L98: Introduction to Computational Semantics

Lecture 6: Truth

Weiwei Sun and Simone Teufel

Natural Language and Information Processing Research Group
Department of Computer Science and Technology
University of Cambridge

Lent 2021/22

Lecture 6: Truth

1. Ferdinand de Saussure
2. World model and discourse referents
3. Functions and λs
4. Truth conditions
5. First-Order Predicate Logic

Ferdinand de Saussure

Sign

De Saussure: The linguistic sign is a two-sided psychological entity:

- signifier: ‘sound-image’
- signified: ‘concept’

Example: 止戈为武

- 止 initially meant foot, walk, go
- 戈 is an old-fashioned weapon
- 武 means military
- When 武 was created, the meaning of was "take your weapon and walk, go to war"
- So initially, there is semantic compositionality
- After many years, the meanings of the parts shifted and people could no longer see the compositionality.
- 止 now means "stop".
- Now some philosophers reinterpret 武 as: Stop using weapons; that is what a military should do.

Arbitrariness of the sign

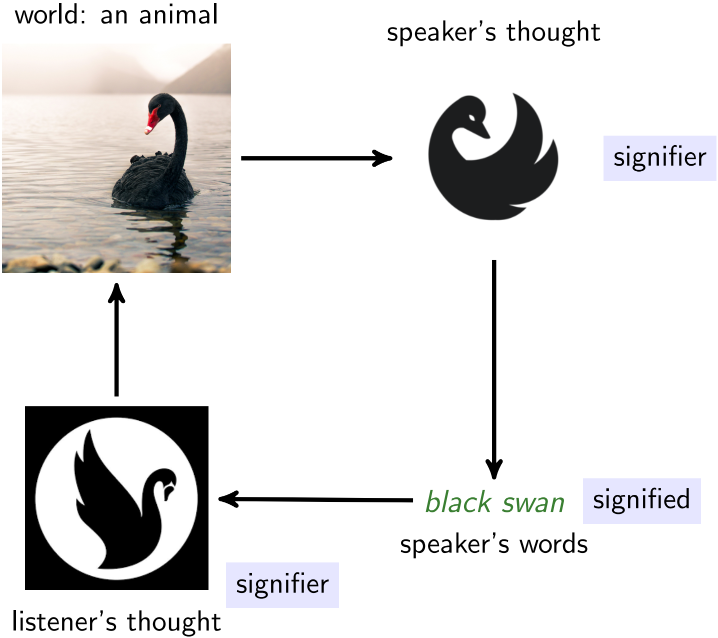
- De Saussure stated that the link between the signified and signifier is *arbitrary*.
- The example of 武 shows that it does not matter if language users know the complicated (non-arbitrary) history of this sign.
- The connection has become arbitrary.
- All you need to know to communicate is to know that 武 means military.
- Triumph of arbitrariness of the sign.

Old: mind and language

Mind

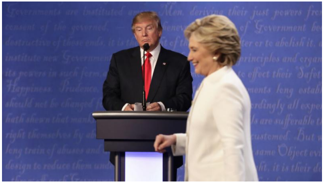
Language

Reality, mind and language



Reality, mind and language

world: Trump elected



speaker's thought

signifier

???

a black swan event

signified

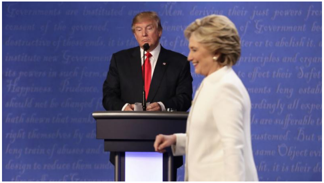
speaker's words

signifier

listener's thought

Reality, mind and language

world: Trump elected



a black swan event

speaker's words

Natural Language Understanding

Example: Visual QA

Something gets lost if a system goes directly from words to images (diagonal)

- Idea of such a system is that the meaning of the language string creates something like a image in the mind.
- But we have just seen that it's not a picture that is in the mind, otherwise the misunderstanding between listener and speaker would not have happened.
- This is why deep NLU needs to model the thought
- Rest of this lecture: how can we model the thought

World Model and Discourse Referent

Domains of interpretation

- the real world
- a part of the real world
- a hypothesized model of the real world ▷world model
  - e.g. Shapeworld: objects with properties in positions
  - or something more complicated
- some constructed model in the case of an artificial language

Desired properties of a world model

A world model is an abstracted, simplified version of our world.

- The world model should be precise.
- reflect the complexity of the phenomena we think are improtant
- drop other things we don't care about
- should have a systematic way to be constructed

- Its components should be transparent, ie, it should be easy to see what in our world is what in the model.
- There should be a close link between our model and those phenomena in the real world we care about

Our world model

It consists of

- **discourse referents.** unique variables standing in for actual people and objects in the world
- **semantic predicates.** functions representing “buckets” (certain nouns) and properties and events

We will start with proper names and simple predicates…

Trump gave Johnson a golden lighter.

The term “predicate” is also often used to describe a particular syntactic elements. We use“semantic predicate” to distinguish these two concepts.

Extension and intension

The extension of a linguistic expression is the set of things it extends to, or applies to.

Example: *politician*

$$\{\text{Trump, Johnson, } \ldots\}$$

Extension and intension

The extension of a linguistic expression is the set of things it extends to, or applies to.

Example: *politician*

$$\{\text{Trump, Johnson, } \ldots\}$$

The above set could also be the extension of *lier*.

Extension and intension

The extension of a linguistic expression is the set of things it extends to, or applies to.

Example: *politician*

$$\{\text{Trump, Johnson, } \ldots\}$$

The above set could also be the extension of *lier*.

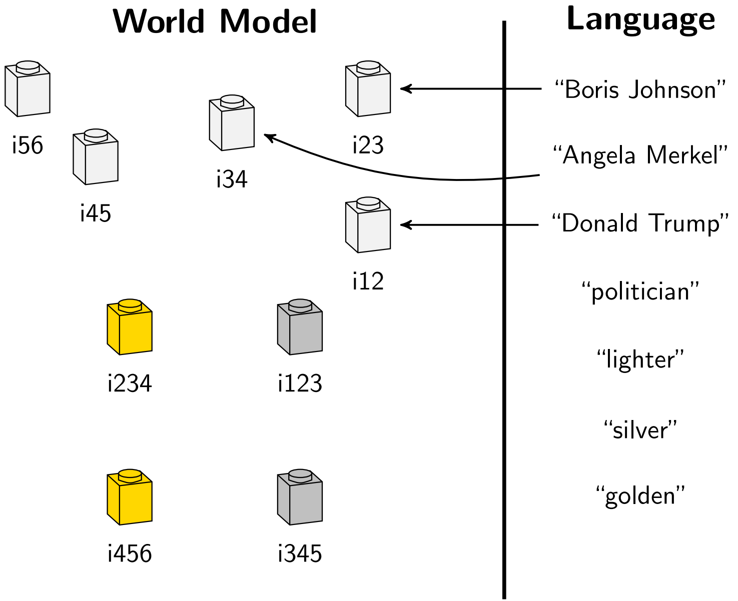
Intensional semantics

e.g. a description from wikipedia

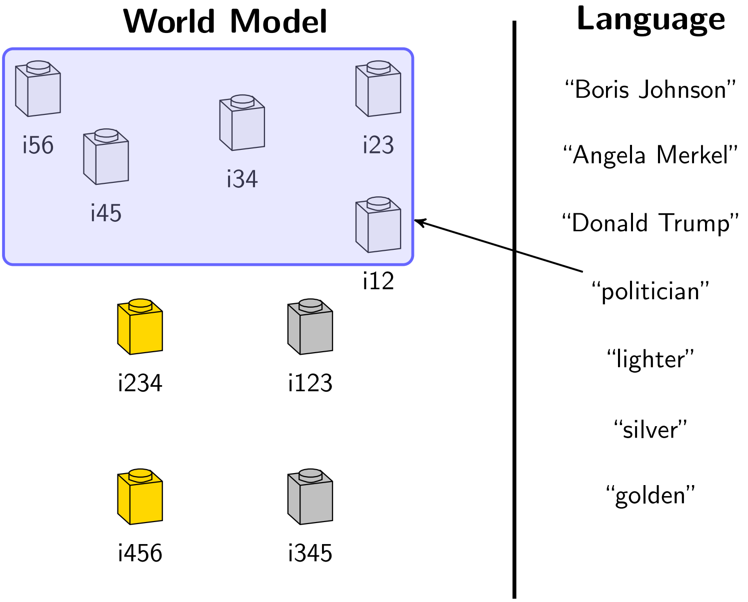
A *politician* is a person active in party politics, or a person holding or seeking an elected seat in government.

