Scope in an incremental context

Lecture 2: formal and theoretical considerations

Asad Sayeed

University of Gothenburg

Part 1: incremental syntax and semntics

The age-old question: representing sentences.

Representing sentences

Brutus stabs Caesar.

Representing sentences

"Brutus stabs Caesar."

- How to represent this?
- Start with the predicate: stab.
- Standard first-order predicate calculus representation:
  - stab(Brutus, Caesar)

That was easy. Let's make it more complicated.

Representing sentences

Brutus stabs Caesar with a knife.

Representing sentences

"Brutus stabs Caesar with a knife."

- The knife is a participant in the action.
- Perhaps...make it an argument of the predicate?
- Possible representation:
  - stab(Brutus, Caesar, knife)
  - ...

Representing sentences

Brutus stabs Caesar in the agora.

Representing sentences

"Brutus stabs Caesar in the agora."

- The agora is a participant in the action.
- Perhaps...make it a member of the predicate?
- Possible representation:
  - stab(Brutus, Caesar, agora)
  - . . .
- But an agora is not a participant in the same way that a knife is.
  - Knife - instrument
  - Agora - location
- The predicate arguments are now ambiguous.

Brutus stabs Caesar with a knife in the agora.

Representing sentences

"Brutus stabs Caesar with a knife in the agora."

- The knife is a participant in the action.
- The agora is a participant in the action.
- Possible representation:
  - stab(Brutus, Caesar, agora, knife) ?
  - ...
- Uh-oh, we require a different arity for all these optional adjuncts.
- Or we need a more flexible way to represent predicates.

Splitting it up

- “Brutus stabs Caesar with a knife in the agora.”
- Possible representation:
  - stabs(Brutus, Caesar) & with(knife) & in(agora)
- Much better now: can have arbitrary adjuncts.

Representing sentences

Brutus stabs Caesar with a knife in the agora and twists it hard.

Representing sentences

- "Brutus stabs Caesar with a knife in the agora and twists it hard."
- Possible representation:
  - stabs(Brutus, Caesar) & with(knife) & in(agora) & twists(Brutus, knife) & hard
- What does the with-predicate apply to?
- What does the 0-arity hard-predicate apply to?

Davidson's problem

Standard predicate calculus representations

- They do not allow us to refer to predicates.
- But need to refer to predicates to describe complex actions.
- Language is very flexible, predicates can have variable #s of arguments without necessarily having strongly different meanings. ("Pass (me) the salt.")
- Adverbial modifiers, relative clauses, oh my...

A rough analogy

Consider the noun phrase: *some clever driver in America*

- Translate this into a possible logical form:

$\exists x$ driver(x) & clever(x) & location(x, America)

- Descriptors of the entity tied together by an existentially-quantified variable x.

Consider the sentence: *Bob drives cleverly in America*

- Now there is an act of driving, rather than a driver.
- Can we still tie together the descriptors of driving?
- (The analogy doesn't quite work because a "former driver" is not a driver who is former. . . .)

Davidson's solution

Davidson 1989 quoted in Maienborn 2010

Adverbial modification is thus seen to be logically on a par with adjectival modification: what adverbial clauses modify is not verbs but the events that certain verbs introduce.

- A verb predicate like "stab" is actually a description of an event.
- We need to represent this event in the semantics: an event variable.
- We don't know what it is beforehand, so we existentially quantify it.

Brutus stabs Caesar with a knife in the agora

=

$\exists e$ **stabs(**e**, Brutus, Caesar) & with(**e**, knife) & in(**e**, agora)**

Now we have a flexible way to talk about the event of stabbing.

Davidson's solution

Brutus stabs Caesar with a knife in the agora and twists it hard.

=

$\exists e_1$ **stabs(**e_1**, Brutus, Caesar) & with(**e_1**, knife) & in(**e_1**, agora) &** $\exists e_2$ **twists(**e_2**, Brutus, knife) & hard(**e_2**)**

Now we have a flexible way to talk about the event of stabbing and twisting hard.

