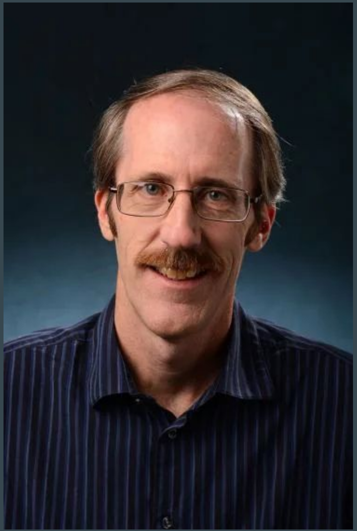
Work with what we have: Bootstrapping from lexical resources for low-resource languages to AMR/UMR annotation

Ongoing Joint Work at CU-Boulder With:



Prof. Alexis Palmer

Prof. Andy Cowell

Julia Bonn

Matt Buchholz

Project Goals

- We want to build meaning representations for Arapaho texts

- Currently, UMR annotation is a lot of duplicative work for annotators
  - A lot of time has been spent already producing an interlinearized database using Toolbox

- **Question:** how can we use the existing materials we have for Arapaho to expedite graph annotation?
  - We're exploring approaches which could be extended to other low-resource languages

```
(t/ taste-01
  :ARG0 (p/ person :wiki "Edmond_Pope"
        :name (n/ name :op1 "Edmund" :op2 "Pope"))
  :ARG1 (f/ free-04
        :ARG1 p)
  :temporal (t2/ today)
  :ord (o/ ordinal-entity :value 1
       :range (m/ more-than
               :op1 (t3/ temporal-quantity :quant 8
                   :unit (m2/ month))))
  :aspect Performance
  :modstr FullAff)
```

“Edmund Pope tasted freedom today for the first time in more than eight months”

```
(t/ taste-01
  :ARG0 (p/ person :wiki "Edmond_Pope"
        :name (n/ name :op1 "Edmund" :op2 "Pope"))
  :ARG1 (f/ free-04
        :ARG1 p)
  :temporal (t2/ today)
  :ord (o/ ordinal-entity :value 1
       :range (m/ more-than
              :op1 (t3/ temporal-quantity :quant 8
                   :unit (m2/ month))))
  :aspect Performance
  :modstr FullAff)
```

“Edmund Pope tasted freedom today for the first time in more than eight months”

```
(s12x / 3o'ohoen-00
        :actor (s12p / person
            :refer-person 3rd
            :refer-number Singular)
        :undergoer (s12p2 / person
            :refer-person 1st
            :refer-number Plural)
        :Aspect Habitual
        :MODSTR FullAff)
```

“3oo3o'ohoenei'eet”

“He is crushing our hands”

```
(s12x / 3o'ohoen-00
      :actor (s12p / person
         :refer-person 3rd
         :refer-number Singular)
      :undergoer (s12p2 / person
         :refer-person 1st
         :refer-number Plural)
      :Aspect Habitual
      :MODSTR FullAff)
```

“3oo3o'ohoenei'eet”

“He is crushing our hands”

What Do You Need for AMR/UMR Annotation?

- Ideally, you'd like a valency lexicon
    - Describes typical argument structures of eventualities
    - Many higher-resource languages have frame files (e.g. English/Chinese PropBank, others)
- **Problem:** we don't have this for Arapaho

Purpose of Frame Files in UMR Annotation

1. List senses of a lemma

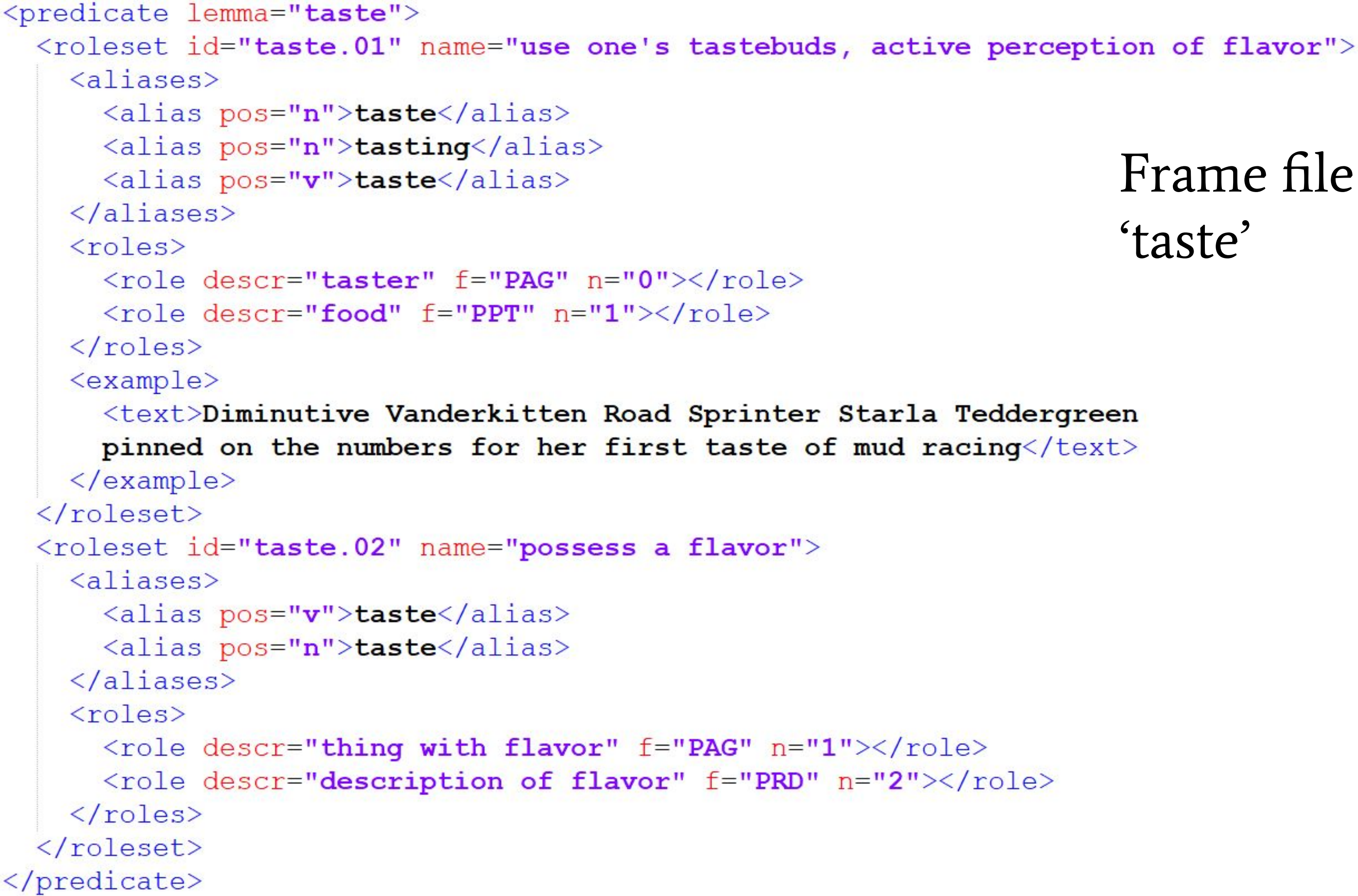
2. Describe the argument structure of those senses

