Computational Intelligence 696i

Language

Lecture 4

Sandiway Fong

Administriva

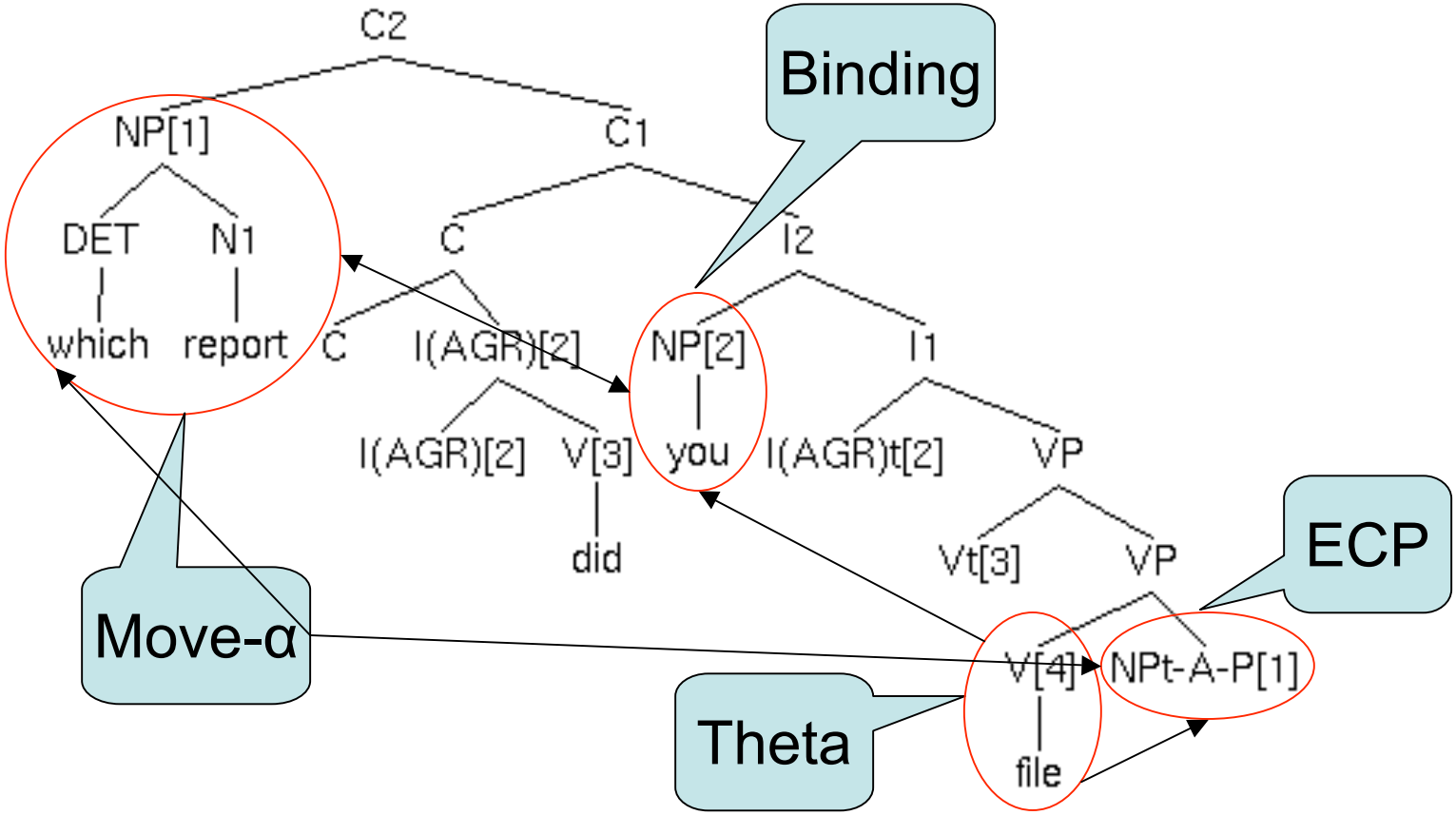
- Homework 1 out today
  - reviewed in class today
    - *so ask clarification questions*!
  - due one week from today
  - submit to sandiway@email.arizona.edu

Last Time

- we talked about the paradigm shift from **“rule-based” systems** to the **principles-and-parameters** (P&P) framework
- the idea that we have UG, a system with some amount of pre-wiring + learning mechanism (including parameter setting)

Principles-and-Parameters

a system of interacting sub-modules



Today's Lecture

- goal is to get you familiarized with PAPPI, a principles-and-parameters (P&P) parser
  - representing one possible instantiation of UG
  - universal part
    - a set of 20–30 principles
  - language-particular part
    - parameters settings instantiated for SVO, SOV, V2 languages
    - small lexicons for a certain number of languages
      - Turkish, Hungarian, Chinese, Japanese, Dutch, German, French, Spanish, Bangla, English
  - system is a parser only
    - there is no learning mechanism

Today's Lecture

- Gotta get through 3 things today...
  1. explain the demo
  2. do one exercise
  3. present the homework

- Reading (optional) for discussion next time:
  - latest thinking on language and linguistic theory
  - download and read 1st 5–6 pages of
  - **On Phases** by N. Chomsky (m.s. 2005)
  - http://dingo.sbs.arizona.edu/~sandiway/mpp/onphases.pdf

Part (1)