Alternative ways?

They're actually surprisingly hard to find.

- However, consider Discourse Representation Theory (DRT).

From the "Handbook of Philosophical Logic"

Luigi was writing to the Department Chairman. He had applied for the job without much hope.

- Can view multiple sentences as one discourse:

$n\ l\ c\ t_1\ s_1\ j\ t_2\ e_2$
Luigi(l)
"the Department Chairman"(c)
$t_1 \prec n$
$t_1 \subseteq s_1$
s_1: PROG(^e. e: write-to(l,c))
"the job"(j)
$t_2 \prec t_1$
$e_2 \subseteq t_2$
e_2: apply-for(l,j)
"without-much-hope"(e_2)

- Events even show up here!

Something that is Neo.

Q: But what makes it NEO-Davidsonian?

A: It got an update by Parsons (1990). But this requires a bit of background in semantics.

Thematic roles

Active voice

Brutus stabbed Caesar.

Brutus is the subject of the sentence.
Caesar is the object of the sentence.

Thematic roles

Passive voice

Caesar was stabbed (by Brutus).

Caesar is the subject of the sentence.
Brutus is the object of the optional preposition *by*.

Thematic roles

Active voice

Brutus stabbed Caesar.

Brutus is the one doing the stabbing.
Caesar is the one getting stabbed.

Thematic roles

Passive voice

Caesar was stabbed (by Brutus).

Brutus is the one doing the stabbing.
Caesar is the one getting stabbed.

Thematic roles

Passive voice

Caesar was stabbed (by Brutus).

Brutus is the one doing the stabbing.
Caesar is the one getting stabbed.

Grammatical roles have switched, but semantic roles have not!

Thematic roles

- There is a clear separation between *syntactic* arguments and *semantic* arguments.
- *Semantic arguments* $\Rightarrow$ thematic roles aka θ-roles.
- A POSSIBLE inventory of roles from Parsons (1995). Given an event e and an entity x:
  - e is by x - Agent
  - x experiences e - Experiencer
  - e is of x - Theme
  - e is from x - Source
  - ...
- The specific inventory is quite controversial but Agent and Theme are minimal members of the set.

Thematic roles

Caesar was stabbed by Brutus.

- Caesar is the Theme (or "Patient").
- Brutus is the Agent.

Caesar was stabbed by Brutus.

A Davidsonian representation:

- $\exists e$ stab(e, Brutus, Caesar)

Caesar was stabbed (by Brutus).

But there's no way to represent the fact that the theme is optional, without an ambiguous lexical entry.

Caesar was stabbed (by Brutus).

Solution: break predicate arguments into thematic roles.

- $\exists e$ stab(e) (& Agent(e, Brutus)) & Patient(e, Caesar)
- The stab-predicate now only has the event argument.
- Optionality of agent in the passive is fully accomodated.

Problems?

"Arcane" semantic issues mentioned by Parsons (1995).

- "I sold you a car for \$5." – what is the Theme here, "a car" or 5?
  - The \$5 is changing hands here, so it is affected by the event.
- Ambiguous prepositions: "I sold a car for Mary for \$5."
  - Two uses of "for" – not exactly a semantic problem for us, but Parsons thought it was.
- "Mary fed her baby" – why is "baby" not an agent? It is feeding!
  - (Consider "riechen" in German. "Es riecht" (it smells) vs "Ich rieche es" (I'm smelling it).

What about non-eventive assertions?

- "Mary is sick."
- Need a state variable rather than an event variable.
  - $\exists s$ being-sick(s) & In(s, Mary)
  - Read: "Mary is in a state of being sick."

Other issues

- "Caesar was stabbed."
  - But what if you really wanted to include the agent?
  - No problem, quantify: $\exists x \exists e$ stab(e) & Agent(e, x) & Theme(e, Caesar)
  - Entirely optional to do it this way. (Why would you?)
- "Destruction" vs. "destroy" – do nouns have events?
  - "the destruction of the city by God"
- Argument/adjunct distinctions.