3. Keep an inventory of some of the morphological variants of those senses

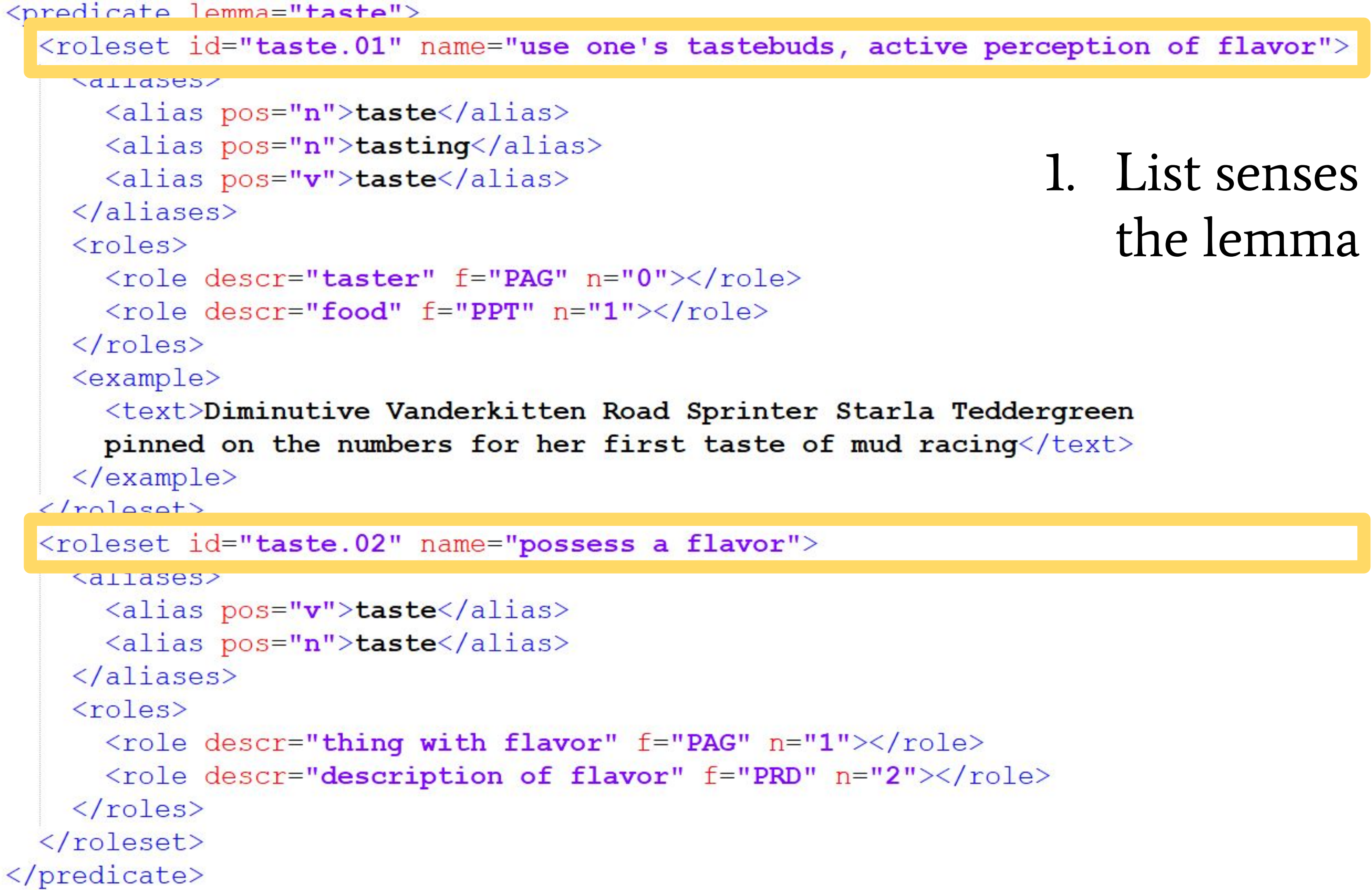
```
(t/ taste-01
  :ARG0 (p/ person :wiki "Edmond_Pope"
        :name (n/ name :op1 "Edmund" :op2 "Pope"))
  :ARG1 (f/ free-04
        :ARG1 p)
  :temporal (t2/ today)
  :ord (o/ ordinal-entity :value 1
       :range (m/ more-than
              :op1 (t3/ temporal-quantity :quant 8
                   :unit (m2/ month))))
  :aspect Performance
  :modstr FullAff)
```

"Edmund Pope tasted freedom today for the first time in more than eight months"

Frame file for ‘taste’

