Semantics

He He

New York University

November 3, 2021

Table of Contents

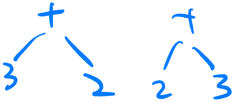
Syntax vs semantics

Syntax: does the string belong to the language?

Semantics: what is the meaning of the string?

Examples in programming languages:

Different syntax, same semantics



$2 + 3$ $\quad$ $3 + 2$

Same syntax, different semantics

3 / 2 (Python 2.7) $\quad$ 3 / 2 (Python 3)

(Slide adapted from Stanford CS221 Lecture 16)

Model-theoretic semantics

An **expression** is a string of mere symbols.

A **model** defines meanings of symbols.

The output of an expression with respect to a model is its **denotation**.

expression	model	denotation
`3 + 2 * 4`	calculator	11
`the red ball`	an image	the red ball in the image
`SELECT Name FROM Student WHERE Id = 0;`	database	John
`Book me a ticket from NYC to Seattle`	database	[action]

We understand the expression if we know how to act (in a world).

Natural language as expressions

Motivating applications:

Question answering

What is the profit of Mulan?

Who is the 46th president of the US?

Personal assistant

Alexa, play my favorite song.

Siri, show me how to get home.

- But natural language is full of ambiguities
- Cannot be directly handled by a computer (unlike programming/formal languages)

Semantic analysis

Goal: convert natural language to meaning representation

John likes fruits. (*informal*)

$\forall x \ \mathrm{FRUIT}(x) \implies \mathrm{LIKES}(x, \mathrm{JOHN})$ (*formal*)

Main tool: first-order logic

Why logic?

- Unambiguity: one meaning per statement
- Knowledge: link symbols to knowledge (entities, relations, facts etc.) (*Take in complex information*)
- Inference: derive additional knowledge given statements (*Reason with the information*)

Logic and semantics: example

Natural language: "John likes Mary's friends"

Logical form: $\forall x \; \textsc{Friends}(x, \textsc{Mary}) \implies \textsc{Likes}(x, \textsc{John})$

World model: state of affairs in the world

People $= \{\text{John}, \text{Mary}, \text{Joe}, \text{Ted}\}$

John is a friend of Mary.

Joe is a friend of Mary.

Given the world model,

- Is $\textsc{Likes}(\textsc{Joe}, \textsc{John})$ true?
- What else can we infer from the statement?

The value of the expression may change given a different world model.

Table of Contents

Propositional logic

A **proposition** is a statement that is either true or false.
Propositional logic deals with propositions and their relations.

Syntax of propositional language:

- **Propositional symbols**: a *primitive* set of propositions
  - p_1: John likes Mary
  - p_2: John is a student
- **Logical connectives**: rules to build up **formulas**

symbol	read	meaning	formula
$\neg$	not	negation	$\neg p$
$\vee$	or	disjunction	$p \wedge q$
$\wedge$	and	conjunction	$p \vee q$
$\implies$	implies / if then	implication	$p \implies q$
$\iff$	equivalent to / iff	equivalence	$p \iff q$

- Parentheses: (,)

Parsing a formula

How would you check if a formula is valid (i.e. grammatical)?

A propositional formula is contructed by connecting propositions using the connectives.

- Formulas can be nested.
- Parentheses are used to disambiguate formulas.

Example:

$$((p \land q) \land \neg p)$$
$$((p \lor q) \land r) \implies p)$$

Try to draw the parse trees of the formulas.

World model for propositional logic

Propositional symbols:

$$p_1 = \text{hot}$$
$$p_2 = \text{John likes ice cream}$$
$$p_3 = \text{John ate an ice cream}$$

Formula: $p_1 \wedge p_2 \implies p_3$ (Is this true?)

