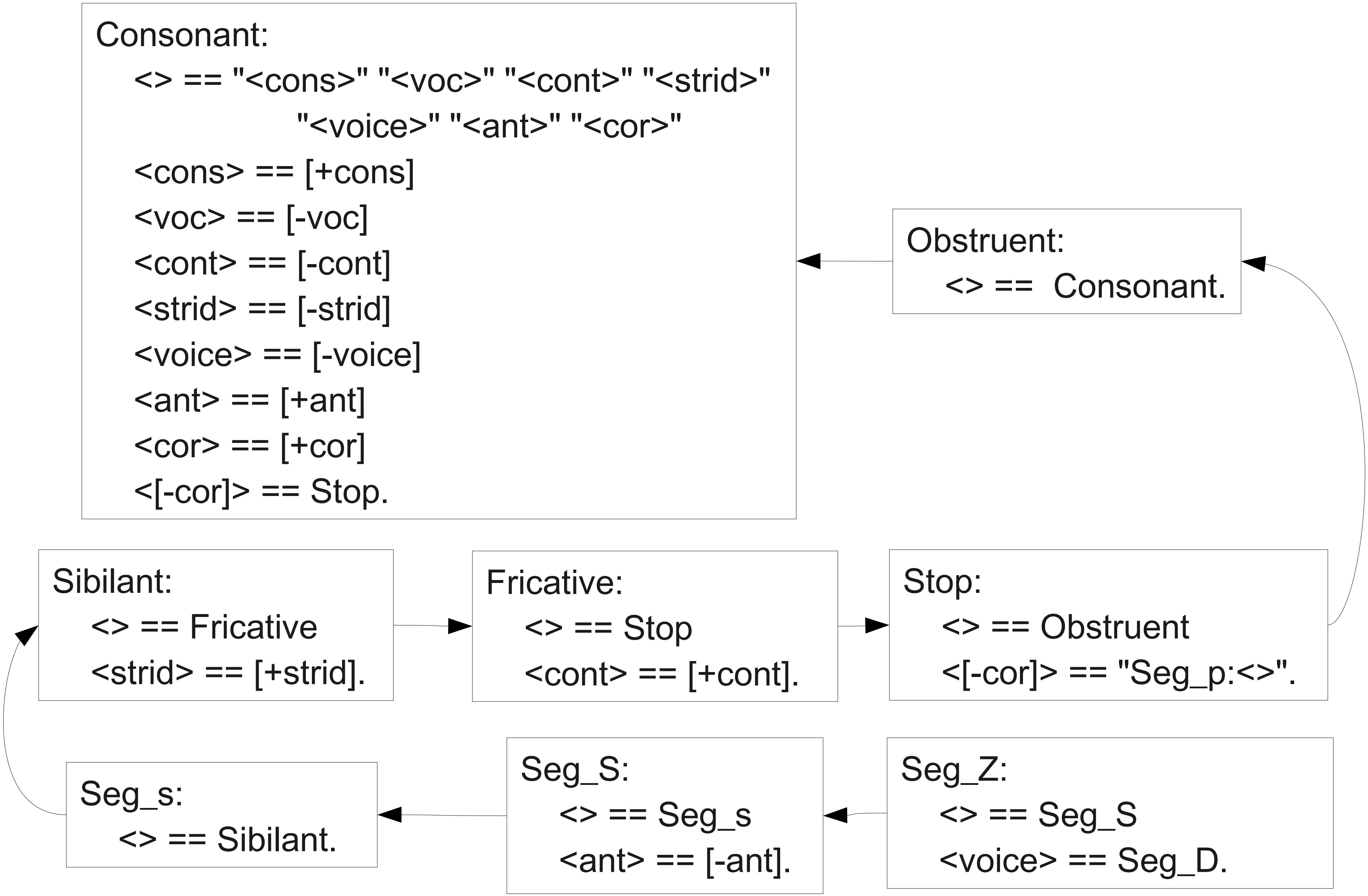
<semantics> ==

'To remove thin outer layer(s), e.g. paint or clothing'

<syntax> == Verb:<optional transitive>

<> == Strip1.

# #1: Segment inheritance hierarchy



# #1: Now for the logical part...

We have notations:

- inheritance network
- a species of attribute-value notation

But a notation does not make a logic

- principles of inference are needed
  - derivation
  - constraint resolution

A logic enables us to

- use facts
- use generalisations
- in order to make inferences (like in a syllogism)
  - about other facts
  - about other generalisations

So:

- <u>infer</u> complete specifications from partial specifications.

# #1: Inference of a complete specification

Requirement:

- From a minimal specification of phonological objects
- Infer a full feature specification, using
  - facts
  - generalisations
  - inheritance from generalisations and prototypes

Needless to say, this inference involves a lot of inference steps.

Thus:

Phon_strip:<> ==

[+cons] [-voc] [+cont] [+strid] [-voice] [+ant] [+cor] ^
[+cons] [-voc] [-cont] [-strid] [-voice] [+ant] [+cor] ^
[+cons] [+voc] [-cont] [-strid] [+voice] [-ant] [+back] [+cor] ^
[-cons] [+voc] [+voice] [+high] [-low] [-back] [-round] ^
[+cons] [-voc] [-cont] [-strid] [-voice] [+ant] [-cor].

