Computational Semantics

LING 571 — Deep Processing for NLP
October 24th, 2018
Ryan Georgi

Miscellanea

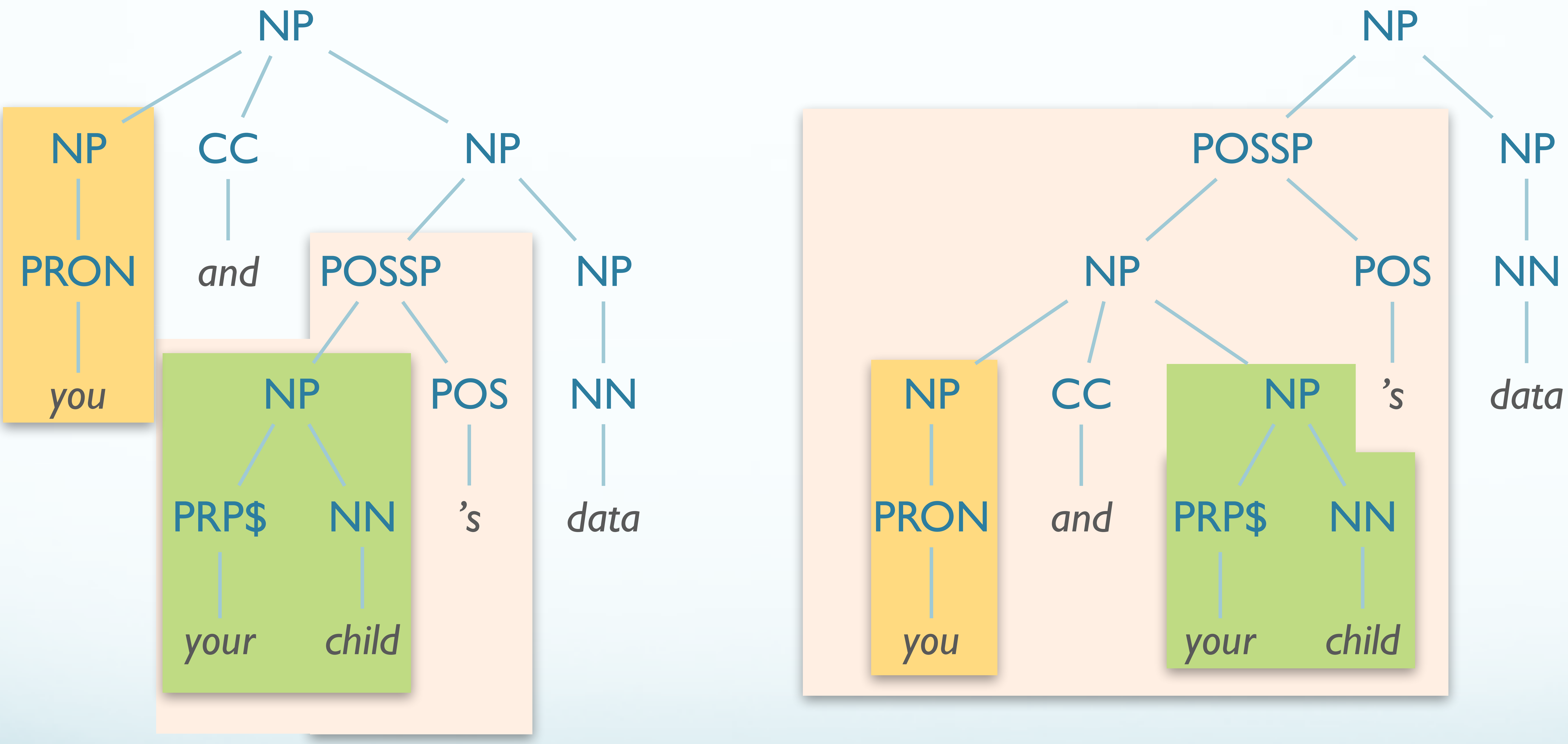
Adventures in Linguistic Ambiguity

- Regarding a learning study from University of Reading, a letter advised potential participants:

"If you are not chosen to take part, you and your child's data will be destroyed."

Source: *The News Quiz*, BBC Radio 4, Feb 2, 2018. [link]

"If you are not chosen to take part,
you and your child's data will be destroyed."



Source: *The News Quiz*, BBC Radio 4, Feb 2, 2018. [link]

