Question Answering with Semantic Parsing

Katharina Kann — CSCI/LING5832

Constituency Grammars

Syntax

- By syntax, we mean the kind of knowledge of your native language **not directly related to meaning** that you had mastered by the time you were 3 years old without any explicit instruction.

Syntax

- By syntax, we mean the kind of knowledge of your native language **not directly related to meaning** that you had mastered by the time you were 3 years old without any explicit instruction.
- The set of rules, principles, and processes that govern the structure of sentences in a language.

Constituents and Words

- Note that each word is a constituent

Constituents and Words

- Note that each word is a constituent
- Words can also act as phrases:
  - *Joe grew potatoes*
  - *Joe* and *potatoes* are both noun phrases

Constituents and Words

- Note that each word is a constituent
- Words can also act as phrases:
  - *Joe grew potatoes*
  - *Joe* and *potatoes* are both noun phrases
- Compare with:
  - *The man from Amherst grew large potatoes*

Some Constituency Tests

1. They appear in similar environments (before a verb)
   Kermit the frog comes on stage
   They come to Massachusetts every summer
   December twenty-sixth comes after Christmas
   The reason he is running for president comes out only now.
   But not each individual word in the consituent
   **The comes our... *is comes out... *for comes out...*

Some Constituency Tests

1. They appear in similar environments (before a verb)
   Kermit the frog comes on stage
   They come to Massachusetts every summer
   December twenty-sixth comes after Christmas
   The reason he is running for president comes out only now.
   But not each individual word in the consituent
   **The comes our...* **is comes out...* **for comes out...*

2. The constituent can be placed in a number of different locations
   Consituent = Prepositional phrase: *On December twenty-sixth*
   On December twenty-sixth I'd like to fly to Florida.
   I'd like to fly on December twenty-sixth to Florida.
   I'd like to fly to Florida on December twenty-sixth.
   But not split apart
   **On December I'd like to fly twenty-sixth to Florida.*
   **On I'd like to fly December twenty-sixth to Florida.*

Common Constituents

The Noun Phrase

Harry *the Horse*

a high-class ***spot*** *such as Mindy's*

the Broadway ***coppers***

the ***reason*** *he comes into the Hot Box*

they

three ***parties*** *from Brooklyn*

The Noun Phrase

Harry *[the <u>Horse</u>]*

*[a high-class **<u>spot</u>**] such as [<u>Mindy's</u>]*

*the Broadway **coppers***

*[the **<u>reason</u>**] he comes into [the Hot <u>Box</u>]*

they

*[three **<u>parties</u>**] from Brooklyn*

- Noun phrases often contain other noun phrases!

The Verb Phrase

prefer *a morning flight.*

leave *Boston in the morning.*

gave *Delta $200.*

flew*.*

would not ***eat*** *Jell-O.*

The Verb Phrase

prefer *[a morning flight].*

leave *[Boston] in [the morning].*

gave *[Delta] [$200].*

flew*.*

would not ***eat*** *[Jell-O].*

- *VPs often contain NPs!*

The Prepositional Phrase

to *Seattle*

on *these flights*

in *Minneapolis*

about *the ground transportation in Chicago*

of *the round trip flight on United Airlines*

The Prepositional Phrase

to *[Seattle]*

on *[these flights]*

in *[Minneapolis]*

about *[[the ground transportation] [in [Chicago]]]*

of *[[the round trip flight] [on [United Airlines]]]*

- *PPs often contain NPs, and can contain lots more!*

Constituency is Recursive

I pet [the cat].

I pet [[the cat] [who bit [the dog]]].

I pet [[the cat] [who bit [[the dog] [with the black spot]]]].

Attachment Ambiguity

- A particular constituent can be attached to the parse tree at more than one place

Attachment Ambiguity

- A particular constituent can be attached to the parse tree at more than one place
  - *Scientists study whales from space*.

Attachment Ambiguity

- A particular constituent can be attached to the parse tree at more than one place
  - *Scientists study whales from space*.
  - *I shot an elephant in my pajamas*.

Attachment Ambiguity

- A particular constituent can be attached to the parse tree at more than one place
    - *Scientists study whales from space*.
    - *I shot an elephant in my pajamas*.
        - *How he got into my pajamas I don't know*. (Groucho Marx, Animal Crackers, 1930)

Coordination Ambiguity

- Different sets of phrases can be conjoined by a conjunction like *and*

Coordination Ambiguity

- Different sets of phrases can be conjoined by a conjunction like *and*
    - *Old men and women*

Coordination Ambiguity

- Different sets of phrases can be conjoined by a conjunction like *and*
  - *Old men and women*
    - *Old* [*men and women*]

Coordination Ambiguity

- Different sets of phrases can be conjoined by a conjunction like *and*
  - *Old men and women*
    - *Old* [*men and women*]
    - [*Old men*] *and women*

Statistical Parsing

Chomsky Normal Form (CNF)

- Restricted to two types of rules:
  - A → B C
  - A → w

Chomsky Normal Form (CNF)

- Restricted to two types of rules:
    - A → B C
    - A → w
- Any context-free grammar can be converted into a corresponding CNF grammar that accepts exactly the same set of strings

Conversion to Chomsky Normal Form (CNF)

- Three things that need to be changed:

Conversion to Chomsky Normal Form (CNF)

- Three things that need to be changed:
  - Rules that mix terminals with non-terminals on the right-hand side

Conversion to Chomsky Normal Form (CNF)

- Three things that need to be changed:
  - Rules that mix terminals with non-terminals on the right-hand side
  - Rules that have a single non-terminal on the right-hand side

Conversion to Chomsky Normal Form (CNF)

- Three things that need to be changed:
  - Rules that mix terminals with non-terminals on the right-hand side
  - Rules that have a single non-terminal on the right-hand side
  - Rules in which the length of the right-hand side is greater than 2

Conversion to Chomsky Normal Form (CNF)

1. Copy all conforming rules to the new grammar unchanged.
2. Convert terminals within rules to dummy non-terminals.
3. Convert unit productions.
4. Make all rules binary and add them to new grammar.

In-Class Exercise

- Convert the following grammar to Chomsky normal form:
  - S → NP VP
  - S → VP
  - VP → V
  - V → *walk* | *run*
  - NP → *the* N
  - N → *dogs* | *cats* | *lions*

Cocke-Kasami-Younger (CKY) Algorithm for Recognition

- Answers the question: can this sentence be generated by the grammar?

J&M, Ch.13

Cocke-Kasami-Younger (CKY) Algorithm for Recognition

- Answers the question: can this sentence be generated by the grammar?

function CKY-PARSE(*words, grammar*) **returns** *table*

for $j \leftarrow$ **from** 1 **to** LENGTH(*words*) **do**
  for all $\{A \mid A \rightarrow words[j] \in grammar\}$
    $table[j-1, j] \leftarrow table[j-1, j] \cup A$
  for $i \leftarrow$ **from** $j-2$ **downto** 0 **do**
    for $k \leftarrow i+1$ **to** $j-1$ **do**
      for all $\{A \mid A \rightarrow BC \in grammar$ **and** $B \in table[i,k]$ **and** $C \in table[k,j]\}$
        $table[i,j] \leftarrow table[i,j] \cup A$

Figure 13.5 The CKY algorithm.