# #1: Inference of a complete specification

198 inference steps

from:

Strip1:<phonology>

to:

[+cons][-voc][+cont][+strid][-voice][+ant][+cor] ^
[+cons][-voc][-cont][-strid][-voice][+ant][+cor] ^
[+cons][+voc][-cont][-strid][+voice][-ant][+back][+cor] ^
[-cons][+voc][+voice][+high][-low][-back][-round] ^
[+cons][-voc][-cont][-strid][-voice][+ant][-cor]

- i.e.:

s = [+cons][-voc][+cont][+strid][-voice][+ant][+cor]
t = [+cons][-voc][-cont][-strid][-voice][+ant][+cor]
r = [+cons][+voc][-cont][-strid][+voice][-ant][+back][+cor]
i = [-cons][+voc][+voice][+high][-low][-back][-round]
p = [+cons][-voc][-cont][-strid][-voice][+ant][-cor]

# Challenge 2:

# Making Sense of German Inflection

# #2: Default/override inheritance hierarchy

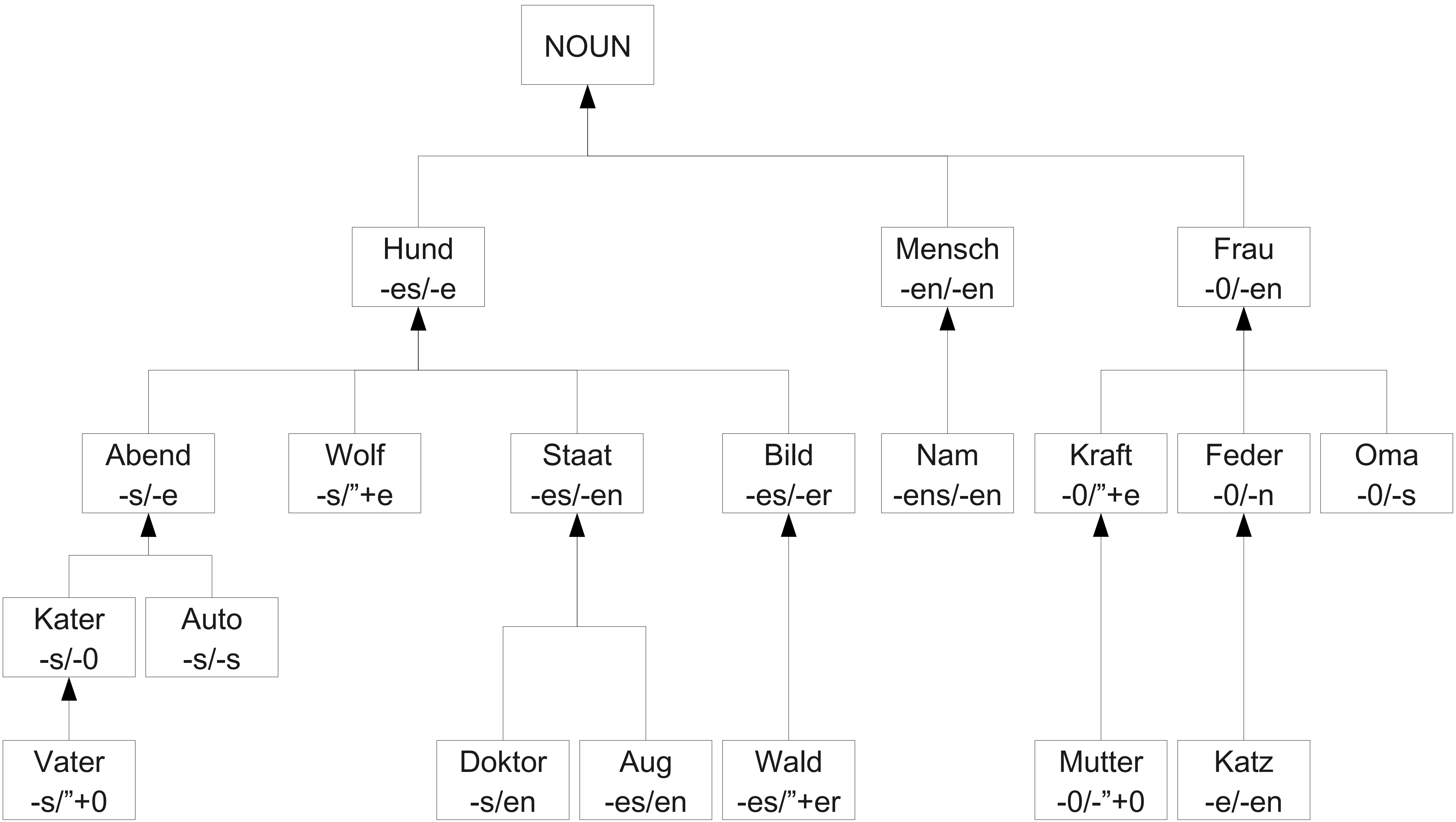
Contrary to common wisdom...

- close analysis shows that German has
  - not just 2 (‘strong/weak)
  - or 3 (e-Plural, en-Plural, weak)
  - but 19 systematically related ‘subdeclensions’.