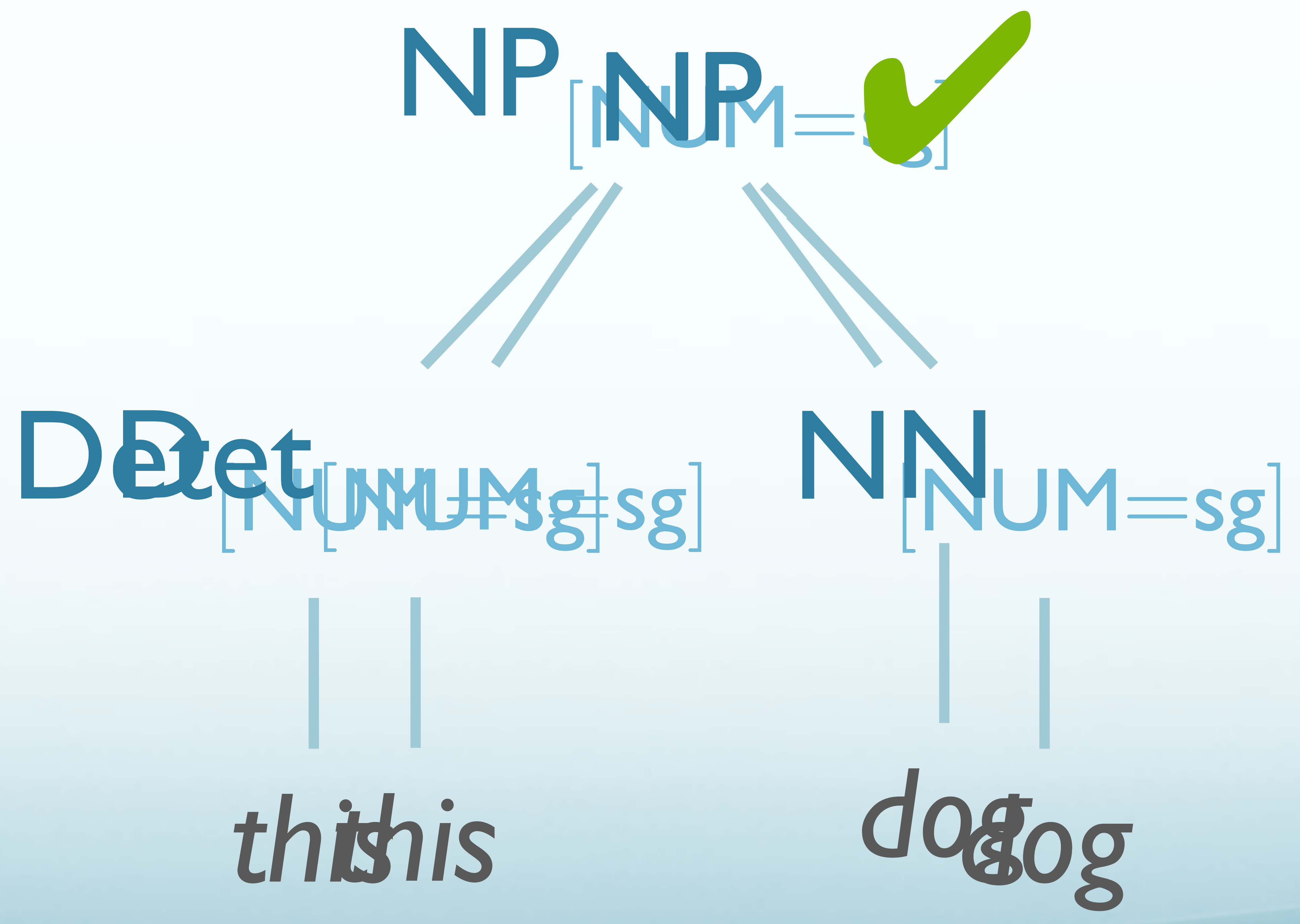
NLTK Feature Syntax

- Basics
  - `X[FEAT1=VALUE1, FEAT2=VALUE2]`
- Variables
  - `X[FEAT=?f]`
- Binary Values
  - `X[-FEAT], Y[+FEAT]`

HW #5: NLTK Feature Syntax

```
NP[NUM=?n] -> Det[NUM=?n] N[NUM=?n]
```

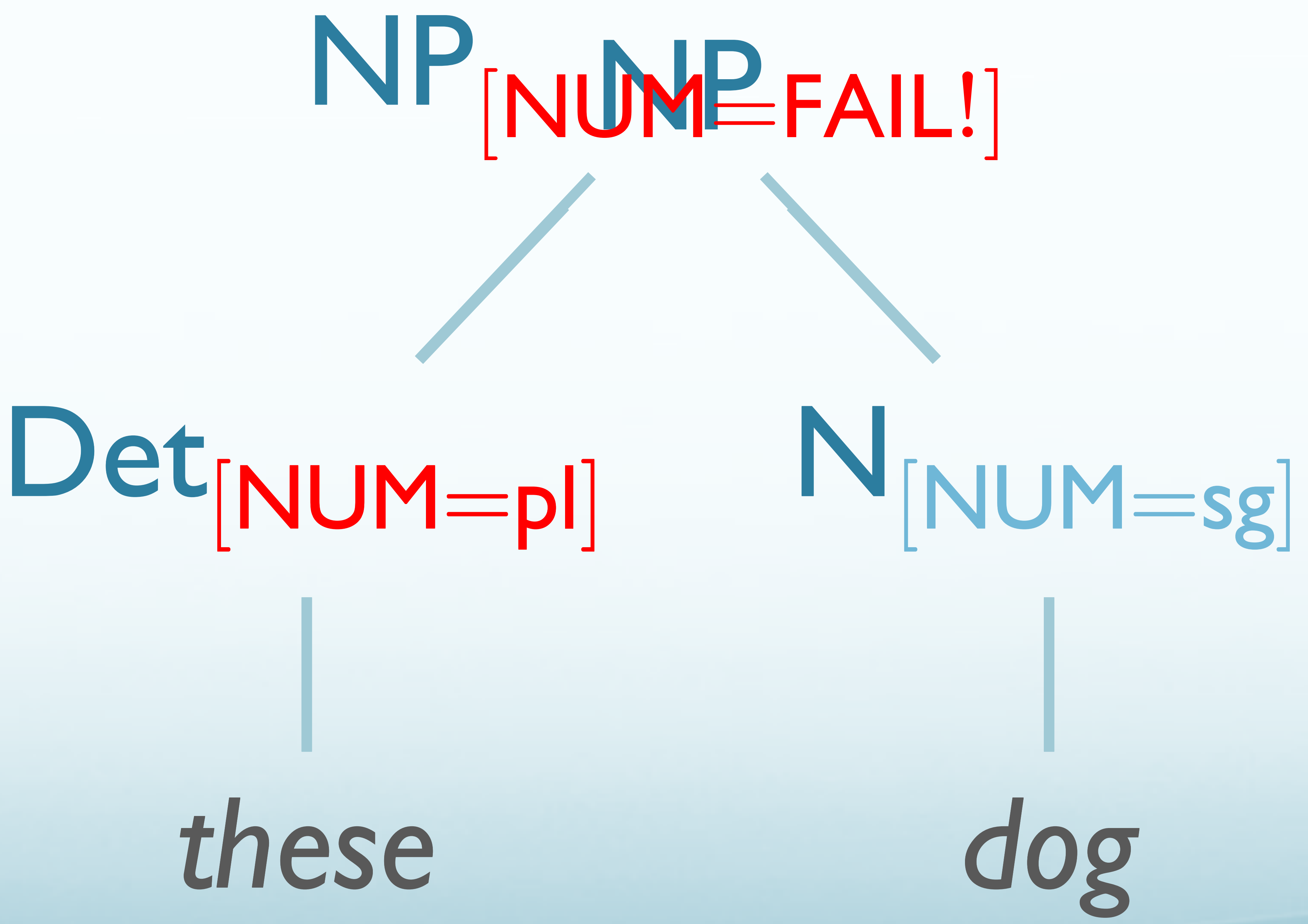
```
Det[NUM=sg] -> 'this' | 'that'
Det[NUM=pl] -> 'these' | 'those'
N[NUM=sg] -> 'dog' | 'cat'
```