Q: But this is a seminar on incremental syntax/semantics. What does neo-Davidsonian semantics have to do with that?

From Allie Brosh's famous blog.

A: A lot.

From Allie Brosh's famous blog.

A: A lot.

- Broken down the representation into atomic components of fixed arity.
- Provided way to connect them via the event variable.

An incremental parse:

|| **Brutus often stabs Caesar.**

=

An incremental parse:

Brutus $\|$ **often stabs Caesar.**

$$=$$

$$\exists e\ \mathbf{Agent}(e, \mathbf{Brutus})$$

An incremental parse:

Brutus often || stabs Caesar.

$$=$$

$$\exists e\ \mathbf{Agent}(e, \mathbf{Brutus})\ \&\ \mathbf{often}(e)$$

An incremental parse:

Brutus often stabs || Caesar.

=

$\exists e$ **Agent(**e**, Brutus) & often(**e**) & stabs(**e**)**

An incremental parse:

Brutus often stabs Caesar. $\|$

$$=$$

$\exists e$ **Agent(**e**, Brutus) & often(**e**) & stabs(**e**) & Theme(**e**, Caesar)**

Incremental parsing

- Previous example: big assumption that the syntactic parser will cooperate.
- The challenge: designing the syntax and the lexicon to work with the semantics.
- But: Neo-Davidsonian approach at least simplifies predicate representation.
- Low recursion: inference rules not required to go back and edit deeply embedded arguments.

Argument/adjunct representation issues

"Brutus stabbed Caesar violently yesterday."

- Possible neo-Davidsonian representation:
  - $\exists e$ stab(e) & Agent(e, Brutus) & Theme(e, Caesar) & violently(e) & yesterday(e)
- Get rid of the last two
  - $\exists e$ stab(e) & Agent(e, Brutus) & Theme(e, Caesar)
  - Still corresponds to grammatical sentence (ignoring tense): "Brutus stabbed Caesar".
  - Doesn't change truth condition, they're adjuncts.

But getting rid of Brutus or Caesar definitely does. They're arguments.

Argument/adjunct representation issues

Implications for the design of the parsing algorithm

- Argument-event relation mediated through thematic role.
- Verb adjuncts: direct relation.
- Syntax already usually aware of optional adjuncts—need principled way to translate to semantics.

Minimal recursion semantics (MRS)

Practical Davidsonian representation (Copestake, 2005 etc).

- "Elementary predications" (EPs) – Davidsonian conjuncts.
- Hooks and slots (roles and fillers) are used for semantic composition through equations (more complicated than this, of course).
- Can be attached to various grammar formalisms
- Underspecification! (we'll deal with this next week too)

the fat cat sat on a mat

MRS representation:

$l0$: _the_q$(x0, h01, h02), l1$: _fat_j$(x1), l2$: _cat_n$(x2), l3$: _sit_v_1$(e3, x3), l4$: _on_p$(e4, e41, x4)$,
$l5$: _a_q$(x5, h51, h52), l6$: _mat_n_1$(x6)$,
$h01 =_q l1, h51 =_q l6$
$x0 = x1 = x2 = x3, e3 = e41, x4 = x5 = x6, l1 = l2, l3 = l4$

Minimal recursion semantics (MRS)

To achieve further underspecification, can be Neo-Davidsonianized (RMRS):

RMRS equivalent to the MRS above:

$l0$: $a0$: _the_q($x0$), $l0$: $a0$: RSTR($h01$), $l0$: $a0$: BODY($h02$), $l1$: $a1$: _fat_j($x1$), $l2$: $a2$: _cat_n($x2$),
$l3$: $a3$: _sit_v_1($e3$), $l3$: $a3$: ARG1($x31$), $l4$: $a4$: _on_p($e4, e41, x4$), $l4$: $a4$: ARG1($e41$), $l4$: $a4$: ARG2($x4$),
$l5$: $a5$: _a_q($x5$), $l5$: $a5$: RSTR($h51$), $l5$: $a5$: BODY($h52$), $l6$: $a6$: _mat_n_1($x6$),
$h01 =_q l1, h51 =_q l6$
$x0 = x1 = x2 = x3, e3 = e41, x4 = x5 = x6, l1 = l2, l3 = l4$