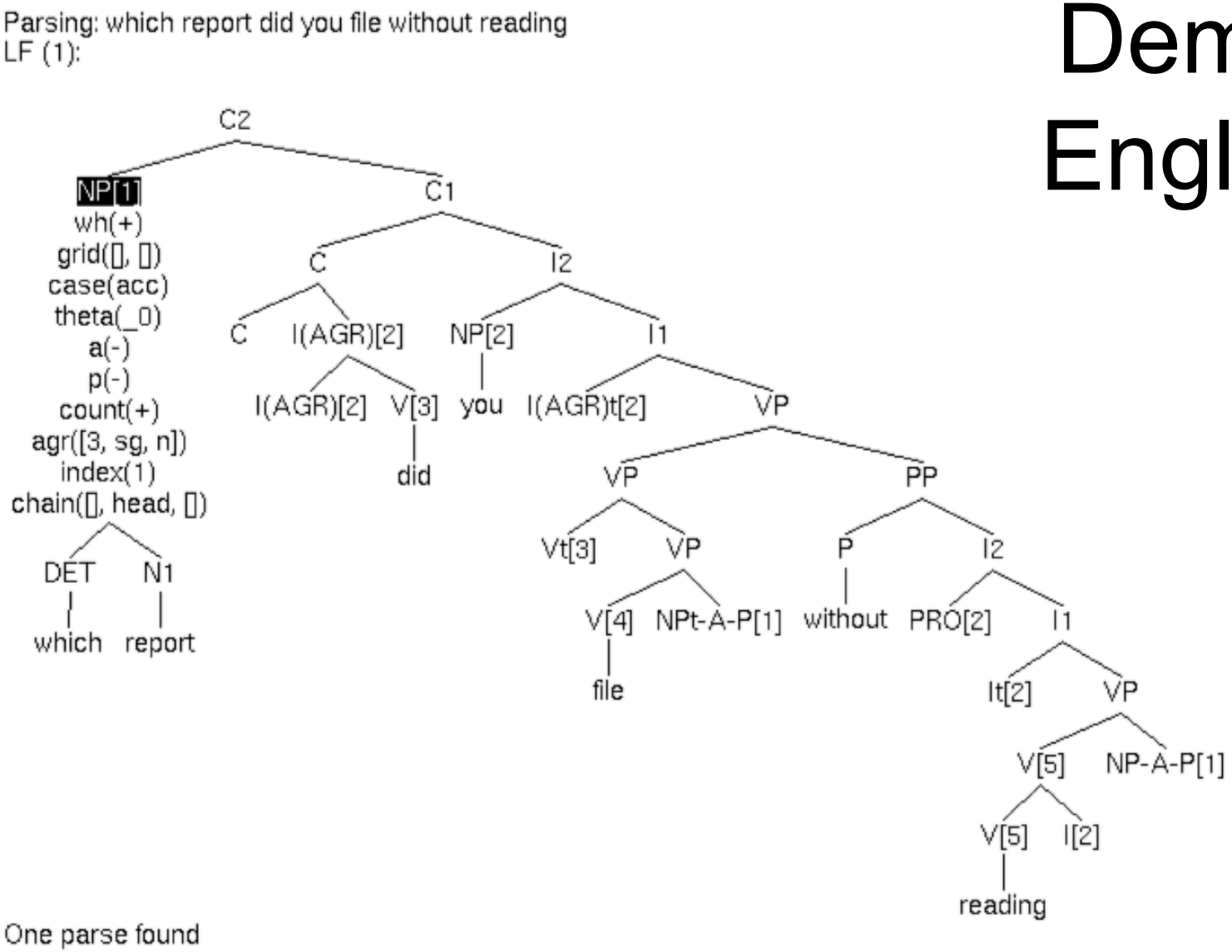
Demo

- description available on webpage
  - http://dingo.sbs.arizona.edu/~sandiway/pappi/macosx/index.html#test
- example of how UG might be instantiated
  - one set of principles
  - three languages
    - English: SVO
    - Japanese: SOV
    - Dutch: V2-language
      - verb is 2nd phrase (roughly resembles SVO),
      - but in embedded clauses verb comes last (SOV)

Demo: English

- Example:
  - Which report did you file without reading?
- Word Order:
  - SVO
- Structure:
  - Which report did you file [the report] without [you] reading [the report]?
  - Which report[1] did [$_{S}$ you[2] [$_{VP}$ file NP*t*[1]] [without [$_{S}$ NP[2] [$_{VP}$ reading NP[1]]]]]?
- Notes:
  - NP indicates noun phrase e-element
  - trace indicated by *t*
  - indices, e.g. [1], are used for coindexation