The challenge:

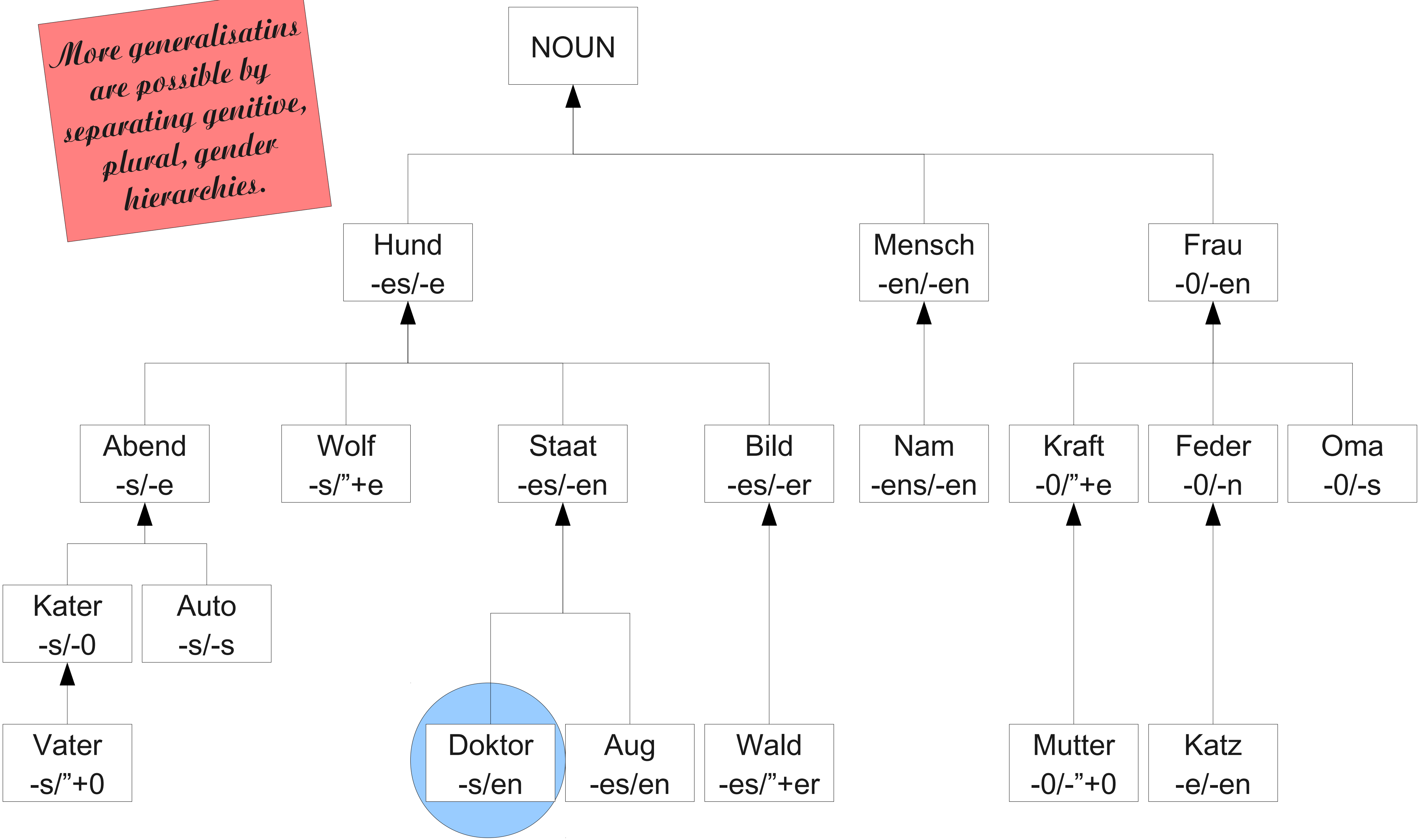
- how to show paradigmatic relations of similarity and difference in between these subdeclensions in a connected manner.

# #2: German Noun Declensions (gen, plur)



# #2: German Noun Declensions (gen, plur)

*More generalisatins are possible by separating genitive, plural, gender hierarchies.*



# Challenge 3:

# German Morphoprosody in Word Formation

# #3: Simplex lexical entries

```
Ost:
 <>                     == N
 <surf phon nuc vow>    == 'O'
 <surf phon cod sib>    == s
 <surf phon cod con>    == t.

See:
 <>                     == N
 <surf phon ons con>    == z
 <surf phon nuc vow>    == e
 <surf phon nuc ext>    == ':'.

Kueste:
 <>                     == N
 <surf phon ons con>    == k
 <surf phon nuc vow>    == 'Y'
 <surf phon cod sib>    == s
 <surf phon cod con>    == t
 <surf phon cod ext>    == @.
```

```
Zug:
 <>                     == N
 <surf phon ons con>    == ts
 <surf phon nuc vow>    == u
 <surf phon nuc ext>    == ':'
 <surf phon cod con>    == k.
```

```
Ver:
 <>                     == Prefix
 <surf phon ons con>    == f
 <surf phon nuc vow>    == @
 <surf phon cod con>    == 'R'.

Bind:
 <>                     == V
 <surf phon ons con>    == b
 <surf phon nuc vow>    == 'I'
 <surf phon cod con>    == n
 <surf phon cod ext>    == d.

Ung:
 <>                     == Suffix
 <surf phon nuc vow>    == 'U'
 <surf phon cod con>    == n
 <surf phon cod ext>    == g.
```