Highly underspecified RMRS output:

$l0$: $a0$: _the_q($x0$), $l1$: $a1$: _fat_j($x1$), $l2$: $a2$: _cat_n($x2$), $l3$: $a3$: _sit_v($e3$), $l4$: $a4$: _on_p($e4$),
$l5$: $a5$: _a_q($x5$), $l6$: $a6$: _mat_n($x6$)

Copestake (2007): use POS tags rather than a lexicon, and get what relations we can from the grammar.

Minimal recursion semantics (MRS)

- RMRS-style representations are used in a lot of projects these days.
- Standard composition algorithm not incremental.
- Incremental versions by Schlangen et al. for dialog systems
- We can try it ourselves using the DELPH-IN project web site—implemented with the “English Resource Grammar”.

Lots of open questions!

Just for example:

- What do about raising constructions? “I want (Brutus) to stab Caesar.”
- Modals? “Brutus may have stabbed Caesar in the agora.” — possible world semantics!!!
- Relative clauses: “The man who Caesar offended stabbed him.”
- And, of course, an actual incremental semantic construction algorithm. . .

Lets try some of our own

- Brutus often stabs Caesar in the chest in the agora.
- Some person often stabs Caesar.
- Every senator who stabbed Caesar is angry.
- Mark Antony saw that every senator stabbed Caesar.
- Brutus wants to stab Caesar.

Part 2: incrementality, syntax, and scope

Q: So how do we use this to represent processing?

A: We need to connect it to SOME incrementality-capable syntactic formalism.

Q: Which one, then?

A: One example: TAG

A: One example: TAG

Sayeed and Demberg (2012): most of the remainder of this lecture is shameless self-promotion. ;)

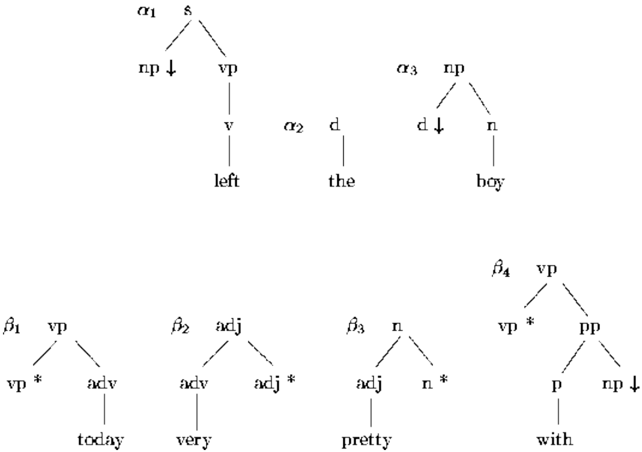
TAG: the recap

What's in a TAG?

- Trees.
- Two operators: adjunction and substitution.
- Some trees are initial and have N substitution nodes.
- Some trees are auxiliary and have the same phrase-type label on the root and foot nodes.

TAG: the recap

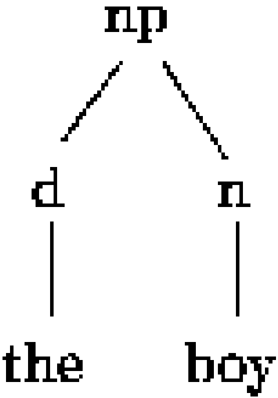
Initial and auxiliary trees:



from `http://www.let.rug.nl/ vannoord/papers/diss/diss/node59.html`

TAG: the recap

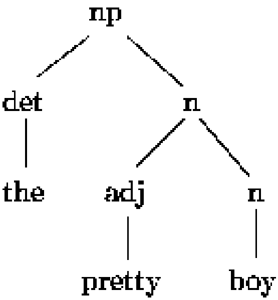
Substitution:



from http://www.let.rug.nl/ vannoord/papers/diss/diss/node59.html

TAG: the recap

Adjunction:



from http://www.let.rug.nl/ vannoord/papers/diss/diss/node59.html

TAG: the recap

What's in an LTAG?