Demo: English

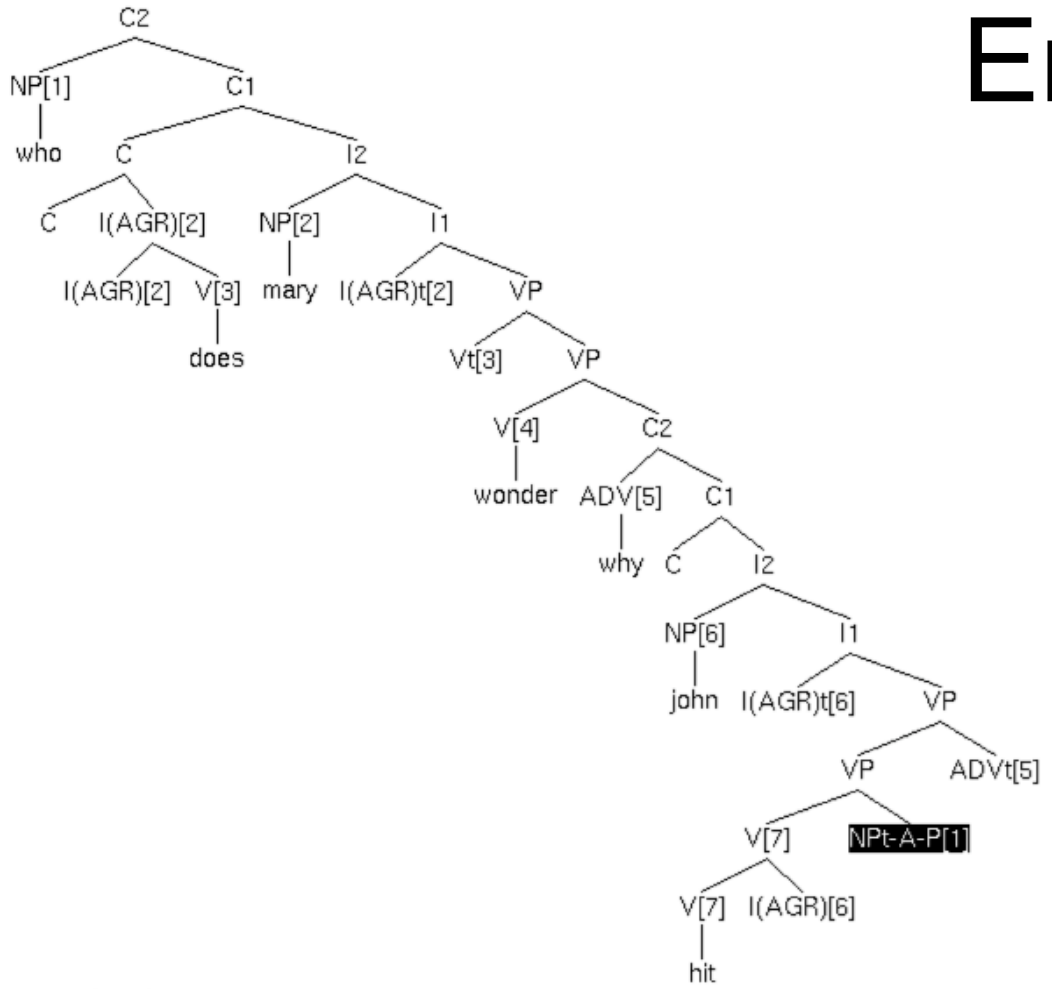


Demo: English

- Example:
  - *Who does Mary wonder why John hit?
- Ungrammatical
  - violates principle of subjacency
    - *can't displace too far in one hop*
- However, you can still recover the meaning...
  - *so it's (considered) a mild violation*
- Underlying structure:
  - Mary wonders why John hit **who**
  - **Who** does Mary wonders why John hit **trace**
- Explanation:
  - interaction with X'-theory:
  - *no intermediate position available as a landing site*
  - *cf. Who does Mary think John hit?*

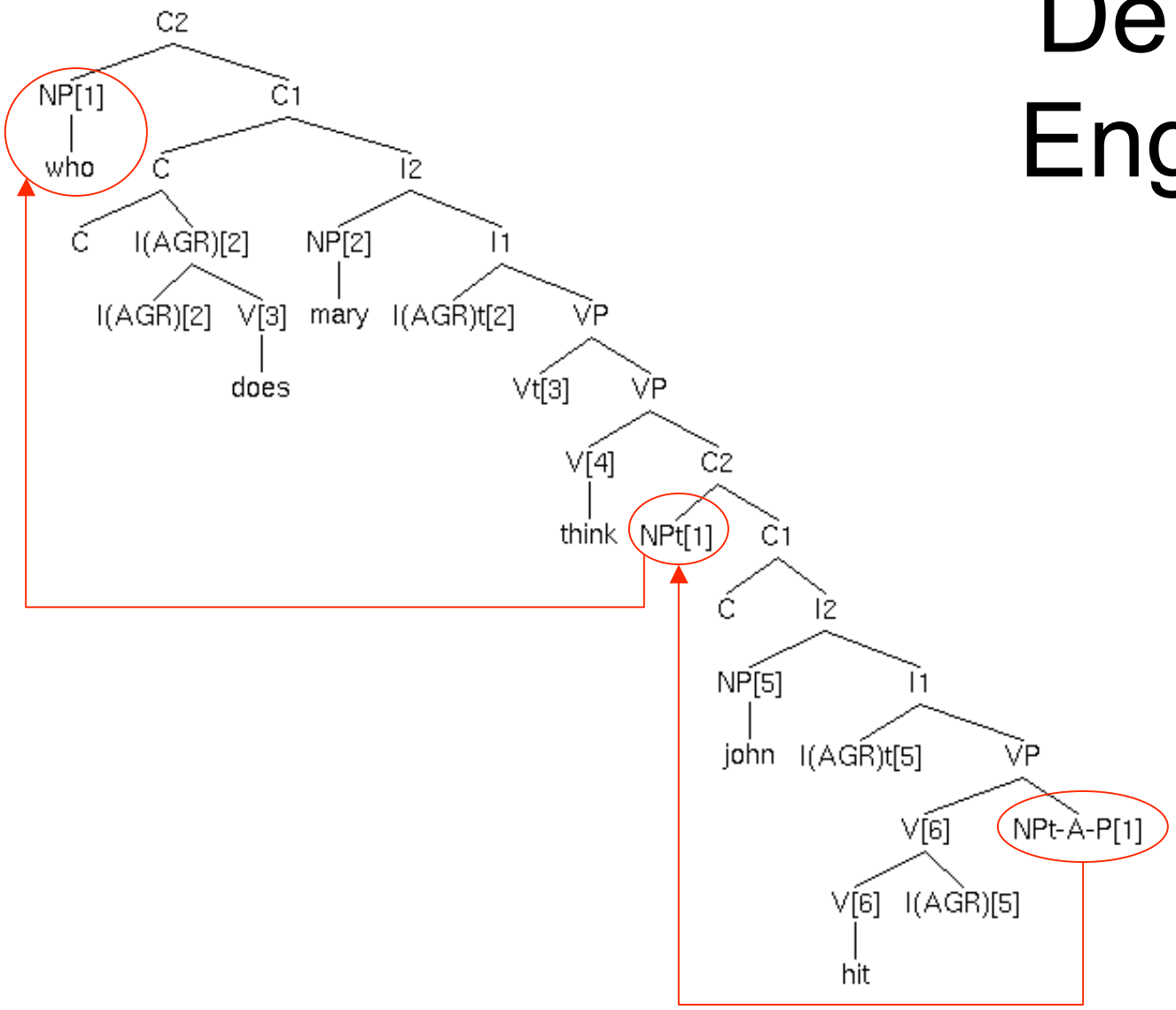
Demo: English

Subjacency not active.
Parsing: who does Mary wonder why John hit
LF (1):



