Semantics

Introduction to Linguistics

Dustin A. Chacón

Lexical meaning

⟦dog⟧ = ?

Using tools from logic, semanticists characterize the meaning of sentences as their **truth conditions**

⟦Dale ate the pie⟧ = **true** iff Dale ate the pie

Semanticists typically focus on the kinds of meaning relations between sentences

- *P* **entails** Q (*P* → Q) if the truth of *P* guarantees the truth of Q:

Dale eats pie entails:

(1) Dale ate something
(2) Somebody ate pie
(3) Somebody ate something
(4) Somebody did something
(5) Dale did something ...

Dustin Alfonso Chacón

Entailments are distinct from **implicatures**

Dale ate the pie implies…

- that there’s no more dessert left
- that Dale might not be hungry anymore
- that we should punish Dale for eating the food that belonged to the police station

Kinds of Meaning

Dustin Alfonso Chacón

Introduction to Linguistics

Distinct kinds of meanings are sensitive to different diagnostics:

Implicatures can be **cancelled**, but entailments cannot

#Dale ate the pie, but Dale didn’t eat anything.
Dale ate the pie, but he’s still hungry

Meanings also have **presuppositions**, which are conditions that are required for determining the truth conditions of a sentence

⟦The leader of the United States is a Republican⟧ = **true**

⟦The leader of Spain is a Republican⟧ = **false**

⟦The leader of Atlantis is a Republican⟧ = **?**

Definite NPs presuppose the existence of the referent:
The leader of <place NP> presupposes that there is a president of that place

Presuppositions **project** under negation, conditionals, etc.:

Dale ate the pie → *Dale ate something*

Dale didn't eat the pie ↛ *Dale ate something*

If Dale ate the pie, then he would be happy ↛ *Dale ate something*

The leader of Spain is a Republican → *There is a leader of Spain*
The leader of Spain isn't a Republican → *There is a leader of Spain*
If the leader of Spain is a Republican, then he would pass this law
→ *There is a leader of Spain*

The truth conditions of a sentence can be denied, but not the presupposed material

The waitress from the RR Diner made the cherry pie.

No, the huckleberry pie.

**No, from the roadhouse.*

Compositionality

Dustin Alfonso Chacón

Introduction to Linguistics

The entailments of ⟦ [$_S$ [$_{NP}$ Dale] [$_{VP}$ ate the pie]] ⟧ overlap with the entailments of ⟦ [$_S$ [$_{NP}$ Harry] [$_{VP}$ ate the pie]] ⟧

Dale ate the pie →
- Dale ate something
- Someone ate pie
- Someone ate something
- Someone did something

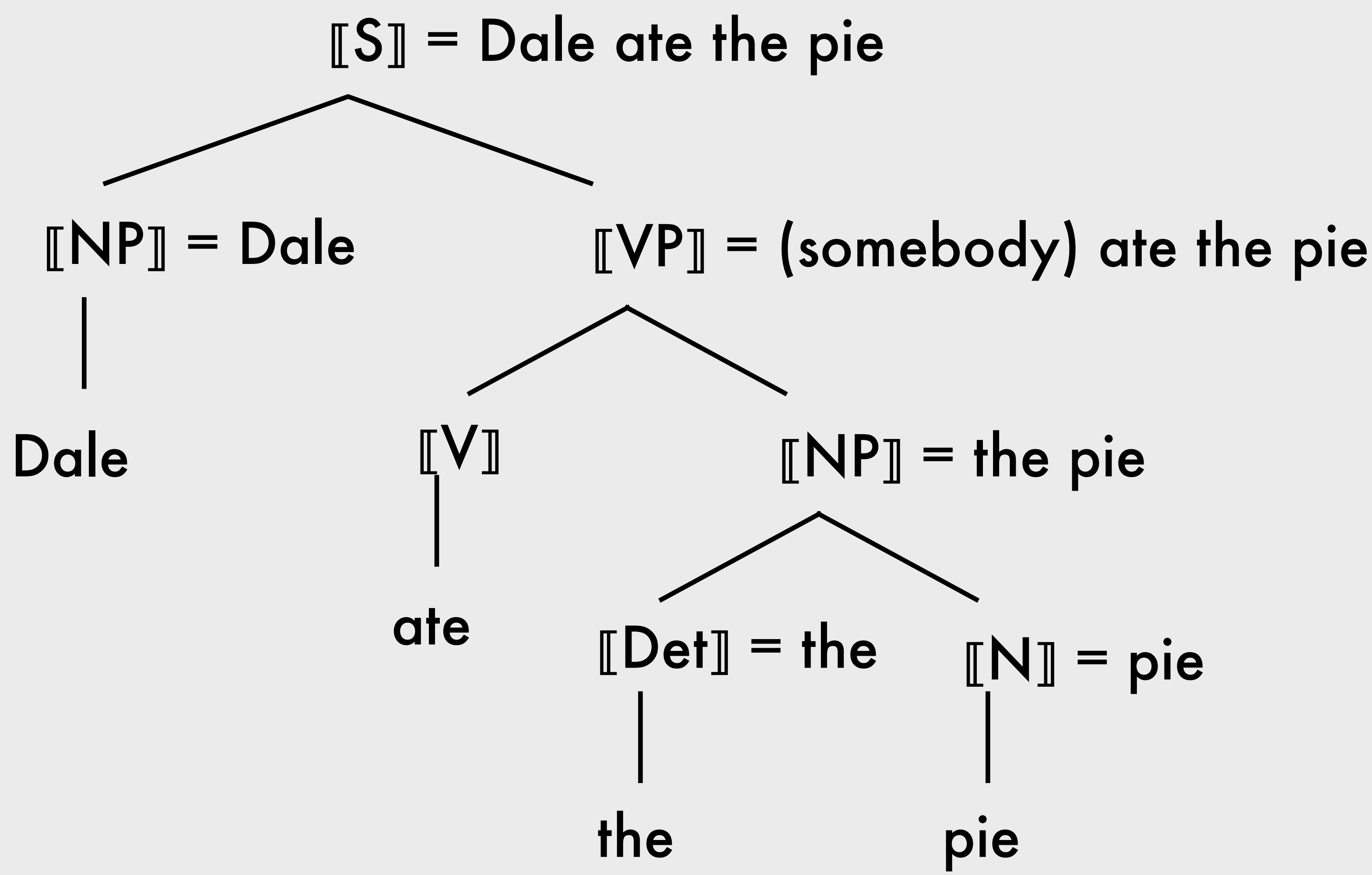
Harry ate the pie →
- Harry ate something
- Someone ate pie
- Someone ate something
- Someone did something