- intensional, but imprecise description

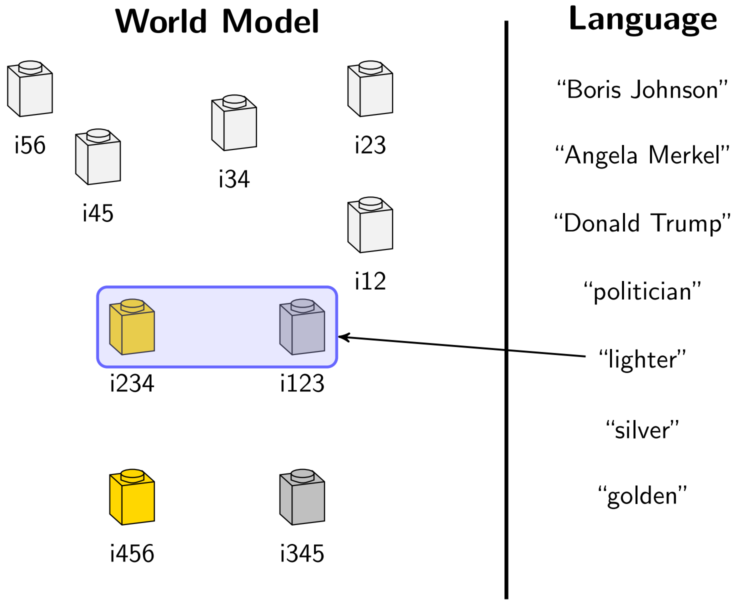
Discourse referents



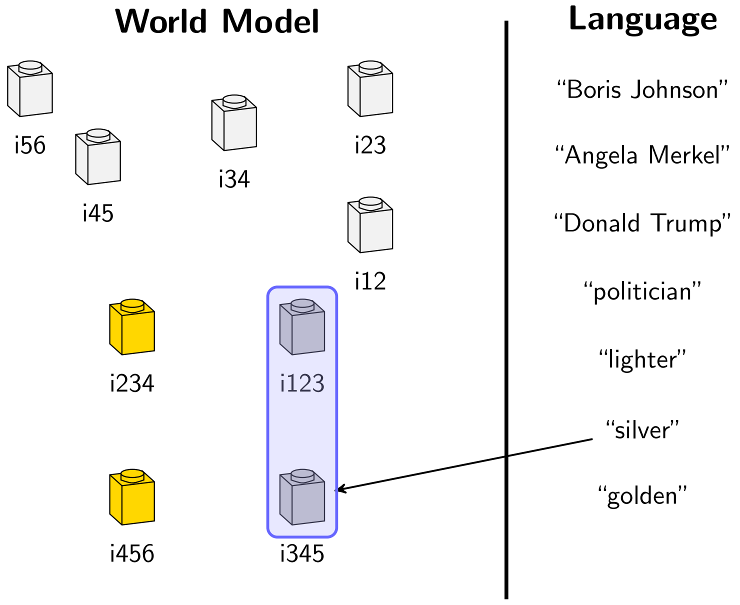
Discourse referents



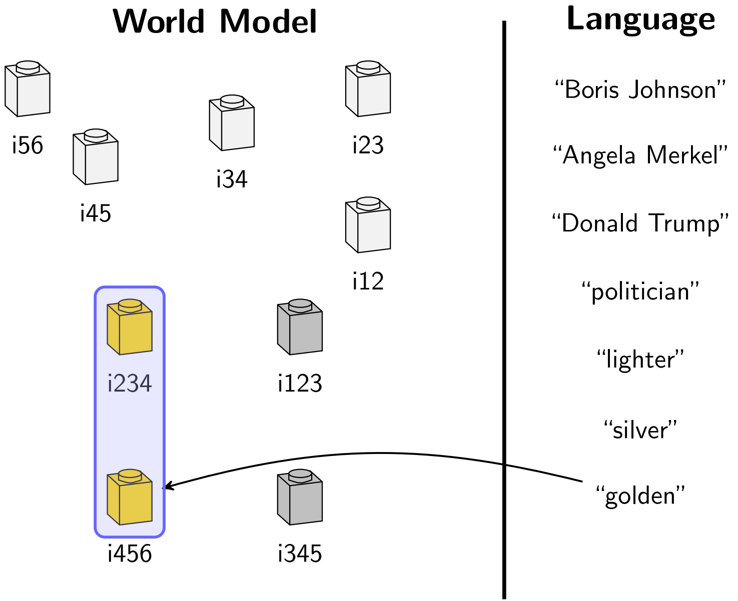
Discourse referents



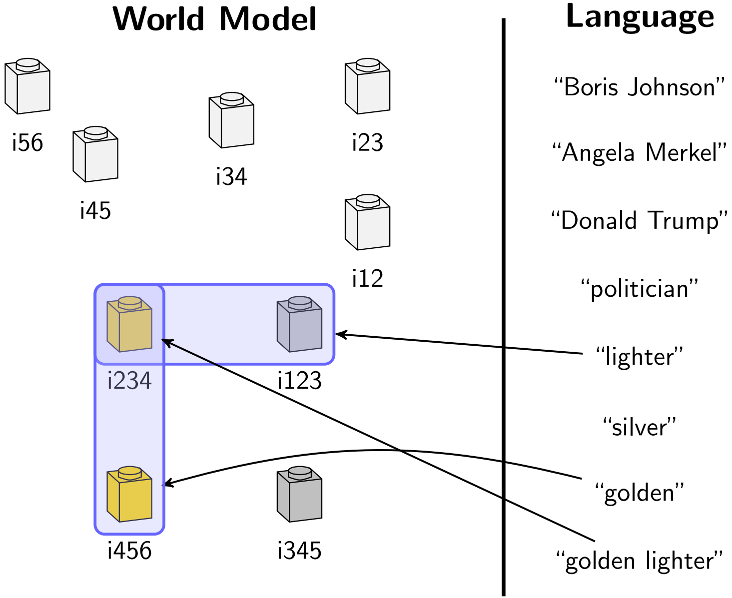
Discourse referents



Discourse referents

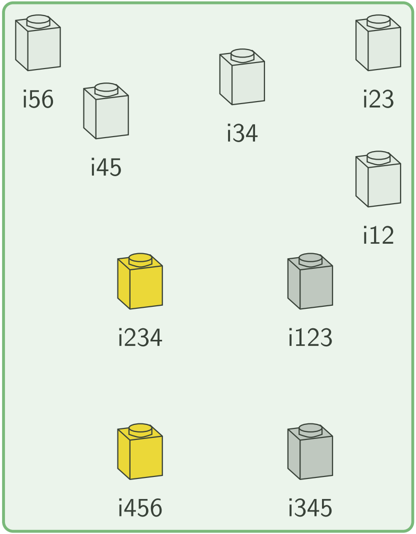


Discourse referents



Discourse referents

World Model



Language

“Boris Johnson”

“Angela Merkel”

“Donald Trump”

“politician”

“lighter”

“silver”

“golden”

. . .