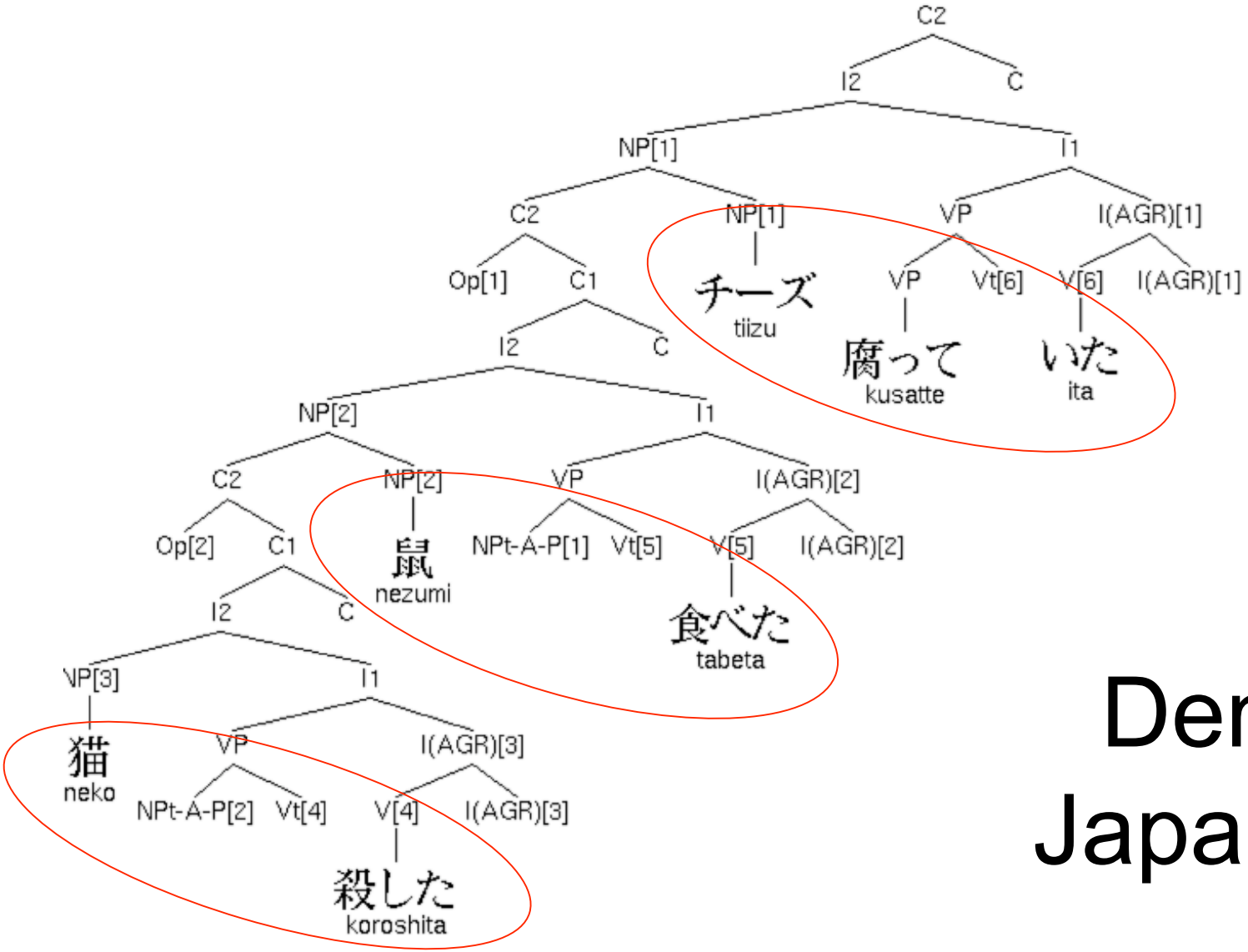
One parse found

Demo: English



Demo: Japanese

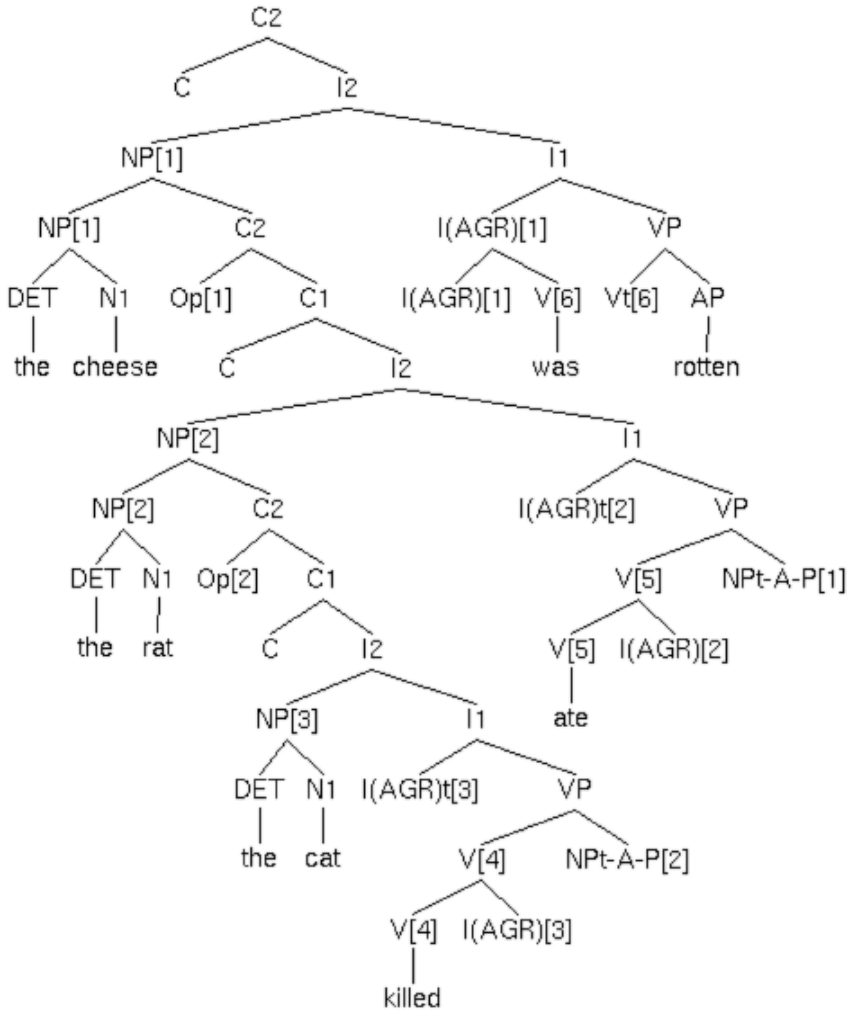
- Example:
  - neko-ga **koroshita** nezumi-ga **tabeta** tiizu-wa **kusatte ita**
  - cat-NOM killed rat-NOM ate cheese-TOP rotten was
  - the cheese the rat the cat killed ate was rotten
- Word Order:
  - SOV
- Center-embedding (English)
  - [the cheese [the rat [the cat killed] ate] was rotten]
  - *resource limitation*
- Left-embedding (Japanese)
  - [cat killed] [rat ate] [cheese was rotten]
  - *no resource limitation*