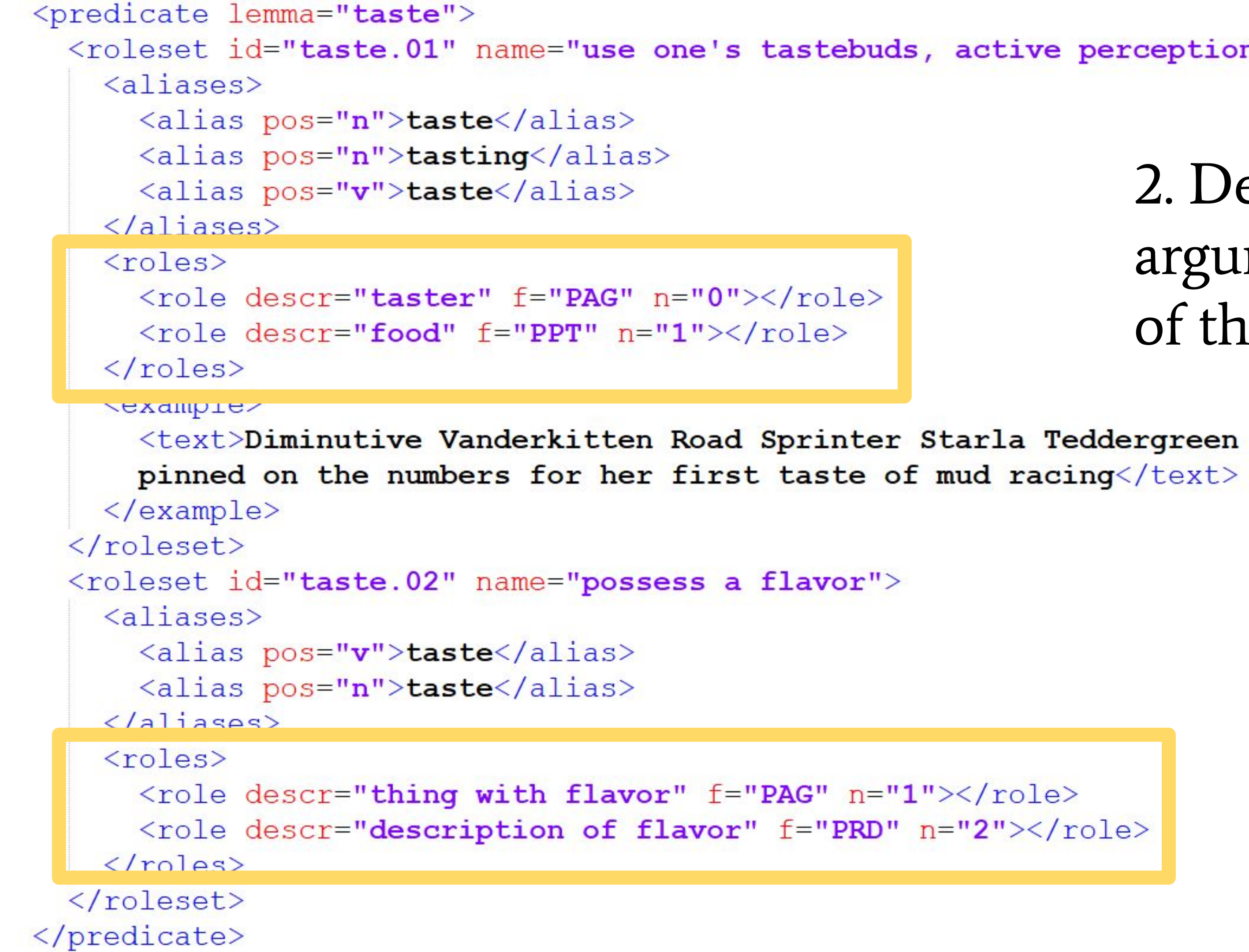


```
<predicate lemma="taste">
  <roleset id="taste.01" name="use one's tastebuds, active perception of flavor">
    <aliases>
      <alias pos="n">taste</alias>
      <alias pos="n">tasting</alias>
      <alias pos="v">taste</alias>
    </aliases>
    <roles>
      <role descr="taster" f="PAG" n="0"></role>
      <role descr="food" f="PPT" n="1"></role>
    </roles>
    <example>
      <text>Diminutive Vanderkitten Road Sprinter Starla Teddergreen
      pinned on the numbers for her first taste of mud racing</text>
    </example>
  </roleset>
  <roleset id="taste.02" name="possess a flavor">
    <aliases>
      <alias pos="v">taste</alias>
      <alias pos="n">taste</alias>
    </aliases>
    <roles>
      <role descr="thing with flavor" f="PAG" n="1"></role>
      <role descr="description of flavor" f="PRD" n="2"></role>
    </roles>
  </roleset>
</predicate>
```