# #3: Complex lexical entries

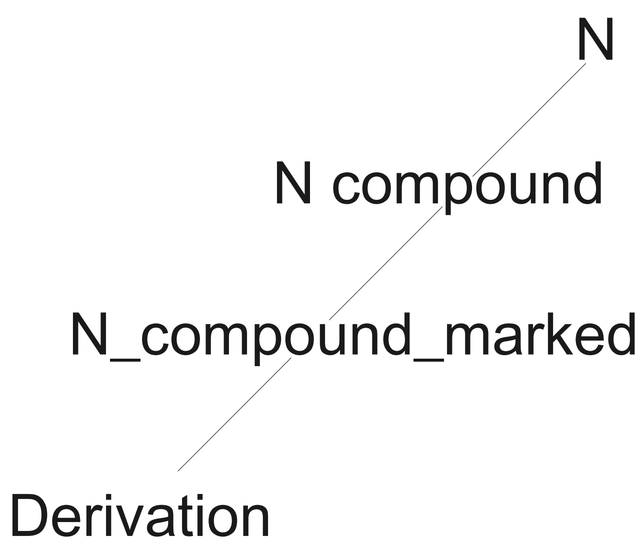
```
Verbind:
 <>                    == V_prefixation
 <determinans>   == "Ver:<>"
 <determinatum> == "Bind:<>".

Verbindung:
 <>                    == N_derivation
 <determinatum> == "Ung:<>"
 <determinans>   == "Verbind:<>".

Zugverbindung:
 <>                    == N_compound
 <determinans>         == "Zug:<>"
 <determinatum>        ==
"Verbindung:<>".
```

```
Ostseekuestenzugverbindung:
 <>                    == N_compound
 <determinans>         == "Ostseekueste:<>"
 <interfix>            == n
 <determinatum>        == "Zugverbindung:<>".

Ostseekueste:
 <>                    == N_compound
 <determinans>         == "Ostsee:<>"
 <determinatum>        == "Kueste:<>".

Ostsee:
 <>                    == N_compound
 <determinans>         == "Ost:<>"
 <determinatum>        == "See:<>".
```

# #3: Paradigmatic & syntagmatic hierarchies

**Challenge**: How to combine 3 syntagmatic hierarchies into 1 paradigmatic hierarchy to make ‘un système où tout se tient’?

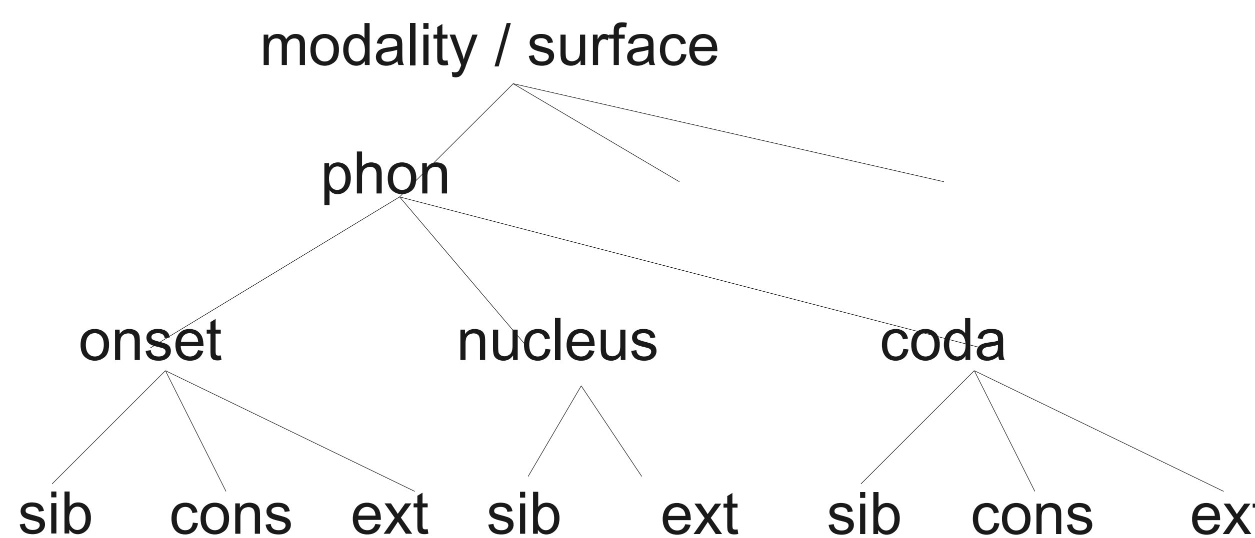
CLASSIFICATION:



# #3: Paradigmatic & syntagmatic hierarchies

**Challenge**: How to combine 3 syntagmatic hierarchies into 1 paradigmatic hierarchy to make ‘un système où tout se tient’?