- Everything that's in a TAG.
- Every tree is usually headed by a lexical item, so the lexicon is full of trees.

PLTAG: “Psycholinguistically-plausible TAG”

What’s in a PLTAG?

- Everything that’s in an LTAG.

PLTAG: "Psycholinguistically-plausible TAG"

What's in a PLTAG?

- Everything that's in an LTAG.
- Plus prediction trees.
  - The special sauce!
  - Unlexicalized.
- Prediction trees are unified with lexical items by verification operation.

PLTAG: "Psycholinguistically-plausible TAG"

What's in a PLTAG (Demberg, 2010)?

- Everything that's in an LTAG.

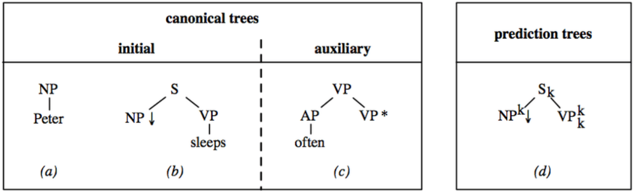
PLTAG: "Psycholinguistically-plausible TAG"

What's in a PLTAG (Demberg, 2010)?

- Everything that's in an LTAG.
- Plus prediction trees.
  - The special sauce!
  - Unlexicalized.
- Prediction trees are unified with lexical items by verification operation.

PLTAG: "Psycholinguistically-plausible TAG"

Prediction trees:



(after Demberg et al., 2013)

Q: How to bridge to a semantic predictive incrementality?

A: “Decorate” the nodes with semantic expressions.

Building a grammatical formalism

One possible methodology:

- Choose a particularly interesting sentence.
  - Preferably one with a psycholinguistically/grammatically/logically interesting characteristic.
- Choose an existing (set of) formalism(s).
  - In our case, PLTAG and a neo-Davidsonian representation.
- Try to come up with the minimum compromise required to get a successful output representation.

Not necessarily the only methodology. . .

Our example sentence:

Send every restaurant a reservation request.

Our example sentence:

Send every restaurant a reservation request.

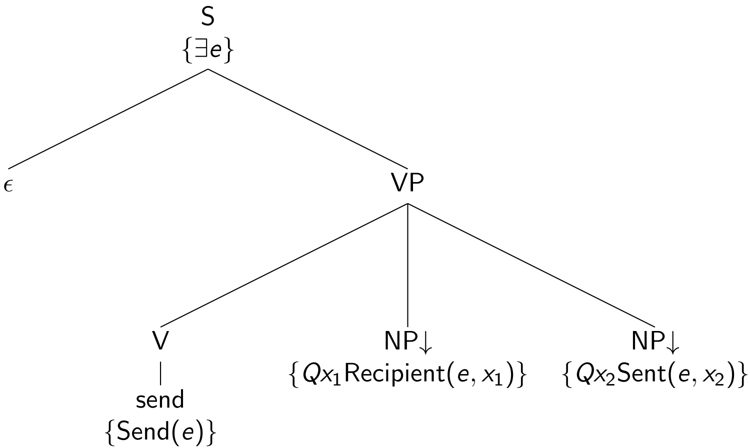
Why this one? Ditransitive alternation of “send”.

- Incremental parsing under ambiguous circumstances.
- A selection of quantifiers.