Demo: Japanese

Demo: Japanese

Parsing: the cheese the rat the cat killed ate was rotten
LF (1):



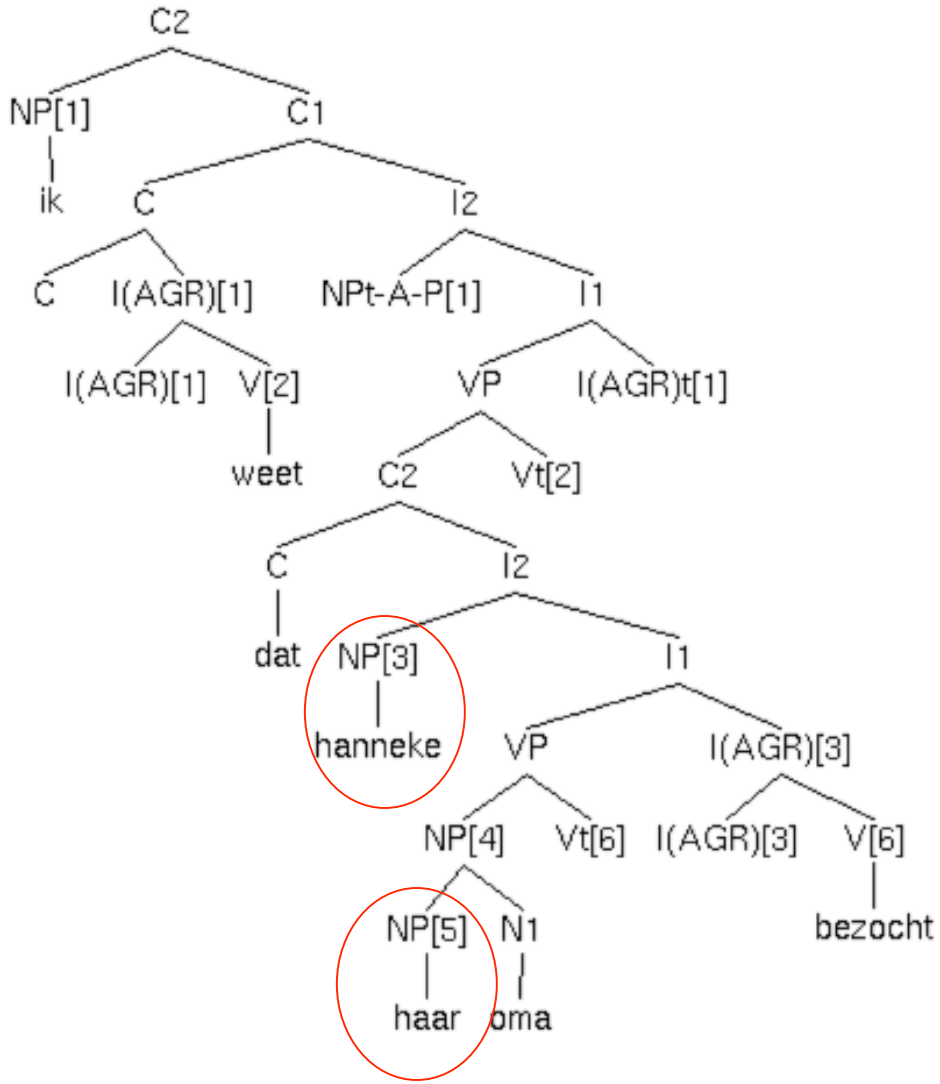
One parse found

Demo: Dutch

- Example:
  - Ik **weet** dat Hanneke haar oma **bezocht**
  - I know that Hanneke her grandma visited
  - *I know that Hanneke visited her Grandma*
- V2 word order:
  - [$_S$ Ik **weet** [$_S$ dat Hanneke haar oma **bezocht**]]
- Pronoun binding ambiguity
  - **whose grandma**?
  - same ambiguity in Dutch as in English
  - determined by the rules of pronoun binding