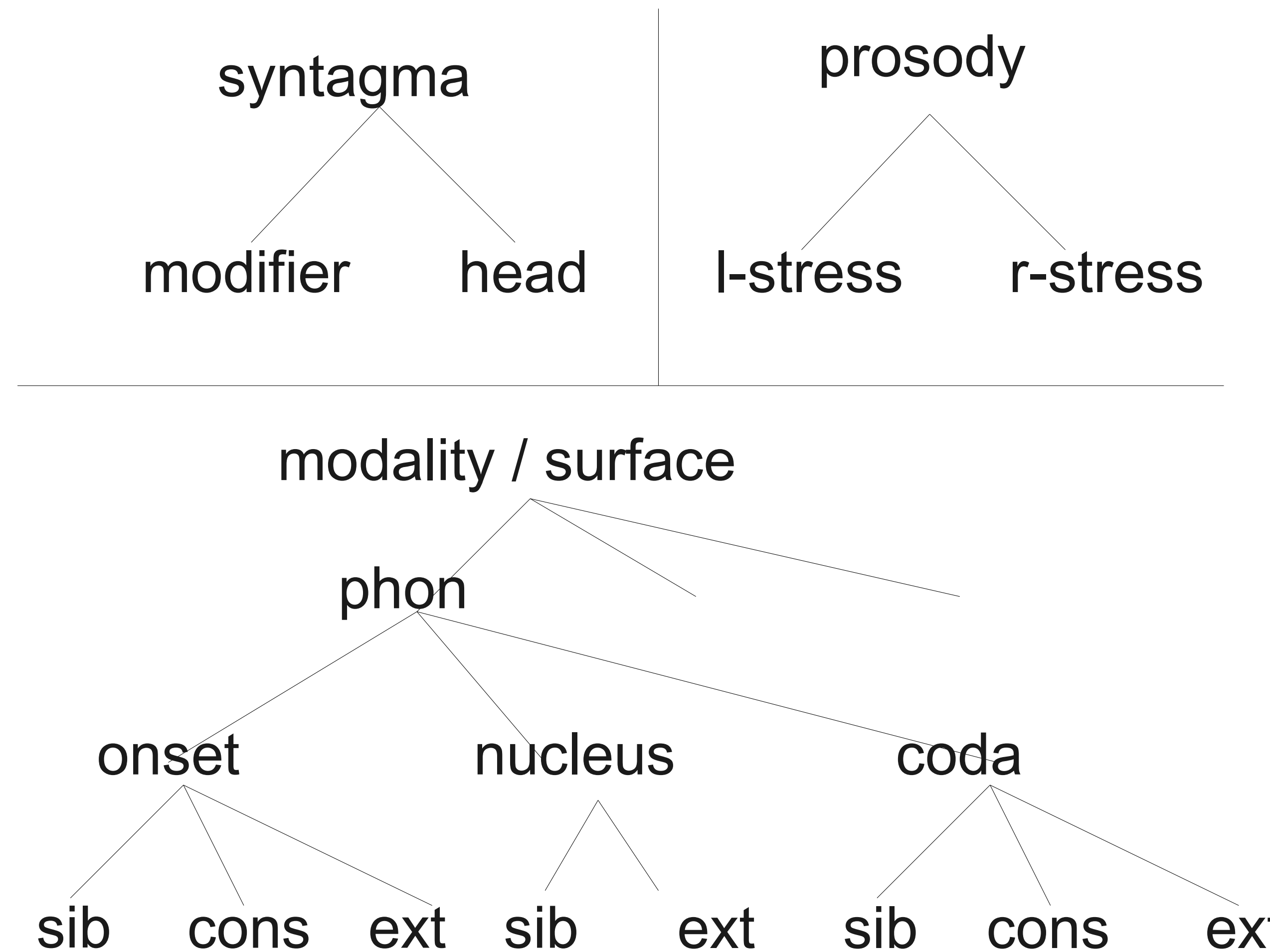
COMPOSITION (more than ‘segmentation’):



# #3: Paradigmatic & syntagmatic hierarchies

**Challenge**: How to combine 3 syntagmatic hierarchies into 1 paradigmatic hierarchy to make ‘un système où tout se tient’?

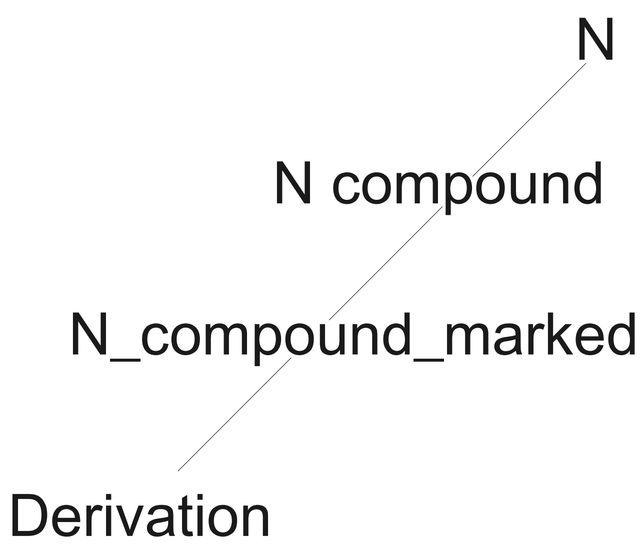
COMPOSITION (more than ‘segmentation’):



