Morphotactics in the Matrix
Morphotactic & lexical inference

Ling 567
Jan 17, 2023

Overview

- Lab 2 grading notes
- Morphotactics in the Grammar Matrix
- Matrix-ODIN Morphology (MOM system)
- Lab 3: Which phenomena?

Lab 2 grading notes

- Please be sure to include: grammar, testsuite.txt, tsdb/home, tsdb/skeletons, write up

- Write ups should include specific choices (later tdl) pasted in, in addition to the prose descriptions

- Write ups should include IGT (not just strings or strings + translations)

- Detailed comments posted as Canvas comments

Morphotactics: Basic concepts

- Position class: A supertype to lexical rules which fit in the same slot
- Lexical rule type: *lex-rule* and its subtypes, all have DTR feature
- Lexical rule instance: A grammar entity (manipulatable by the LKB) which inherits from a lexical rule type and specifies a spelling change (including no change).
- Forbids constraint: A specification in the customization system stating that a stem lexical rule type (including a position class) cannot co-occur with another lexical rule type, instance, pc or stem.
- Requires constraint: A specification in the customization system stating that a stem lexical rule type (including a position class) must co-occur with another lexical rule type, instance, pc or stem.

Position classes, inputs and lexical rule hierarchies

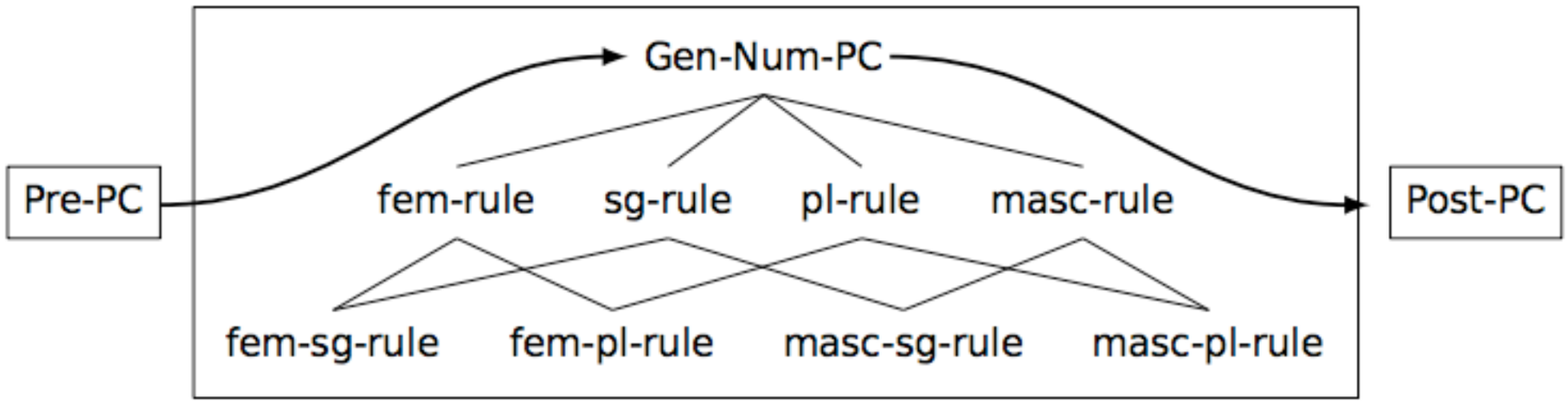
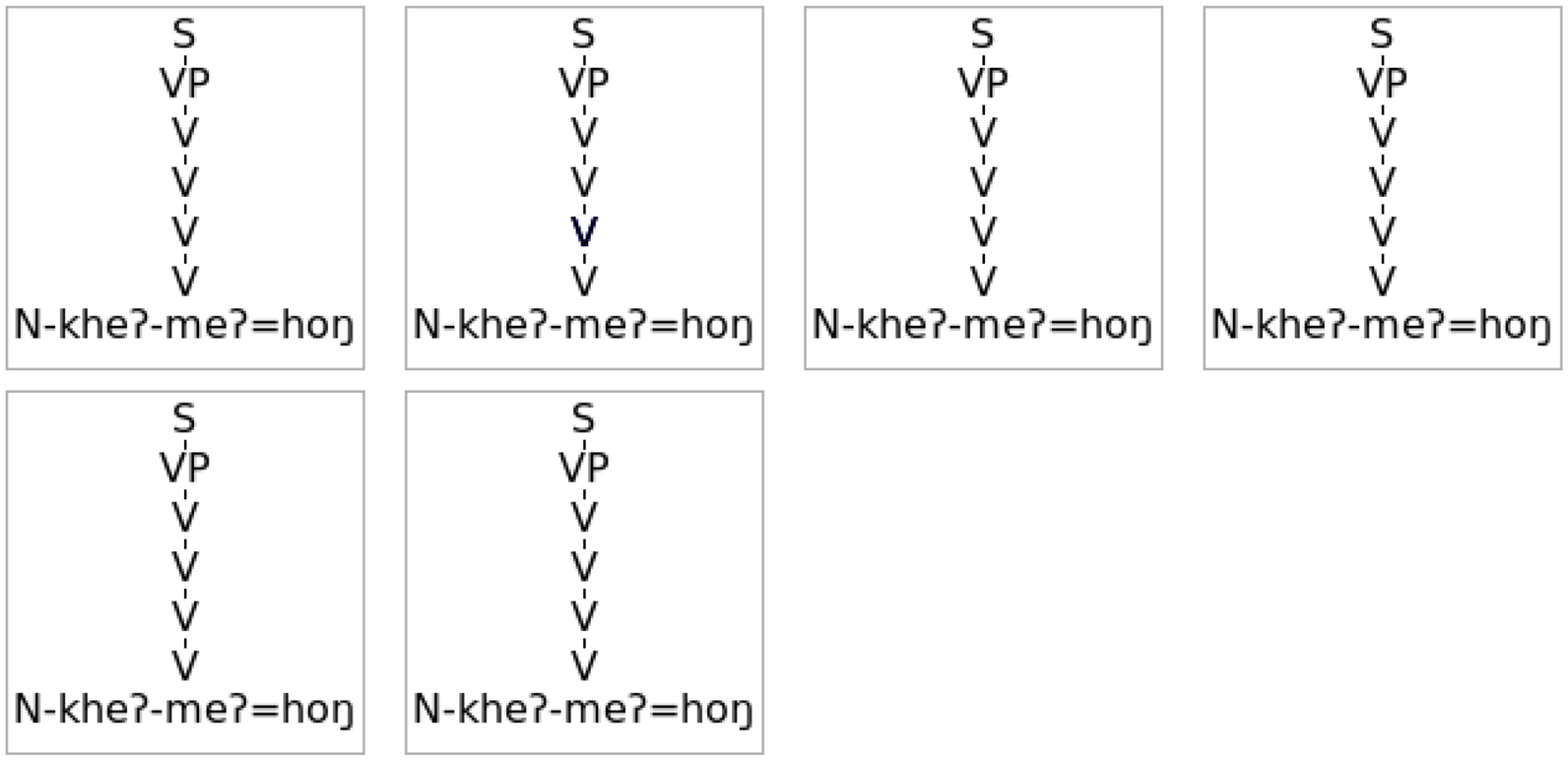