Since this is a lexicalized formalism, the first thing we need is a lexicon

Our toy lexicon

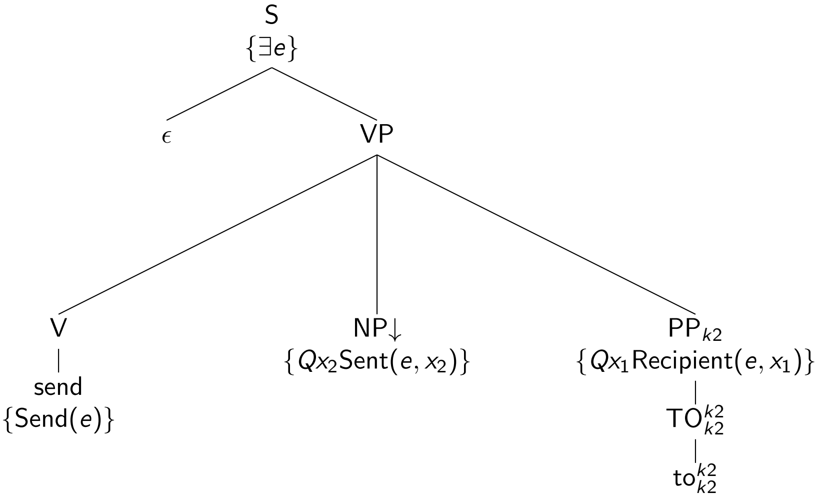
Ditransitive alternation of imperative "send":



With the recipient first.

Our toy lexicon

Ditransitive alternation of imperative "send":



The "to" version.

The semantic extension

We see now how the strategy unfolds:

- Decorate the root of sentence-type items with existentially quantified events ($\exists e$).
- The head gets the main predicate.
- The substitution nodes get the θ-roles.

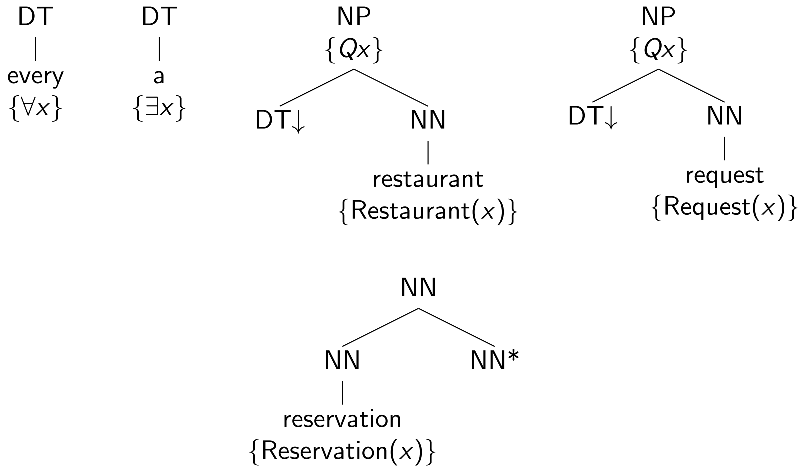
The semantic extension

A bit of notation:

- Q_n – Unspecified quantifiers, valued by unification with determiners.
- x_n – variables representing entities.
- e_n – variables representing events.
- _ – empty predicate from prediction tree, filled by verification.

Our toy lexicon

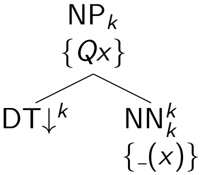
Nominals and determiners:

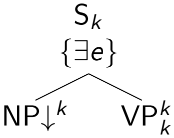


No surprises here, but do notice the unquantified auxiliary tree!

Our toy lexicon

Prediction trees:





- Superscripts and subscripts just like from Vera's PLTAG lecture.
- Headed items occupied by _ predicate variables—filled through verification/unification.

But now that we have a lexicon, we need a combinatory process.

But now that we have a lexicon, we need a combinatory process.

Fortunately, just like PLTAG process. Except. . .

Semantic extension to PLTAG parsing

1. As new elementary and prediction trees are introduced, emit predicates as conjuncts in output semantic expression.
2. Integration process (substitution / adjunction / verification) in parse tree
   - Corresponding variables coindexed across syntactic and semantic expressions
3. Calculate correct scope for $\forall$, scope ambiguity, etc.
   - Need to identify arguments and adjuncts.

Scope issues

Preserving semantic well-formedness: particularly difficult with quantifiers.