HW #5: NLTK Feature Syntax

```
NP[NUM=?n] -> Det[NUM=?n] N[NUM=?n]
```

```
Det[NUM=sg] -> ‘this’ | ‘that’
Det[NUM=pl] -> ‘these’ | ‘those’
N[NUM=sg] -> ‘dog’ | ‘cat’
```



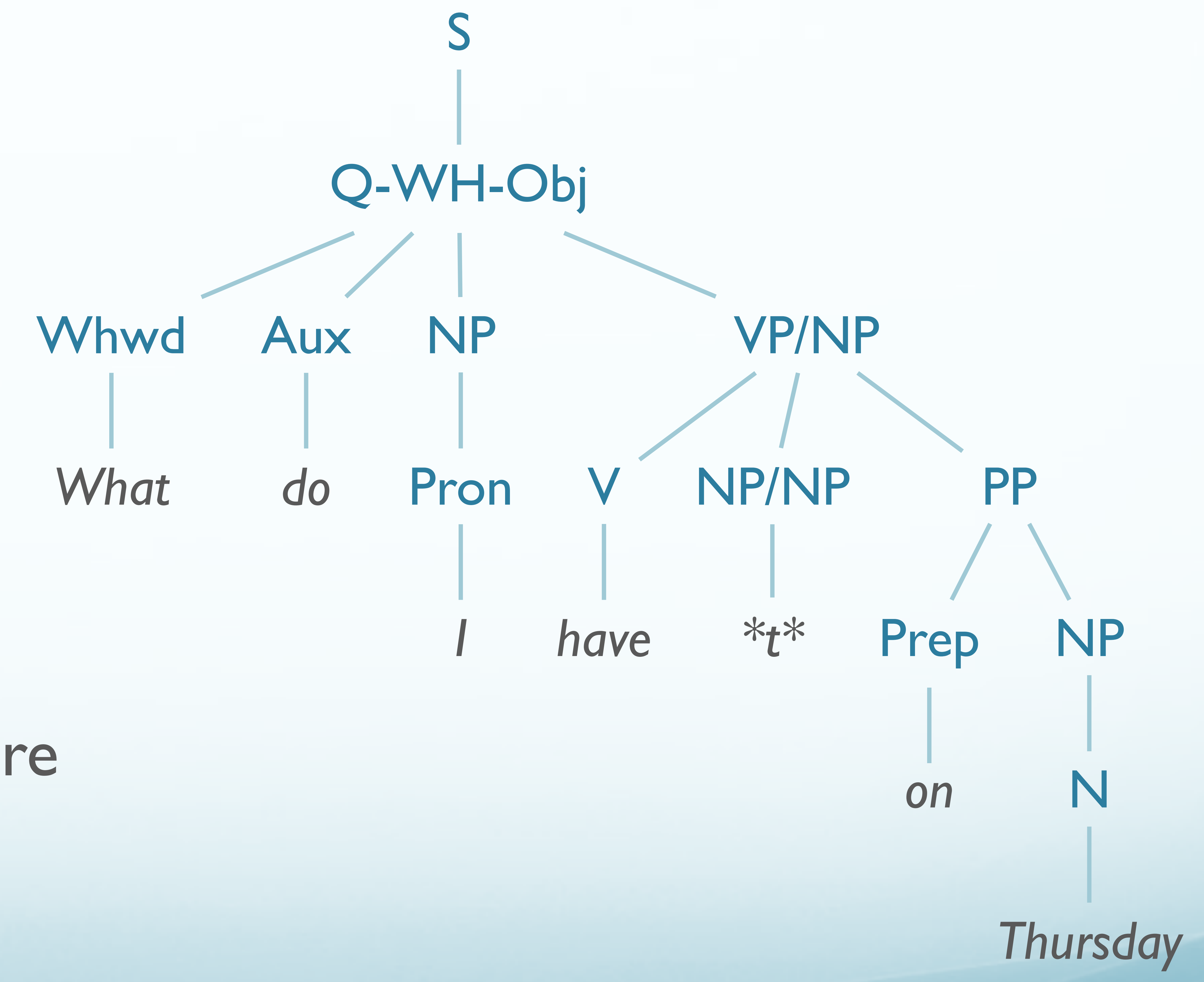
HW #5: Grammars

- It's possible to get the grammar to work with completely arbitrary rules, BUT…
- We would prefer them to be linguistically motivated!
  - instead of `[IT_OK=yes]` or `[PRON_AGR=it]`
  - `[GENDER=neut, PERSON=3rd, NUMBER=sg]`