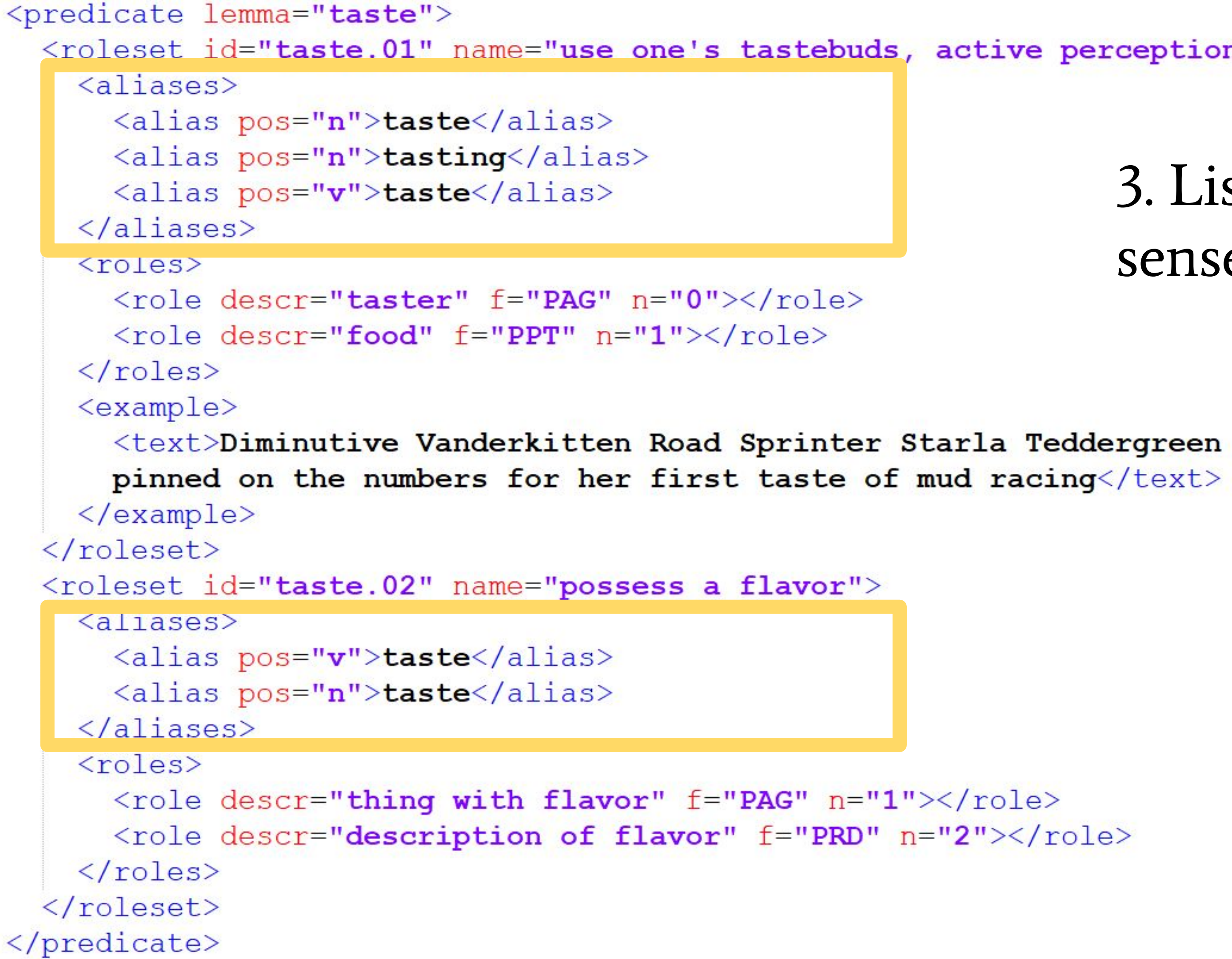
```
<predicate lemma="taste">
  <roleset id="taste.01" name="use one's tastebuds, active perception of flavor">
    <aliases>
      <alias pos="n">taste</alias>
      <alias pos="n">tasting</alias>
      <alias pos="v">taste</alias>
    </aliases>
    <roles>
      <role descr="taster" f="PAG" n="0"></role>
      <role descr="food" f="PPT" n="1"></role>
    </roles>
    <example>
      <text>Diminutive Vanderkitten Road Sprinter Starla Teddergreen
      pinned on the numbers for her first taste of mud racing</text>
    </example>
  </roleset>
  <roleset id="taste.02" name="possess a flavor">
    <aliases>
      <alias pos="v">taste</alias>
      <alias pos="n">taste</alias>
    </aliases>
    <roles>
      <role descr="thing with flavor" f="PAG" n="1"></role>
      <role descr="description of flavor" f="PRD" n="2"></role>
    </roles>
  </roleset>
</predicate>
```

1. List senses of the lemma



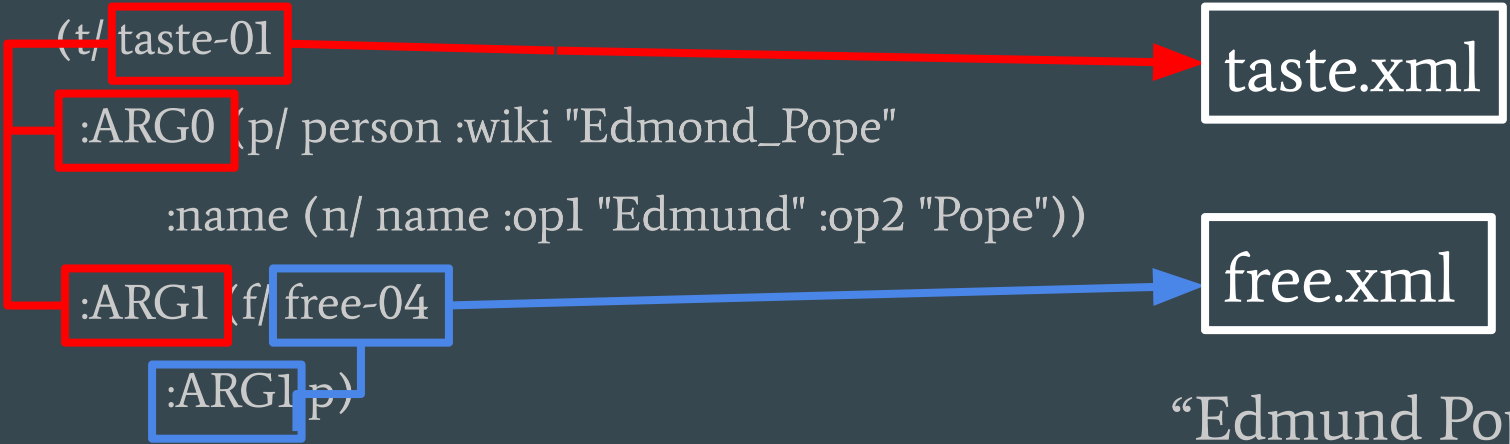
```
<predicate lemma="taste">
  <roleset id="taste.01" name="use one's tastebuds, active perception of flavor">
    <aliases>
      <alias pos="n">taste</alias>
      <alias pos="n">tasting</alias>
      <alias pos="v">taste</alias>
    </aliases>
    <roles>
      <role descr="taster" f="PAG" n="0"></role>
      <role descr="food" f="PPT" n="1"></role>
    </roles>
    <example>
      <text>Diminutive Vanderkitten Road Sprinter Starla Teddergreen
      pinned on the numbers for her first taste of mud racing</text>
    </example>
  </roleset>
  <roleset id="taste.02" name="possess a flavor">
    <aliases>
      <alias pos="v">taste</alias>
      <alias pos="n">taste</alias>
    </aliases>
    <roles>
      <role descr="thing with flavor" f="PAG" n="1"></role>
      <role descr="description of flavor" f="PRD" n="2"></role>
    </roles>
  </roleset>
</predicate>
```

