Semantics and First-Order Predicate Calculus

11-711 Algorithms for NLP

28 November 2017

(With thanks to Noah Smith)

Key Challenge of Meaning

- We actually *say* very little - much more is left unsaid, because it's assumed to be widely known.

- Examples:
  - Reading newspaper stories
  - Using restaurant menus
  - Learning to use a new piece of software

Meaning Representation Languages

- Symbolic representation that does two jobs:
  - Conveys the meaning of a **sentence**
  - Represents (some part of) the **world**
- We're assuming a very literal, context-independent, inference-free version of meaning!
  - Semantics vs. linguists' "pragmatics"
  - "Meaning representation" vs some philosophers' use of the term "semantics".
- Today we'll use **first-order logic.** Also called First-Order Predicate Calculus. Logical form.

A MRL Should Be Able To ...

- Verify a query against a knowledge base: *Do CMU students follow politics?*
- Eliminate ambiguity: *CMU students enjoy visiting Senators.*
- Cope with vagueness: *Sally heard the news.*
- Cope with many ways of expressing the same meaning (canonical forms): *The candidate evaded the question* vs. *The question was evaded by the candidate*.
- Draw conclusions based on the knowledge base: *Who could become the 45th president?*
- Represent all of the meanings we care about

Model-Theoretic Semantics

- Model: a simplified representation of (some part of) the world: objects, properties, relations (**domain**).
- Non-logical vocabulary
  - Each element **denotes** (maps to) a well-defined part of the model
  - Such a mapping is called an **interpretation**

A Model

- **Domain**: Noah, Karen, Rebecca, Frederick, Green Mango, Casbah, Udipi, Thai, Mediterranean, Indian
- **Properties**: Green Mango and Udipi are crowded; Casbah is expensive
- **Relations**: Karen likes Green Mango, Frederick likes Casbah, everyone likes Udipi, Green Mango serves Thai, Casbah serves Mediterranean, and Udipi serves Indian

- n, k, r, f, g, c, u, t, m, i
- Crowded = {g, u}
- Expensive = {c}
- Likes = {(k, g), (f, c), (n, u), (k, u), (r, u), (f, u)}
- Serves = {(g, t), (c, m), (u, i)}

Some English

- Karen likes Green Mango and Frederick likes Casbah.
- Noah and Rebecca like the same restaurants.
- Noah likes expensive restaurants.
- Not everybody likes Green Mango.

- What we want is to be able to represent these statements in a way that lets us compare them to our model.
- **Truth-conditional semantics**: need operators and their meanings, given a particular model.

First-Order Logic

- **Terms** refer to elements of the domain: **constants**, **functions**, and **variables**
  - Noah, SpouseOf(Karen), X
- **Predicates** are used to refer to sets and relations; predicate applied to a term is a **Proposition**
  - Expensive(Casbah)
  - Serves(Casbah, Mediterranean)
- Logical connectives (**operators**):

  $\wedge$ (and), $\vee$ (or), $\neg$ (not), $\Rightarrow$ (implies), ...

- **Quantifiers** ...

Quantifiers in FOL

- Two ways to use variables:
  - refer to one anonymous object from the domain (**existential**; $\exists$; "there exists")
  - refer to all objects in the domain (**universal**; $\forall$; "for all")

- A restaurant near CMU serves Indian food
  $\exists x\ \text{Restaurant}(x) \wedge \text{Near}(x, \text{CMU}) \wedge \text{Serves}(x, \text{Indian})$
- All expensive restaurants are far from campus
  $\forall x\ \text{Restaurant}(x) \wedge \text{Expensive}(x) \Rightarrow \neg\text{Near}(x, \text{CMU})$

Inference

- Big idea: extend the knowledge base, or check some proposition against the knowledge base.
- **Forward chaining** with modus ponens: given α and $\alpha \Rightarrow \beta$, we know β.
- **Backward chaining** takes a query β and looks for propositions α and $\alpha \Rightarrow \beta$ that would prove β.
  - Not the same as backward reasoning (*abduction*).
  - Used by Prolog
- Both are sound, neither is complete by itself.

Inference example

- Starting with these facts:

  Restaurant(Udipi)

  $\forall$x Restaurant(x) $\Rightarrow$ Likes(Noah, x)

- We can “turn a crank” and get this *new* fact:

  Likes(Noah, Udipi)

FOL: Meta-theory

- Well-defined set-theoretic semantics
- **Sound:** can’t prove false things
- **Complete:** can prove everything that logically follows from a set of axioms (e.g., with “resolution theorem prover”)
- Well-behaved, well-understood
- Mission accomplished?