Figure 9: Example lexical rule type hierarchy in a position class

(Goodman 2013)

`N-khe?-me?=hoŋ' (6 Trees)



`derivation' for `N-khe?-me?=hoŋ' [i-id == 7150]

1 (54 verb-pc4_lrt1-suffix3 0.0 0 1 (53 verb-pc68_lrt2-suffix3 0.0 0 1 (52 verb-pc157_lrt2-prefix 0.0 0 1 (29 khe?_3 0.

0 1 (57 verb-pc4_lrt1-suffix3 0.0 0 1 (56 verb-pc5_lrt1-suffix 0.0 0 1 (55 verb-pc157_lrt1-prefix 0.0 0 1 (14 khe?_3 0.0

0 1 (60 verb-pc4_lrt1-suffix3 0.0 0 1 (59 verb-pc5_lrt1-suffix 0.0 0 1 (58 verb-pc157_lrt2-prefix 0.0 0 1 (23 khe?_3 0.0

0 1 (63 verb-pc4_lrt1-suffix3 0.0 0 1 (62 verb-pc21_lrt1-suffix 0.0 0 1 (61 verb-pc157_lrt1-prefix 0.0 0 1 (17 khe?_3 0.0

1 (66 verb-pc4_lrt1-suffix3 0.0 0 1 (65 verb-pc68_lrt2-suffix3 0.0 0 1 (64 verb-pc157_lrt1-prefix 0.0 0 1 (20 khe?_3 0.