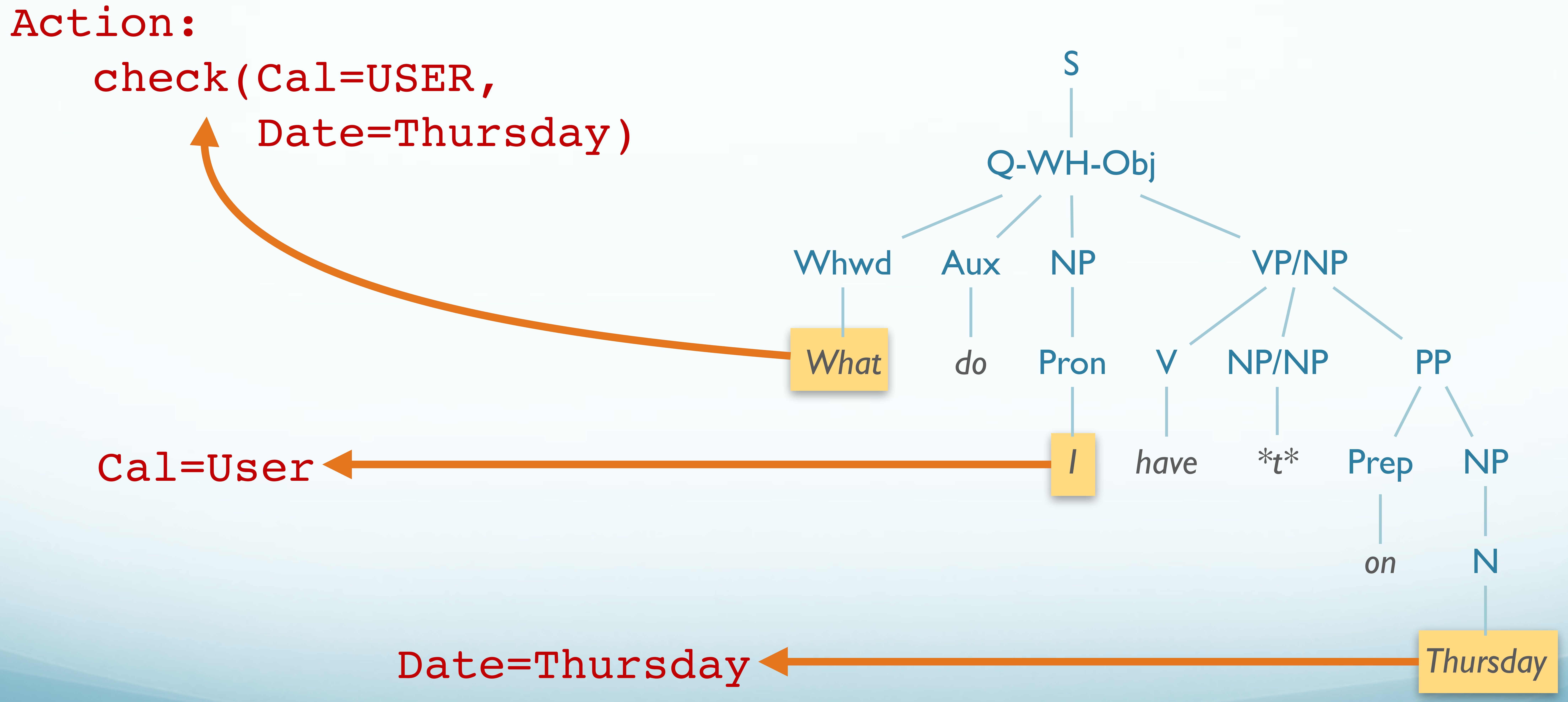
Computational Semantics

Dialogue System

- User: *What do I have on Thursday?*
- Parser:
  - Yes! It's grammatical!
  - Here's the structure!
- System:
  - Great, but what do I *DO* now?
- Need to associate meaning w/structure