Compositionality

Dustin Alfonso Chacón

Introduction to Linguistics

Each word has its own semantic value, and the semantic value of phrases and sentences are deterministically composed of their constituent parts



Modeling Truth Conditions

Truth conditions are typically represented using **first-order predicate logic**

Let D_e be the domain of **entities:**

D_e = {**dale, ernie, dustin, chomsky,** ...}

Let D_t be the domain of **truth values:**

D_t = {**true, false**}

The semantic value of sentences are truth values:

⟦Dustin teaches linguistics⟧ = **true**

⟦Ernie teaches philosophy⟧ = **false**

The semantic value of NPs are entities:

⟦Dale⟧ = **dale**

⟦the pie⟧ = **the pie**

(we'll ignore how the meaning of *the pie* is composed)

Modeling Truth Conditions

Dustin Alfonso Chacón

Introduction to Linguistics

Intuitively, we want the meaning of *Ernie is a cat* to be **true** if the entity **ernie** is a cat

$[\![$cat$]\!]$ = {**ernie, garfield, heathcliff, ...**}

$[\![$cat$]\!]$ is a subset ($\subset$) of D_e, meaning that every member in $[\![$cat$]\!]$ is also in D_e

This gives us a way to model entailments:

If everything that's a cat is in the set of entities, then
if Garfield is a cat, then Garfield is an entity.

$[\![$cat$]\!] \subset D_e$, and
garfield $\in [\![$cat$]\!]$
then **garfield** $\in D_e$

We can also represent **garfield** $\in [\![$cat$]\!]$ as
cat(**garfield**)

The meaning of sentences like *Ernie is a cat* checks to see whether the entity denoted by the subject (**ernie**) is in the set of cats

⟦Ernie is a cat⟧ = cat(**ernie**)
⟦Garfield is a cat⟧ = cat(**garfield**)
⟦Heathcliff is a cat⟧ = cat(**heathcliff**)

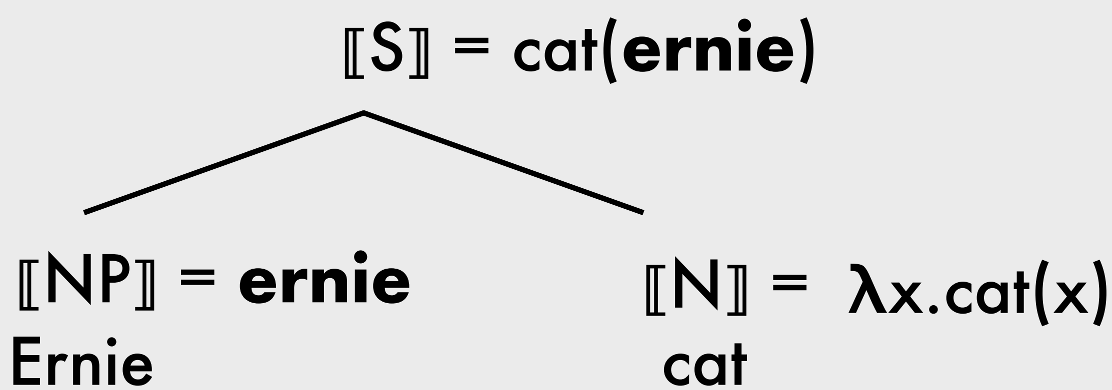
We can abstract away the meaning of *cat* as a **function** from entities to truth values; We feed it an entity, and get **true** if that entity is in the set of cats, **false** otherwise