J&M, Ch.13

Cocke-Kasami-Younger (CKY) Recognition

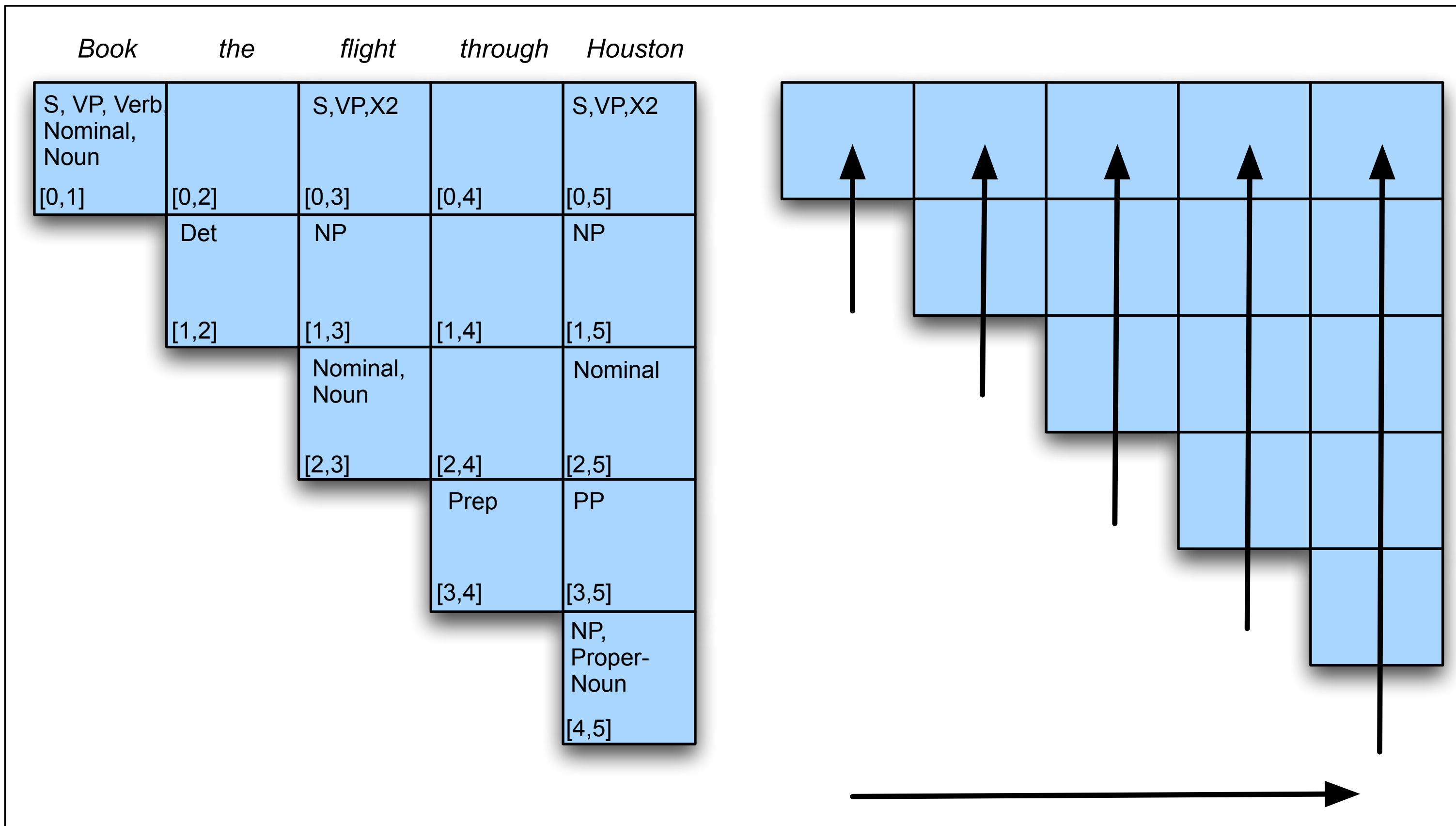


Figure 13.4 Completed parse table for *Book the flight through Houston.*

J&M, Ch.13

Cocke-Kasami-Younger (CKY) Parsing

- Can return the most likely parse tree for a sentence, together with its probability
  - Small changes from the recognizer necessary
- Can be implemented with dynamic programming

Constituency Parsing with Neural Networks

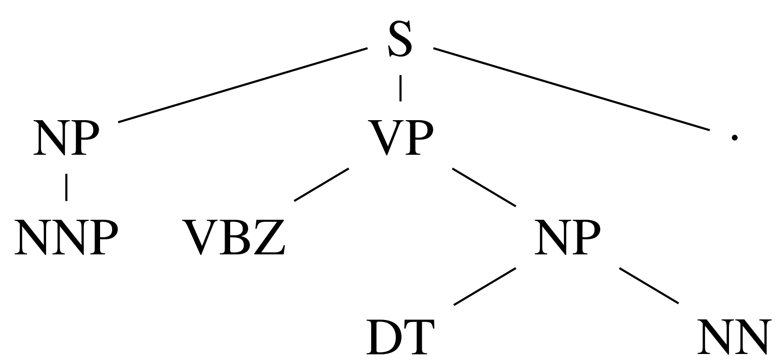
Grammar as a Foreign Language

- Vinyals et al. (2015):
  - Treat parsing as a sequence-to-sequence task
  - Input is the original sentence
  - Output is the parse tree in bracketed notation

Grammar as a Foreign Language

- Vinyals et al. (2015):

John has a dog . $\rightarrow$



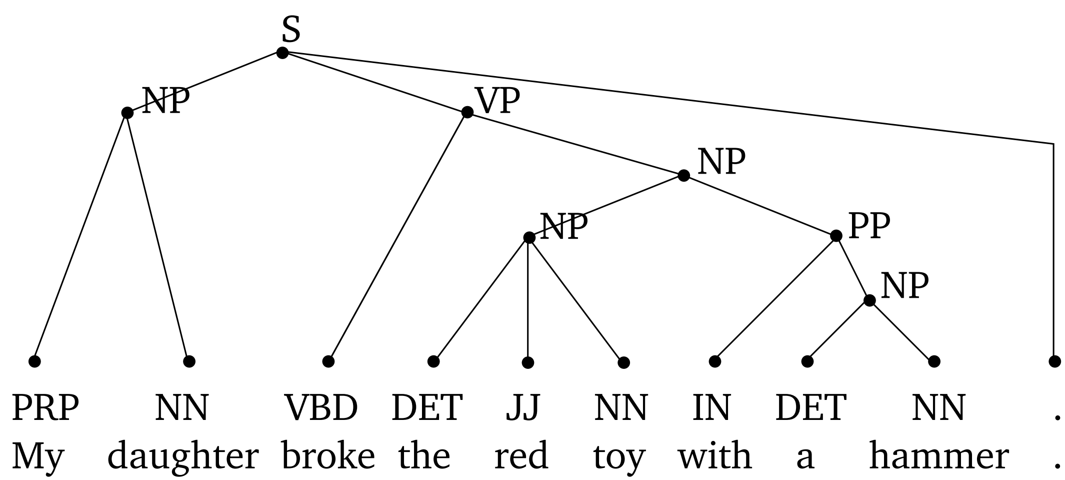
John has a dog . $\rightarrow$ (S (NP NNP $)_{\mathrm{NP}}$ (VP VBZ (NP DT NN $)_{\mathrm{NP}}$ $)_{\mathrm{VP}}$. $)_{\mathrm{S}}$

Constituent Parsing as Sequence Labeling

- Gomez-Rodriguez and Vilares (2018):
    - Reduce constituent parsing to sequence labeling
    - For each word w_t, generate a label that encodes:
        - The number of ancestors in the tree that the words w_t and w_{t+1} have in common
        - The non-terminal symbol at the lowest common ancestor

Constituent Parsing as Sequence Labeling

- Gomez-Rodriguez and Vilares (2018):



Linearized tree (absolute scale):

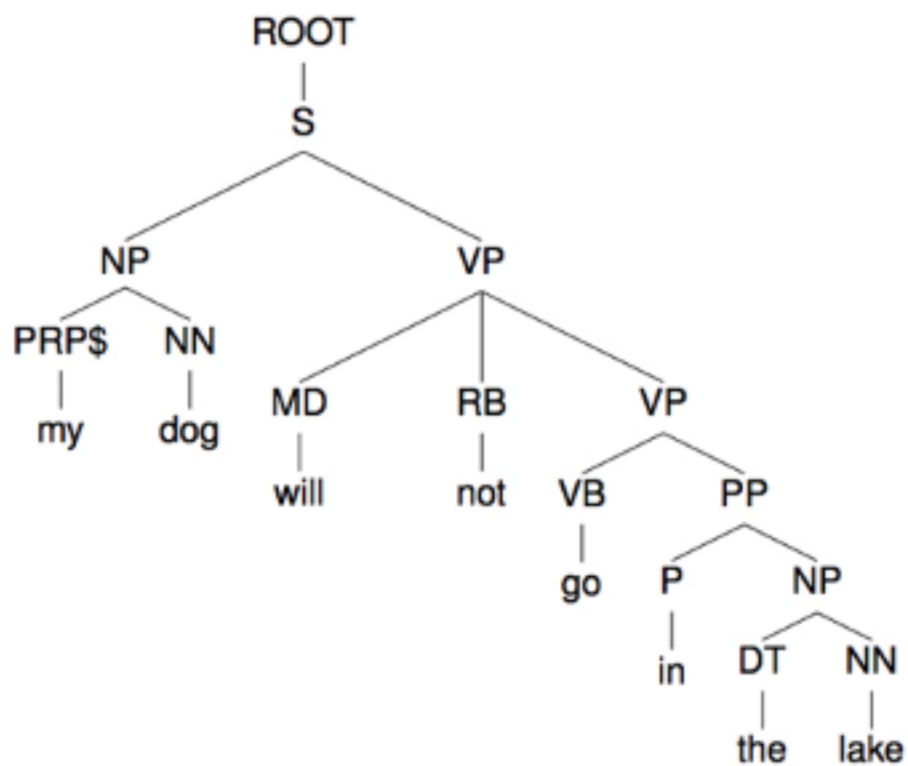
2NP 1S 2VP 4NP 4NP 3NP 4PP 5NP 1S

Linearized tree (relative scale):