Dialogue System



Syntax vs. Semantics

- Syntax:
  - Determine the ***structure*** of natural language input

- Semantics:
  - Determine the ***meaning*** of natural language input

High-Level Overview

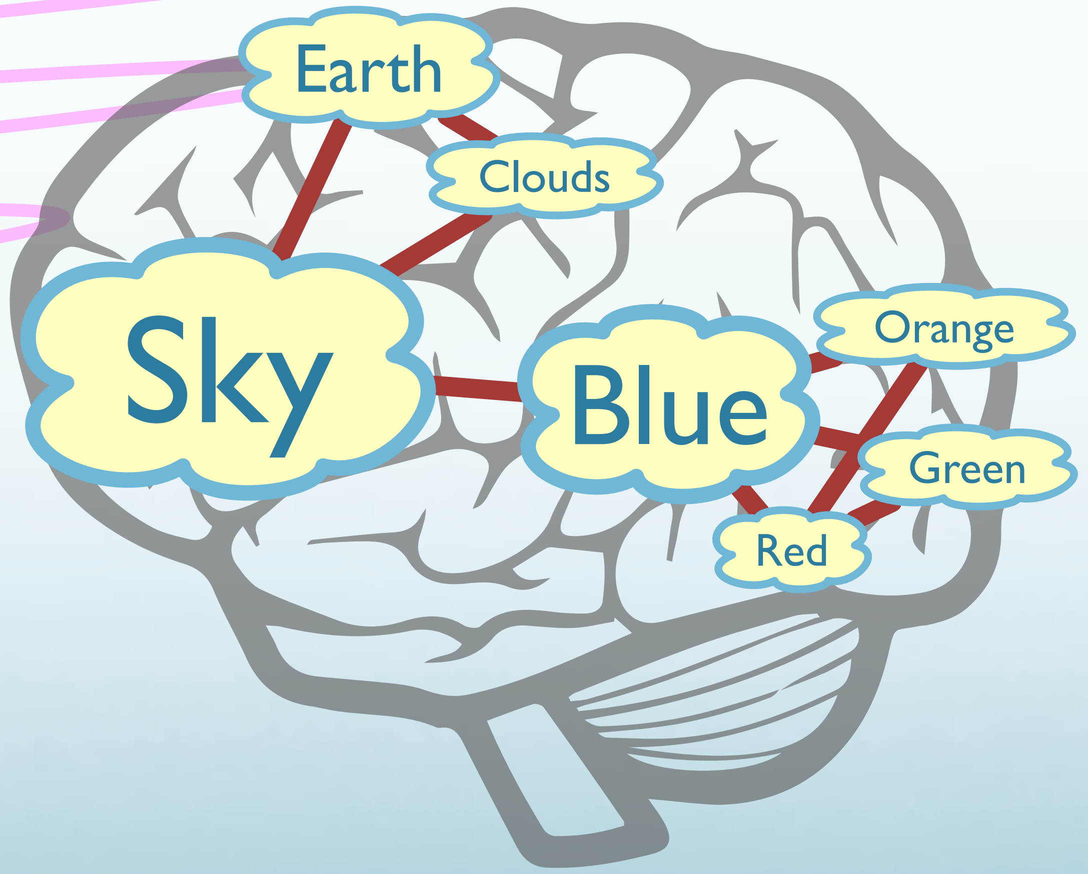
- Semantics = meaning
  - …but what does "meaning" mean?

"The sky is blue."