FOL: But there are also “Issues”

- “Meanings” of sentences are ***truth values***.
- Only ***first-order*** (no quantifying over *predicates* [which the book does without comment]).
- Not very good for ***“fluents”*** (time-varying things, real-valued quantities, etc.)
- Brittle: ***anything*** follows from *any* contradiction(!)
- ***Goedel incompleteness***: “This statement has no proof”!
  - (Finite axiom sets are incomplete w.r.t. the real world.)
- **So:** Most systems use its descriptive apparatus (with extensions) but not its inference mechanisms.

First-Order Worlds, Then and Now

- Interest in this topic (in NLP) waned during the 1990s and 2000s.
- It has come back, with the rise of semi-structured databases like Wikipedia.
  - Lay contributors to these databases may be helping us to solve the knowledge acquisition problem.
- Also, lots of research on using NLP, information extraction, and machine learning to grow and improve knowledge bases from free text data.
  - “Read the Web” project here at CMU.
- And: Semantic embedding/NN/vector approaches.

Lots More To Say About MRLs!

- See chapter 17 for more about:
  - Representing events and states in FOL
  - Dealing with optional arguments (e.g., “eat”)
  - Representing time
  - Non-FOL approaches to meaning

Connecting Syntax and Semantics

Semantic Analysis

- Goal: transform a NL statement into MRL (today, FOL).
- Sometimes called “semantic parsing.”
- As described earlier, this is the literal, context-independent, inference-free meaning of the statement

“Literal, context-independent, inference-free” semantics

- Example: *The ball is red*
- Assigning a specific, grounded meaning involves deciding *which* ball is meant
- Would have to resolve *indexical terms* including pronouns, normal NPs, etc.
- Logical form allows compact representation of such indexical terms (vs. listing all members of the set)
- To retrieve a specific meaning, we combine LF with a particular context or situation (set of objects and relations)
- So LF is a function that maps an initial discourse situation into a new discourse situation (from *situation semantics*)

Compositionality

- The meaning of an NL phrase is determined by combining the meaning of its sub-parts.
- There are obvious exceptions (“hot dog,” “straw man,” “New York,” etc.).
- Note: your book uses an event-based FOL representation, but I’m using a simpler one without events.
- Big idea: start with parse tree, build semantics on top using FOL with λ-expressions.

Extension: Lambda Notation

- A way of making anonymous functions.
- λx. (*some expression mentioning x*)
  - Example: λx.Near(x, CMU)
  - Trickier example: λx.λy.Serves(y, x)
- Lambda reduction: substitute for the variable.
  - (λx.Near(x, CMU))(LulusNoodles)
    becomes
    Near(LulusNoodles, CMU)

Lambda reduction: order matters!