Extensional interpretation

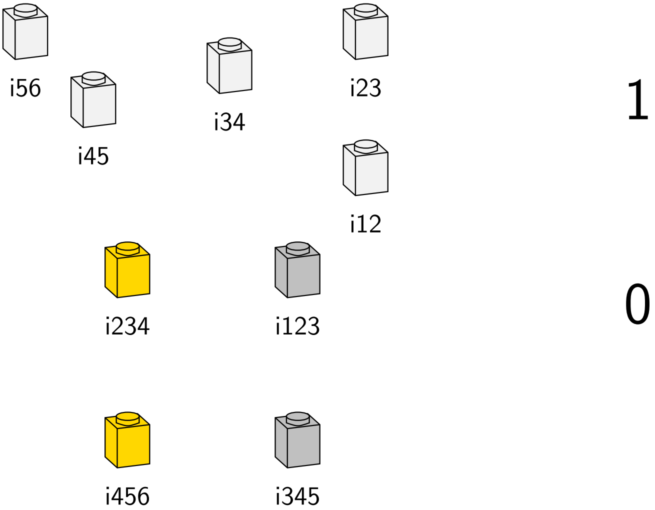
- An interpretation function ($[\![\;]\!]$) maps language expressions onto objects, sets of objects, sets of sets of..., of the world model.

  e.g $[\![$*politician*$]\!] = \{i12, i23, i34, i45, i56\}$
- In this lecture, objects of the world model are discourse referents.
- To simplify, proper names are mapped to **unique** discourse referents.

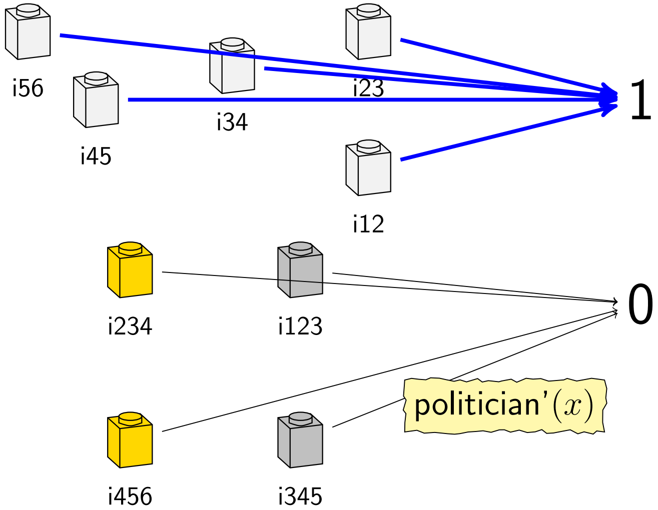
  e.g $[\![$*Angela Merkel*$]\!] = i34$

Functions and λs

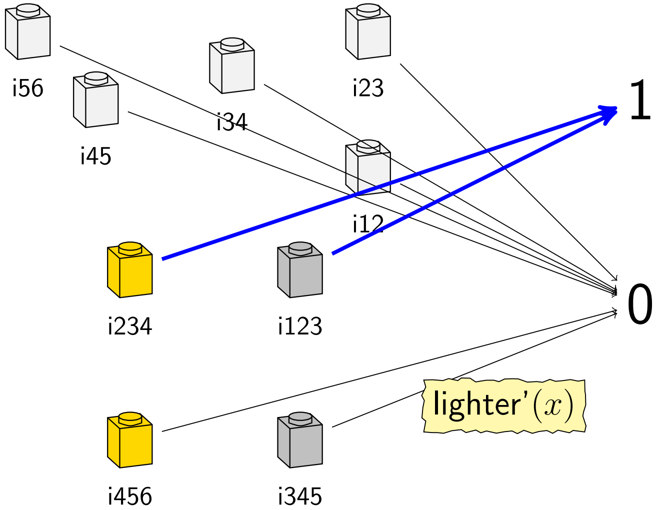
Buckets/sets → functions



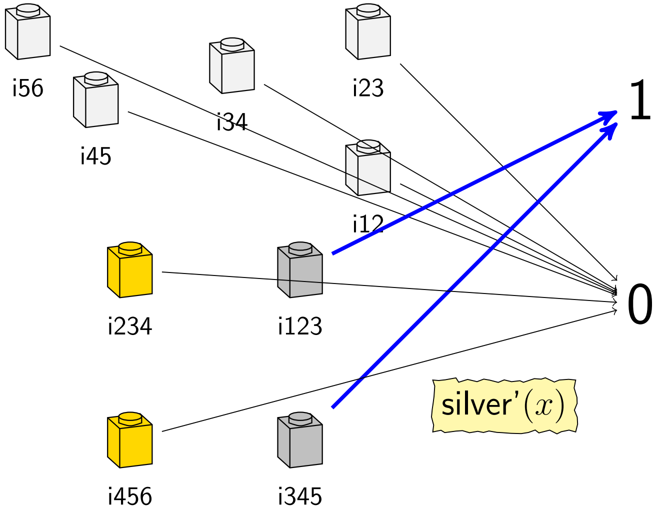
Buckets/sets → functions



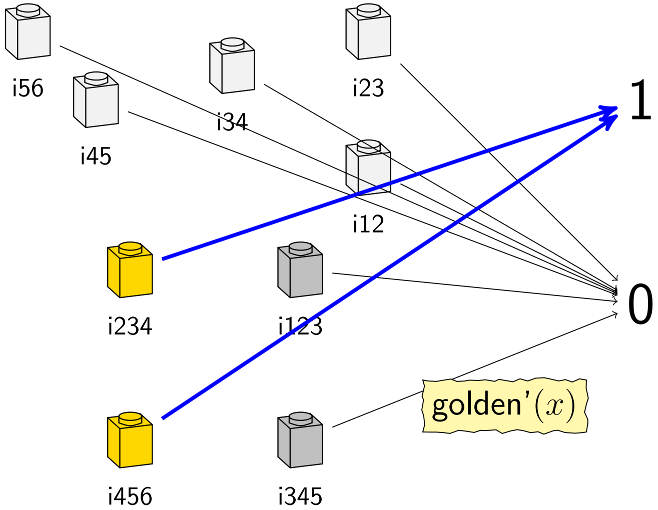
Buckets/sets → functions



Buckets/sets → functions



Buckets/sets → functions



Predicates are functions; predicates are sets.

Q What is the meaning of *politician*?

A politician'

- politician' is a semantic predicate which is a set and also a function.
- Discourse referents are mapped to either 0 or 1 through politician'. The referents mapped to 1 indicate politicians.
- It is a great idea to define functions with a minimal programming language — λ-calculus.

Building functions

λ-calculus — a simple notation for functions and application

- β-reduction/function application:

$$[\lambda x.M](N) \longrightarrow M[x := N]$$

- Apply a λ-term to an argument, and get a value.

More online: `https://plato.stanford.edu/entries/lambda-calculus/`

Example

- $f(x) = x^2 \longleftrightarrow [\lambda x.[x^2]]$
- $f(5) = 25 \longleftrightarrow [\lambda x.[x^2]](5) = 25$
- $g(x, y) = x^2 + y^2 \longleftrightarrow [\lambda x.[\lambda y.[x^2 + y^2]]]$
- $g(2, 1) = 5 \longleftrightarrow [\lambda x.[\lambda y.[x^2 + y^2]]](2)(1) = 5$

Simple types

From a nonempty set $\mathbf{BasTyp}$ of *basic types*, the set $\mathbf{Typ}$ is the smallest set such that

- $\mathbf{BasTyp} \subseteq \mathbf{Typ}$,
- $\langle \sigma, \tau \rangle \in \mathbf{Typ}$ if $\sigma, \tau \in \mathbf{Typ}$.

A type of the form $\langle \sigma, \tau \rangle$ is said to be a *functional type*.

Example

- Assume $\mathbf{e}$ for individuals and $\mathbf{t}$ for `true`/`false`,
- then $\langle \mathbf{e}, \mathbf{t} \rangle$ is the type for unary relations,
- and $\langle \langle \mathbf{e}, \mathbf{t} \rangle, \langle \mathbf{e}, \mathbf{t} \rangle \rangle$ is for the type of a function mapping unary relations into unary relations.

Simple types

From a nonempty set $\mathbf{BasTyp}$ of *basic types*, the set $\mathbf{Typ}$ is the smallest set such that

- $\mathbf{BasTyp} \subseteq \mathbf{Typ}$,
- $\langle \sigma, \tau \rangle \in \mathbf{Typ}$ if $\sigma, \tau \in \mathbf{Typ}$.

A type of the form $\langle \sigma, \tau \rangle$ is said to be a *functional type*.