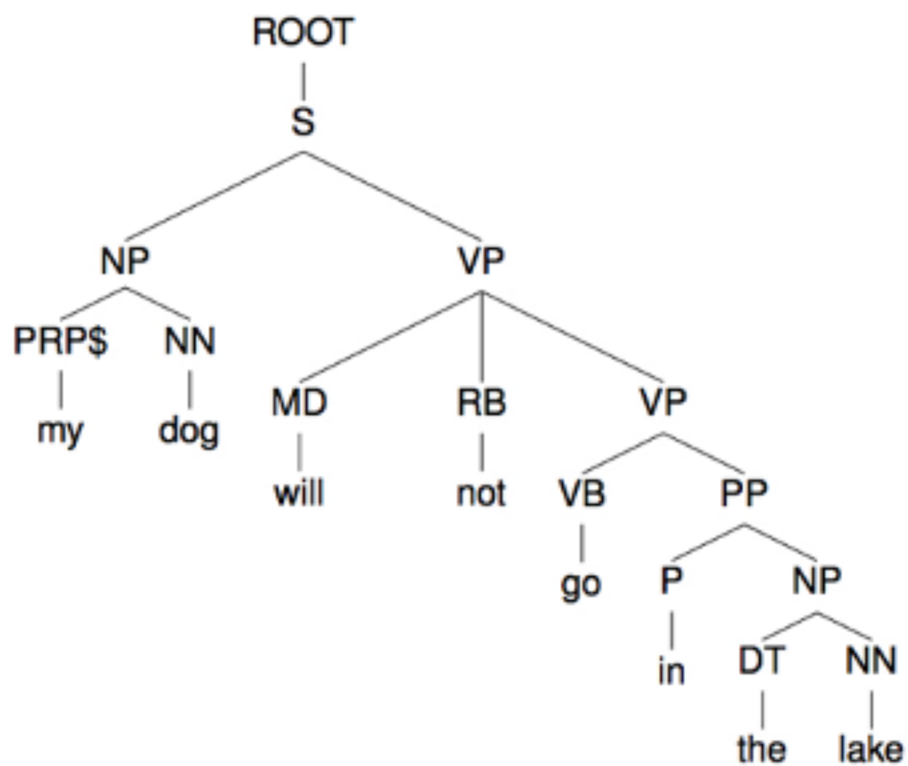
2NP -1S 1VP 2NP 0NP -1NP 1PP 1NP -4S

Dependency Parsing

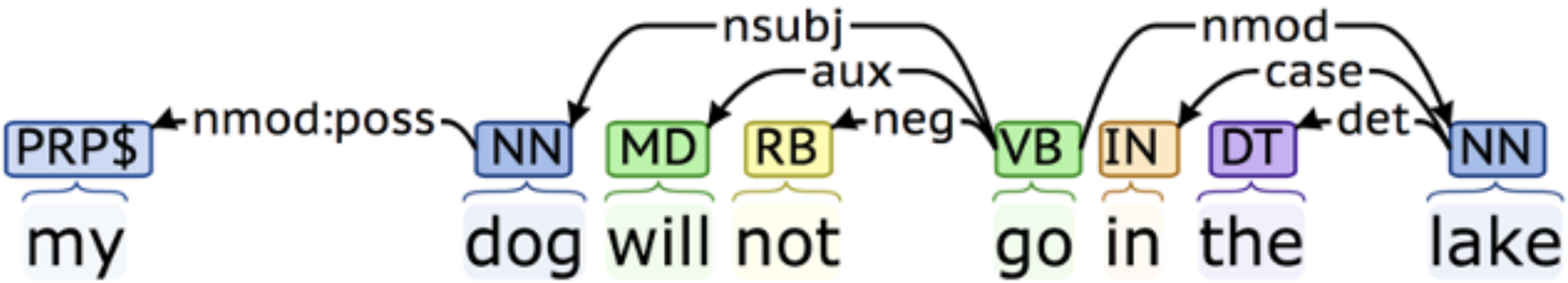
What's the Main Verb of this Sentence?



What’s the Subject of the Main Verb of this Sentence?



How About Now?



Dependency Trees

- ...highlight relationships between words.

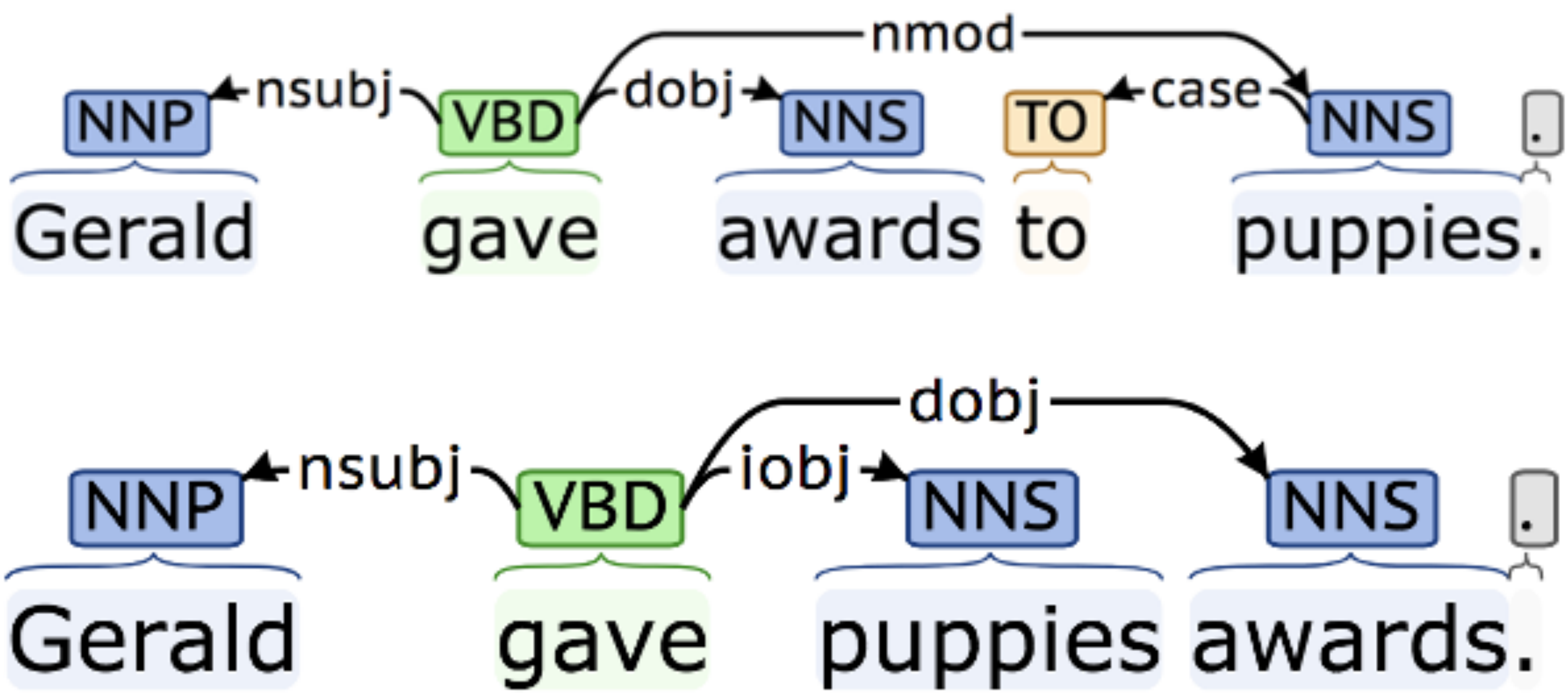
Dependency Trees

- ...highlight relationships between words.
- ...can be usually derived deterministically from constituency trees.

Dependency Trees

- ...highlight relationships between words.
- ...can be usually derived deterministically from constituency trees.
- ...often highlight the similarities between superficially different sentences

Dependency Trees



Dependency vs. Constituency

- Ability to deal with languages that have a relatively free word order.
  - A phrase-structure grammar needs a separate rule for each possible place in the parse tree where a phrase could occur
  - A dependency-based approach just has one link type representing this particular relation

Universal Dependencies

- acl: clausal modifier of noun (adjectival clause)
- advcl: adverbial clause modifier
- advmod: adverbial modifier
- amod: adjectival modifier
- appos: appositional modifier
- aux: auxiliary
- case: case marking
- cc: coordinating conjunction
- ccomp: clausal complement
- clf: classifier
- compound: compound
- conj: conjunct
- cop: copula
- csubj: clausal subject
- dep: unspecified dependency
- det: determiner
- discourse: discourse element
- dislocated: dislocated elements
- expl: expletive
- fixed: fixed multiword expression
- flat: flat multiword expression
- goeswith: goes with
- iobj: indirect object
- list: list
- mark: marker
- nmod: nominal modifier
- nsubj: nominal subject
- nummod: numeric modifier
- obj: object
- obl: oblique nominal
- orphan: orphan
- parataxis: parataxis
- punct: punctuation
- reparandum: overridden disfluency
- root: root
- vocative: vocative
- xcomp: open clausal complement

Universal Dependencies

Afrikaans	1	49K		IE, Germanic
Akkadian	1	1K		Afro-Asiatic, Semitic
Amharic	1	10K		Afro-Asiatic, Semitic
Ancient Greek	2	417K		IE, Greek
Arabic	3	1,042K	W	Afro-Asiatic, Semitic
Armenian	1	22K		IE, Armenian
Bambara	1	13K		Mande
Basque	1	121K		Basque
Belarusian	1	8K		IE, Slavic
Breton	1	10K	W	IE, Celtic
Bulgarian	1	156K		IE, Slavic
Buryat	1	10K		Mongolic
Cantonese	1	6K		Sino-Tibetan
Catalan	1	531K		IE, Romance
Chinese	4	160K	W	Sino-Tibetan
Classical Chinese	1	34K		Sino-Tibetan
Coptic	1	22K		Afro-Asiatic, Egyptian
Croatian	1	197K	W	IE, Slavic
Czech	5	2,222K	W	IE, Slavic
Danish	2	100K		IE, Germanic
Dutch	2	307K	W	IE, Germanic
English	6	586K	W	IE, Germanic
Erzya	1	15K		Uralic, Mordvin
Estonian	1	434K		Uralic, Finnic

~ 70 languages!

Projectivity

- *Projective* dependency trees have no crossing dependency arcs.
- Projectivity makes parsing easier (most parsers assume projectivity), but it limits how we can handle some things:

Projectivity

- *Projective* dependency trees have no crossing dependency arcs.
- Projectivity makes parsing easier (most parsers assume projectivity), but it limits how we can handle some things:

- The same concept applies in constituency parsing, too: Projective parse trees have no crossing branches.