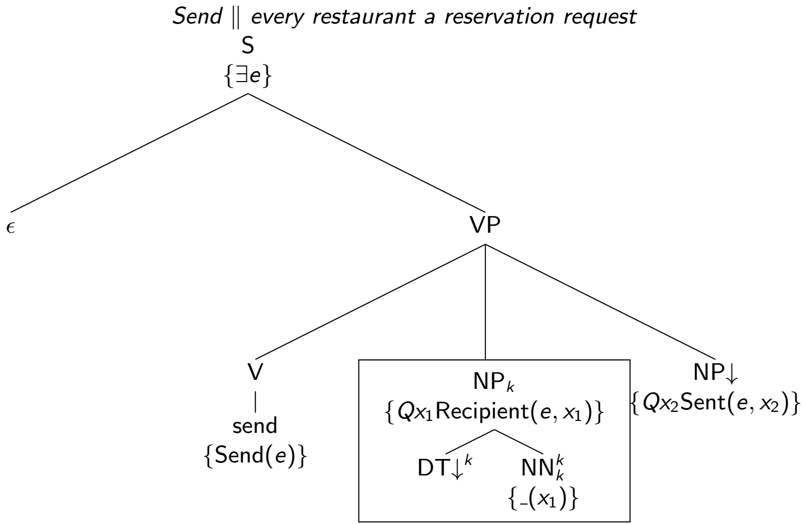
- Restrictor vs. nuclear scope – characteristics of the thing being quantified vs. the proposition it is scoping.

(1) a. Some flower that some bride holds wilts.
b. $\exists x_1 \text{Flower}(x_1) \& [\exists x_2 \text{Bride}(x_2) \& \exists e_2 \text{Hold}(e_2) \& \text{Holder}(e_2, x_2) \& \text{Held}(e_2, x_1)]$
$\& \exists e_1 \text{Wilt}(e_1) \& \text{Wilter}(e_1, x_1)$

(2) a. Every flower that some bride holds wilts.
b. $\forall x_1 \text{Flower}(x_1) \& [\exists x_2 \text{Bride}(x_2) \& \exists \mathbf{e_2} \text{Hold}(\mathbf{e_2}) \& \text{Holder}(\mathbf{e_2}, x_2) \& \text{Held}(\mathbf{e_2}, \mathbf{x_1})]$
$\rightarrow \exists e_1 \text{Wilt}(e_1) \& \text{Wilter}(e_1, x_1)$

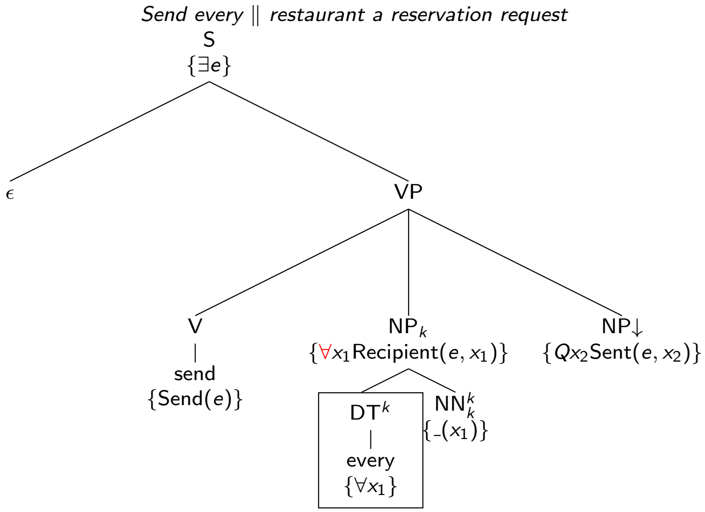
Universal ($\forall x$) quantification requires a conditional ($\rightarrow$). Where to put it? Can events help us?