2. Describe argument structure of those senses



```
<predicate lemma="taste">
  <roleset id="taste.01" name="use one's tastebuds, active perception of flavor">
    <aliases>
      <alias pos="n">taste</alias>
      <alias pos="n">tasting</alias>
      <alias pos="v">taste</alias>
    </aliases>
    <roles>
      <role descr="taster" f="PAG" n="0"></role>
      <role descr="food" f="PPT" n="1"></role>
    </roles>
    <example>
      <text>Diminutive Vanderkitten Road Sprinter Starla Teddergreen
      pinned on the numbers for her first taste of mud racing</text>
    </example>
  </roleset>
  <roleset id="taste.02" name="possess a flavor">
    <aliases>
      <alias pos="v">taste</alias>
      <alias pos="n">taste</alias>
    </aliases>
    <roles>
      <role descr="thing with flavor" f="PAG" n="1"></role>
      <role descr="description of flavor" f="PRD" n="2"></role>
    </roles>
  </roleset>
</predicate>
```

3. List aliases of the senses



```
(t/ taste-01
  :ARG0 (p/ person :wiki "Edmond_Pope"
        :name (n/ name :op1 "Edmund" :op2 "Pope"))
  :ARG1 (f/ free-04
        :ARG1 p)
  :temporal (t2/ today)
  :ord (o/ ordinal-entity :value 1
       :range (m/ more-than
             :op1 (t3/ temporal-quantity :quant 8
                  :unit (m2/ month))))
  :aspect Performance
  :modstr FullAff)
```

taste.xml

free.xml

“Edmund Pope tasted freedom today for the first time in more than eight months”

What Resources Do We Have For Arapaho?

- Traditional lexicon
  - JSON-formatted file with ~20,000 entries

- Text database of IGT
  - Toolbox format
  - ~100,000 sentences
  - Variety of genres

- A grammar + the expertise of Prof. Andy Cowell

- **Question:** How do we build frame files for Arapaho verbs from these?

Relevant Arapaho Background

Four main verb classes, divided by grammatical animacy of arguments and transitivity

Class	Abbreviation	Subject	Object	Example
Animate intransitive	VAI	Animate	∅	*nihooneihi-* ‘to be yellow’ (animate)
Inanimate intransitive	VII	Inanimate	∅	*nihooyoo-* ‘to be yellow’ (inanimate)
Transitive animate	VTA	Animate	Animate	*noohow-* ‘to see someone’
Transitive inanimate	VTI	Animate	Inanimate	*noohoot-* ‘to see something’

Relevant Arapaho Background

- Verbs are marked for agreement with subject / object via cumulative suffix

heetbiinein.

heet-	biin	-ein
FUT-	give s.t. to s.o.	-3S/2S
pref-	vta	-infl

“He will give it to you”

Relevant Arapaho Background

- Other arguments might appear obliquely (esp. for ditransitives, etc.)

Niiseti'	bei'ci3ei'i	nihbiineit.		
niiseti -'	bei'ci3ei'i	nih-	biin	-eit
one -0S	money	PAST-	give	-4/3S
vii -infl	ni.pl	pref-	vta	-infl

“He gave [Wovoka] a dollar”

Arapaho has many derivational strategies

We can modify the verb root with many different types of affixes:

- Valency changes (voice, making objects indefinite, etc.)
- Changing the manner of the action (rapid / violent, iterative, etc.)
- Derive different parts of speech (adverbials, nominalizations)
- ... a whole lot more

Arapaho has many, many derivational strategies

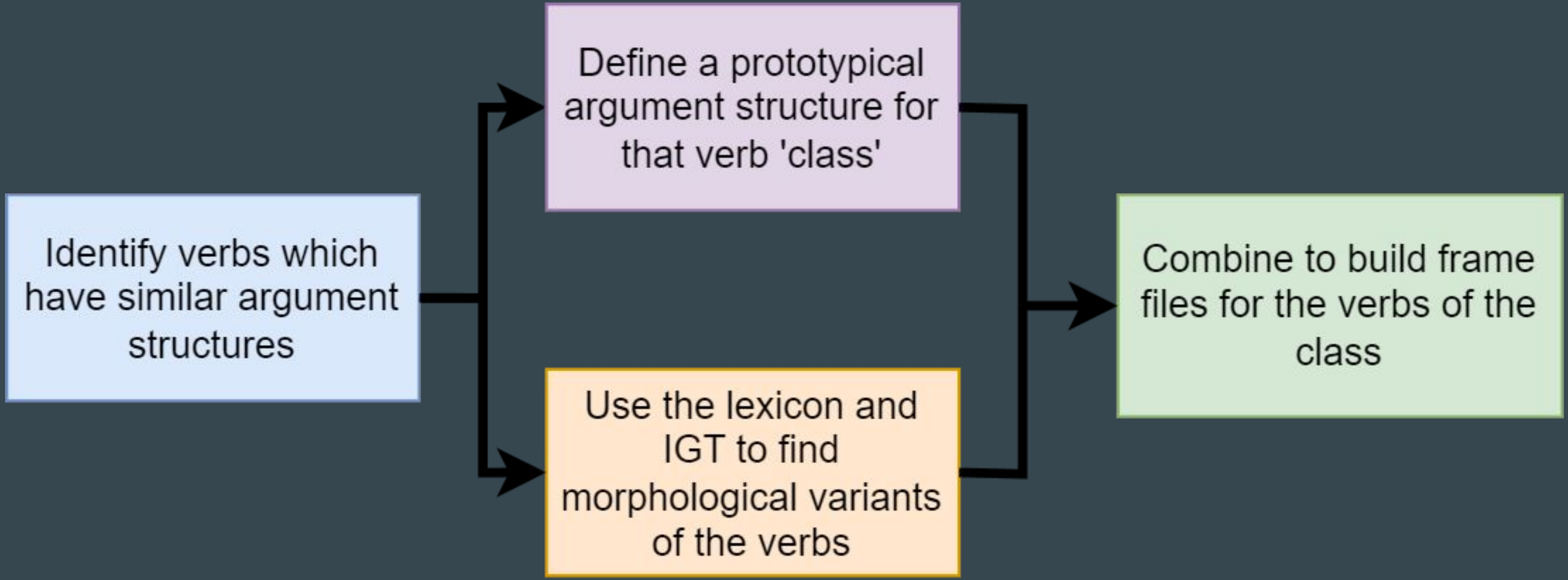
Take a root like *noohow-* ‘see s.o.’

