# Human string-generating algorithms

Seminar week 3: Understanding the Theory of Syntax, Summer 2014

Asad Sayeed

Uni-Saarland

## Where did we leave off last week?

We talked about:

- the notion of linguistic intuition.
- a language-user's internal knowledge.
- the use of grammaticality judgements.
- going beyond description to explanation.
- examples of anaphora and binding in English — making our own homegrown linguistic generalization.

# So what's next?

The big questions:

- What are the elements of a linguistic generalization?
- What are the mechanisms by which words connect to one another (the "meat" of syntax)?
- How are these connections resolved into the sequences that human beings produce?

## So what's next?

The big questions:

- What are the elements of a linguistic generalization?
- What are the mechanisms by which words connect to one another (the "meat" of syntax)?
- How are these connections resolved into the sequences that human beings produce?

**The hairyest question of them all: how is this even learnable?**

## Yeah, Asad, how is this even learnable?

# Let's go back to our binding theory generalisation.

**X $\leftarrow$ VERB $\rightarrow$ Y-reflexive if Y $=$ X.**
**X $\leftarrow$ VERB $\rightarrow$ Y-nonreflexive if Y $\neq$ X.**

## Let's go back to our binding theory generalisation.

$$\mathbf{X} \leftarrow \mathbf{VERB} \rightarrow \mathbf{Y\text{-}reflexive\ if\ Y = X.}$$
$$\mathbf{X} \leftarrow \mathbf{VERB} \rightarrow \mathbf{Y\text{-}nonreflexive\ if\ Y \neq X.}$$

It works at arbitrarily large distances . . .

## . . . as long as we stay within the same finite predicate!

Bob hates himself.
Bob wants to hate himself.
Bob needs to want to hate himself.
Bob has to need to want to hate himself.
Bob has to need to want those pills to hate himself.

## (A side note.)

(Let's get something out of the way right now. **How** it is that you *can* say something is a different question from **why** it is that you *would* say something.)

## (A side note.)

(Let's get something out of the way right now. **How** it is that you *can* say something is a different question from **why** it is that you *would* say something.)
(OK, fine. At least, you can't assume that they're the same question off the bat. Later in the semester we might get a contrasting view. . . )

# Anyway, back to our reflexives.

Bob hates himself.
Bob wants to hate himself.
Bob needs to want to hate himself.
Bob has to need to want to hate himself.
Bob has to need to want those pills to hate himself.

But generally not

*Bob has to need to want those pills to hate him. (where Bob $=$ him)

**I just picked these because we know reflexives now.**

## But there's a heck of a lot of "action-at-a-distance".

Who did Bob wish Jill hated?
Who did Bob wish Mary regretted Jill hated?

But this is worse:

?Who did Bob wish Mary regretted hated Jill?

(You can coerce an interpretation of the latter – syntax $\neq$ semantics! – but it's degraded to me, at least.)

## So we need a superstructure.

An architecture outside the string itself.

- Of course, being comp. ling. people, we have the architecture, right?
- (P)CFGs, TAGs, etc.
- These can (and have) been used to account for *many* phenomena.

## But existing formalisms won't cut it.

What formal approaches *don't* currently do:

- They don't solve the learning (child acquisition) problem.
- To be tractable, they have to make *assumptions*.
- These assumptions are often application-driven, or they accommodate adult processing facts, or . . .

## But nobody solves these problems!

**Certainly nothing I'm going to teach today is a real solution!**

## So let's start over.

- Cut away the assumptions intended to *do* something or *describe* something for a specific purpose.
- Try to find the minimum assumptions that make language language, **before** answering learnability questions.

"Virtual conceptual necessity."

- Now we're getting into the Uriagereka reading.
- Why is 2+2, 4?
- Well, not quite. Instead: "What are the things without which we couldn't describe the universe?"

## Q: So, what capacities MUST human language generation have?

**A: At least one thing: the ability to combine strings.**

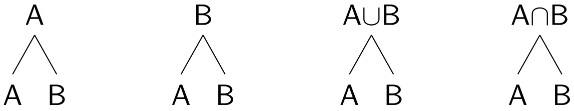
## And here we finally get Merge.

**Finally** climbing out of the pit of methodological discussion.

- We need to be able to put *at least* two things together.
- So we need a "merge" operation.
- But then we need the two things we put together to be put together with other things.
- Otherwise we'd only get two-word sentences . . .

# What object does Merge make?

Some options for merge(A, B):



Uriagereka favours the first two as possible outcomes of merge – "projection". (Our first trees of the semester!) Why?

# What are the options for Merge?

Let's take "Bob hates". Bob – N, hates – V.

- Union - merge(Bob, hates) has both noun and verb characteristics.

## What are the options for Merge?

Let's take "Bob hates". Bob – N, hates – V.

- Union - merge(Bob, hates) has both noun and verb characteristics.
    - Can merge again with another verb? "(Bob hates) grows."

## What are the options for Merge?

Let's take “Bob hates”. Bob – N, hates – V.

- Union - merge(Bob, hates) has both noun and verb characteristics.
  - Can merge again with another verb? “(Bob hates) grows.”
- Intersection - merge(Bob, hates) has neither verb nor noun characteristics.

## What are the options for Merge?

Let's take "Bob hates". Bob – N, hates – V.

- Union - merge(Bob, hates) has both noun and verb characteristics.
  - Can merge again with another verb? "(Bob hates) grows."
- Intersection - merge(Bob, hates) has neither verb nor noun characteristics.
  - Can't say: "(Bob hates) Jack."
  - (Cuts off features too early.)

## What are the options for Merge?

Let's take "Bob hates". Bob – N, hates – V.