World model for propositional logic

Propositional symbols:

$$p_1 = \text{hot}$$
$$p_2 = \text{John likes ice cream}$$
$$p_3 = \text{John ate an ice cream}$$

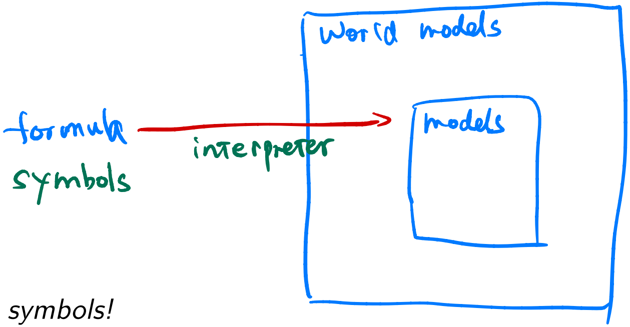
Formula: $p_1 \wedge p_2 \implies p_3$ (Is this true?)

The **world model** in propositional logic is an *assigment* of truth values to propositional symbols.

	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8
p_1	T	T	T	T	F	F	F	F
p_2	T	T	F	F	T	T	F	F
p_3	T	F	T	F	T	F	T	F

In which world(s) is the above formula false?

Meaning of a formula



Propositional symbols:

p_1 = hot

p_2 = John likes ice cream

p_3 = John ate an ice cream

Formula: $p_1 \wedge p_2 \implies p_3$ *Just symbols!*

Semantics is given by *interpreting* the formula against a world model.

A formula specifies *a set of world models* where it is true.

A set of formulas is a knowledge base (constraints on the world model).

Making inference given formulas and the world model: take a course in AI.

Limitations of propositional logic

How do we represent knowledge of a *collection* of objects?

"Everyone who likes ice cream ate an ice cream."

p_{JOHN} (John likes ice cream) $\implies$ q_{JOHN} (John ate an ice cream)

p_{JOE} (Joe likes ice cream) $\implies$ q_{JOE} (Joe ate an ice cream)

p_{ALICE} (Alice likes ice cream) $\implies$ q_{ALICE} (Alice ate an ice cream)

p_{CAROL} (Carol likes ice cream) $\implies$ q_{CAROL} (Carol ate an ice cream)

. . .

[ ] likes ice cream $\implies$ [ ] ate an ice cream

Need a compact way to represent a collection of objects!

First-order logic

First-order logic generalizes propositional logic with several new symbols:

Represent objects:

Constants	Primitive objects, e.g. John
Variables	Placeholder for some object, e.g. x
Functions	A map from object(s) to an object, e.g. John $\rightarrow$ John's farther

Group objects:

Predicate	Properties of a set of objects, e.g. students, couples

Quantify a (infinite) set of objects:

Quantifiers	Specify the number of objects with a certain property, e.g. *all* people are mortal.

Constants, variables, functions

Constants refer to primitive objects such as named entities:

$\textsc{John}, \textsc{IceCream}, \textsc{Hot}$

A **variable** refers to an unspecified object:

x, y, z

$\textsc{Student}(x)$

$\textsc{Friends}(x, \textsc{John})$

A *n*-ary **function** maps *n* objects to an object:

$\textsc{Mother}(x)$

$\textsc{Friends}(\textsc{Mother}(x), \textsc{Mother}(y))$

Predicates

A **predicate** is an indicator function $P\colon X \to \{\text{true}, \text{false}\}$.

- Describes properties of object(s)
- $P(x)$ is an atomic formula

  $\textsc{Student}(\textsc{Mary})$

  $\textsc{Smaller}(\textsc{Desk}, \textsc{Computer})$

  $\textsc{Friends}(\textsc{John}, \textsc{Mary}) \implies \textsc{Friends}(\textsc{Mary}, \textsc{John})$

Quantifiers

Universal quantifier $\forall$:

- The statement is true for *every* object
- $\forall x\ P(x)$ is equivalent to $P(A) \land P(B) \land \ldots$
- All people are mortal: $\forall x\ \textsc{Person}(x) \implies \textsc{Mortal}(x)$

Existential quantifier $\exists$:

- The statement is true for *some* object
- $\exists x\ P(x)$ is equivalent to $P(A) \lor P(B) \lor \ldots$
- Some people are mortal: $\exists x\ \textsc{Person}(x) \land \textsc{Mortal}(x)$

Order matters, e.g., "everyone speaks a language":

$$\forall x \exists y\ \textsc{speaks}(x, y)$$
$$\exists y \forall x\ \textsc{speaks}(x, y)$$