noohobeihi-	‘visible’	(passive voice)
noohobee-	‘seen’	(middle voice)
neenoohob-	‘keep seeing s.o, intensely look at s.o..’	(intensity)
noohowootiihi’	‘by seeing, by watching’	(adverbial)
noohowkuu3-	‘catch a glimpse of s.o.’	(rapid action)
noohowkuu3ei-	‘catch a glimpse of things’	(+ indefinite object)

Making Arapaho Frame Files (the quick and partially-correct way)

- We want to create frame files where:
  - Rolesets have reasonable argument structures
  - Many morphological variants are easily discoverable
- We want to do this with as little effort as possible
  - Avoid getting too deep into morpho-phonology

Our process for Making Frame Files



Identify verbs which have similar argument structures

Define a prototypical argument structure for that verb 'class'

Use the lexicon and IGT to find morphological variants of the verbs

Combine to build frame files for the verbs of the class

Identify Verbs with Similar Argument Structures

- We seed the process with verb classes identified by an expert in the language

Class Description	Experiencer-type verbs, with stimulus	Give-type transfer verbs
Super Class	VTA or VTI	VTA.Ditransitive
Examples	*noohow-* 'see s.o.' *noohoot-* 'see s.t.' *niiton-* 'hear s.o.' *benoot-* 'smell s.t."	*biin-* 'give s.t. to s.o.' *bexoow- 'bestow s.t. on s.o.'* *tou3e'ein-* 'give s.t. as gift to s.o.'

Describe a Typical Argument Structure

- Focus is on arguments which are syntactically supported or implicit but semantically essential

Class Description	Experiencer-type verbs, with stimulus	Give-type transfer verbs
Super Class	VTA or VTI	VTA.Ditransitive
Typical argument structure	Experiencer (Proto-agent) Stimulus (Proto-patient)	Giver (Proto-agent) Recipient (Goal) Thing given (Proto-patient)

Discover Morphological Variants for a Verb Root

```
"L19445": {
        "base form": "noohób-",
        "pos": "vta",
        "morphology": "nooh-ow-",
        "gloss": "see s.o.",
        "allolexemes": [
            "nonoohób- IC",
            "noohob-"
        ],
        "senses": [
                "definition": "see s.o.",
                ]
}
```

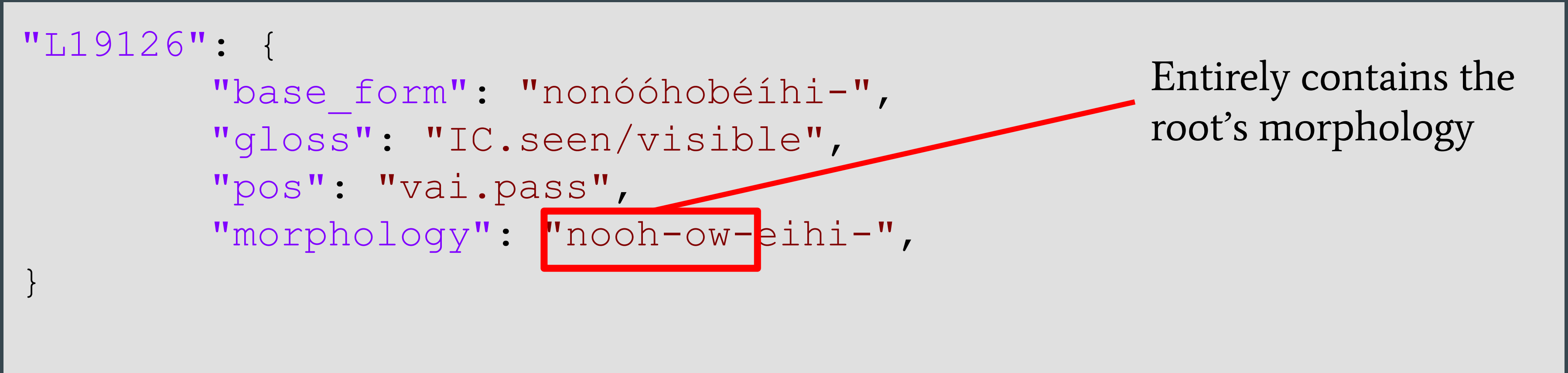
Discover Morphological Variants for a Root

- From the lexicon:
  - Get allolexemes
  - Look for other entries which contain the root’s entire morphology

- From the IGT:
  - Perform fuzzy matching against items in the morpheme breakdown

Discover Morphological Variants for a Root

- Looking for an alias of *noohob-* ‘see s.o.’



```
"L19126": {
         "base_form": "nonóóhobéíhi-",
         "gloss": "IC.seen/visible",
         "pos": "vai.pass",
         "morphology": "nooh-ow-eihi-",
}
```

Entirely contains the root’s morphology

Discover Morphological Variants for a Root

- Looking for an alias of *noohob-* ‘see s.o.’

Fuzzy match with the verb root

hooxou'ei'oo3i',		niinoohobeihi3i'.		
hooxou'ei'oo	-3i'	nii-	noohobeihi	-3i'
IC.smile	-3PL	IMPERF-	to.be.seen	-3PL
vai	-infl	pref-	vai.pass	-infl

Potential Alias	Gloss / Translation	Part of Speech
noohowkuu3ei-	‘catch a glimpse of things’	VAI w/ indefinite object
hoownoohobeihi	‘invisible’	VAI.passive.neg
nonoohobeihi-	‘IC.seen/visible’	VAI.passive
noohobetiitooni-	‘people are looking at each other’	VII.impers
niitnoohobeihiinoo' hee3ou'use'	‘scoreboard’	NI
noohobeti-	‘see one’s self; see each other’	VAI.reflexive

Discover Morphological Variants for a Root

- Post-processing
    - Eliminate things which aren’t straightforward derivations
    - Create new rolesets where appropriate
        - Object incorporation
        - Directional vector-adding affixes
        - etc.