What to Use

- Popular/effective Python tools:
  - StanfordNLP for UD
  - Kitaev Parser for PTB
  - spaCy for non-UD dependencies (especially fast/easy)

What to Use

- Popular/effective Python tools:
  - StanfordNLP for UD
  - Kitaev Parser for PTB
  - spaCy for non-UD dependencies (especially fast/easy)
- Performance in the 90s on most intuitive measures of accuracy, for clean English data. Usable performance on many more languages/ settings.

Question Answering with Semantic Parsing

The Big Question

What kind of thing is the meaning of a sentence?

The Big Question

~~What kind of thing is the meaning of a sentence?~~

What can you do with a sentence if you know its meaning?

The Big Question

~~What kind of thing is the meaning of a sentence?~~

What can you do with a sentence if you know its meaning?

If it’s a question, answer it!

Early Methods: SHRDLU (Winograd 1972)

- A natural language understanding computer program, developed by Terry Winograd at MIT
- User can have a conversation with the computer, move objects, name collections and query the state of a simplified blocks world

Early Methods: SHRDLU (Winograd 1972)

Find a block which is taller than the one you are holding and put it into the box.

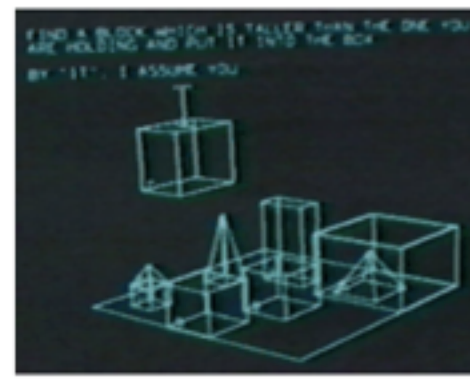
OK.

How many blocks are not in the box?

FOUR OF THEM.

Is at least one of them narrower than the one which I told you to pick up?

YES, THE RED CUBE.



Early Methods: CHAT-80

- Developed 1979-1982 by Fernando Pereira and David Warren
- Could answer questions about geography
- Hand-built lexicon and grammar
- Implemented in prolog

Early Methods: CHAT-80

- Is there more than one country in each continent?

Early Methods: CHAT-80

- Is there more than one country in each continent?
- What countries border Denmark?

Early Methods: CHAT-80

- Is there more than one country in each continent?
- What countries border Denmark?
- What are the countries from which a river flows into the Black_Sea?

Early Methods: CHAT-80

- Is there more than one country in each continent?
- What countries border Denmark?
- What are the countries from which a river flows into the Black_Sea?
- What is the total area of countries south of the Equator and not in Australasia?

Early Methods: CHAT-80

- Is there more than one country in each continent?
- What countries border Denmark?
- What are the countries from which a river flows into the Black_Sea?
- What is the total area of countries south of the Equator and not in Australasia?
- Which country bordering the Mediterranean borders a country that is bordered by a country whose population exceeds the population of India?

Early Methods: CHAT-80

- Is there more than one country in each continent?
- What countries border Denmark?
- What are the countries from which a river flows into the Black_Sea?
- What is the total area of countries south of the Equator and not in Australasia?
- Which country bordering the Mediterranean borders a country that is bordered by a country whose population exceeds the population of India?
- How far is London from Paris?

Early Methods: CHAT-80

```
% Facts about countries.
% country(Country, Region, Latitude, Longitude,
%   Area(sqmiles), Population, Capital, Currency)
country(andorra, southern_europe, 42, -1, 179, 25000,
andorra_la_villa, franc_peseta).
country(angola, southern_africa, -12, -18, 481351, 5810000, luanda,
?).
country(argentina, south_america, -35, 66, 1072067, 23920000,
buenos_aires, peso).

capital(C,Cap) :- country(C,_,_,_,_,Cap,_).
```

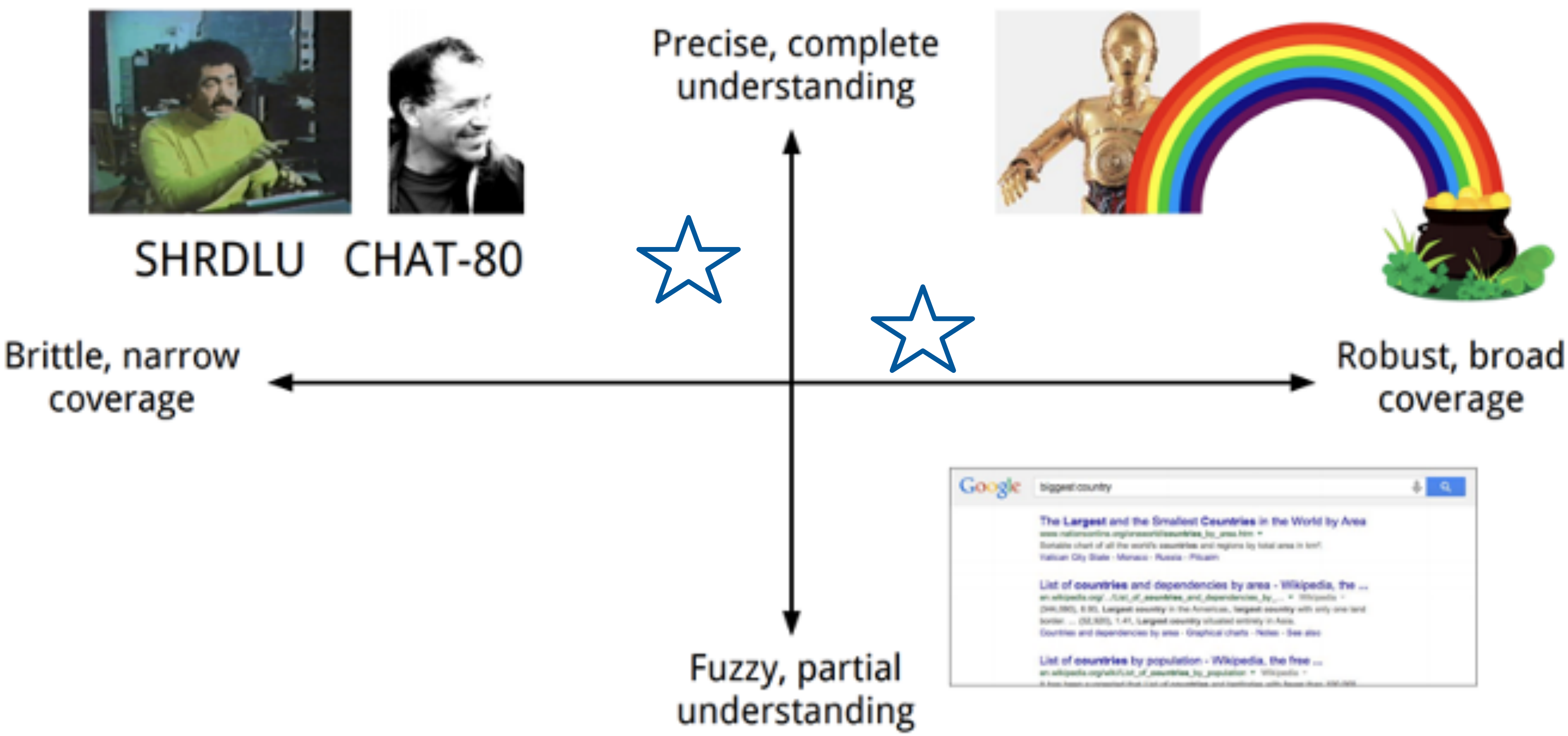
Early Methods: CHAT-80

```
/* Sentences */
sentence(S) --> declarative(S), terminator(.) .
sentence(S) --> wh_question(S), terminator(?) .
sentence(S) --> yn_question(S), terminator(?) .
sentence(S) --> imperative(S), terminator(!) .

/* Noun Phrase */
np(np(Agmt,Pronoun,[]),Agmt,NPCase,def,_,Set,Nil) -->
  {is_pp(Set)},
  pers_pron(Pronoun,Agmt,Case),
  {empty(Nil), role(Case,decl,NPCase)}.

/* Prepositional Phrase */
pp(pp(Prep,Arg),Case,Set,Mask) -->
  prep(Prep),
  {prep_case(NPCase)},
  np(Arg,_,NPCase,_,Case,Set,Mask).
```

Question Answering: The Landscape



Semantic Parsing

Example Questions

Which country has the highest CO2 emissions?
What about highest per capita?
Which had the biggest increase over the last five years?
What fraction was from European countries?

Meaning Representations

- If we want to understand natural language completely and precisely, we need to do **semantic parsing**
  - Convert natural language into a **formal meaning representation** a machine can understand

Meaning Representations

- If we want to understand natural language completely and precisely, we need to do **semantic parsing**
  - Convert natural language into a **formal meaning representation** a machine can understand
- We need to define our goal
  - How should this representation look like?