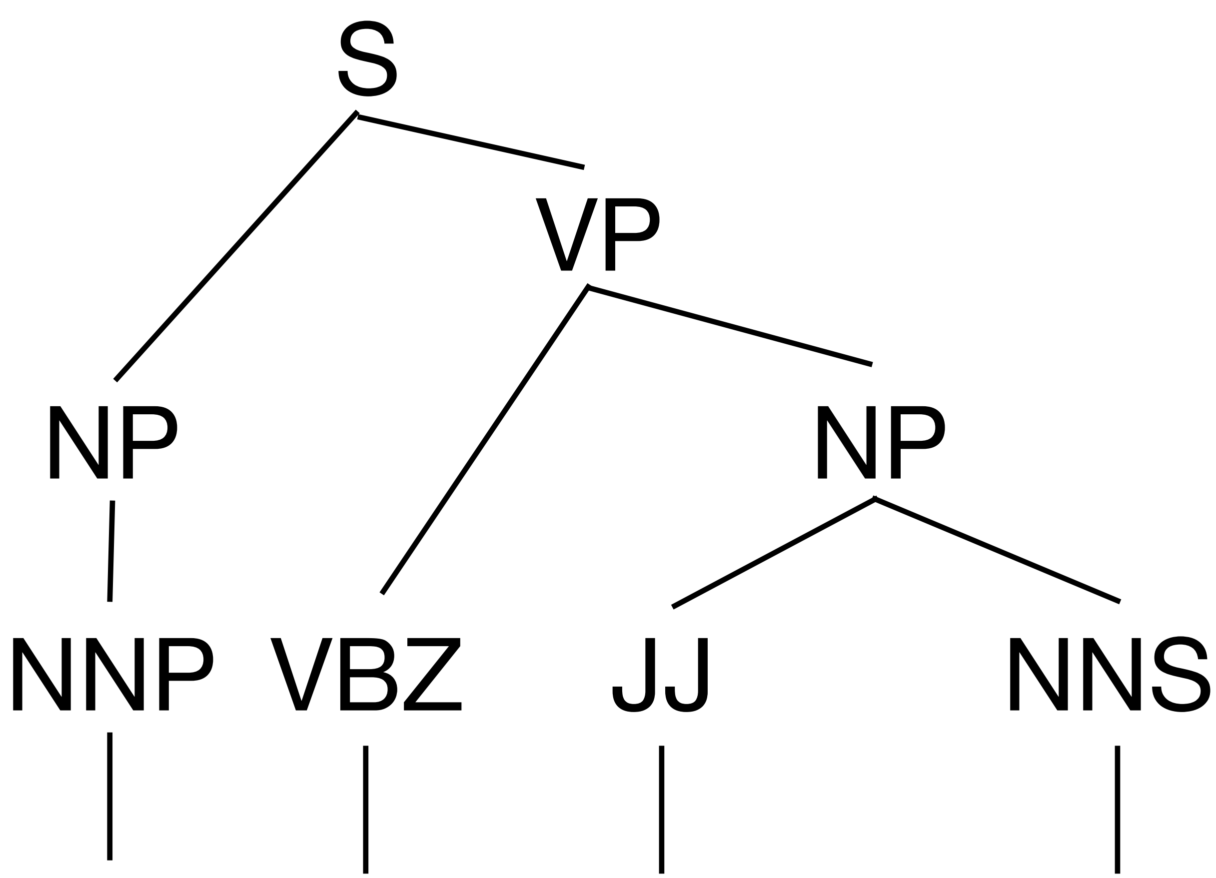
- **λx.λy.**Serves(y, x) (Bill)(Jane) becomes λy.Serves(y, Bill)(Jane)

  Then λy.Serves(y, Bill) (Jane) becomes Serves(Jane, Bill)