Other Difficulties

- Hard to handle shared roots

Potential Alias	Gloss / Translation	Part of Speech
biin-	‘give something to someone’	VTA.ditransitive
biineti-	‘give each other things’	VAI.reflexive
biin-	‘eat something animate’	VTA
niibiineet	‘cannibal’	NA/VAI.imperfective

Other Difficulties

- Shorter roots tend to have many false positives

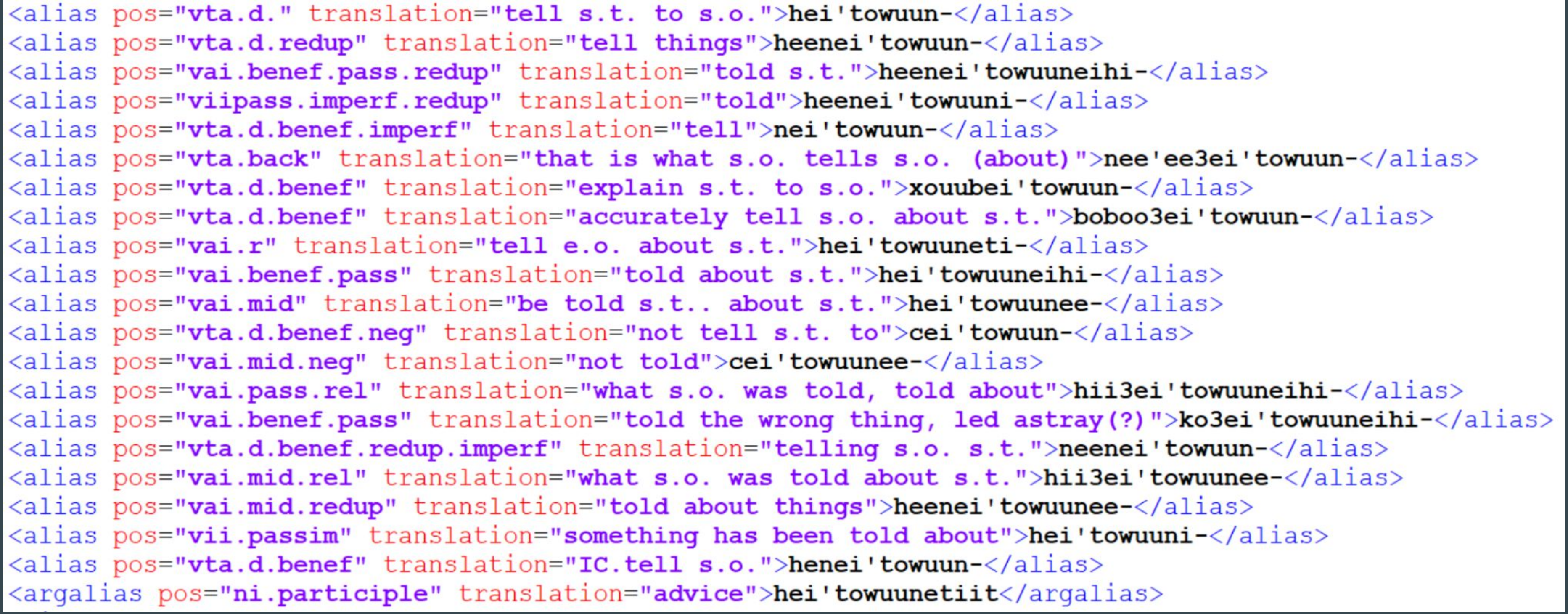
Potential Alias	Gloss / Translation	Part of Speech
biin-	‘give something to someone’	VTA.ditransitive
biinoho'o	‘digging stick; crowbar’	NA
biinohooo	‘Hardin, MT’	Placename
biinohei-	‘paw ground, like a bull’	VAI with implicit object

Other Difficulties

- Hard to define boundaries just based on etymological and semantic relatedness

- Many derivations which share a root, but derivations might get farther away in meaning; are they still describing the same event for annotation?

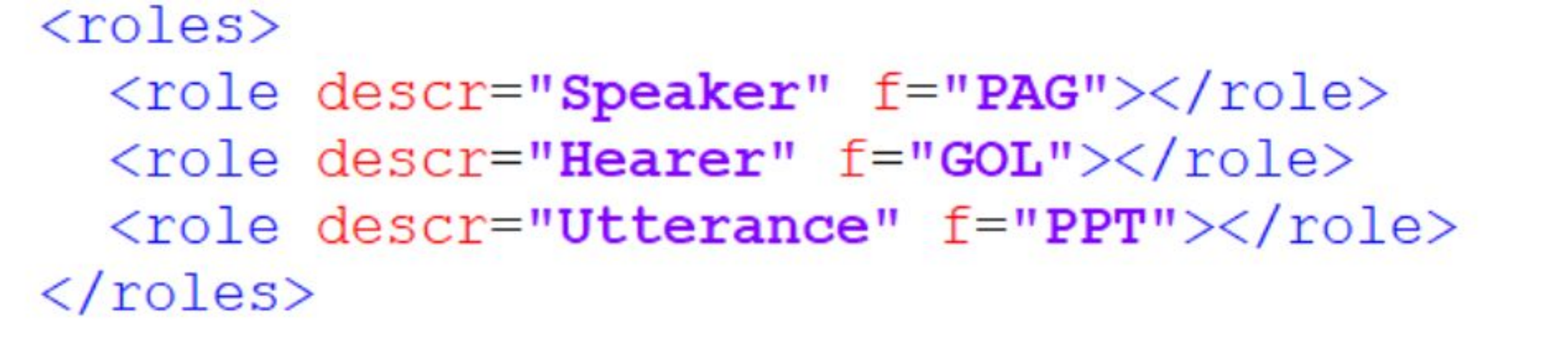
Example Frame File for *hei'towuun-* 'tell s.t. to s.o'



```
<alias pos="vta.d." translation="tell s.t. to s.o.">hei'towuun-</alias>
<alias pos="vta.d.redup" translation="tell things">heenei'towuun-</alias>
<alias pos="vai.benef.pass.redup" translation="told s.t.">heenei'towuuneihi-</alias>
<alias pos="viipass.imperf.redup" translation="told">heenei'towuuni-</alias>
<alias pos="vta.d.benef.imperf" translation="tell">nei'towuun-</alias>
<alias pos="vta.back" translation="that is what s.o. tells s.o. (about)">nee'ee3ei'towuun-</alias>
<alias pos="vta.d.benef" translation="explain s.t. to s.o.">xouubei'towuun-</alias>
<alias pos="vta.d.benef" translation="accurately tell s.o. about s.t.">boboo3ei'towuun-</alias>
<alias pos="vai.r" translation="tell e.o. about s.t.">hei'towuuneti-</alias>
<alias pos="vai.benef.pass" translation="told about s.t.">hei'towuuneihi-</alias>
<alias pos="vai.mid" translation="be told s.t.. about s.t.">hei'towuunee-</alias>
<alias pos="vta.d.benef.neg" translation="not tell s.t. to">cei'towuun-</alias>
<alias pos="vai.mid.neg" translation="not told">cei'towuunee-</alias>
<alias pos="vai.pass.rel" translation="what s.o. was told, told about">hii3ei'towuuneihi-</alias>
<alias pos="vai.benef.pass" translation="told the wrong thing, led astray(?)">ko3ei'towuuneihi-</alias>
<alias pos="vta.d.benef.redup.imperf" translation="telling s.o. s.t.">neenei'towuun-</alias>
<alias pos="vai.mid.rel" translation="what s.o. was told about s.t.">hii3ei'towuunee-</alias>
<alias pos="vai.mid.redup" translation="told about things">heenei'towuunee-</alias>
<alias pos="vii.passim" translation="something has been told about">hei'towuuni-</alias>
<alias pos="vta.d.benef" translation="IC.tell s.o.">henei'towuun-</alias>
<argalias pos="ni.participle" translation="advice">hei'towuunetiit</argalias>
```