0 1 (69 verb-pc4_lrt1-suffix3 0.0 0 1 (68 verb-pc21_lrt1-suffix 0.0 0 1 (67 verb-pc157_lrt2-prefix 0.0 0 1 (26 khe?_3 0.0

(generated b

To define a position class

- Required:
    - Whether or not it is obligatory
    - Possible inputs and prefix/suffix
        - = position in the string
- Optional:
    - Requires/forbids constraints

Position class inputs in choices files

- ybh

```
verb-pc157_name=verb-pc157
verb-pc157_order=prefix
verb-pc157_inputs=verb8, verb108, verb118, verb122,
    verb152, verb184
```

Position class definitions in tdl

- ybh

```
verb-pc157-rule-dtr := word-or-lexrule.

verb-pc157-lex-rule-super := add-only-no-ccont-rule
& infl-lex-rule & verb-pc1-rule-dtr & verb-pc10-
rule-dtr & [...] &
  [ DTR verb-pc157-rule-dtr ].

verb8-verb-lex := intransitive-verb-lex & verb-pc1-
rule-dtr & verb-pc10-rule-dtr & [...] & verb-pc157-
rule-dtr & [...] & verb-pc99-rule-dtr.
```

To define a lex rule type

- Required
    - Nothing (though defaults fill in)
- Optional
    - Name
    - Supertype (if it doesn’t inherit directly from its position class)
    - Feature/value pairs (optional, but this is usually the point!)
    - Requires/forbids constraints

To define a lex rule instance

- Required
  - Affix v. no affix
  - Spelling for affix
- Optional
  - Nothing

Lexical rule types & instances in choices files

- ybh

```
verb-pc157_name=verb-pc157
verb-pc157_order=prefix
verb-pc157_inputs=verb8, verb108, verb118, verb122,
    verb152, verb184

verb-pc157_lrt1_name=verb-pc157_lrt1
      verb-pc157_lrt1_lri1_inflecting=yes
      verb-pc157_lrt1_lri1_orth=n-

verb-pc157_lrt2_name=verb-pc157_lrt2
      verb-pc157_lrt2_lri1_inflecting=yes
      verb-pc157_lrt2_lri1_orth=n-
```

Lexical rule types & instances in tdl

- yakkha.tdl:

```
verb-pc157_lrt1-lex-rule := verb-pc157-lex-rule-super.

verb-pc157_lrt2-lex-rule := verb-pc157-lex-rule-super.
```

- irules.tdl

```
verb-pc157_lrt1-prefix :=
%prefix (* n-)
verb-pc157_lrt1-lex-rule.

verb-pc157_lrt2-prefix :=
%prefix (* n-)
verb-pc157_lrt2-lex-rule.
```

tdl files

- matrix.tdl: Supertypes for lex-rules, which handle the copying up of everything you're not changing

- my_language.tdl: Position classes and lex rule types defined through the customization system; features for inside INFLECTED

- lrules.tdl: Instances for non-spelling-changing lex rules (zero morphemes)

- irules.tdl: Instances for spelling-changing lex rules

Handling of morphotactics

- Rule order handled through super types and typing the DTR feature
- Requires/forbids through the INFLECTED feature

```
num-lex-rule-super := add-only-no-ccont-rule &
   [ INFLECTED [ NUM-FLAG +,
                 PERNUMTENSE-FLAG #pernumtense ],
     DTR common-noun-lex &
         [ INFLECTED.PERNUMTENSE-FLAG #pernumtense ] ].
```

Matrix-ODIN Morphology (MOM) system

(Wax et al 2012, Wax 2014, Zamaraeva et al 2017, 2019, Howell 2020)

- Read in IGT, identify verbs (resp. nouns), identify roots
- Observe root-affix co-occurences
- Merge affixes with shared inputs above threshold into position classes
- Previously: assumed all words of a given POS were in one class; currently iterate between merging words into classes (constrained by valence) and merging affixes into PCs
- Previously: all lexical rules are optional, no non-inflecting rules
- Currently: posit non-inflecting rules for certain types of glosses & make associated position classes obligatory

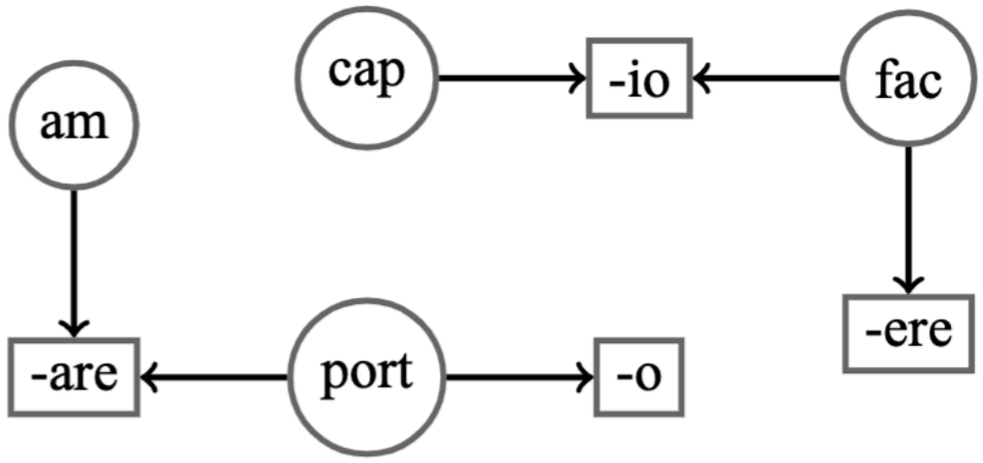


Figure 1: Sample graph MOM will initially build on the example training data consisting of Latin verbs *am-are* ('to love'), *port-are* ('to carry'), *port-o* ('I carry'), *cap-io* ('I take'), *fac-io* ('I do'), and *fac-ere* ('to do').