⟦cat⟧ = λx.cat(x)	type **<e,t>** (function from entities to truth values)
⟦ernie⟧ = ernie	type **<e>** (entity)
⟦ernie is a cat⟧ = cat(ernie)	type **<t>** (truth value)

We compose the meaning of ⟦ernie is a cat⟧ through **function application**

⟦cat⟧ is a function that requires an **argument**

The syntactic representation supplies the function its argument



Modeling Truth Conditions

Dustin Alfonso Chacón

Introduction to Linguistics

Subset relations are useful for modeling the entailment patterns of adjectives

Ernie is a grey cat → Ernie is a cat; Ernie is grey

⟦cat⟧ = **{ernie, garfield, heathcliff, …}**; λx.cat(x)
⟦grey⟧ = **{ernie**, **dumbo,** …}; λx.grey(x)

Let functions of type <e,t> compose to make a new function of type <e,t>, as below:

⟦grey cat⟧ = λx.grey(x) & cat(x)
-> this is **true** if the entity is in both the set of grey things and the set of cats, or,
That it's a member of the **intersection** of the two sets: ⟦grey⟧ ∩ ⟦cat⟧

⟦ernie is a grey cat⟧ = grey(**ernie**) & cat(**ernie**)
-> this captures the entailment relationships, because if **ernie** is both sets, then he
he obviously needs to be in each of the two sets

Once we add **functions** to our semantic type inventory, we can have functions from functions. In set talk…

⟦like⟧ = {<dale, audrey>, <dustin,ernie>, …}

The meaning of like is the set of **ordered pairs** <x,y> such that x loves y
(you can think of this as the set of "liking relations")

In function talk…

⟦like⟧ = λx.λy.like(y,x)

"The semantic value of *like* is the function from entity *x* to the function from entity *y* to **true** iff *y* likes *x*"

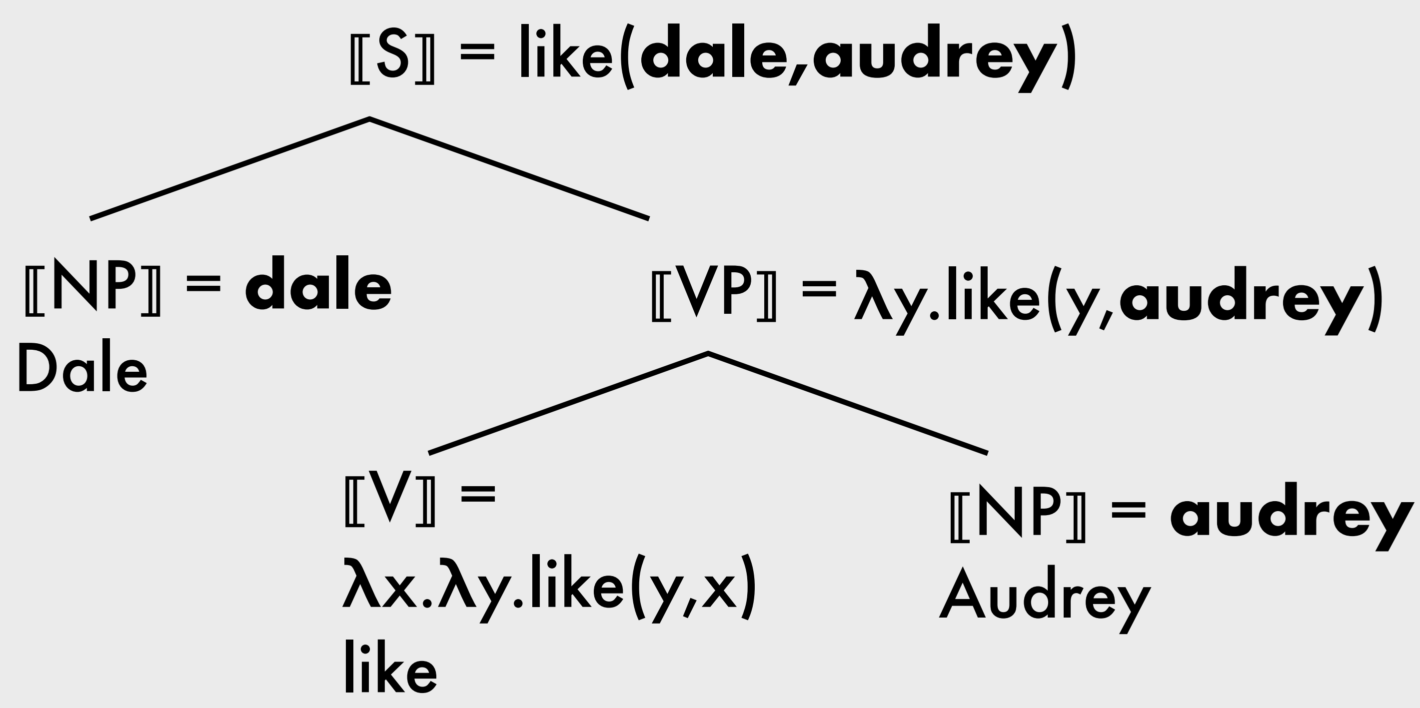
The semantic type of *like* is <e,<e,t>> – first, we *like* eats an entity and returns a function, which in turn eats an entity and returns a truth value