Example

- Assume $\mathbf{e}$ for individuals and $\mathbf{t}$ for `true`/`false`,
- then $\langle \mathbf{e}, \mathbf{t} \rangle$ is the type for unary relations,
- and $\langle \langle \mathbf{e}, \mathbf{t} \rangle, \langle \mathbf{e}, \mathbf{t} \rangle \rangle$ is for the type of a function mapping unary relations into unary relations.

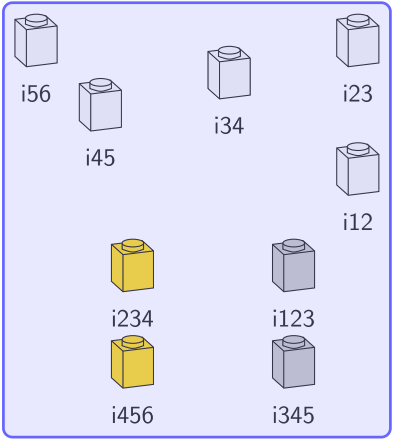
`C/C++/Java/Typescript` vs `Python/Javascript`

e, **t** and **e** to **t**

Gottlob Frege

There are only two atomic things, truth values and individuals. All other things are created by function application.

entity **e**

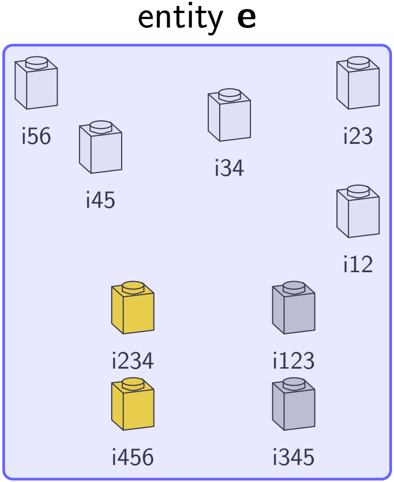


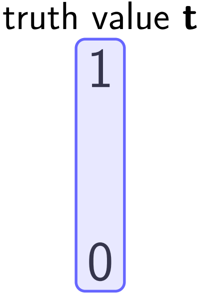
truth value **t**

e, t and e to t

Gottlob Frege

There are only two atomic things, truth values and individuals. All other things are created by function application.





λx.politician'(x) : $\langle \mathbf{e}, \mathbf{t} \rangle$
λx.lighter'(x) : $\langle \mathbf{e}, \mathbf{t} \rangle$
λx.silver'(x) : $\langle \mathbf{e}, \mathbf{t} \rangle$
λx.golden'(x) : $\langle \mathbf{e}, \mathbf{t} \rangle$

Syntactico-semantic composition

S
$[\![\textit{VP}]\!]([\![\textit{NP}]\!])$
$=$ smoke'$(i23)$

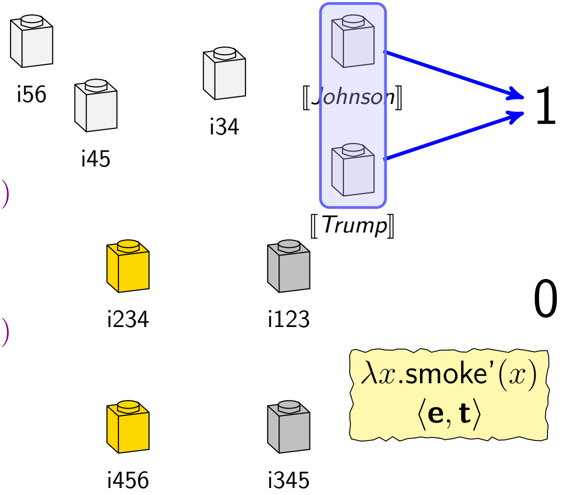
NP
$[\![\textit{NP}]\!]$
$= i23$

VP
$[\![\textit{VP}]\!]$
$= \lambda x.$smoke'(x)

N
$[\![\textit{Johnson}]\!]$
$=i23$

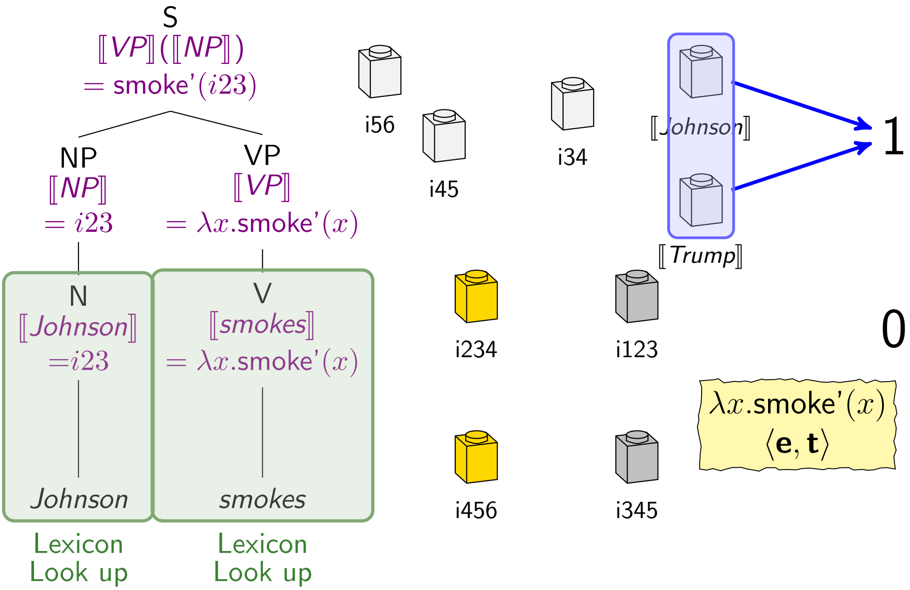
V
$[\![\textit{smokes}]\!]$
$= \lambda x.$smoke'(x)