Meaning Representations

- To facilitate data exploration and analysis, you might want to parse natural language into database queries

Meaning Representations

- To facilitate data exploration and analysis, you might want to parse natural language into database queries

which country had the highest carbon emissions last year

```
SELECT    country.name
FROM      country, co2_emissions
WHERE     country.id = co2_emissions.country_id
AND       co2_emissions.year = 2014
ORDER BY  co2_emissions.volume DESC
LIMIT     1;
```

Meaning Representations

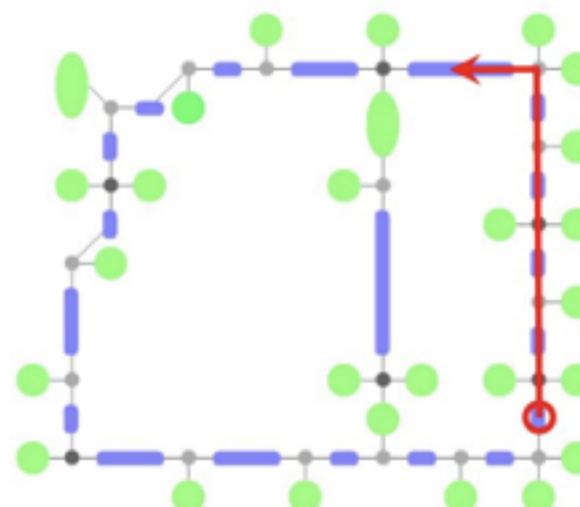
- For a robot control application, you might want a custom-designed procedural language

Meaning Representations

- For a robot control application, you might want a custom-designed procedural language

Go to the third junction and take a left.

```
(do-sequentially
  (do-n-times 3
    (do-sequentially
      (move-to forward-loc)
      (do-until
        (junction current-loc)
        (move-to forward-loc))))
  (turn-left))
```



Meaning Representations

- For smartphone voice commands, you might want relatively simple meaning representations, with *intents* and *arguments*

directions to SF by train

```
(TravelQuery
  (Destination /m/0d6lp)
  (Mode TRANSIT))
```

angelina jolie net worth

```
(FactoidQuery
  (Entity /m/0f4vbz)
  (Attribute /person/net_worth))
```

weather friday austin tx

```
(WeatherQuery
  (Location /m/0vzm)
  (Date 2013-12-13))
```

text my wife on my way

```
(SendMessage
  (Recipient 0x31cbf492)
  (MessageType SMS)
  (Subject "on my way"))
```

play sunny by boney m

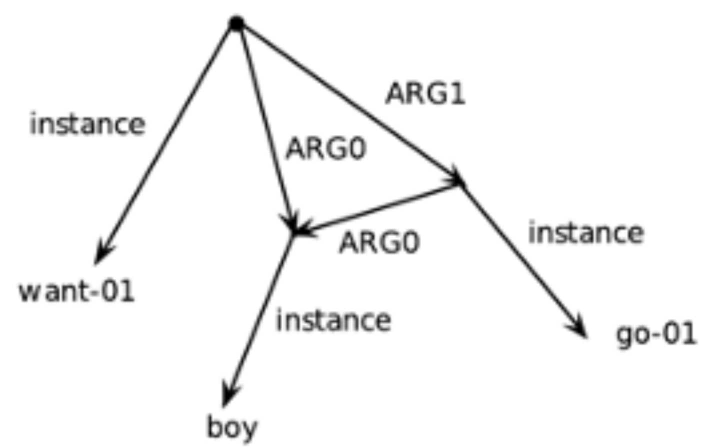
```
(PlayMedia
  (MediaType MUSIC)
  (SongTitle "sunny")
  (MusicArtist /m/017mh))
```

is REI open on sunday

```
(LocalQuery
  (QueryType OPENING_HOURS)
  (Location /m/02nx4d)
  (Date 2013-12-15))
```

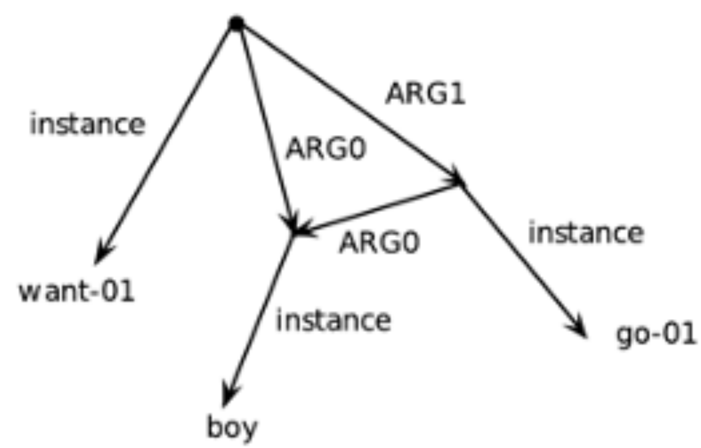
Meaning Representations

- Ongoing efforts to create fine-grained, general-purpose symbolic meaning representations for NLP: AMR (below) and DRT feature reasonably complete semantic parsers:



Meaning Representations

- Ongoing efforts to create fine-grained, general-purpose symbolic meaning representations for NLP: AMR (below) and DRT feature reasonably complete semantic parsers:



- For most typical question answering applications, though, the information in these formalisms is neither necessary nor sufficient to issue a query to your knowledge source

Semantic Parsing and Machine Translation

- Both involve translating from one semantic representation into another

Semantic Parsing and Machine Translation

- Both involve translating from one semantic representation into another
- Both involve complex structures, often related in complex ways

Semantic Parsing and Machine Translation

- Both involve translating from one semantic representation into another
- Both involve complex structures, often related in complex ways
- Some techniques are transferable: word sense disambiguation, alignment, …

Semantic Parsing and Machine Translation

- Both involve translating from one semantic representation into another
- Both involve complex structures, often related in complex ways
- Some techniques are transferable: word sense disambiguation, alignment, …
- But: in semantic parsing, the output is not human-readable, but machine-readable!

Today’s Approach

- Define possible syntactic structures using a context-free grammar

Today's Approach

- Define possible syntactic structures using a context-free grammar
- Construct semantics bottom-up, following syntactic structure

Today’s Approach

- Define possible syntactic structures using a context-free grammar
- Construct semantics bottom-up, following syntactic structure
- Score parses with a machine learning model (learned from training data)

Today's Approach

- Define possible syntactic structures using a context-free grammar
- Construct semantics bottom-up, following syntactic structure
- Score parses with a machine learning model (learned from training data)
- Leverage annotators for names, numbers, places, etc.

Today's Approach

- Define possible syntactic structures using a context-free grammar
- Construct semantics bottom-up, following syntactic structure
- Score parses with a machine learning model (learned from training data)
- Leverage annotators for names, numbers, places, etc.
- Use grammar induction to avoid manual grammar annotation

Syntax

- The syntactic part of the grammar is a fairly conventional CFG:

```
$Loc → Google
$Loc → NY
$Loc → $Loc in $Loc
$Opt → me
$Mode → bike
$Mode → car
$ROOT → route ($Opt)? to $Loc by $Mode
```

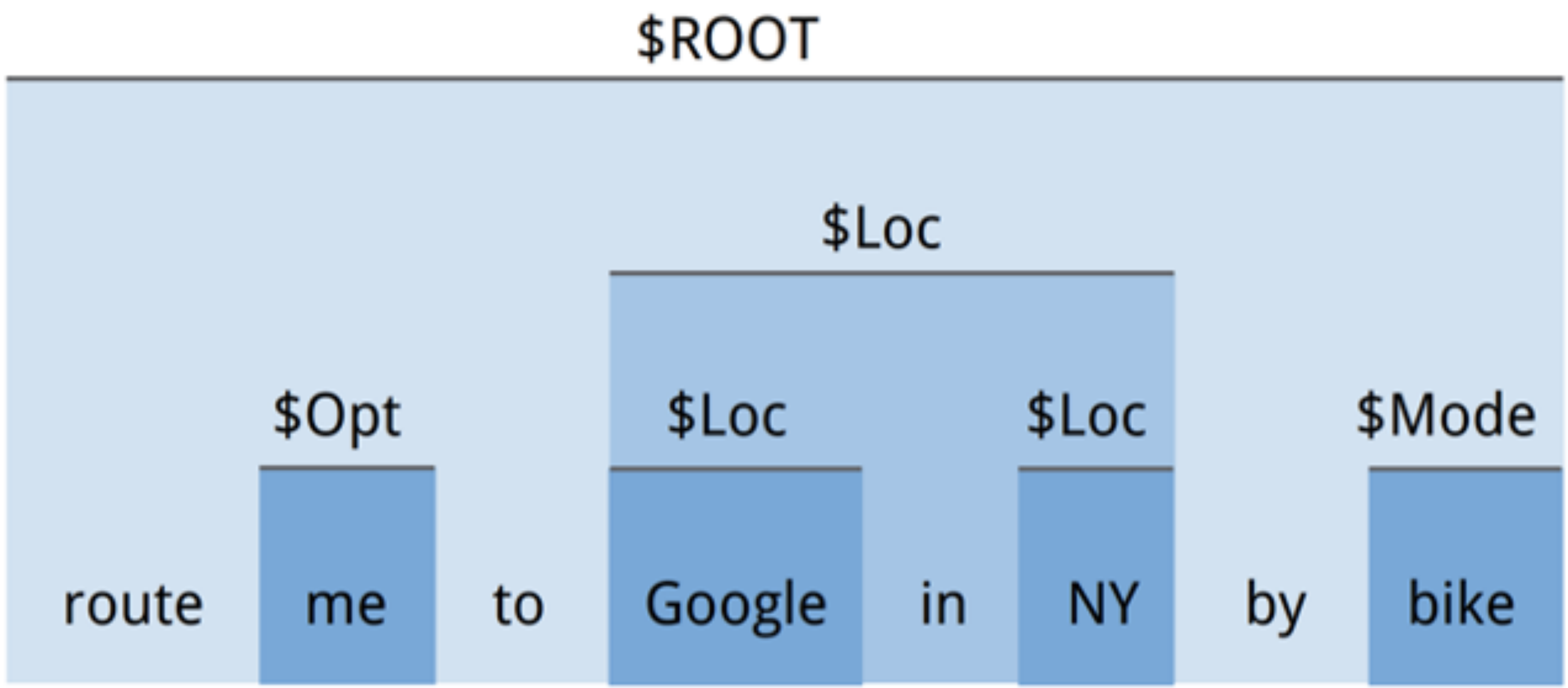
Syntax

- The syntactic part of the grammar is a fairly conventional CFG:

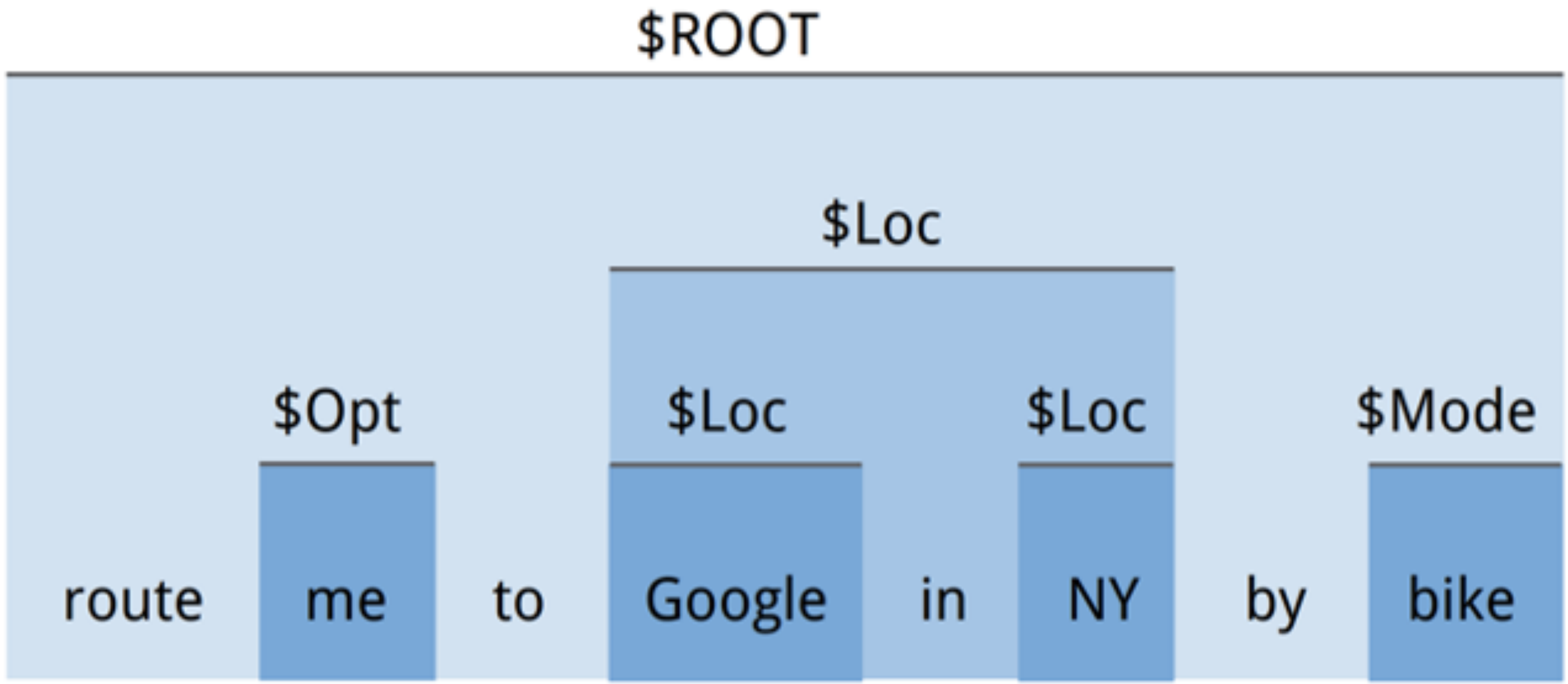
```
$Loc → Google
$Loc → NY
$Loc → $Loc in $Loc
$Opt → me
$Mode → bike
$Mode → car
$ROOT → route ($Opt)? to $Loc by $Mode
```

- Usually not deterministic: many possible derivations per input

Syntax



Syntax

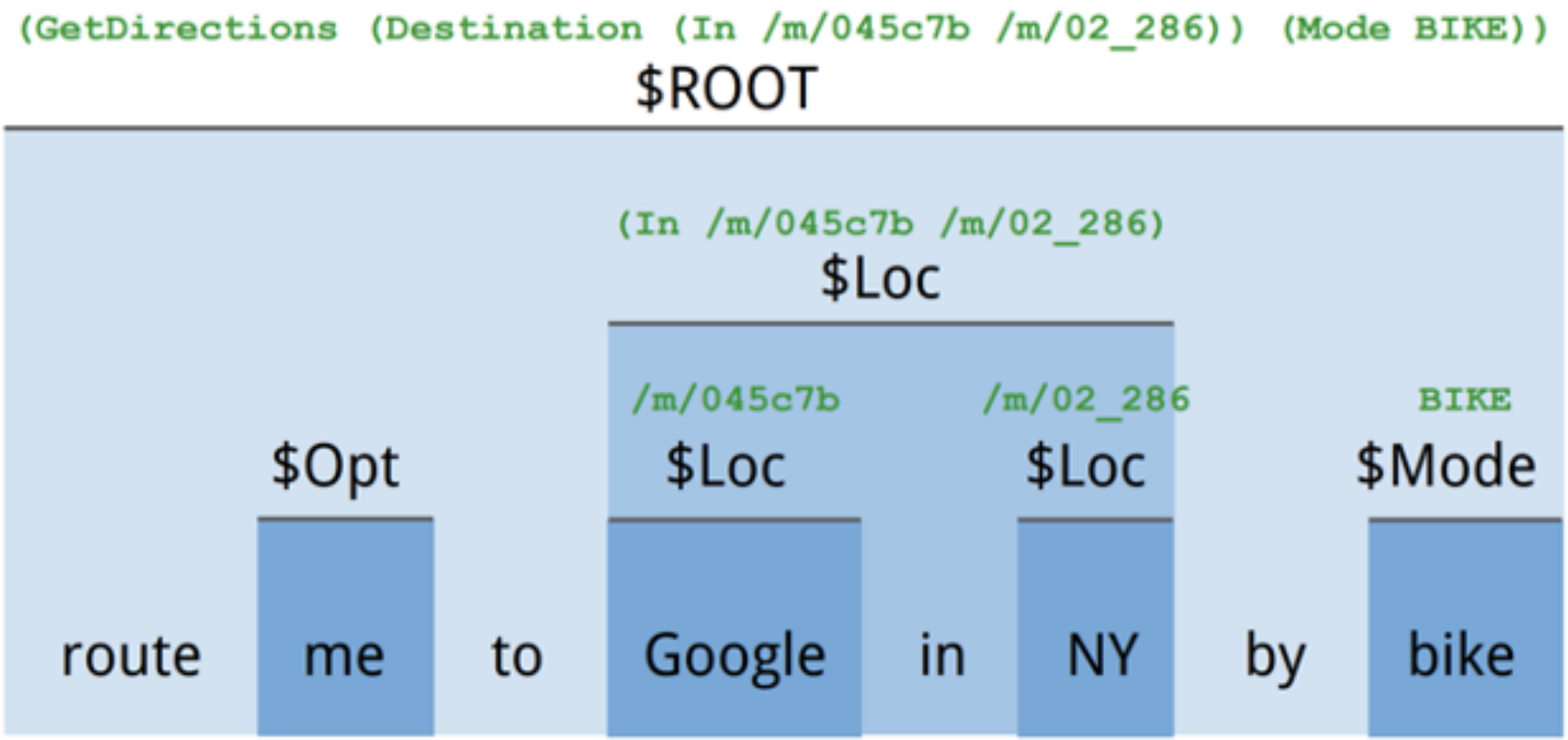


Use a syntactic parser!

Syntax with Semantics

```
$Loc → Google [/m/045c7b]
$Loc → NY [/m/02_286]
$Loc → $Loc in $Loc [(In $1 $2)]
$Opt → me []
$Mode → bike [BIKE]
$Mode → car [CAR]
$ROOT → route ($Opt)? to $Loc by $Mode
  [(GetDirections (Destination $2) (Mode $3))]
```

Syntax with Semantics



Where does Semantics Come from?