Modeling Truth Conditions

Dustin Alfonso Chacón

Introduction to Linguistics

With higher types like <e,<e,t>>, we can see how the syntax “guides" function application to compute the meaning of a whole sentence. Notice that this theory gives us a meaning for the VP node – λy.like(y,**audrey**)



We also use **logical operators:**

$\exists x[P(x)]$

there is at least one entity that satisfies the condition $\lambda x.P(x)$

"For some *x*, ..."

$\forall x[P(x)]$

every entity satisfies the condition $\lambda x.P(x)$

"For all *x*, ... "

$\neg P$

flip the truth value; **true** becomes **false**; **false** becomes **true**

"It is not the case that *P*"

Modeling Truth Conditions

Dustin Alfonso Chacón

Introduction to Linguistics

Ernie is a grey cat entails *Something is a grey cat*

⟦ernie is a grey cat⟧ = grey(**ernie**) & cat(**ernie**)

⟦something is a grey cat⟧ = ∃x[grey(x) & cat(x)]

⟦something is grey⟧ = ∃x[grey(x)]

⟦something is cat⟧ = ∃x[cat(x)]

Ernie (*is a*) *grey cat* entails *some cat* (*is*) *grey* (we continue to ignore *is* and *a*)

⟦ernie⟧ = **ernie** <e>
⟦grey⟧ = λx.grey(x) <e,t>
⟦cat⟧ = λx.cat(x) <e,t>

⟦S⟧ = grey(**ernie**) & cat(**ernie**)

⟦NP⟧ = **ernie**
Ernie

⟦NP⟧ = λx.grey(x) & cat(x)

⟦Adj⟧ =
λx.grey(x)
grey

⟦N⟧ =
λx.cat(x)
cat

← two <e,t> compose this way

Modeling Truth Conditions

Dustin Alfonso Chacón

Introduction to Linguistics

Ernie (*is a*) *grey cat* entails *some cat* (*is*) *grey* (we continue to ignore *is* and *a*)

⟦ernie⟧ = **ernie** <e>
⟦grey⟧ = λx.grey(x) <e,t>
⟦cat⟧ = λx.cat(x) <e,t>
⟦some⟧ = **?**

⟦S⟧ = ∃x[cat(x) & grey(x)]

⟦NP⟧ = **?**

⟦Adj⟧ = λx.grey(x)

⟦D⟧ =
?
some

⟦N⟧ =
λx.cat(x)
cat

Ernie (*is a*) *grey cat* entails *some cat* (*is*) *grey* (we continue to ignore *is* and *a*)

$[\![\text{ernie}]\!] = \textbf{ernie}$ <e>

$[\![\text{grey}]\!] = \lambda x.\text{grey}(x)$ <e,t>

$[\![\text{cat}]\!] = \lambda x.\text{cat}(x)$ <e,t>

$[\![\text{some}]\!] = \lambda P.\lambda Q.\exists x[P(x)\ \&\ Q(x)]$

type: <<e,t>,<<e,t>,t>>

$[\![\text{S}]\!] = \exists x[\text{cat}(x)\ \&\ \text{grey}(x)]$

$[\![\text{NP}]\!] = \lambda Q.\exists x[\text{cat}(x)\ \&\ Q(x)]$

$[\![\text{Adj}]\!] = \lambda x.\text{grey}(x)$

$[\![\text{D}]\!] =$

$\lambda P.\lambda Q.\exists x[P(x)\ \&\ Q(x)]$

some

$[\![\text{N}]\!] =$

$\lambda x.\text{cat}(x)$

cat