Johnson *smokes*

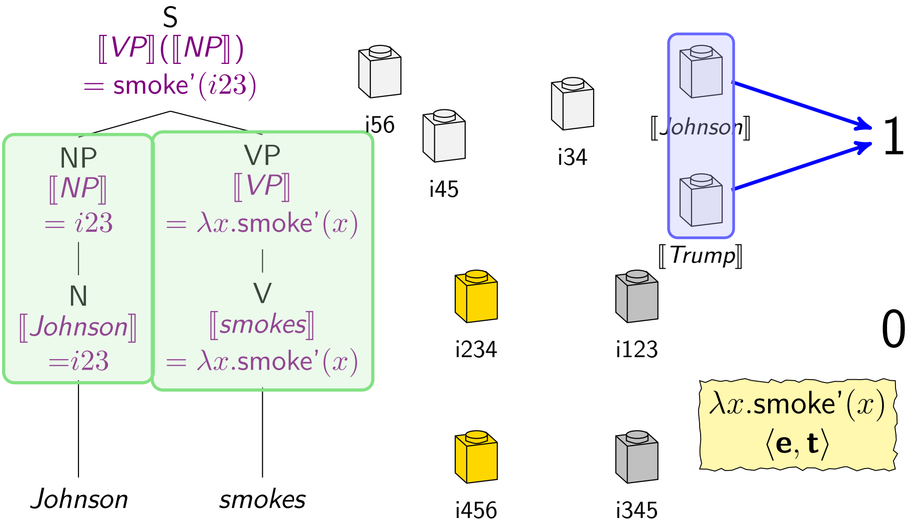


$\lambda x.$smoke'(x)
$\langle \mathbf{e}, \mathbf{t} \rangle$

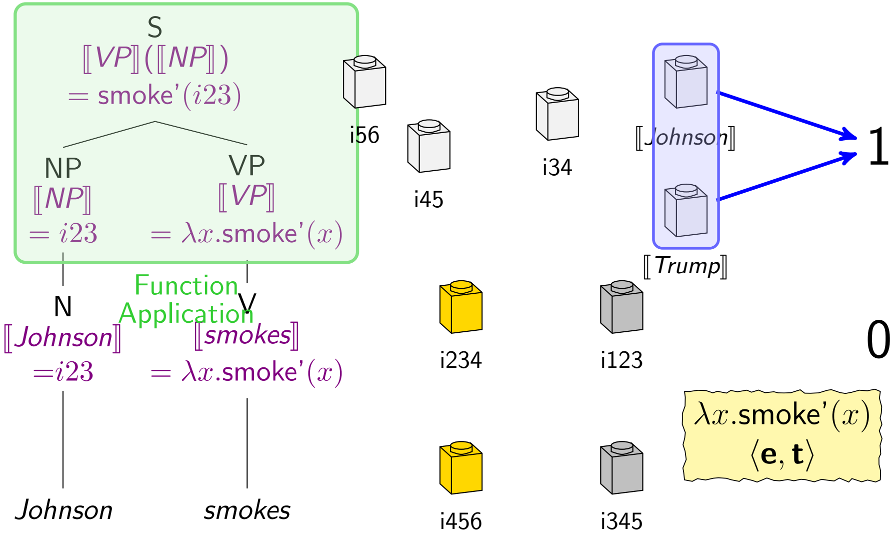
Syntactico-semantic composition



Syntactico-semantic composition



Syntactico-semantic composition

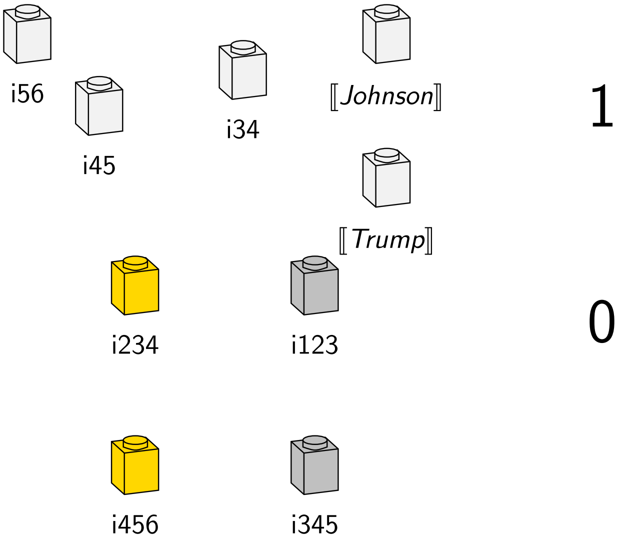


Compositional semantics

- ⟦*Johnson smokes*⟧ is not listed in the lexicon.
- But the interpretation of *Johnson smokes* can still be derived from its parts along with a syntactic analysis.
- Finite means make infinite interpretation possible.
- This is exactly the point of compositional semantics
- and note that we have remained precise
- This means we can use this thing we just built as a **meaning representation** of the kind we wanted in Lecture 1.

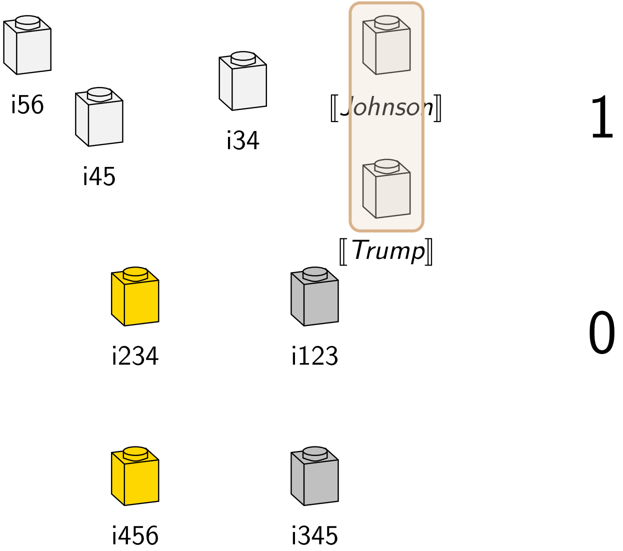
Transitive verbs

Johnson *kissed* *Trump*



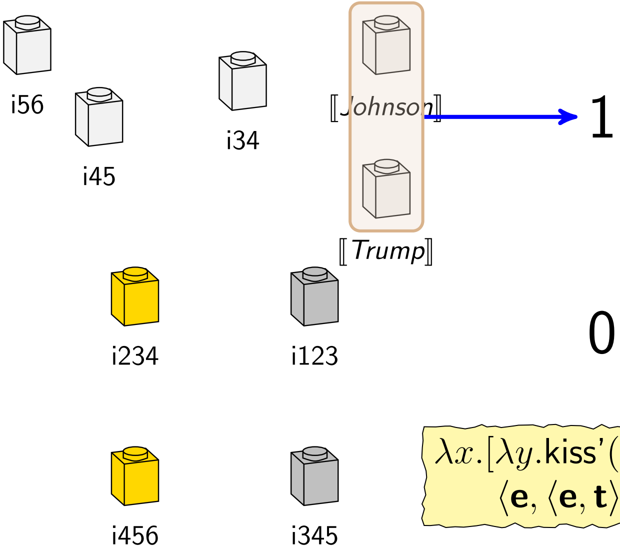
Transitive verbs

Johnson kissed Trump



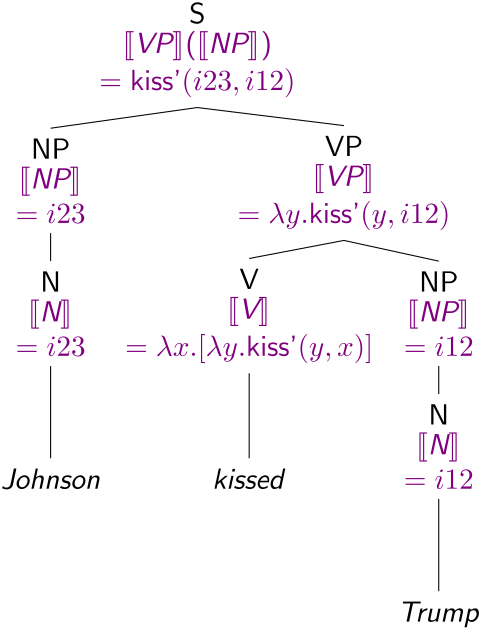
Transitive verbs

Johnson kissed Trump

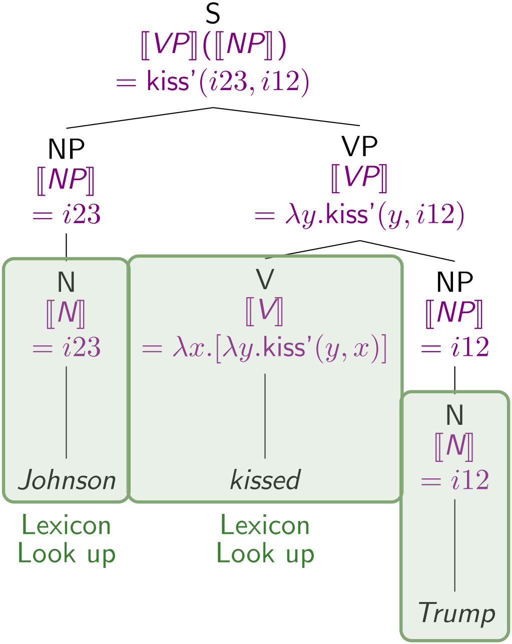


$$\lambda x.[\lambda y.\text{kiss'}(y,x)]$$
$$\langle \mathbf{e}, \langle \mathbf{e}, \mathbf{t} \rangle \rangle$$