- We don't want a million individual rules like

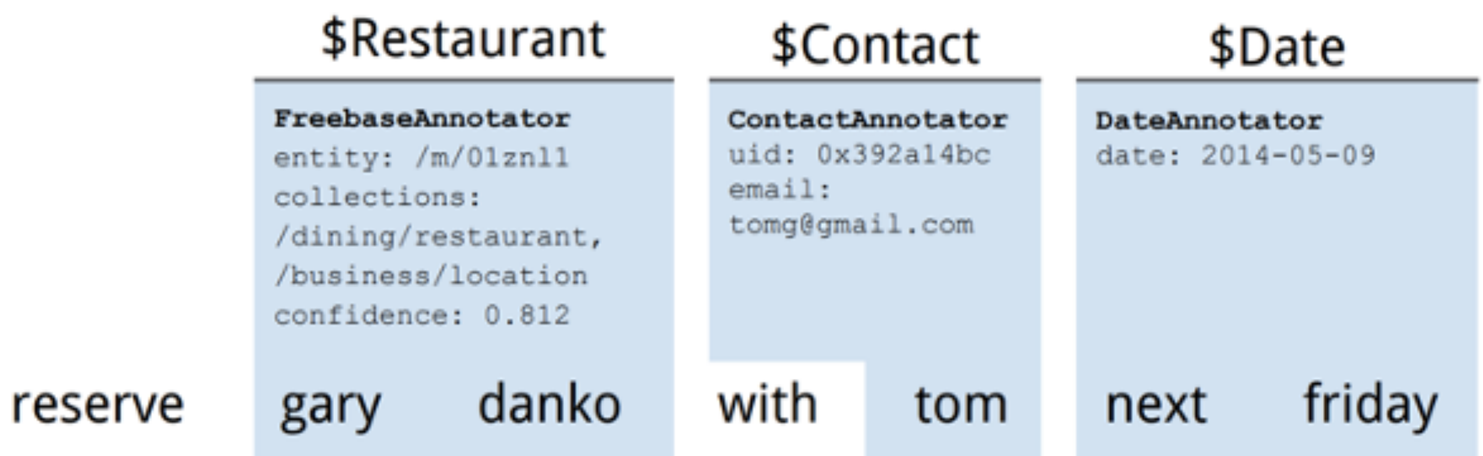
```
$Loc → NY [/m/02_286]
```

Where does Semantics Come from?

- We don't want a million individual rules like

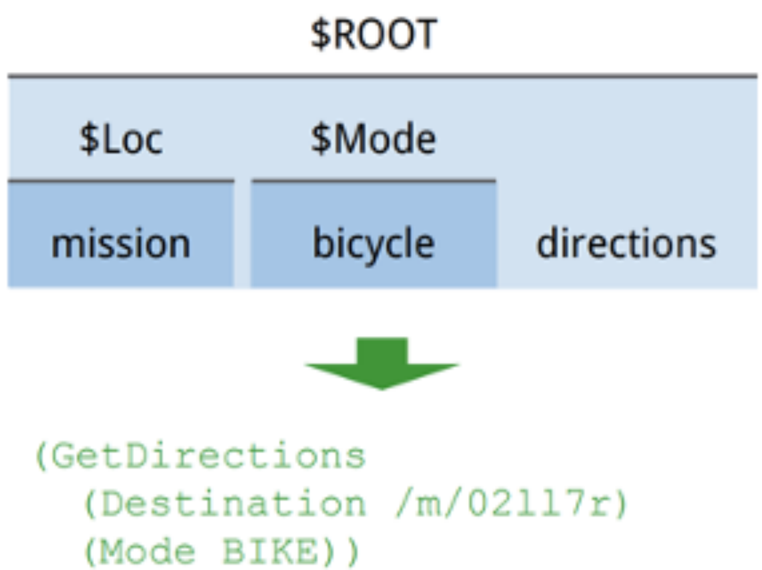
```
$Loc → NY [/m/02_286]
```

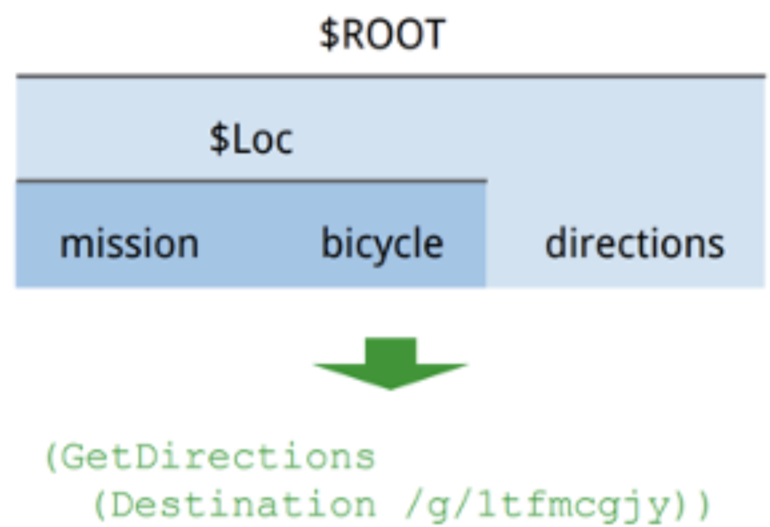
- Instead, leverage intelligence of special-purpose annotators



Ambiguity

- When grammar supports multiple interpretations, how to choose?





Ambiguity

- A log-linear model to score alternative parses

Ambiguity

- A log-linear model to score alternative parses
- Features from the input, the semantic yield, and the derivation

Ambiguity

- A log-linear model to score alternative parses
- Features from the input, the semantic yield, and the derivation
  - E.g., co-occurrence of "*to*" in input and *Destination* in semantics

Ambiguity

- A log-linear model to score alternative parses
- Features from the input, the semantic yield, and the derivation
  - E.g., co-occurrence of "*to*" in input and *Destination* in semantics
  - E.g., occurrence of specific CFG rules or categories in derivation

In-Class Exercise 2

- Find three examples of ambiguous sentences or headlines on the internet (Wikipedia, news, …). Write them down in a Google doc, together with a short description of the sources of the ambiguities.

Building the Syntax and Semantics

- Where do grammar rules come from?

Building the Syntax and Semantics

- Where do grammar rules come from?
  - For small domains, we can write the rules manually

Building the Syntax and Semantics

- Where do grammar rules come from?
  - For small domains, we can write the rules manually
  - For large, complex domains, this doesn't scale!

Building the Syntax and Semantics

- Where do grammar rules come from?
  - For small domains, we can write the rules manually
  - For large, complex domains, this doesn't scale!
- Rules can be induced automatically from data (Liang et al., 2011)

Paraphrase-driven Methods*

- Even collecting a diverse/interesting set of questions with their answers can be hard
  - Requires annotators to know what's in the database.
  - The paradox of information retrieval research:
    Users will only ask the system things they think it can answer.

*Building a Semantic Parser Overnight ([Wang et al., 2015](#))

Paraphrase-driven Methods

- A workaround: Paraphrasing
- Use the a simple hand-built syntax and semantics to generate a large random set of valid questions with their semantics (you get the answers for free).

article that has the largest publication date

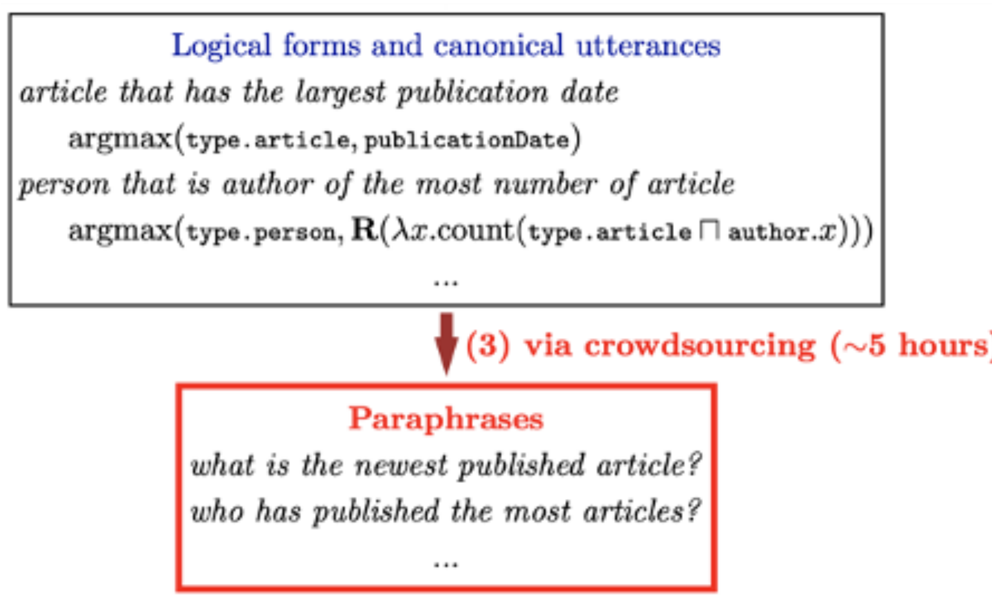
$$\mathrm{argmax}(\texttt{type.article}, \texttt{publicationDate})$$

person that is author of the most number of article

$$\mathrm{argmax}(\texttt{type.person}, \mathbf{R}(\lambda x.\mathrm{count}(\texttt{type.article} \sqcap \texttt{author}.x)))$$

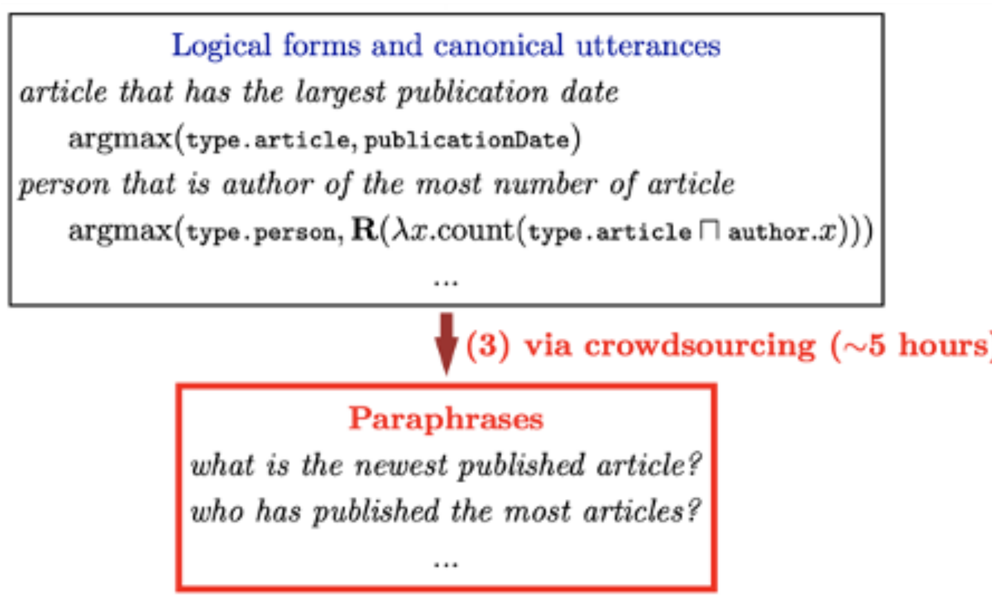
Paraphrase-driven Methods

- **Problem**: These questions may follow a different syntax from real user questions.
- So ask annotators to paraphrase the questions.
- **Result**: Lots of valid questions with their semantics.