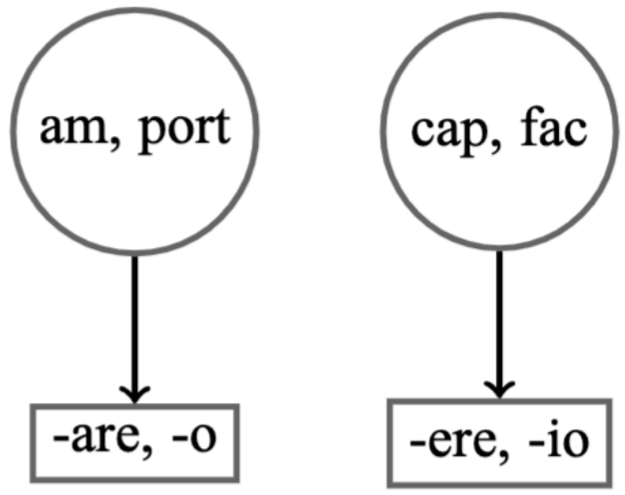


Figure 2: Sample MOM output, after compressing the graph in Figure 1 with a 50% overlap value.

Figure from
Zamareva et al 2019

Lab 3 goals: Morphotactics

- Refine/untangle the rats nest of position classes
- Do you have duplicates? Are there classes that could be merged?
- Can you rename at least some of them after their functions (e.g. subj-agr instead of verb-pc7182)
- NOT: Dealing with the feature constraints that should go on the rules
- Expected results: Cleaner grammars, less ambiguity, maybe less coverage

While in class: Steps to merging position classes

- Find a set of PCs to combine
    - Find union of their LRTs
    - Find union of their inputs
- Create single PC with ^^
- Find all others PCs which took orig PCs as input and replace that with new PC.
- Dedup LRTs with identical affixes (if warranted)

Choosing phenomena for Lab 3

- Additional phenomena this week are only about testsuite extension, because primary development work is on morphotactics

- Testsuite development this week will serve choices & tdl modifications in future weeks

- Testsuite development this week can also illustrate existing functionality of choices files

- To the extent that we can coordinate on phenomena, that is desirable, but not all languages evince all phenomena

List of phenomena

- http://courses.washington.edu/ling567/testsuites.html#phenomena

  - Basic Word Order
  - Pronouns
  - The Rest of the NP
  - Argument Optionality
  - Agreement
  - Case
  - Negation
  - Matrix yes-no questions
  - Embedded complement clauses
  - Adverbial clausal modifiers
  - Modals
  - Coordination
  - Agreement in NP Coordination
  - Tense/aspect
  - Demonstratives/definiteness
  - Possessives
  - Attributive adjectives
  - Adverbs
  - Non-verbal predicates
  - Information Structure
  - Matrix wh questions
  - Valence-Changing Lexical Rules
  - Evidentials

MMT sentences

Dogs sleep
Dogs chase cars
I chase you
Dogs eat
The dogs dont chase cars
I think that you know that dogs chase cars
I ask whether you know that dogs chase cars
Cats and dogs chase cars
Dogs chase cars and cats chase dogs
Cats chase dogs and sleep
Do cats chase dogs
Hungry dogs eat
Dogs in the park eat
Dogs eat in the park
The dogs are hungry
The dogs are in the park
The dogs are the cats
The dog s car sleeps
My dogs sleep
Who sleeps
What do the dogs chase
What do you think the dogs chase
Who asked what the dogs chase
I asked what the dogs chased
The dog sleeps because the cat sleeps
The dog sleeps after the cat sleeps
I can eat glass
It doesnt hurt me

Testsuite reminders

- Examples should be maximally simple: illustrating just the phenomenon in question and the bare minimum otherwise

- This should be true for negative examples well:
  - Exactly one thing wrong
  - Grammar would otherwise parse it, if parsing the simple positive examples

- Stealth interacting phenomena: non-verbal predicates, adverbs, adjectives, coordination, commands, questions…

- You will have to make up examples!