- **λy.λx.**Serves(y, x) (Bill)(Jane) becomes λx.Serves(Bill, x)(Jane)

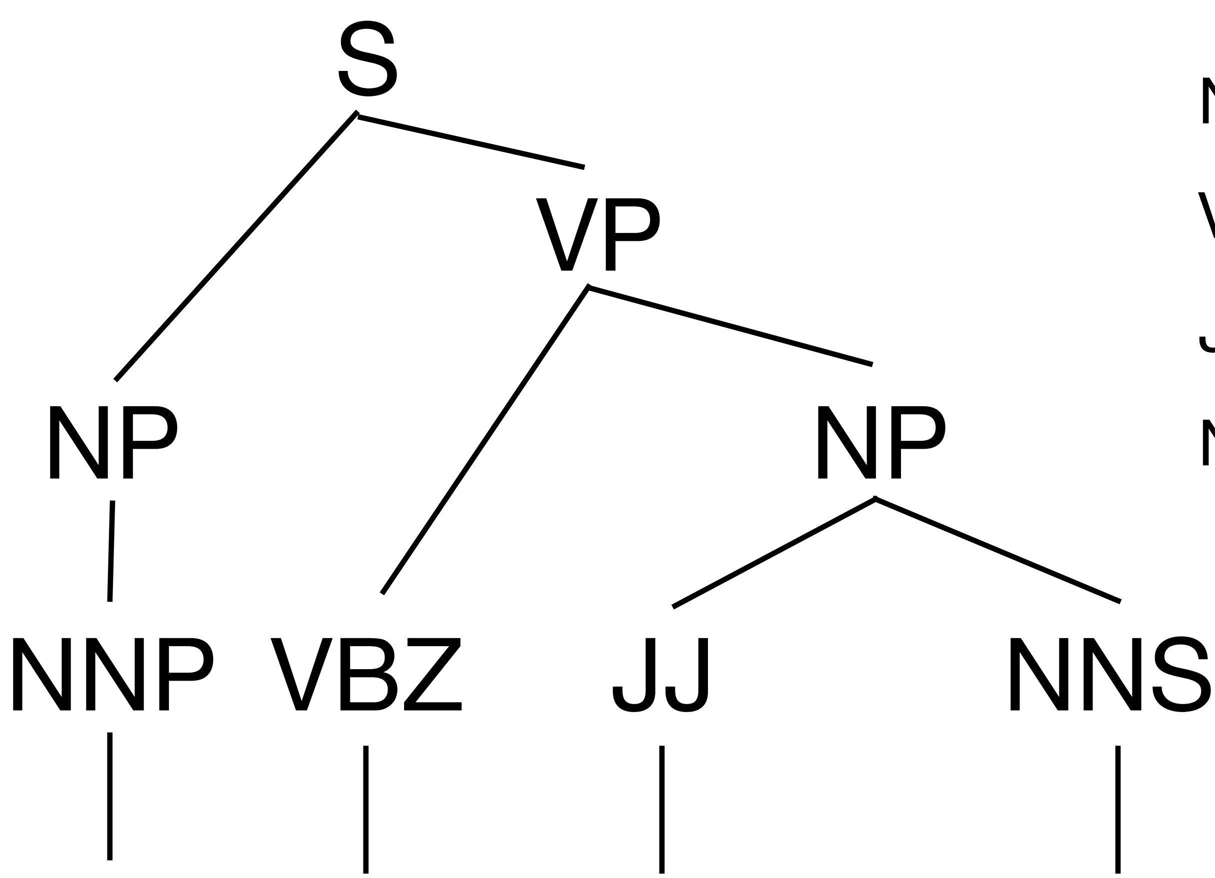
  Then λx.Serves(Bill, x) (Jane) becomes Serves(Bill, Jane)