Example Frame File for *hei’towuun-* ‘tell s.t. to s.o’

- All of those aliases fall under this roleset:



```
<roles>
  <role descr="Speaker" f="PAG"></role>
  <role descr="Hearer" f="GOL"></role>
  <role descr="Utterance" f="PPT"></role>
</roles>
```

What’s next?

- Goal is to make UMR graph annotation easier when you have existing IGT

- We want to (partially) automate graph generation using the IGT and generated frame files

Automatic graph generation

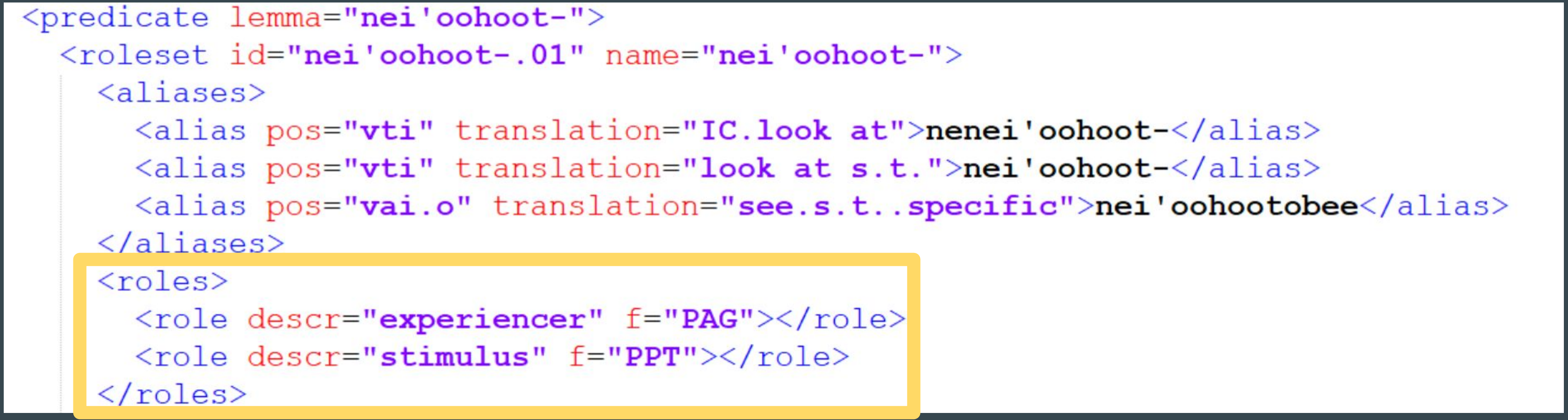
EvLB.045

Nih'oowunei'oohooto' biito'owu'.

nih-	'oowu-	**nei'oohoot**	- o'	biito'owu'
PAST-	down-	look.at	- 3S	land

“He looked down at the world.”

Automatic graph generation



```
<predicate lemma="nei'oohoot-">
  <roleset id="nei'oohoot-.01" name="nei'oohoot-">
    <aliases>
      <alias pos="vti" translation="IC.look at">nenei'oohoot-</alias>
      <alias pos="vti" translation="look at s.t.">nei'oohoot-</alias>
      <alias pos="vai.o" translation="see.s.t..specific">nei'oohootobee</alias>
    </aliases>
    <roles>
      <role descr="experiencer" f="PAG"></role>
      <role descr="stimulus" f="PPT"></role>
    </roles>
```

nei'oohoot- frame file defines argument structure

1. Verb expects experiencer and stimulus

2. Verbal agreement marker must be first argument (experiencer)

3. Second noun is probably stimulus

EvLB.045

Nih'oowunei'oohooto' biito'owu'.

nih-	'oowu-	**nei'oohoot**	- o'	biito'owu'
PAST-	down-	look.at	- 3S	land

"He looked down at the world."

Future Work: Automatic graph generation

```
(s1x / nei'oohoot-00
    :actor (s1p / person
        :refer-person 3rd
        :refer-number Singular)
    :stimulus (s1x2 / biito'owu')
    :direction (s1x3 / hoowu-)
    :Aspect Activity
    :MODSTR FullAff)
```

Results

- As a test, we created frame files for about 30 verbs
    - Discarded the ones with obvious problems
- Frame file aliases matched 1.1% of morphemes in the text database
- 5.9% of sentences in the database had at least one morpheme which mapped to a frame file

Conclusion

You can get reasonably far in UMR graph annotation with just a lexicon, IGT, and a little bit of knowledge of the language