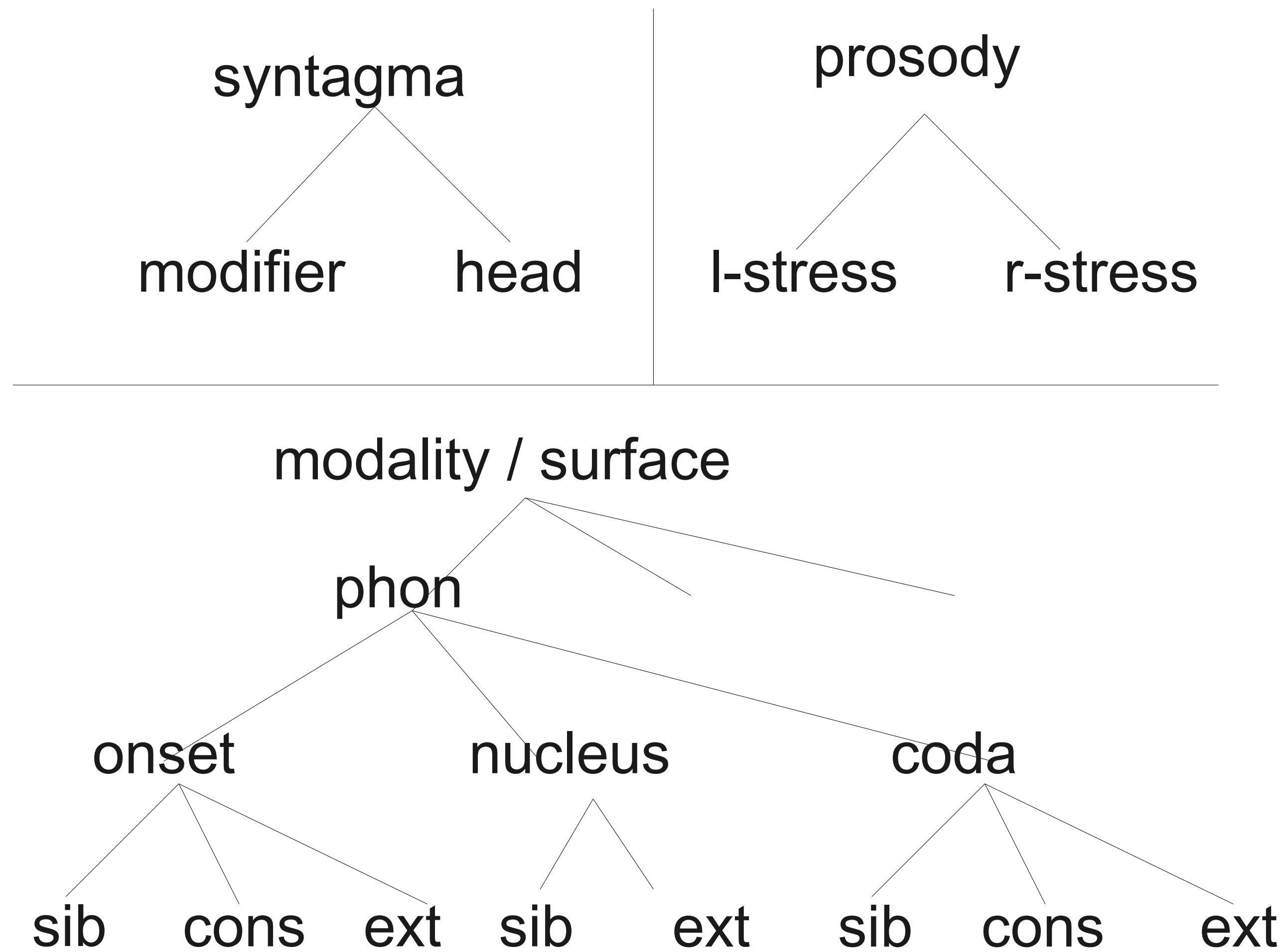
# #3: Paradigmatic & syntagmatic hierarchies

**Challenge**: How to combine 3 syntagmatic hierarchies into 1 paradigmatic hierarchy to make ‘un système où tout se tient’?

CLASSIFICATION:




COMPOSITION (more than ‘segmentation’):



# #3: Output as theorem

Ostseekuestenzugverbindung:< surf phon qlp > =
[ " O s t ] ^ [ z e : ] ^ [ k Y s t @ ] n ^
[ % ts u : k ] ^ [ f @ R ] ^ [ % b I n d ] ^ [ U n g ] .

Ostseekuestenzugverbindung:< surf phon qlp > =

| | |
|---|---|
| Ost | [ " O s t ] ^ |
| See | [ z e : ] ^ |
| Küsten | [ k Y s t @ ] n ^ |
| Zug | [ % ts u : k ] ^ |
| Ver | [ f @ R ] ^ |
| bind | [ % b I n d ] ^ |
| ung | [ U n g ] . |

## 465 inference steps

# #3: Inheritance of stress assignment

```
Stress:
    <>                              == ""
    <modi right_stress>             ==
    <modi right_stress head>== <modi left_stress head>
    <modi left_stress head>         == '%'
    <head right_stress modi>        == <modi>
    <head right_stress head>        == <head left_stress head>
    <head left_stress>              == <modi right_stress>
    <head left_stress head>         == <head>.
```

# Challenge 4:

# Feature displacement in Kikuyu

# #4: Feature displacement in Kikuyu

The challenge:

- The lexical tone on a given syllable is realised one the next syllable (with a number of additional adjustments).
- e.g.:

to + mo + ror + íre → tomororiré
'we looked-at him/her'

to + má + ror + íre → tomaróriré
'we looked-at them'

# #4 Output as theorems

Lexical: Kikuyu:< we him look_at past > = [ L ° to ] [ L ° mo ] [ L ° rOr ] [ H ° i ] [ H ° rE ] .

Delayed: Kikuyu:< d we him look_at past > = [ L ° to ] [ L ° mo ] [ L ° rOr ] [ L ° i ] [ H ° rE ] .

Lexical: Kikuyu:< we them look_at past > = [ L ° to ] [ H ° ma ] [ L ° rOr ] [ H ° i ] [ H ° rE ] .

Delayed: Kikuyu:< d we them look_at past > = [ L ° to ] [ L ° ma ] [ H ° rOr ] [ L ° i ] [ H ° rE ] .

120 – 134 – 119 - 133 inference steps

# Challenge 5:

# Tonal lookahead in Baule

# #5: The problem: linear context sensitivity

Baule_q0:< h l h l > = hc l !h l .