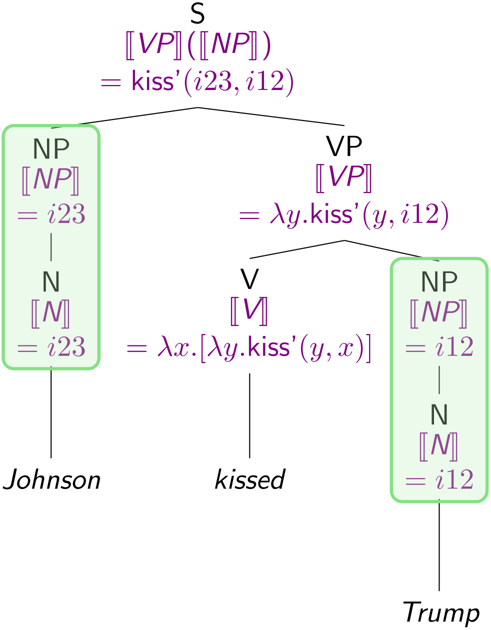
Syntactico-semantic composition



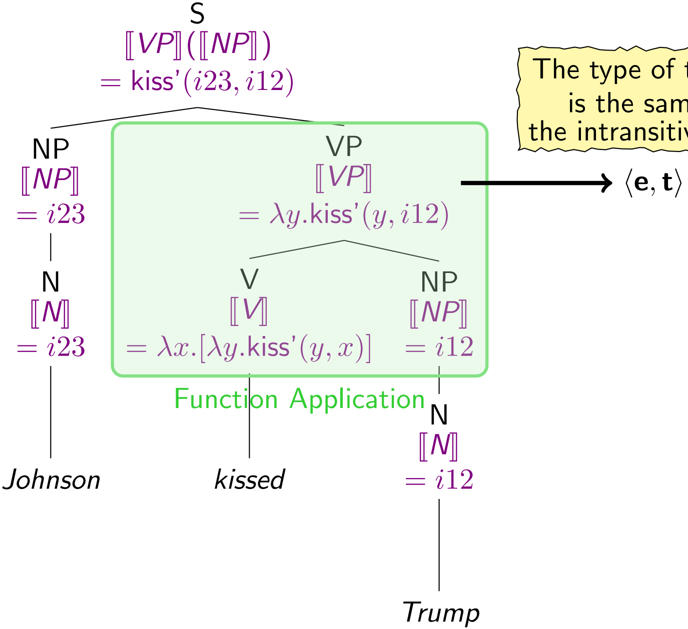
Syntactico-semantic composition



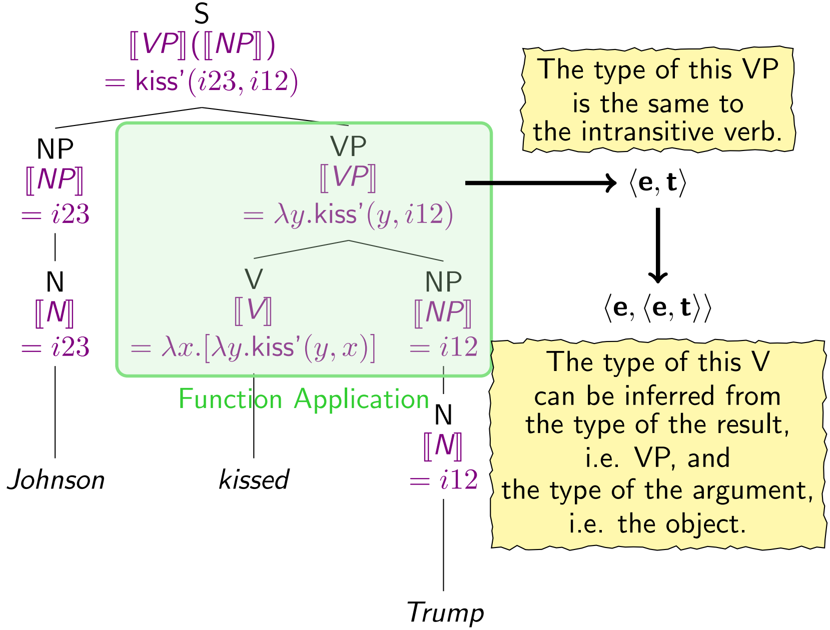
Syntactico-semantic composition



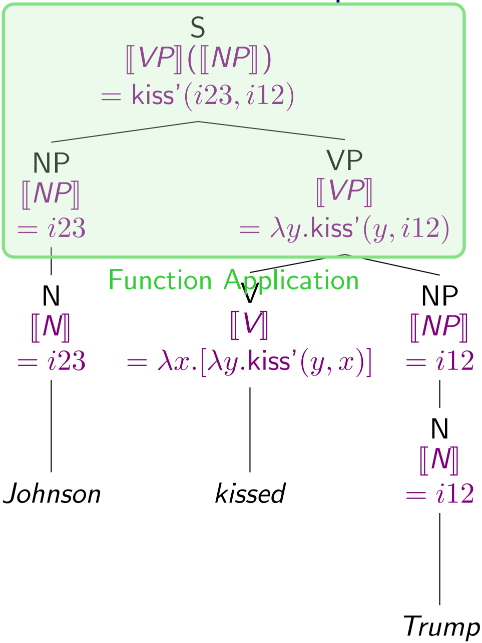
Syntactico-semantic composition



Syntactico-semantic composition



Syntactico-semantic composition



Lexicalised grammar

What should we know for a lexical entry?

- *kissed*
- syntactic category: V
- semantic type: $\langle \mathbf{e}, \langle \mathbf{e}, \mathbf{t} \rangle \rangle$
- semantic interpretation: $\lambda x.[\lambda y.\text{kiss'}(y, x)]$

Truth-Conditions

Meanings as truth conditions

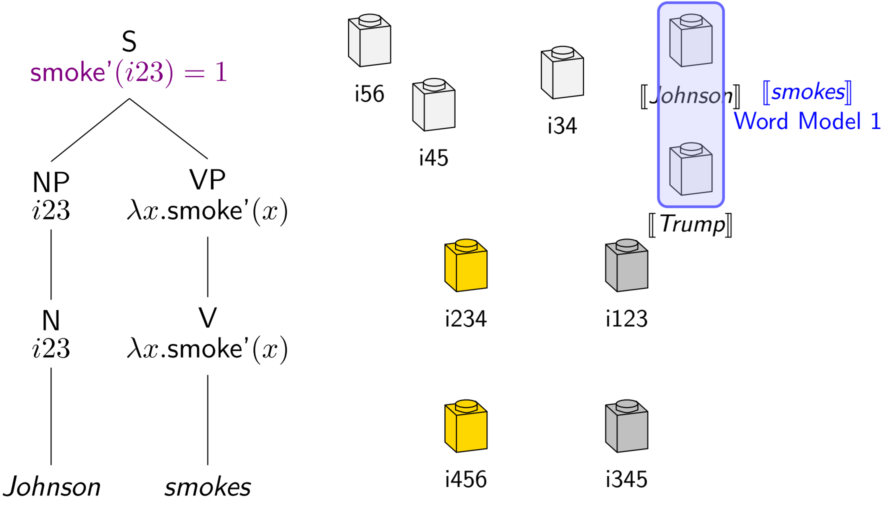
Ludwig Wittgenstein

> *To know the meaning of a sentence is to know how the world would have to be for the sentence to be true.*

The meaning of words and sentence parts is their contribution to the truth-conditions of the whole sentence.