A sample parse



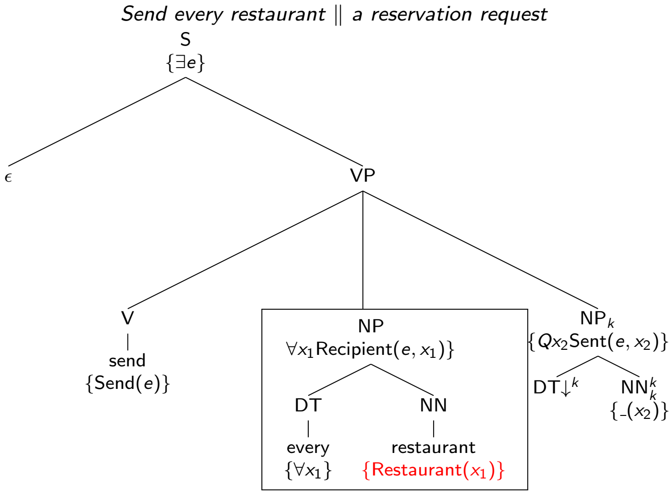
Semantics: $Qx_1\,\_(x_1)\&Qx_2\exists e\,\text{Recipient}(e,x_1)\&\text{Sent}(e,x_2)\&\text{Send}(e)$

A sample parse



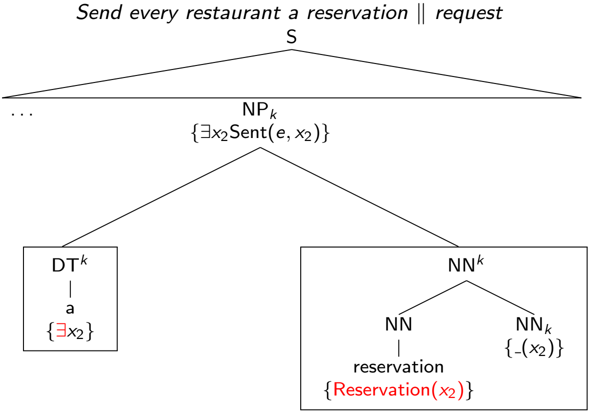
Semantics: $\forall x_1 \_(x_1) \rightarrow Q x_2 \exists e \text{Recipient}(e, x_1) \& \text{Sent}(e, x_2) \& \text{Send}(e)$

A sample parse



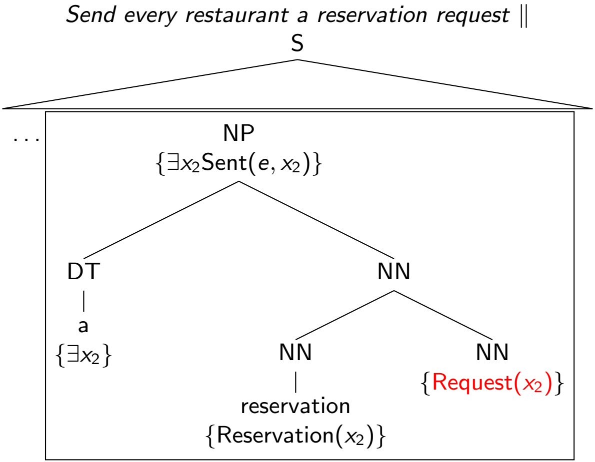
Semantics: $\forall x_1 Restaurant(x_1) \rightarrow Qx_2 \_(x_2) \& \exists e Recipient(e, x_1) \& Sent(e, x_2)$
$\& Send(e)$

A sample parse



Semantics: $\forall x_1 \text{Restaurant}(x_1) \rightarrow \exists x_2 \_(x_2) \& \text{Reservation}(x_2)$
$\& \exists e \text{Recipient}(e, x_1) \& \text{Sent}(e, x_2) \& \text{Send}(e)$

A sample parse



Semantics: $\forall x_1 \text{Restaurant}(x_1) \rightarrow \exists x_2 \text{Request}(x_2) \& \text{Reservation}(x_2)$
$\& \exists e \text{Recipient}(e, x_1) \& \text{Sent}(e, x_2) \& \text{Send}(e)$

Next lecture: psycholinguistic matters