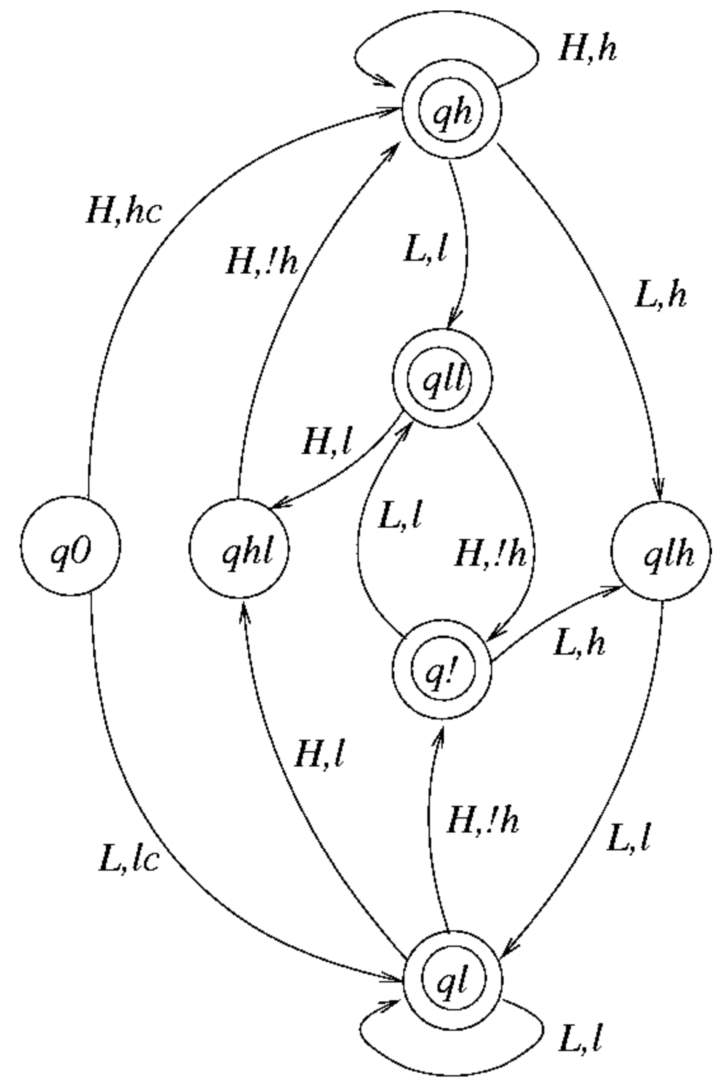
Baule_q0:< l h l h > = lc !h l !h .

Baule_q0:< h h l l h h l l >= hc h h l l !h h l .

Baule_q0:< l l h h l l h h >= lc l l !h h l l !h .

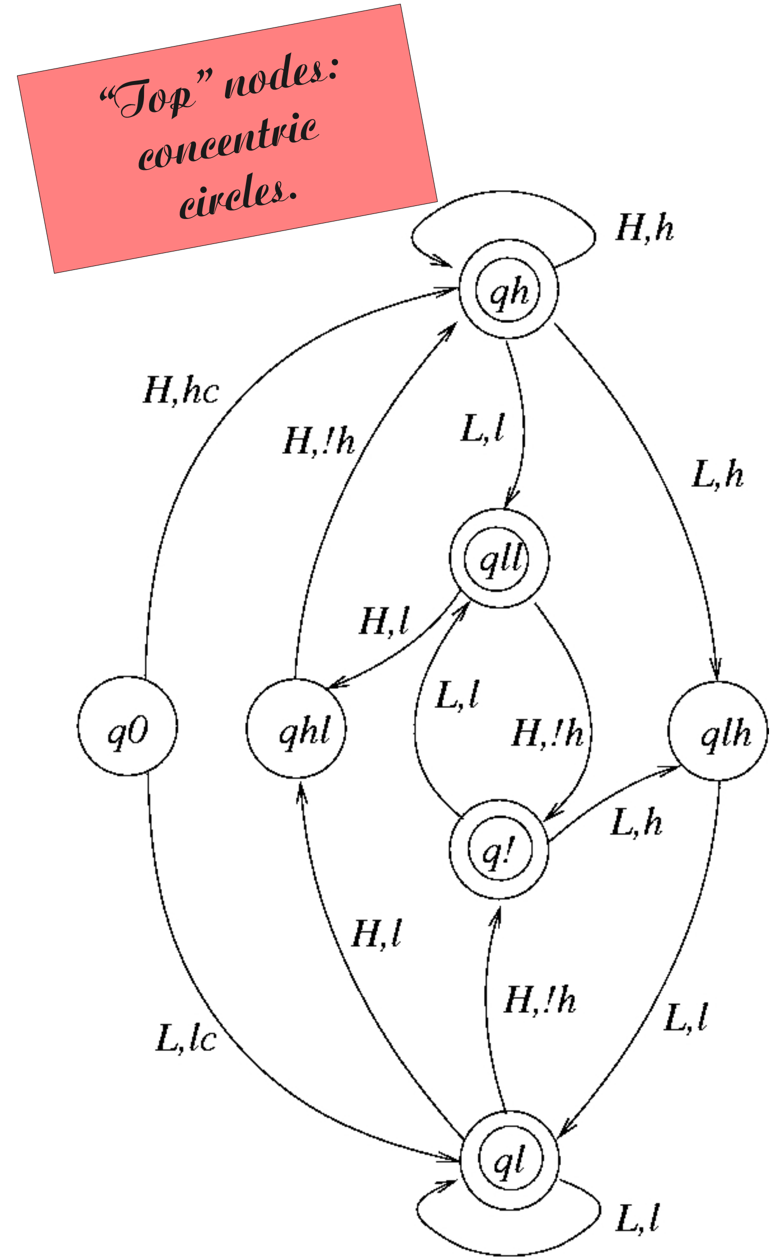
# #5: Modelled as nondeterministic FSN

Baule_q0:< h l h l > = hc l !h l .
Baule_q0:< l h l h > = lc !h l !h .
Baule_q0:< h h l l h h l l >= hc h h l l !h h l .
Baule_q0:< l l h h l l h h >= lc l l !h h l l !h .