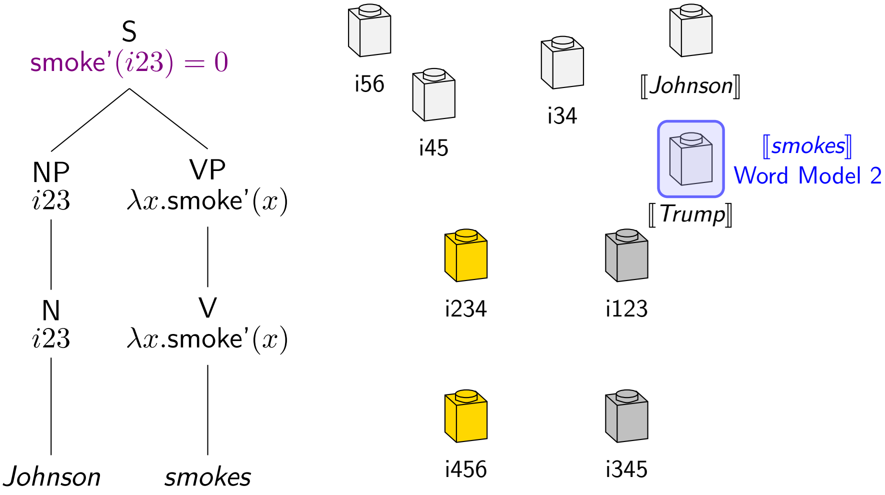
The truth-conditional tradition

Consider three different word models: Different people smoke



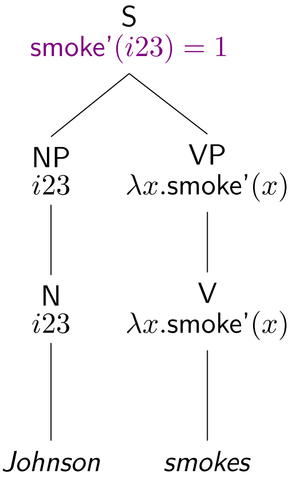
The truth-conditional tradition

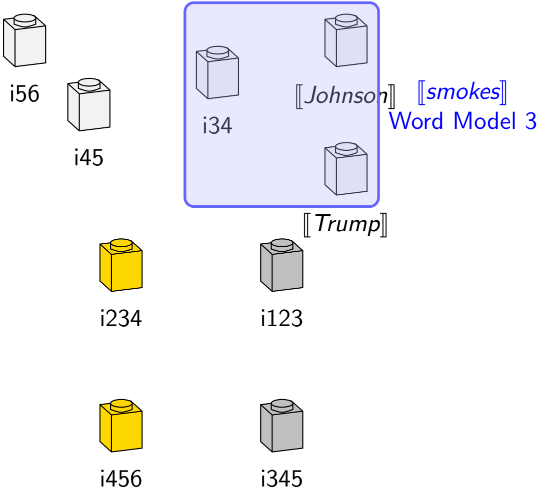
Consider three different word models: Different people smoke



The truth-conditional tradition

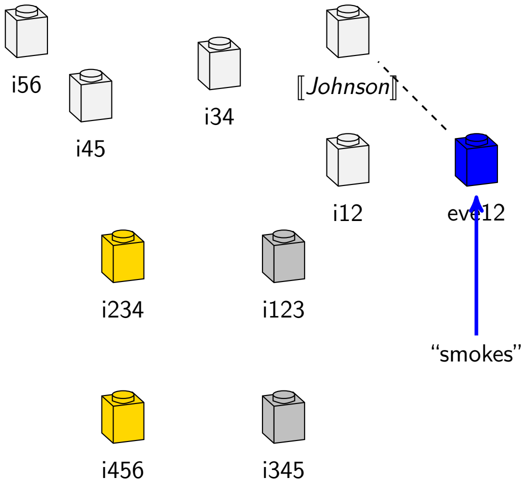
Consider three different word models: Different people smoke



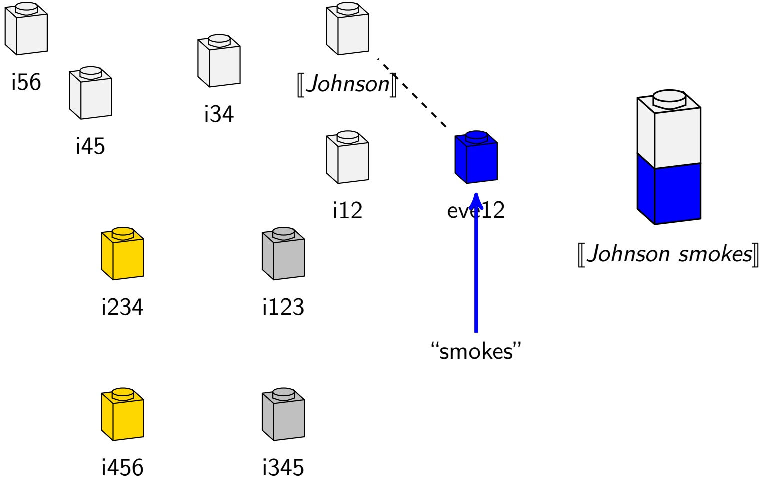


First-Order Predicate Logic (FOPL)

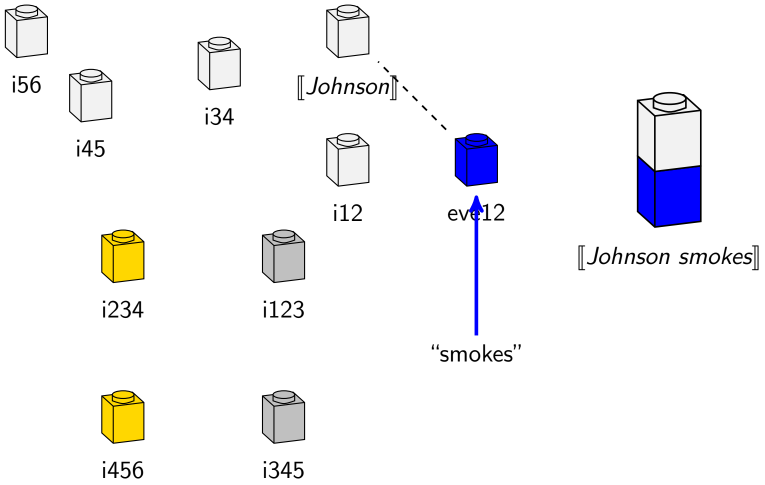
Davidsonian semantics: Adding event variables



Davidsonian semantics: Adding event variables

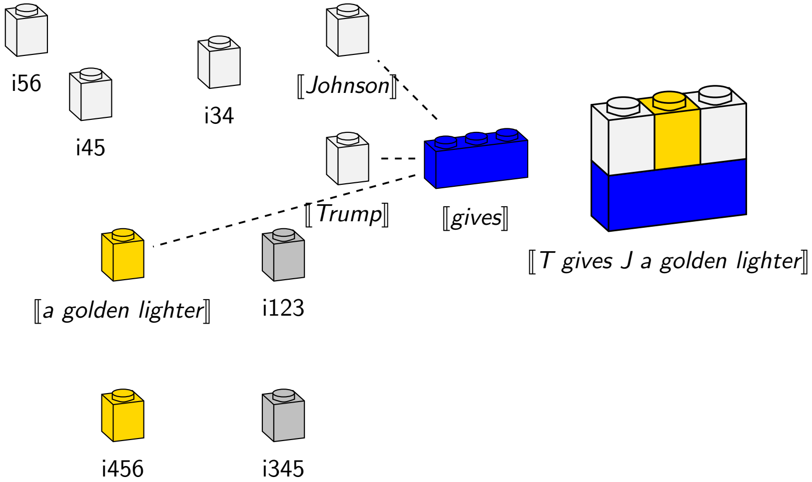


Davidsonian semantics: Adding event variables

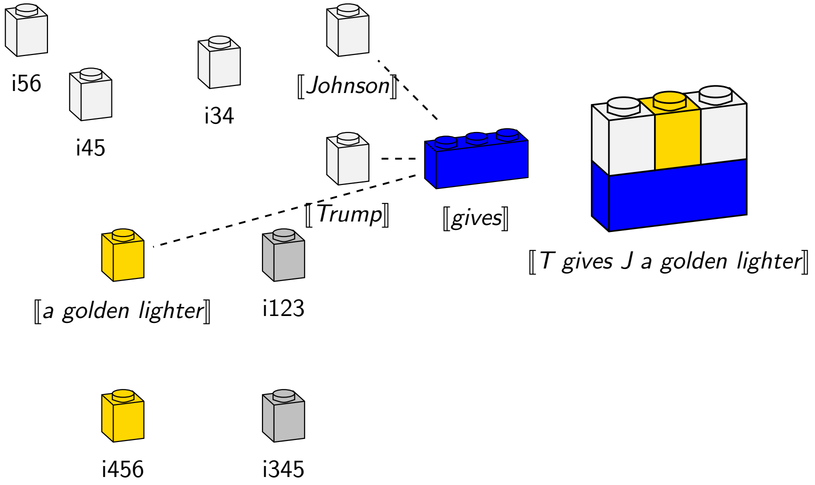


What is the type of ⟦*gives*⟧? **e** — individual.