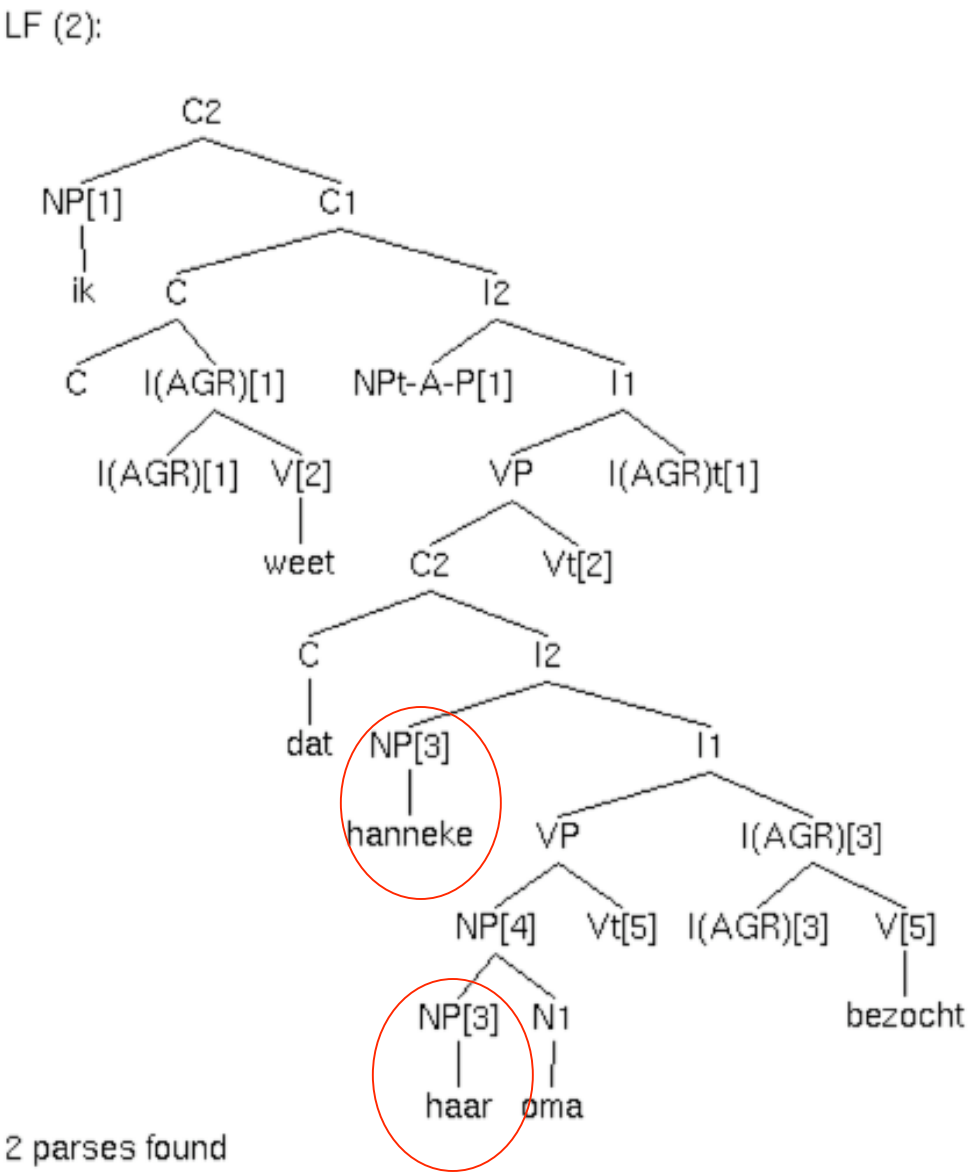
Demo: Dutch

Parsing: ik weet dat Hanneke haar oma bezocht
LF (1):



[3] ≠ [5]

Demo: Dutch



[3] = [3]

Part (2)