Syntax of first-order logic

Terms refer to objects:

- Constant symbol, e.g. $\textsc{John}$
- Variable symbol, e.g. x
- Function of terms, e.g. $\textsc{Mother}(x)$, $\textsc{Capital}(\text{NY})$

Formula evaluates to true or false:

- Predicate over terms is an atomic formula, e.g. $\textsc{Student}(\textsc{Mother}(\textsc{John}))$
- Connectives applied to formulas (similar to propositional logic)

$$\textsc{Student}(x) \quad \wedge \quad \textsc{Happy}(x)$$

- Quantifiers applied to formulas

$$\forall x\, \textsc{Student}(x) \implies \textsc{Happy}(x)$$
$$\exists x\, \textsc{Student}(x) \wedge \textsc{Happy}(x)$$

World model of first-order logic

How do we know if $\textsc{Friends}(\textsc{John}, \textsc{Mary})$ is true?

World model of propositional logic: propositions

proposition	truthful value
John is a friend of Mary	True
John is a friend of Joe	False

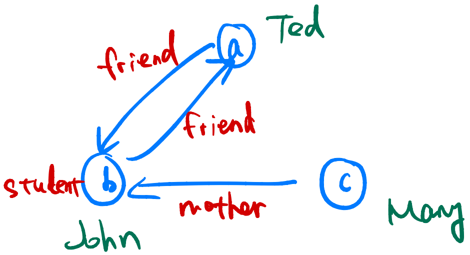
World model of first-order logic: objects and their relations

constant symbol	object
$\textsc{John}$	a
$\textsc{Mary}$	b

predicate symbol	set of n-tuples
$\textsc{Friends}$	$\{(a, b), (b, a)\}$

Graph representation of the world model

Const symbols



$$friend = \{(a, b), (b, a)\}$$

$$mother = \{(c, b)\}$$

$$student = \{a, b\}$$

$$families = \{(a, b, c), \ldots\}$$

Summary

Syntax produces symbols and well-formed formulas.

Semantics grounds symbols to a world and allows for evaluation of formulas.

We have seen how it works for formal languages such as propositional logic and first-order logic.

Next, formal language to natural language.

Table of Contents

System overview

Utterance Linguistic expression.
"Call John, please."

parsing

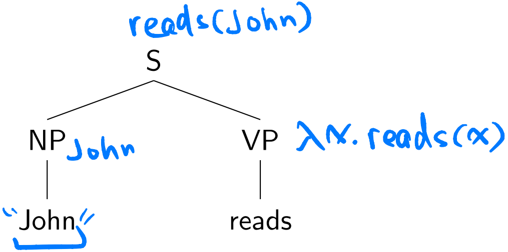
Logical form Formal meaning representation of the utterance
CALL(JOHN) *program*

executor

Denotation Output of the meaning representation with respect to the model
Calling XXX-XXX-XXXX ... *execution result*

Translate NL to logical language

Key idea: *compositionality*



- Sentence: READS(JOHN) (What's the denotation?)
- We would like to construct it recursively
  - John: JOHN (a unique entity)
  - reads: a predicate (function) that takes an entity (one argument)

A brief introduction to lambda calculus

Lambda calculus / λ-calculus

- A notation for applying a function to an argument

$$\lambda x.x^2 + x$$

$f: x \mapsto x^2 + x$

- A function that is waiting for the value of a variable to be filled
- Function application by β-reduction

$$(\lambda x.x^2 + x)(2) = 2^2 + 2 = 6$$

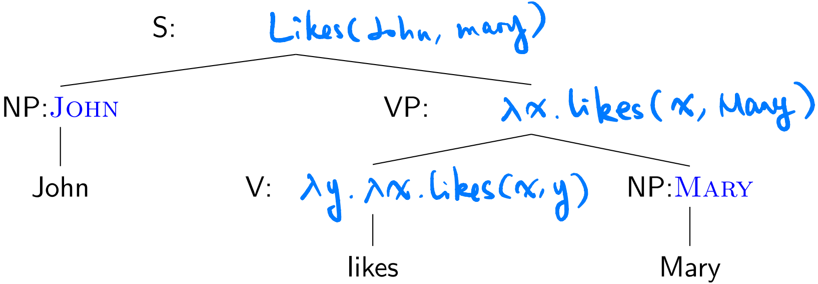
- Takes multiple arguments by "currying"

$$(\lambda x.\lambda y.xy)(2) = \lambda y.2y$$
$$(\lambda x.\lambda y.xy)(3)(2) = (\lambda y.2y)(3) = 6$$