Ditransitive verb

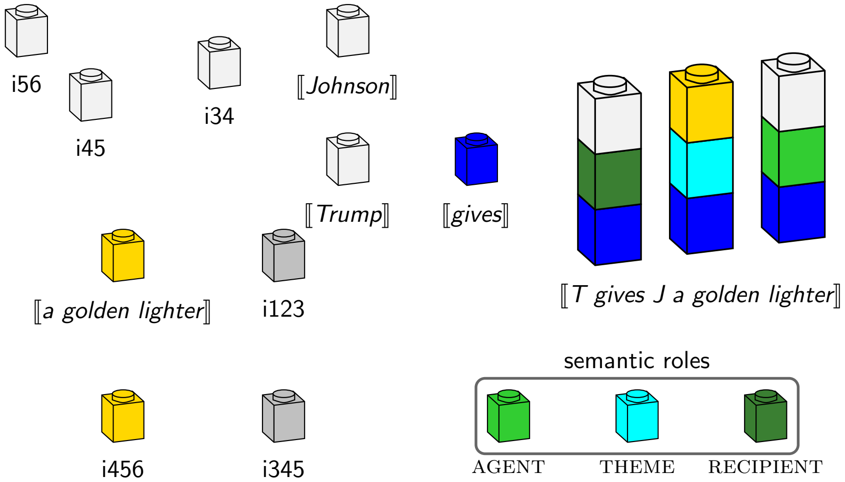


Ditransitive verb

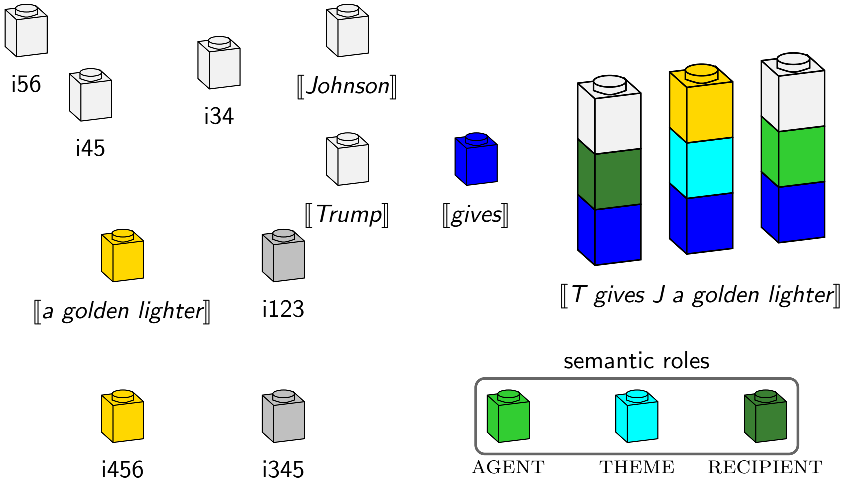


What is the type of ⟦*gives*⟧? **e** — individual.

Neo-Davidsonian semantics: Further decomposition



Neo-Davidsonian semantics: Further decomposition



Further decomposition of the event structure

Lexicalised vs unlexicalised

Before Davidson

- $[\![\textit{gives}]\!]([\![\textit{Trump}]\!], [\![\textit{Johnson}]\!], [\![\textit{a golden lighter}]\!])$
- $\lambda x.[\lambda y.[\lambda z.\text{give'}(z, x, y)]]$
- $\langle \mathbf{e}, \langle \mathbf{e}, \langle \mathbf{e}, \mathbf{t} \rangle \rangle \rangle$
- Lexicalised: the lexical entry contains rich information of arguments.

Davidsonian

- $[\![\textit{gives}]\!](e, [\![\textit{Trump}]\!], [\![\textit{Johnson}]\!], [\![\textit{a golden lighter}]\!])$
- Lexicalised

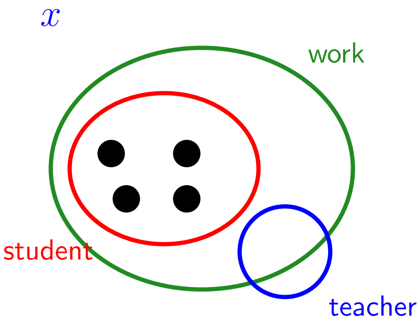
Neo-Davidsonian

- $[\![\textit{gives}]\!](e) \wedge \textsc{agent}(e, [\![\textit{Trump}]\!]) \wedge \textsc{recipient}(e, [\![\textit{Johnson}]\!]) \wedge \textsc{theme}(e, [\![\textit{a golden lighter}]\!])$
- Modularisation of information
- Unlexicalised: the lexical entry doesn't need to know argument structure.

First-order predicate logic

- What is $[\![$*every student smokes*$]\!]$? $\forall x(\text{student'}(x) \rightarrow \text{smoke'}(x))$
- What is $[\![$*some students smoke*$]\!]$? $\exists x(\text{student'}(x) \wedge \text{smoke'}(x))$

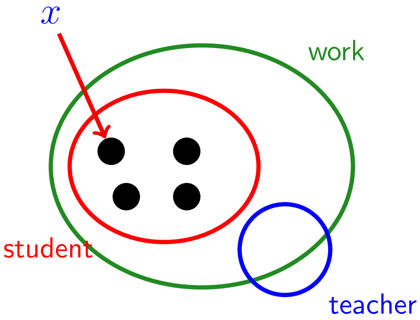
M1



First-order predicate logic

- What is $[\![$*every student smokes*$]\!]$? $\forall x(\text{student'}(x) \rightarrow \text{smoke'}(x))$
- What is $[\![$*some students smoke*$]\!]$? $\exists x(\text{student'}(x) \wedge \text{smoke'}(x))$

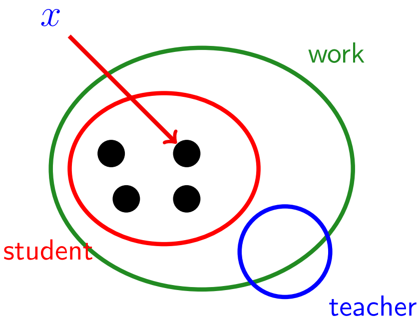
M1



First-order predicate logic

- What is ⟦*every student smokes*⟧? $\forall x(\text{student'}(x) \rightarrow \text{smoke'}(x))$
- What is ⟦*some students smoke*⟧? $\exists x(\text{student'}(x) \wedge \text{smoke'}(x))$

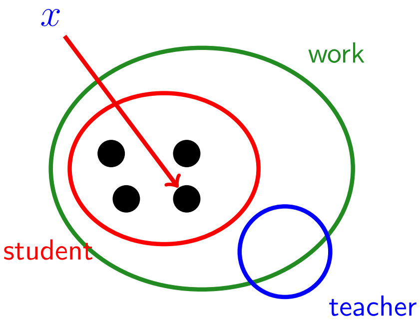
M1



First-order predicate logic

- What is ⟦*every student smokes*⟧? $\forall x(\text{student'}(x) \rightarrow \text{smoke'}(x))$
- What is ⟦*some students smoke*⟧? $\exists x(\text{student'}(x) \wedge \text{smoke'}(x))$

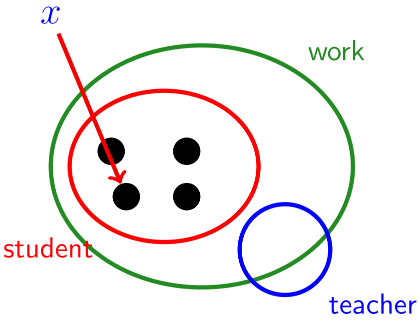
M1



First-order predicate logic

- What is ⟦*every student smokes*⟧? $\forall x(\text{student'}(x) \rightarrow \text{smoke'}(x))$
- What is ⟦*some students smoke*⟧? $\exists x(\text{student'}(x) \wedge \text{smoke'}(x))$

M1



Truth of these statements in our world model?

In the world where *Trump gave Johnson a golden lighter* is true, which one of the following is true?

- Johnson gave Trump a lighter
- Trump gave Johnson a silver lighter
- Johnson was given a lighter

Readings

- Heim and Kratzer. *Semantics in Generative Grammar*. Chapter 1–3.