Paraphrase-driven Methods

- **Bold claim**: This works with a small seed lexicon and a moderate amount of crowdsourcing-based annotation, so it's possible to build a working semantic parser for a new domain over night.



What We Want

Yes, hi, I need to book a flight for myself and my husband from SFO to Boston. Actually Oakland would be OK too. We need to fly out on Friday the 12th, and then I could come back on Sunday evening or Monday morning, but he won't return until Wednesday the 18th, because he's staying for business. No flights with more than one stop, and we don't want to fly on United because we hate their guts.

What We Want (Cont'd)

Six sculptures — C, D, E, F, G, H — are to be exhibited in rooms 1, 2, and 3 of an art gallery.

- Sculptures C and E may not be exhibited in the same room.
- Sculptures D and G must be exhibited in the same room.
- If sculptures E and F are exhibited in the same room, no other sculpture may be exhibited in that room.
- At least one sculpture must be exhibited in each room, and no more than three sculptures may be exhibited in any room.

If sculpture D is exhibited in room 3 and sculptures E and F are exhibited in room 1, which of the following may be true?

A. Sculpture C is exhibited in room 1.
B. Sculpture H is exhibited in room 1.
C. Sculpture G is exhibited in room 2.
D. Sculptures C and H are exhibited in the same room.
E. Sculptures G and F are exhibited in the same room.

Some Remarks

- Semantic parsing is ideal in settings where:

Some Remarks

- Semantic parsing is ideal in settings where:
  - You need a natural language interface to a structured knowledge source like a database, or to a software API.

Some Remarks

- Semantic parsing is ideal in settings where:
  - You need a natural language interface to a structured knowledge source like a database, or to a software API.
  - You need your system to handle recursive, compositional language, not just fixed phrases.

Some Remarks

- Semantic parsing is ideal in settings where:
  - You need a natural language interface to a structured knowledge source like a database, or to a software API.
  - You need your system to handle recursive, compositional language, not just fixed phrases.
  - You're willing to work in a limited domain (geography, airline tickets, movie trivia, …).

Some Remarks

- Semantic parsing can be hard:

Some Remarks

- Semantic parsing can be hard:
  - You'll need to build a backend that's able to handle any reasonable user query.

Some Remarks

- Semantic parsing can be hard:
  - You’ll need to build a backend that’s able to handle any reasonable user query.
  - Current methods can work in restricted settings, but break down in broader domains when the number of possible meanings for a question starts to explode.

Some Remarks

- Semantic parsing can be hard:
  - You'll need to build a backend that's able to handle any reasonable user query.
  - Current methods can work in restricted settings, but break down in broader domains when the number of possible meanings for a question starts to explode.
  - Broad-domain formalisms like AMR are not yet mature enough to support most applications directly.

Some Remarks

- Semantic parsing can be hard:
    - You’ll need to build a backend that’s able to handle any reasonable user query.
    - Current methods can work in restricted settings, but break down in broader domains when the number of possible meanings for a question starts to explode.
    - Broad-domain formalisms like AMR are not yet mature enough to support most applications directly.
    - Many domains seem simple, but start to feel quite broad when you start dealing with real user data

Some Remarks

- Semantic parsing can be hard:
  - You'll need to build a backend that's able to handle any reasonable user query.
  - Current methods can work in restricted settings, but break down in broader domains when the number of possible meanings for a question starts to explode.
  - Broad-domain formalisms like AMR are not yet mature enough to support most applications directly.
  - Many domains seem simple, but start to feel quite broad when you start dealing with real user data
  - Resolving ambiguity often requires world knowledge, which can be hard to use in pipelines like these

Meaning Representations for QA with a KB

Requirements for Meaning Representations

- Such representations need to be able to support the practical computational requirements of semantic processing:

Requirements for Meaning Representations

- Such representations need to be able to support the practical computational requirements of semantic processing:
  - Determine the truth of propositions

Requirements for Meaning Representations

- Such representations need to be able to support the practical computational requirements of semantic processing:
  - Determine the truth of propositions
  - Support unambiguous representations

Requirements for Meaning Representations

- Such representations need to be able to support the practical computational requirements of semantic processing:
  - Determine the truth of propositions
  - Support unambiguous representations
  - Represent variables

Requirements for Meaning Representations

- Such representations need to be able to support the practical computational requirements of semantic processing:
  - Determine the truth of propositions
  - Support unambiguous representations
  - Represent variables
  - Be sufficiently expressive

Requirements for Meaning Representations

- Such representations need to be able to support the practical computational requirements of semantic processing:
  - Determine the truth of propositions
  - Support unambiguous representations
  - Represent variables
  - Be sufficiently expressive
  - Support inference

First-Order Logic

- First-Order Logic (FOL) is a flexible, well-understood, and computationally tractable meaning representation language

First-Order Logic

- First-Order Logic (FOL) is a flexible, well-understood, and computationally tractable meaning representation language
- It provides a sound computational basis for the verifiability, inference, and expressiveness requirements

First-Order Logic

$$
\begin{array}{rcl}
\textit{Formula} & \rightarrow & \textit{AtomicFormula} \\
 & | & \textit{Formula Connective Formula} \\
 & | & \textit{Quantifier Variable}, \ldots \textit{ Formula} \\
 & | & \neg \textit{ Formula} \\
 & | & (\textit{Formula}) \\
\textit{AtomicFormula} & \rightarrow & \textit{Predicate}(\textit{Term}, \ldots) \\
\textit{Term} & \rightarrow & \textit{Function}(\textit{Term}, \ldots) \\
 & | & \textit{Constant} \\
 & | & \textit{Variable} \\
\textit{Connective} & \rightarrow & \wedge \mid \vee \mid \Longrightarrow \\
\textit{Quantifier} & \rightarrow & \forall \mid \exists \\
\textit{Constant} & \rightarrow & A \mid \textit{VegetarianFood} \mid \textit{Maharani} \cdots \\
\textit{Variable} & \rightarrow & x \mid y \mid \cdots \\
\textit{Predicate} & \rightarrow & \textit{Serves} \mid \textit{Near} \mid \cdots \\
\textit{Function} & \rightarrow & \textit{LocationOf} \mid \textit{CuisineOf} \mid \cdots
\end{array}
$$

Figure 16.3 A context-free grammar specification of the syntax of First-Order Logic representations. Adapted from Russell and Norvig (2002).

First-Order Logic

P	Q	$\neg P$	$P \wedge Q$	$P \vee Q$	$P \implies Q$
False	*False*	*True*	*False*	*False*	*True*
False	*True*	*True*	*False*	*True*	*True*
True	*False*	*False*	*False*	*True*	*False*
True	*True*	*False*	*True*	*True*	*True*

Figure 16.4 Truth table giving the semantics of the various logical connectives.

Inference with FOL

- Inference can be done via *if-then* reasoning

$$\frac{\begin{array}{l} VegetarianRestaurant(Leaf) \\ \forall x VegetarianRestaurant(x) \implies Serves(x, VegetarianFood) \end{array}}{Serves(Leaf, VegetarianFood)}$$

How to Use Semantic Parsing for QA

Given:
- A representation of the world (e.g., a KB)
- A question

How to Use Semantic Parsing for QA

Given:

- A representation of the world (e.g., a KB)
- A question

Wanted:

- The answer to the question according to the current state of the world

The Model or World

- Meaning representations have two parts:
  - The **non-logical vocabulary** consists of names for objects, properties, and relations that make up the world we're trying to represent

The Model or World

- Meaning representations have two parts:
  - The **non-logical vocabulary** consists of names for objects, properties, and relations that make up the world we're trying to represent
  - The **logical vocabulary** consists of a closed set of symbols, operators, quantifiers, links, etc.

How to Use Semantic Parsing for QA

1. Decide on a meaning representation

How to Use Semantic Parsing for QA

1. Decide on a meaning representation
2. Train your semantic parser

How to Use Semantic Parsing for QA

1. Decide on a meaning representation
2. Train your semantic parser
3. Parse your questions

How to Use Semantic Parsing for QA

1. Decide on a meaning representation
2. Train your semantic parser
3. Parse your questions
4. Evaluate against an existing knowledge base

Wrapping up

- Discussed today:
    - Syntactic parsing
    - Semantic parsing
    - Some QA

- On Wednesday: Question Answering without Semantic Parsing