Speech & Text

$\exists x \; Sky(x) \wedge Blue(x)$

Logic



Psychology

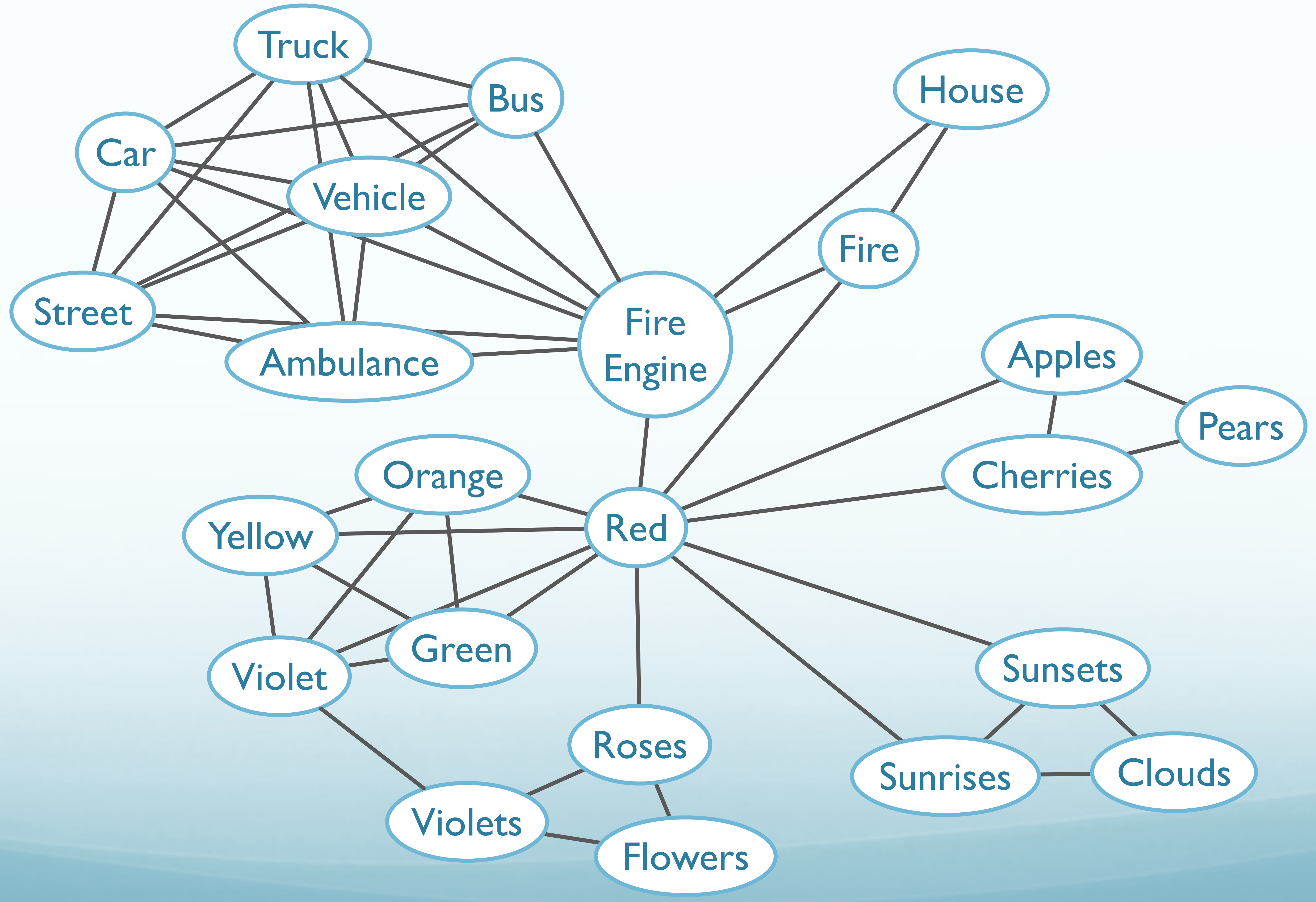
Epistemology

We Will Focus On:

- Concepts that we believe to be true about the world.
- How to connect strings and those concepts.

We ***Won't*** Focus On:

1. Building knowledge bases / semantic networks



Semantics: an Introduction

Uses for Semantics

- Semantic interpretation required for many tasks
  - Answering questions
  - Following instructions in a software manual
  - Following a recipe
- Requires more than phonology, morphology, syntax
- Must link linguistic elements to world knowledge

Semantics is Complex

- Sentences have many entailments, presuppositions
- *Instead, the protests turned bloody, as anti-government crowds were confronted by what appeared to be a coordinated group of Mubarak supporters.*
  - The protests *became* bloody.
  - The protests *had been* peaceful.
  - Crowds oppose the government.
  - Some support Mubarak.
  - There was a confrontation between two groups.
  - Anti-government crowds are not Mubarak supporters
  - …etc.

Challenges in Semantics

- **Semantic Representation**:
  - What is the appropriate formal language to express propositions in linguistic input?
  - e.g.: predicate calculus: $\exists x \left(dog(x) \wedge disappear(x) \right)$
- **Entailment**:
  - What are all the valid conclusions that can be drawn from an utterance?
    - *Lincoln was assassinated* ⊨ *Lincoln is dead*
    - ⊨ "semantically entails"

Challenges in Semantics

- **Reference**
  - How do linguistic expressions link to objects/concepts in the real world?
    - ‘the dog,’ ‘the evening star,’ ‘The Superbowl’
- **Compositionality**
  - How can we derive the meaning of a unit from its parts?
  - How do syntactic structure and semantic composition relate?
  - ‘rubber duck’ vs. ‘rubber chicken’ vs. ‘rubberneck’
  - *kick the bucket*

Tasks in Computational Semantics

- ***Extract***, ***interpret***, and ***reason*** about utterances.

- Define a **meaning representation**
- Develop techniques for **semantic analysis**
  - …convert strings from natural language to meaning representations
- Develop methods for **reasoning** about these representations
  - …and performing inference

Tasks in Computational Semantics

- Semantic similarity (words, texts)
- Semantic role labeling
- Semantic analysis
- Semantic “Parsing”
- Recognizing textual entailment
- Sentiment analysis

Complexity of Computational Semantics

- Knowledge of **language**
  - words, syntax, relationships between structure & meaning, composition procedures
- Knowledge of **the world**:
  - what are the objects that we refer to?
  - How do they relate?
  - What are their properties?
- **Reasoning**
  - Given a representation and world, what new conclusions (bits of meaning) can we infer?

Complexity of Computational Semantics

- Effectively AI-complete
  - Needs representation, reasoning, world model, etc.

Representing Meaning

Representing Meaning

First-Order Logic: $\exists e, y \left(Having(e) \wedge Haver(e, Speaker) \wedge HadThing(e, y) \wedge Car(y) \right)$

Semantic Network:

$$\begin{tikzcd}
 & Having \arrow[dl, bend right] \arrow[dr, bend left] & \\
Haver \arrow[d] & & Had\text{-}Thing \arrow[d] \\
Speaker & & Car
\end{tikzcd}$$

Conceptual Dependency:

$$\begin{array}{l} \textit{Car} \\ \Uparrow \textsc{Poss-By} \\ \textit{Speaker} \end{array}$$

Frame-Based:

```
Having
    Haver: Speaker
    HadThing: Car
```

Meaning Representations

- All consist of structures from set of symbols
  - Representational vocabulary
- Symbol structures correspond to:
  - Objects
  - Properties of objects
  - Relations among objects
- Can be viewed as:
  - Representation of meaning of linguistic input
  - Representation of state of world
- Here we focus on **literal** meaning

Representational Requirements

- Verifiability
  - Can compare representation of sentence to KB model
- Unambiguous representations
  - Semantic representation itself is unambiguous
- Canonical Form
  - Alternate expressions of same meaning map to same representation
- Inference and Variables
  - Way to draw valid conclusions from semantics and KB
- Expressiveness
  - Represent any natural language utterance

Meaning Structure of Language

- Human Languages:
  - Display basic predicate-argument structure
  - Employ variables
  - Employ quantifiers
  - Exhibit a (partially) compositional semantics

Predicate-Argument Structure

- Represent concepts and relationships
- Some words behave like predicates
  - $\textcolor{red}{\boldsymbol{Book}}(John,\ United);\ \textcolor{red}{\boldsymbol{Non\text{-}stop}}(Flight)$
- Some words behave like arguments
  - $Book(\textcolor{green}{\boldsymbol{John}},\ \textcolor{green}{\boldsymbol{United}});\ Non\text{-}stop(\textcolor{green}{\boldsymbol{Flight}})$
- Subcategorization frames indicate:
  - Number, Syntactic category, order of args