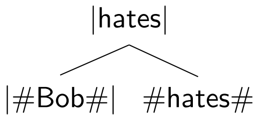
- Union - merge(Bob, hates) has both noun and verb characteristics.
  - Can merge again with another verb? "(Bob hates) grows."
- Intersection - merge(Bob, hates) has neither verb nor noun characteristics.
  - Can't say: "(Bob hates) Jack."
  - (Cuts off features too early.)
- merge(Bob, hates) emits characteristics of "Bob" – just like union (in this instance only)

# What are the options for Merge?

Let's take "Bob hates". Bob – N, hates – V.

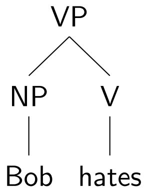
- Union - merge(Bob, hates) has both noun and verb characteristics.
  - Can merge again with another verb? "(Bob hates) grows."
- Intersection - merge(Bob, hates) has neither verb nor noun characteristics.
  - Can't say: "(Bob hates) Jack."
  - (Cuts off features too early.)
- merge(Bob, hates) emits characteristics of "Bob" – just like union (in this instance only)
- **Right answer:** merge(Bob, hates) emits characteristcs of "hates". "(Bob hates) Jack."

## We have a structure!



- Uriagereka uses $|x|$ to mark off the *maximal projection* – the last result of Merge.
- $\#x\#$ for the *head* – the position that originally came with lexical content.
- Uriagereka's own notation.

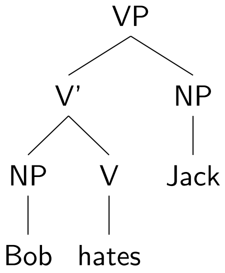
## A more “standard” way of putting it.



In more traditional accounts, there’d be more nodes. Merge eliminates this.

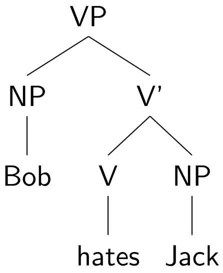
## What if we want to include an object?

By the process we've invented so far, it might look like:



## What if we want to include an object?

But most accounts prefer:



Meaning that the object has to "arrive" in the derivation first!

## Which, of course, is odd.

- Clearly we parse (or generate) the subject first in English!
- But: we're not (yet) talking about parsing or generation, just what structures we need.
- The "dynamic" construction with Merge is really intended to allow us to decide what structure we *don't* need.

(Think of it as the iPhone of syntactic theories, heh.)

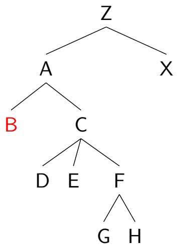
## But we still need to yield a string.

We need to impose some kind of ordering between elements of the structure.

- Many relations are asymmetrical.
  - Like reflexives. “Bob hates himself” vs. “*Himself hates Bob”.
  - In fact symmetry is massively disfavoured in linguistic structures!
- Ubiquitous in syntax: c-command (U. calls it just “command”).

## A c-commands B iff. . .

Uriagereka defines (c-)command formally, but what you need to know is this:

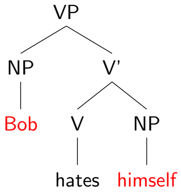


- B c-commands: C D E F G H
- B does not c-command: A X Z

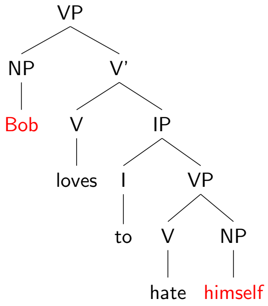
**c-command(X, Y) just means: X is sister or (great-*)aunt of Y**

# Structural asymmetry is powerful.

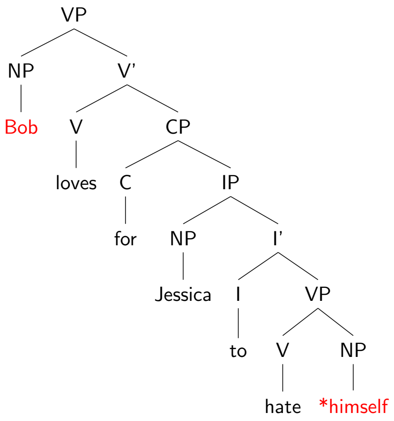
- c-command is implicit in many other frameworks.
- But the “sister” part still alows for the possibility of symmetry, so we often use “proper” c-command (the great-aunt part only).



# Or, to get ambitious. . .



# Or, to get even more ambitious. . .



Complementizer phrases (CP) appear to have special powers.

# What else can we do with c-command?

Define linear precedence relations!

- Uriagereka spends a lot of time on the Linear Correspondence Axiom (LCA).
- (This was an idea proposed in 1994 by Richard Kayne.)
- Basic idea: c-command helps us define linear order: we now know how structures yield strings.

All kinds of interesting predictions follow, except that there are a number of exceptions.

## Conjunctions are hugely problematic.

John bought and Mary read the book. (Wilders, 1999)

This is called “right-node raising” and happens here and there.

- How do we “draw” conjunctions asymetrically?
- Are we now FORCED to admit ternary branching? (Since Merge conveniently is always binary.)
- Should we just admit that nodes are inherently ordered?

## And is that enough to dismiss it?

## That is yet another methodological question!

What is the role of counterexamples?

- If something covers most cases, does it require special technology?
- Is it just a matter of performance?
- If so, why is right-node raising so systematic?
- How much elegance should we sacrifice?

**Gertrude Stein said: “There is no there there.” (You knew I’d work in a quote somewhere.)**

**The problem is: which “there” isn’t there?**

## Next week: recent literature “shock therapy.”