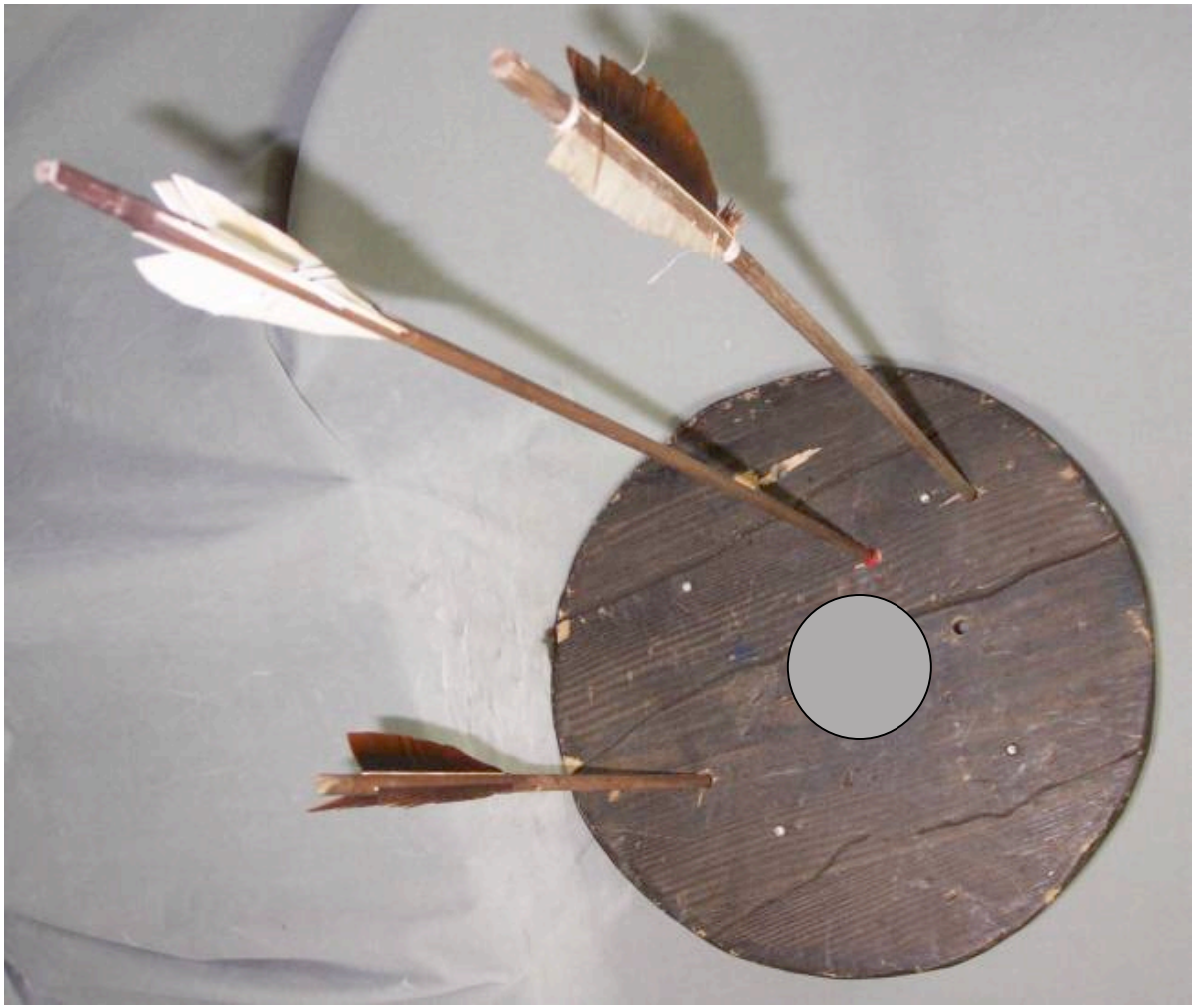
Using PAPPI

- description available on
  - *Introduction to the Theory of PAPPI* http://dingo.sbs.arizona.edu/~sandiway/pappi/macosx/pgap.html
- how to use PAPPI to see what UG is doing
  - you will do a very similar exercise for homework 1
- let's look at the parasitic gap sentence again
    - which report did you file without reading?
  - and look at Move-alpha (*displacement property*)

Using PAPPI

- Example:
    - which report did you file without reading?
- Move-alpha (*displacement property*)
  - you filed which report without reading
  - which report did you file *trace* without reading
- Why isn't it?
  - you filed without reading **which report**
  - which report did you file *trace* without reading *trace*
- Why isn't it?
  - you filed without reading **which report**
  - which report did you file without reading *trace*
- What rules out these derivation?
  - *PAPPI considers all possible derivations*

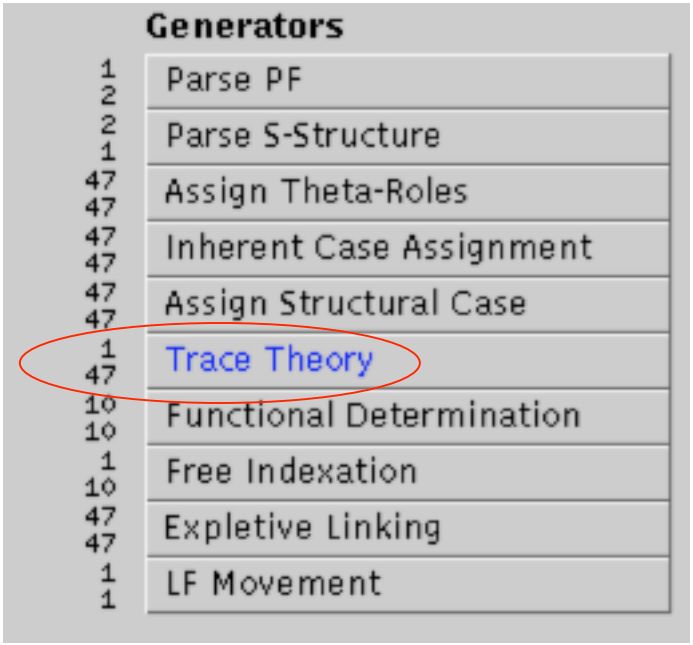
PAPPI: *Computation*



think of
derivations
running
a gauntlet of
constraints
and only the
grammatical
ones make it

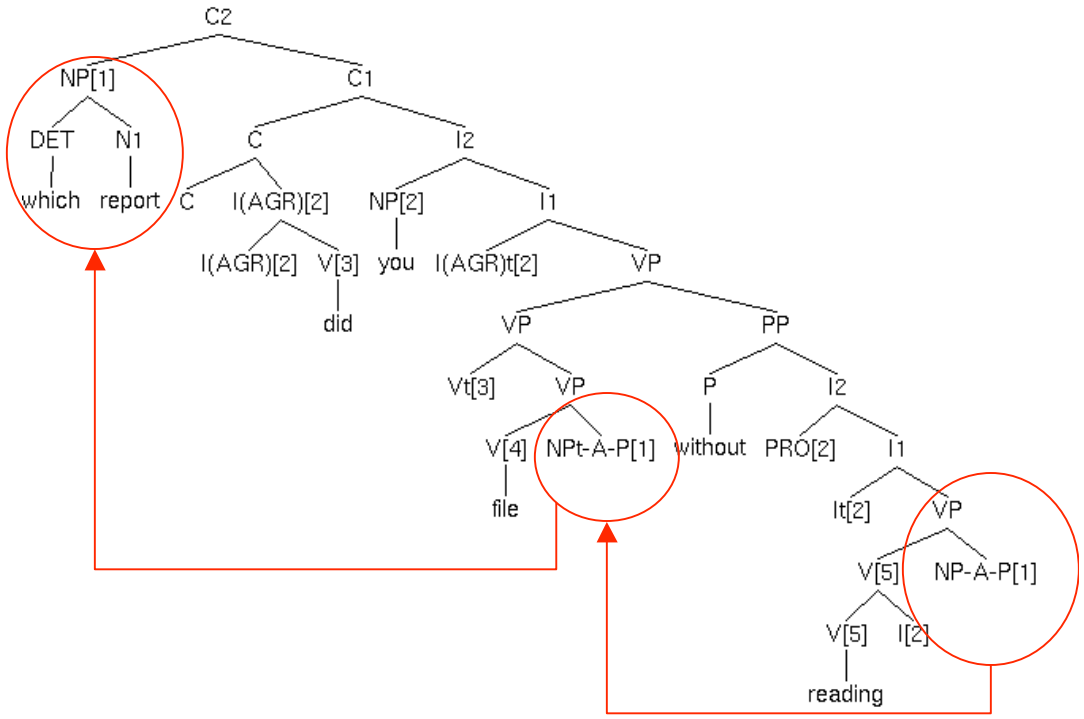
PAPPI: *Computation*

- 47 structures
- 1 admitted
- 46 ruled out
- including
  - which report did you file *trace* without reading *trace*
  - which report did you file without reading *trace*