Translate NL to logical language

Verbs are predicates

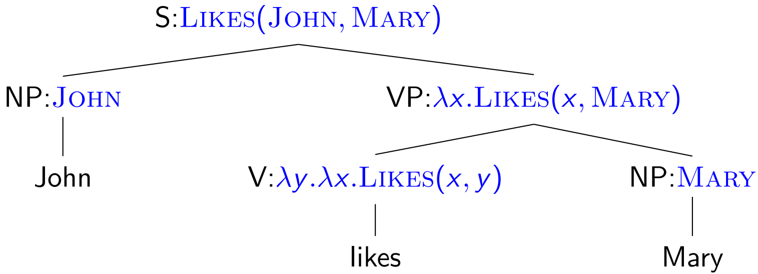
- reads: $\lambda x.\textsc{Reads}(x)$ (waiting for an NP)
- likes: $\lambda x.\lambda y.\textsc{Likes}(x, y)$ (waiting for two NPs)



Translate NL to logical language

Verbs are predicates

- reads: $\lambda x.\textsc{Reads}(x)$ (waiting for an NP)
- likes: $\lambda x.\lambda y.\textsc{Likes}(x, y)$ (waiting for two NPs)



Compositional semantics

Bottom up parsing:

- Start with the semantics of each word
- Combine semantics of spans according to certain rules
  - Associate a combination rule with each grammar rule

$$
\begin{array}{lcl}
\text{V:}\lambda y.\lambda x.\text{LIKES}(x, y) & \rightarrow & \text{likes} \\
\text{NP:}\text{JOHN} & \rightarrow & \text{John} \\
\text{VP:}\alpha(\beta) & \rightarrow & \text{V:}\alpha \ \text{NP:}\beta \\
\text{S:}\beta(\alpha) & \rightarrow & \text{NP:}\alpha \ \text{VP:}\beta
\end{array}
$$

  - Get semantics by function applcation
- Lexical rules can be complex!

Quantification

John bought a book

$\textsc{Bought}(\textsc{John}, \textsc{Book})$?

"book" is not a unique entity! $\textsc{Bought}(\textsc{Mary}, \textsc{Book})$

Correct logical form: $\exists x \textsc{Book}(x) \wedge \textsc{Bought}(\textsc{John}, x)$

But what should be the semantics of "a"? $\lambda P.\lambda Q.\exists x\ P(x) \wedge Q(x)$

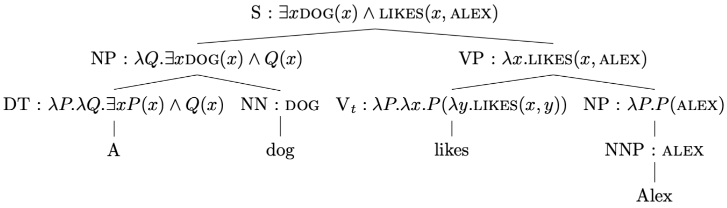
"a book": $\lambda Q.\exists x\ \textsc{Book}(x) \wedge Q(x)$. (Need to change other NP rules)

What about "the", "every", "most"?