First-Order Logic

First-Order Logic

- Meaning representation:
  - Provides sound computational basis for verifiability, inference, expressiveness
- Supports determination of propositional truth
- Supports compositionality of meaning
- Supports inference
- Supports generalization through variables

First-Order Logic Terms

- **Constants**: specific objects in world;
  - $A,\ B,\ John$
  - Refer to exactly one object
  - Each object can have multiple constants refer to it
    - $WAStateGovernor$ and $JayInslee$
- **Functions**: relate ***objects*** → ***concepts***
  - $LocationOf(SFO)$
  - Refer to objects, avoid using constants
- **Variables**:
  - $x,\ e$
  - Refer to any potential object in the world

First-Order Logic Terms

- **Predicates**
  - Relate ***objects*** to other ***objects***
  - *'United serves Chicago'*
    - $Serves(United, Chicago)$
- **Logical Connectives**
  - $\{\wedge, \vee, \Rightarrow\} =$ {and, or, implies}
  - Allow for compositionality of meaning
  - *'Frontier serves Seattle and is cheap.'*
    - $Serves(Frontier, Seattle) \wedge Cheap(Frontier)$

Quantifiers

- $\exists$: **existential** quantifier: *"there exists"*
- Indefinite NP
  - $\geq$**one** such object required for truth
- **A non-stop flight** that **serves Pittsburgh**:

$\exists x\ Flight(x) \wedge Serves(x, Pittsburgh) \wedge Non\text{-}stop(x)$

Quantifiers

- $\forall$: **universal** quantifier: *"for all"*
  - **All flights include beverages.**

    $\forall \boldsymbol{x}\ Flight(\boldsymbol{x}) \Rightarrow Includes(\boldsymbol{x},\ beverages)$

FOL Syntax Summary

$$
\begin{array}{rcl}
\textbf{\textit{Formula}} & \rightarrow & \textit{AtomicFormula} \\
 & | & \textit{Formula Connective Formula} \\
 & | & \textit{Quantifier Variable, ... Formula} \\
 & | & \neg\ \textit{Formula} \\
 & | & (\textit{Formula}) \\
\textbf{\textit{AtomicFormula}} & \rightarrow & \textit{Predicate}(\textit{Term},\ldots) \\
\textbf{\textit{Term}} & \rightarrow & \textit{Function}(\textit{Term},\ldots) \\
 & | & \textit{Constant} \\
 & | & \textit{Variable}
\end{array}
\qquad
\begin{array}{rcl}
\textbf{\textit{Connective}} & \rightarrow & \land \mid \lor \mid \Rightarrow \\
\textbf{\textit{Quantifier}} & \rightarrow & \forall \mid \exists \\
\textbf{\textit{Constant}} & \rightarrow & \textit{VegetarianFood} \mid \textit{Maharani} \mid \ldots \\
\textbf{\textit{Variable}} & \rightarrow & x \mid y \mid \ldots \\
\textbf{\textit{Predicate}} & \rightarrow & \textit{Serves} \mid \textit{Near} \mid \ldots \\
\textbf{\textit{Function}} & \rightarrow & \textit{LocationOf} \mid \textit{CuisineOf} \mid \ldots
\end{array}
$$

J&M p. 556 *(Not in 3rd Ed Yet)*

Compositionality

- The meaning of a complex expression is a function of the meaning of its parts, and the rules for their combination.
- Formal languages **are** compositional.
- Natural language meaning is ***largely compositional,*** though not fully.

Compositionality

- …how can we derive:
  - $loves(John,\ Mary)$
- from:
  - $John$
  - $loves(x,\ y)$
  - $Mary$
- Lambda expressions!

Lambda Expressions

- Lambda (λ) notation (Church, 1940)
  - Just like lambda in Python, Scheme, etc
  - Allows abstraction over FOL formulae
  - Supports compositionality

- Form: (λ) + variable + FOL expression
  - $\lambda x.P(x)$ “Function taking x to $P(x)$”
  - $\lambda x.P(x)(A) = P(A)$

λ-Reduction

- λ-reduction: Apply λ-expression to logical term
  - Binds formal parameter to term

$$\lambda x.P(x)$$
$$\lambda x.P(x)(A)$$
$$P(A)$$

- Equivalent to function application

Nested λ-Reduction

- Lambda expression as body of another

$$\lambda x.\lambda y.Near(x, y)$$

$$\lambda x.\lambda y.Near(x, y)(Midway)$$

$$\lambda y.Near(Midway, y)$$

$$\lambda y.Near(Midway, y)(Chicago)$$

$$Near(Midway, Chicago)$$

Nested λ-Reduction

- If it helps, think of λs as binding sites:

Chicago

Midway

=

$$\lambda x.\lambda y.Near(x,\ y)$$

Nested λ-Reduction

- If it helps, think of λs as binding sites:

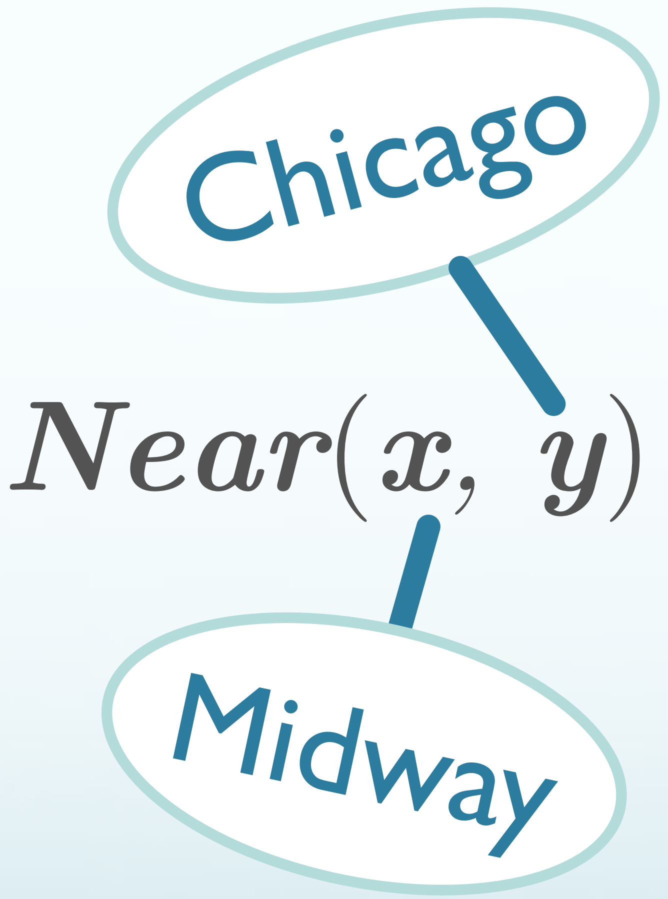
Chicago

$$\lambda y.Near(x,\ y)$$

Midway

Nested λ-Reduction

- If it helps, think of λs as binding sites:



Lambda Expressions

- ***Currying***
  - Converting multi-argument predicates to sequence of single argument predicates
  - Why?
    - Incrementally accumulates multiple arguments spread over different parts of parse tree
- …or *Schönkfinkelization*

Logical Formulae

- FOL terms (objects): denote elements in a domain
- Atomic formulae are:
  - If properties, sets of domain elements
  - If relations, sets of tuples of elements
- Formulae based on logical operators:

P	Q	$\neg P$	$P \wedge Q$	$P \vee Q$	$P \Rightarrow Q$
F	F	T	F	F	T
F	T	T	F	T	T
T	F	F	F	T	F
T	T	F	T	T	T

Logical Formulae: Finer Points

- $\vee$ is not disjunctive:
  - *Your choice is pepperoni or sausage*
  - …use $\veebar$ or $\oplus$

- $\Rightarrow$ is the logical form
  - Does not mean causality, just that if LHS=T, then RHS=T

Inference

1. α
2. $\alpha \Rightarrow \beta$
3. $\therefore \beta$

Inference

1. $VegetarianRestaurant(Leaf)$

2. $\forall x\ VegetarianRestaurant(x) \Rightarrow Serves(x, VegetarianFood)$

3. $\therefore Serves(Leaf,\ VegetarianFood)$

Inference

- Standard AI-type logical inference procedures
  - Modus Ponens
  - Forward-chaining, Backward Chaining
  - Abduction
  - Resolution
  - Etc…
- We'll assume we have a theorem prover.

Events

Representing Events

- Initially, single predicate with some arguments
  - $Serves(United, Houston)$
  - Assume # of args = # of elements in subcategorization frame
- Example:
  - *The flight arrived*
  - *The flight arrived in Seattle*
  - *The flight arrived in Seattle on Saturday.*
  - *The flight arrived on Saturday.*
  - *The flight arrived in Seattle from SFO.*
  - *The flight arrived in Seattle from SFO on Saturday.*

Representing Events

- ***Arity***:
  - How do we deal with different numbers of arguments?

- *The flight arrived in Seattle from SFO on Saturday.*
  - Davidsonian:
    - $\exists \mathbf{e}\ Arrival(\mathbf{e}, Flight, Seattle, SFO) \wedge Time(\mathbf{e}, Saturday)$
  - Neo-Davidsonian:
    - $\exists \mathbf{e}\ Arrival(\mathbf{e}) \wedge Arrived(\mathbf{e}, Flight) \wedge Destination(\mathbf{e}, Seattle) \wedge Origin(\mathbf{e}, \mathrm{SFO}) \wedge Time(\mathbf{e}, Saturday)$

Neo-Davidsonian Events

- Neo-Davidsonian representation:
  - Distill event to single argument for event itself
  - Everything else is additional predication
- Pros
  - No fixed argument structure
  - Dynamically add predicates as necessary
  - No unused roles
  - Logical connections can be derived

Meaning Representation for Computational Semantics

- Requirements
  - Verifiability
  - Unambiguous representation
  - Canonical Form
  - Inference
  - Variables
  - Expressiveness
- Solution:
  - First-Order Logic
    - Structure
    - Semantics
    - Event Representation

Summary

- FOL can be used as a meaning representation language for natural language
- Principle of compositionality:
  - The meaning of a complex expression is a function of the meaning of its parts
- λ-expressions can be used to compute meaning representations from syntactic trees based on the principle of compositionality
- In next classes, we will look at syntax-driven approach to semantic analysis in more detail

Feature Grammar Practice: Animacy

Feature Grammar Practice

- **Initial Grammar:**

```
S -> NP VP
VP[subcat=ditrans] -> V NP NP
NP -> NNP
NP -> Det N
NNP[animacy=True] -> 'Alex' | 'Ahmed'
V -> 'gifted'
Det -> 'a' | 'the'
N[animacy=False] -> 'book' | 'rock'
```

Feature Grammar Practice