An Example



- Noah likes expensive restaurants.
- $\forall x\ \text{Restaurant}(x) \wedge \text{Expensive}(x) \Rightarrow \text{Likes}(\text{Noah}, x)$

An Example



NNP → Noah { Noah }

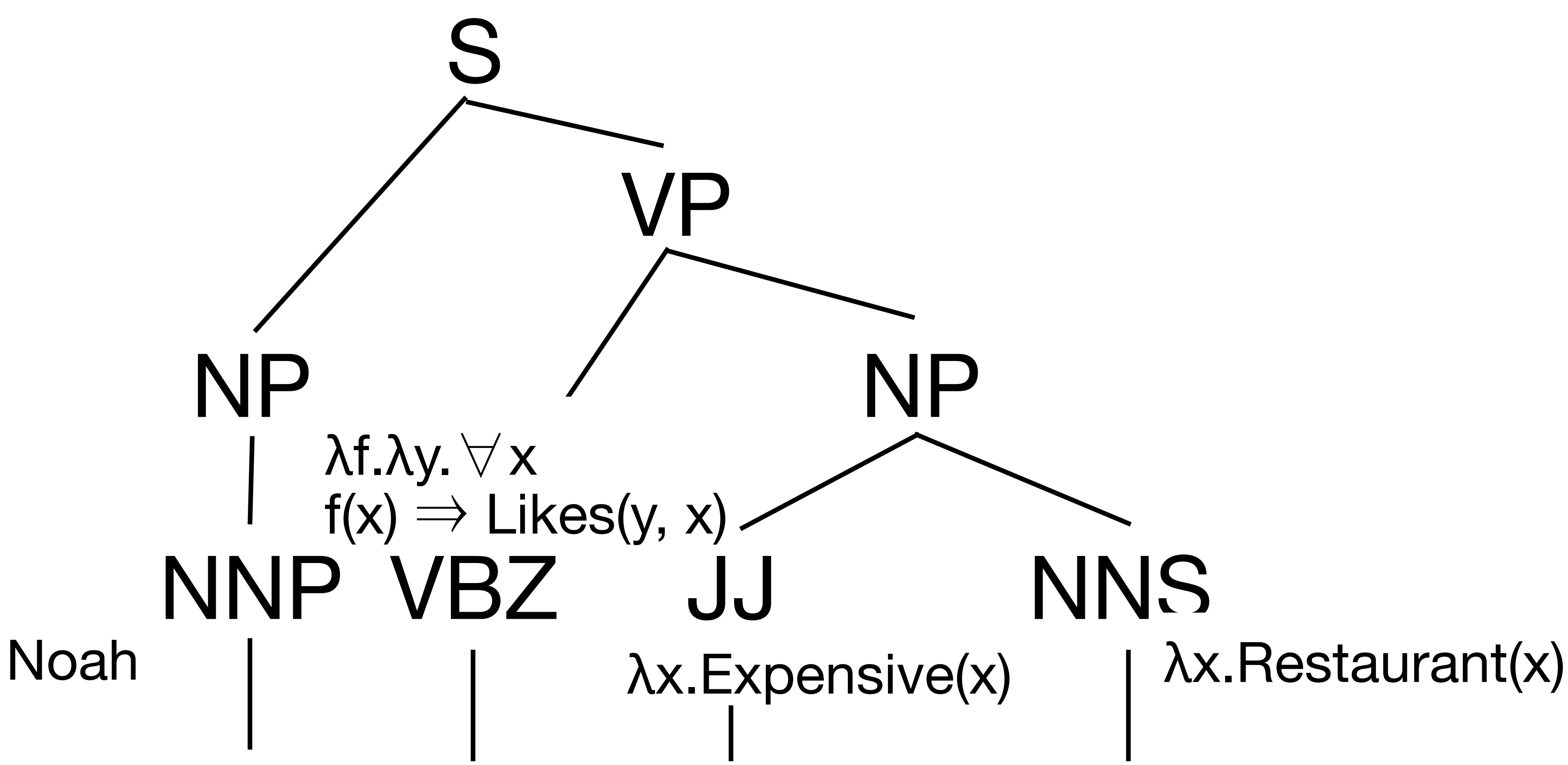
VBZ → likes { $\lambda f.\lambda y.\forall x\ f(x) \Rightarrow Likes(y, x)$ }