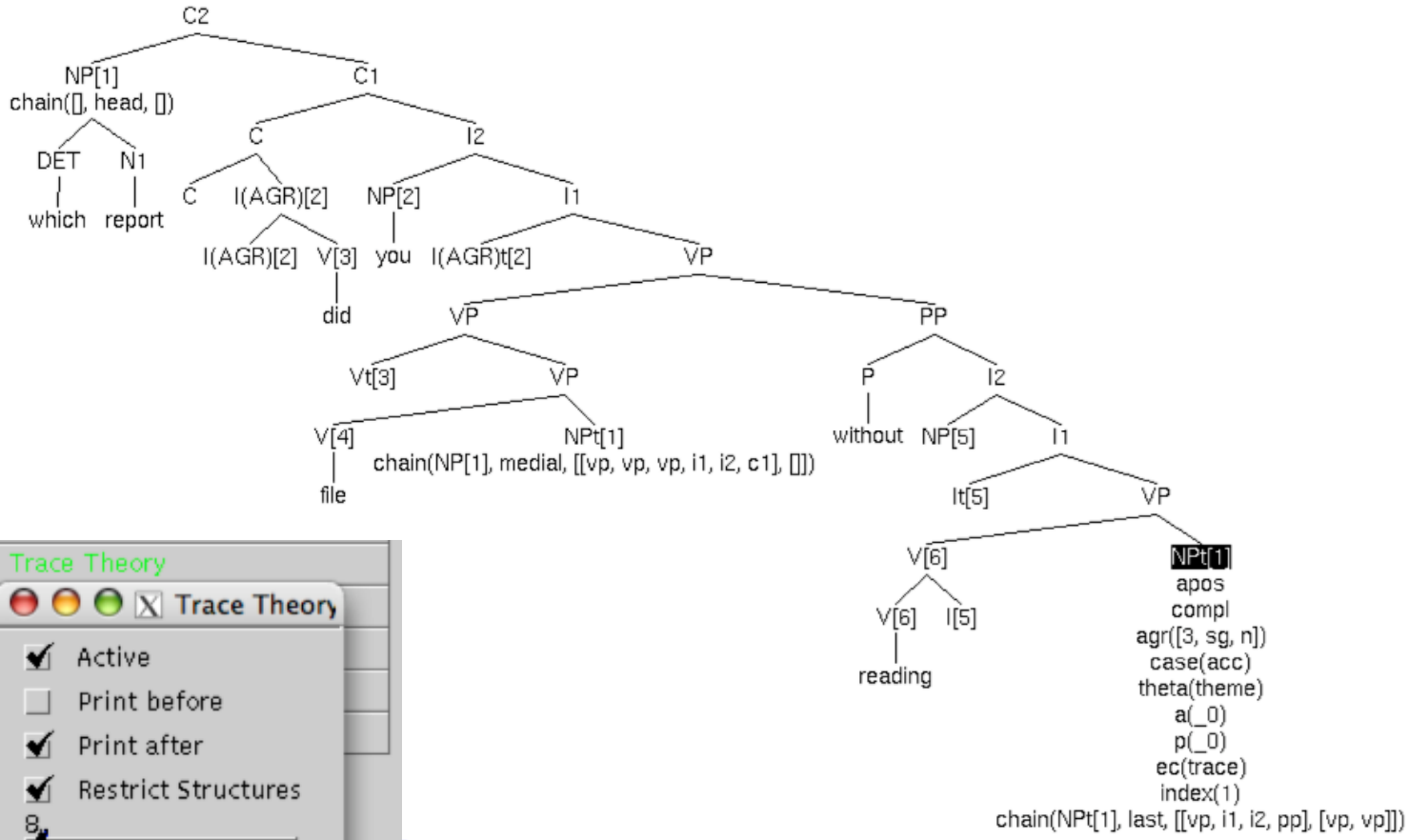
PAPPI: *Computation*

- Why isn't it?
  - you filed without reading **which report**
  - which report did you file *trace* without reading *trace*
- This is tree #8 out of 47
  - look at the chain feature
  - `chain(NP[1],Type,Path)`
    - `Type` = {head, medial, last}
    - `Path` = list of intermediate nodes to antecedent



PAPPI: *Computation*

- **Idea**:
- isolate tree #8
- and see what blocks it



PAPPI: *Computation*

- What blocks a derivation?
  - a principle that when turned off allows a parse to be generated
  - [this is not necessarily the same as the stopping principle reported by the parser]

- Let's test this on #8...
  - Case Condition on Traces (*reported*)
  - Theta Criterion

Part (3)

Homework 1

- Minimal Pair:
  - (1) a. John is too stubborn to talk to
  - b. John is too stubborn to talk to Bill
- It's an interesting example:
  - *just adding one word Bill provokes a big change in gap-filling*
- PAPPI parses:
  - (2) a. John**[1]** is too stubborn Op[1] **PRO[2]** to talk to **t[1]**
  - b. John**[1]** is too stubborn PRO**[1]** to talk to Bill**[2]**
- Readings:
  - (3) a. John is too stubborn **for some arbitrary person** to talk to John
  - b. John is too stubborn **for John** to talk to Bill

Homework 1

- Question 1: 2pts (*giveaway*)
  - *how many structures did it consider for each sentence*?
- Question 2: (6pts)
  - Consider the sentence:
    - (4) John is too stubborn [for John] to talk to himself
    - PAPPI parses both versions of this sentence
    - why is this interpretation unavailable for (1a)?
      - what principle(s) rules it out?
      - your answer should report which parse numbers and the steps required to drill down to the answer
- Question 3: (4 pts)
  - *Think of another example of a minimal pair where the interpretation of a gap in terms of reference must change when a noun (or preposition+noun) is added*