We also want to represent tense: "bought" vs "will buy". (event variables)

Learning from derivations

Text: John bought a book (utterance)

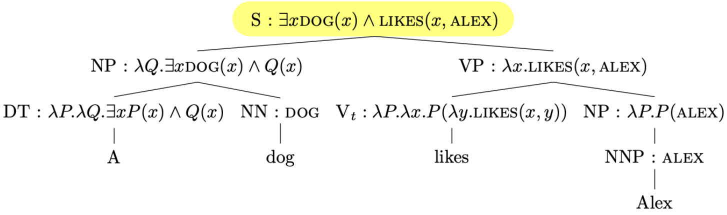


Annotation:

Use approaches from (discriminative) constituent parsing

Learning from derivations

Text: John bought a book (utterance)



Annotation:

Use approaches from (discriminative) constituent parsing

Obstacles:

- Derivations are rarely annotated.
- Unlike syntactic parsing, cannot obtain derivations from logical forms.
- Spurious derivation: wrong derivations that reach the correct logical form.

Learning from logical forms

Text: John bought a book (utterance)
Annotation: $\exists x \textsc{Book}(x) \wedge \textsc{Bought}(\textsc{John}, x)$ (logical form)

Key idea: model derivation as a latent variable z [Zettlemoyer and Collins, 2005]

Learning: maximum marginal likelihood

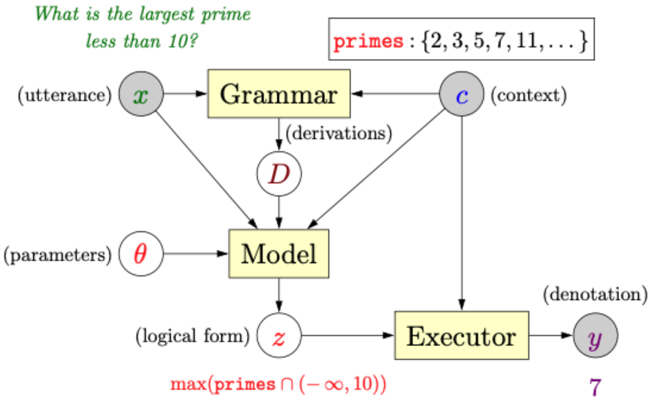
$$\log p(y \mid x) = \log \sum_z p(y, z \mid x)$$
$$= \log \sum_z \frac{\exp\left(\theta \cdot \Phi(x, y, z)\right)}{\sum_{z', y'} \exp\left(\theta \cdot \Phi(x, y', z')\right)}$$

- Need to learn both the lexicon and the model parameters (for CCG)
- Use EM algorithm (with approximation)

Learning from denotations

Text: What states border Georgia?
Annotation: Alabama, Florida, North Carolina, South Carolina, Tennessee

Key idea: model the logical form as a latent variable z [Liang, 2013]



Datasets

Geo880

- 880 questions and database queries about US geography
- “what is the highest point in the largest state?”
- Compositional utterances in a clean, narrow domain

ATIS

- 5418 utterances of airline queries and paired logical forms
- “show me information on american airlines from fort worth texas to philadelphia”
- More flexible word order but simpler logic

Free917, WebQuestions

- Questions and paired logical forms on Freebase
- Logically less complex but scales to many more predicates

Text to SQL

Spider (Yu et al. 2018)

Expert-annotated, cross-domain, complex text-to-SQL dataset

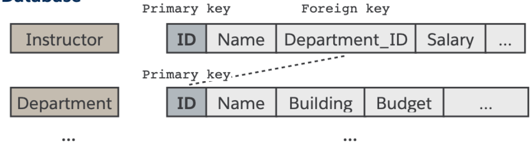
Assumption:

- For each

Hidden

	Train	Dev	Test
# DBs	146	20	40
# Examples	8,659	1,034	2,147

Database



Question What are the name and budget of the departments with average instructor salary above the overall average?

SQL

```
SELECT T2.name, T2.budget
FROM Instructor AS T1 JOIN Department AS T2 ON
T1.Department_ID = T2.ID
GROUP BY T1.Department_ID
HAVING AVG(T1.salary) >
    (SELECT AVG(Salary) FROM Instructor)
```

(Slide from Victoria Lin)

Challenges

Design the logical representation and grammar

- Expressivity vs computation efficiency
- Domain-specific vs domain-general
- Interacts with annotation and learning

Learning from different supervision signals

- End-to-end (utterance to action)
- Reinforcement learning (robotics, visual grounding)
- Interactive learning (obtain user feedback)