JJ → expensive { $\lambda x.Expensive(x)$ }

NNS → restaurants { $\lambda x.Restaurant(x)$ }

- Noah likes expensive restaurants.
- $\forall x\ Restaurant(x) \wedge Expensive(x) \Rightarrow Likes(Noah, x)$

An Example

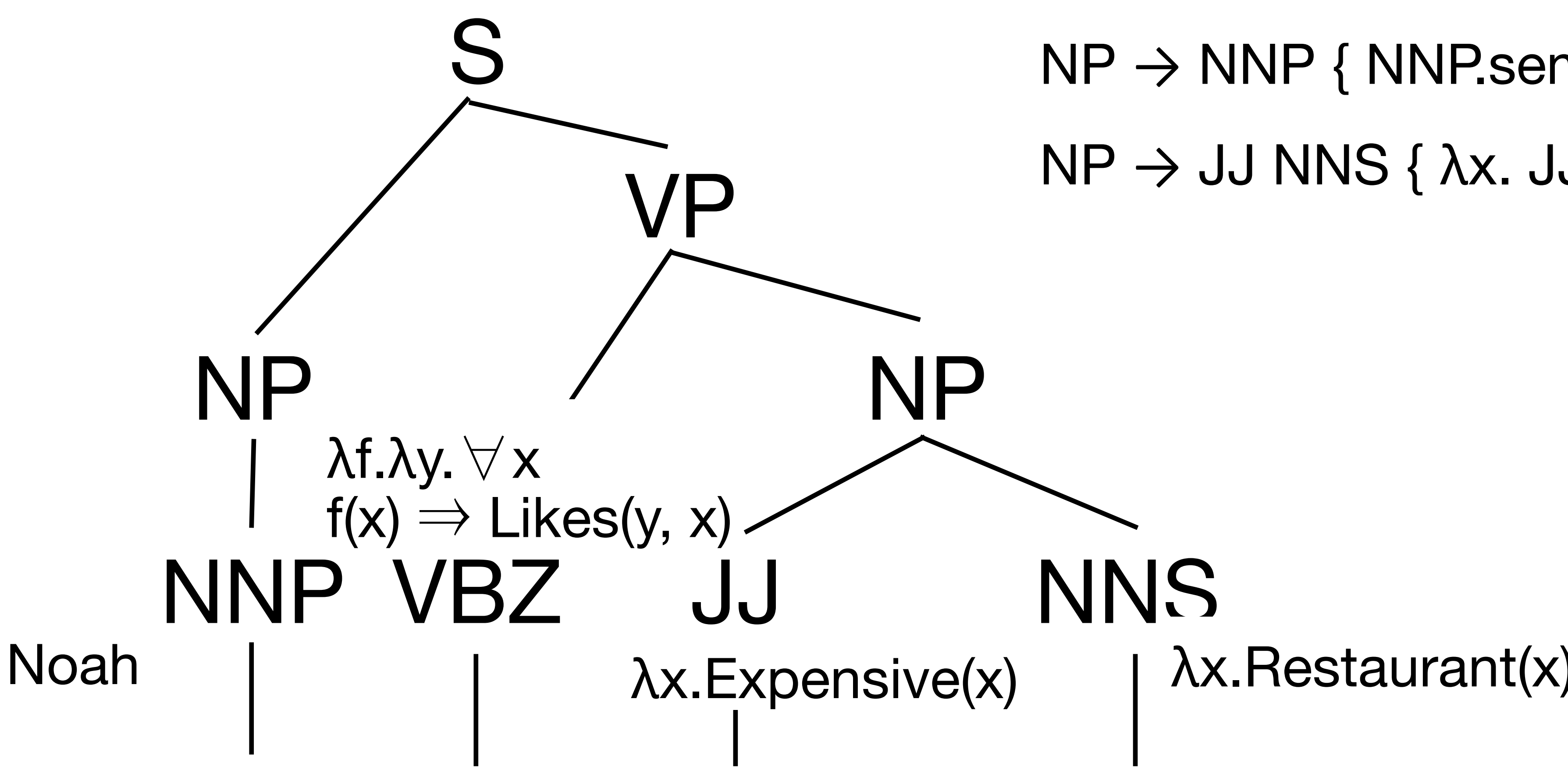


- Noah likes expensive restaurants.
- $\forall x \; Restaurant(x) \wedge Expensive(x) \Rightarrow Likes(Noah, x)$