# #5: Modelled as nondeterministic FSN

Baule_q0:< h l h l > = hc l !h l .
Baule_q0:< l h l h > = lc !h l !h .
Baule_q0:< h h l l h h l l >= hc h h l l !h h l .
Baule_q0:< l l h h l l h h >= lc l l !h h l l !h .



# #5: Simpler model with 1-place lookahead

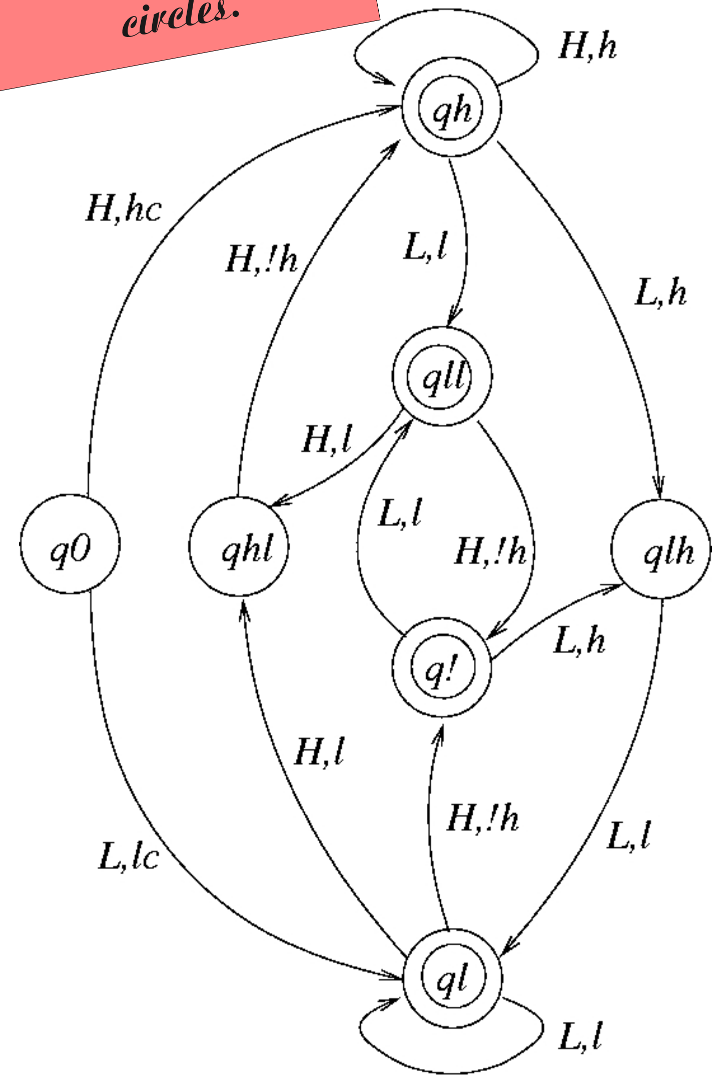
Baule_q0:< h l h l > = hc l !h l .
Baule_q0:< l h l h > = lc !h l !h .
Baule_q0:< h h l l h h l l >= hc h h l l !h h l .
Baule_q0:< l l h h l l h h >= lc l l !h h l l !h .

“Top” nodes: concentric circles.

Baule_q0:<h> == hc Baule_q1:<>
<l> == lc Baule_q2:<>
<> == .

Baule_q1:<h> == h <>
<l l> == h l Baule_q2:<>
<> == Baule_q2.

Baule_q2:<l> == l <>
<h h> == l !h Baule_q1:<>
<h> == !h Baule_q1:<>
<> == Baule_q0.



# #5: Default logical inference

```
Baule_q0:<h> == hc Baule_q1:<>
        <l> ==         lc Baule_q2:<>
        <>  == .

Baule_q1:<h> == h <>
        <l l> ==       h l Baule_q2:<>
        <>    ==       Baule_q2.

Baule_q2:<l> ==  l <>
        <h h> ==     l !h Baule_q1:<>
        <h>   ==     !h Baule_q1:<>
        <>    ==     Baule_q0.
```

```
=0,0,0> LOCAL Baule_q0:< h || h l l h h l l > == hc Baule_q1:<>
        GLOBAL Baule_q0:< h h l l h h l l >

=1,0,1> LOCAL Baule_q1:< h || l l h h l l > == h <>
        GLOBAL Baule_q0:< h h l l h h l l >

=2,0,1> LOCAL Baule_q1:< l l || h h l l > == h l Baule_q2:<>
        GLOBAL Baule_q0:< h h l l h h l l >

=3,0,2> LOCAL Baule_q2:< h h || l l > == l !h Baule_q1:<>
        GLOBAL Baule_q0:< h h l l h h l l >

=4,0,2> LOCAL Baule_q1:< l l > == h l Baule_q2:<>
        GLOBAL Baule_q0:< h h l l h h l l >

=5,0,2> LOCAL Baule_q2:< > == Baule_q0
        GLOBAL Baule_q0:< h h l l h h l l >

=6,0,0> LOCAL Baule_q0:< > ==
        GLOBAL Baule_q0:< h h l l h h l l >

[Query 1 (15 Inferences)] Baule_q0:< h h l l h h l l >
 = hc h h l l !h h l .
```

# Conclusion

# *Mutatis mutandis ...*

There is (typically) a way to account for preferences, defeasible constraints, in a way which satisfies the ‘système où tout se tient’ condition:

1. (typically) formulate implications in terms of inheritance hierarchies
2. (typically) implement inheritance hierarchies with suitable software such as DATR
3. (typically) in order to check the consistency of the resulting complex models efficiently

# Typically ...

# ... thank you for your attention!