An Example

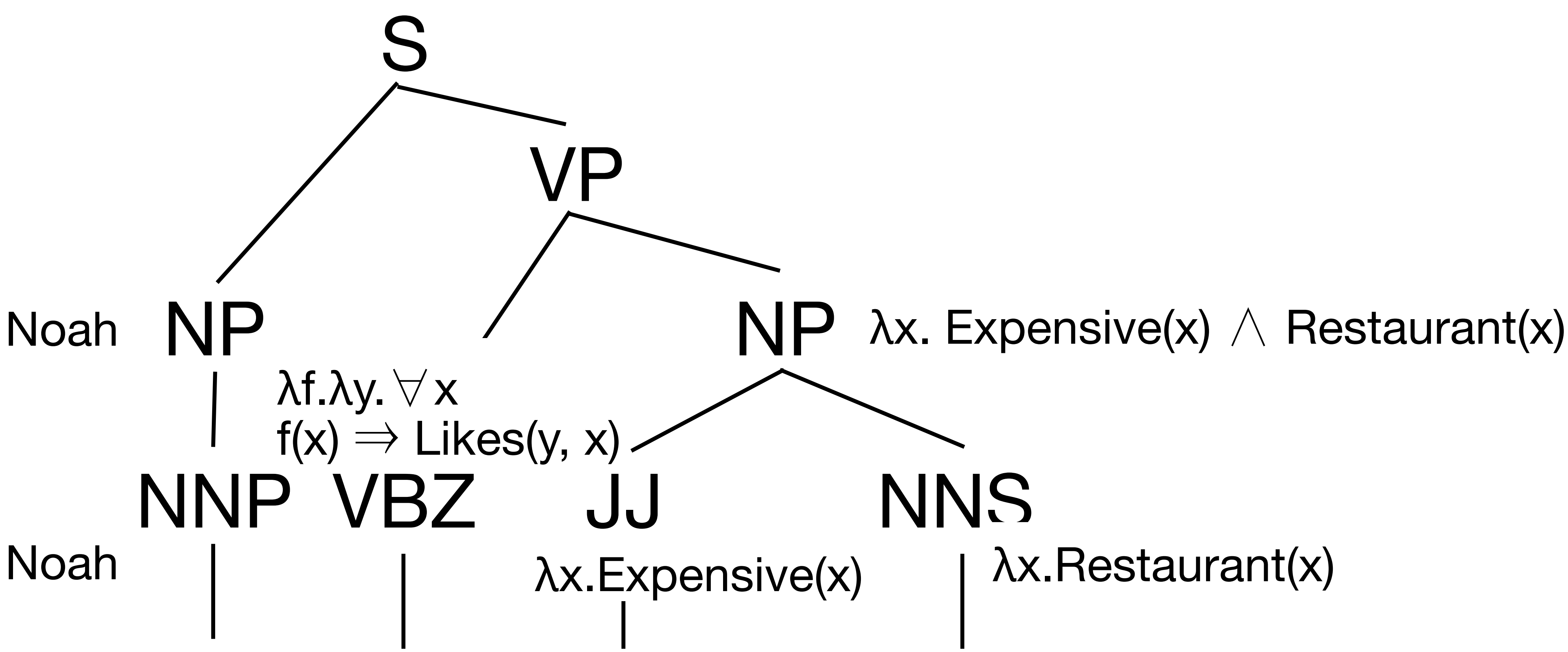
NP → NNP { NNP.sem }

NP → JJ NNS { λx. JJ.sem(x) ∧ NNS.sem(x) }



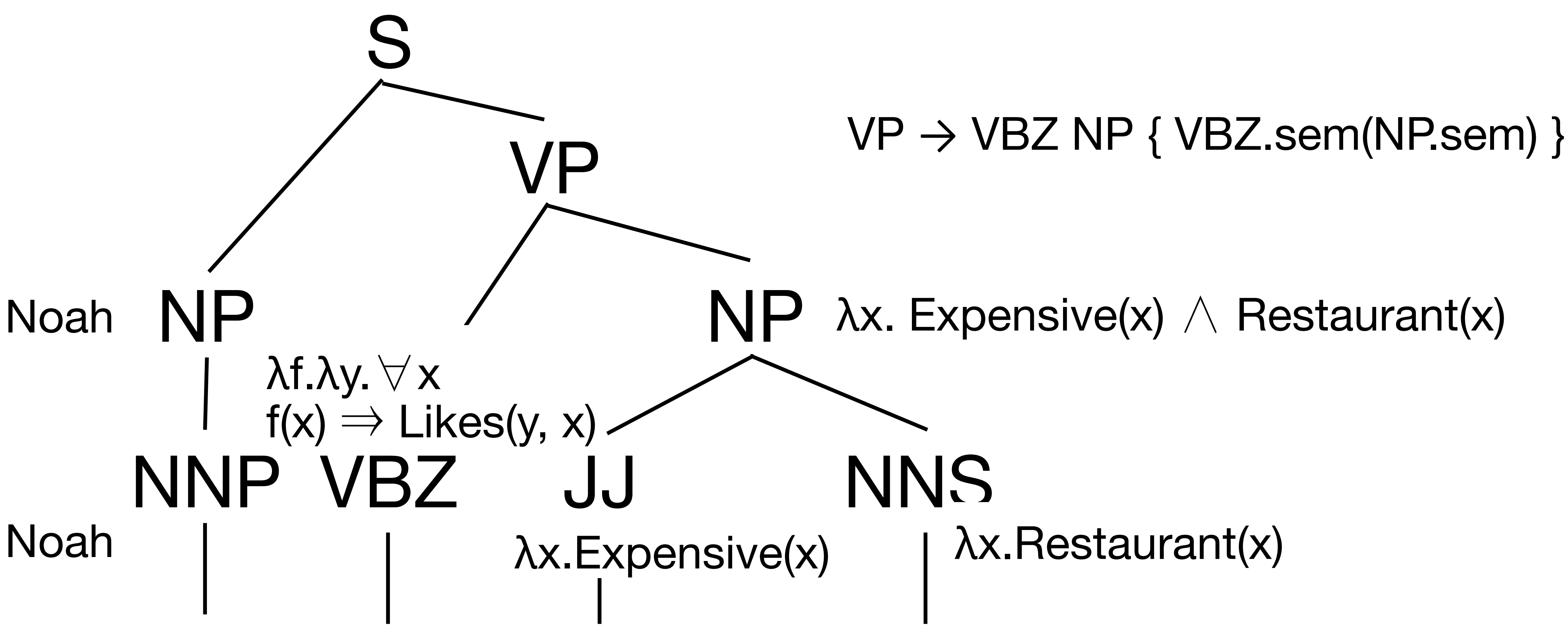
- Noah likes expensive restaurants.
- $\forall x\ Restaurant(x) \land Expensive(x) \Rightarrow Likes(Noah, x)$

An Example



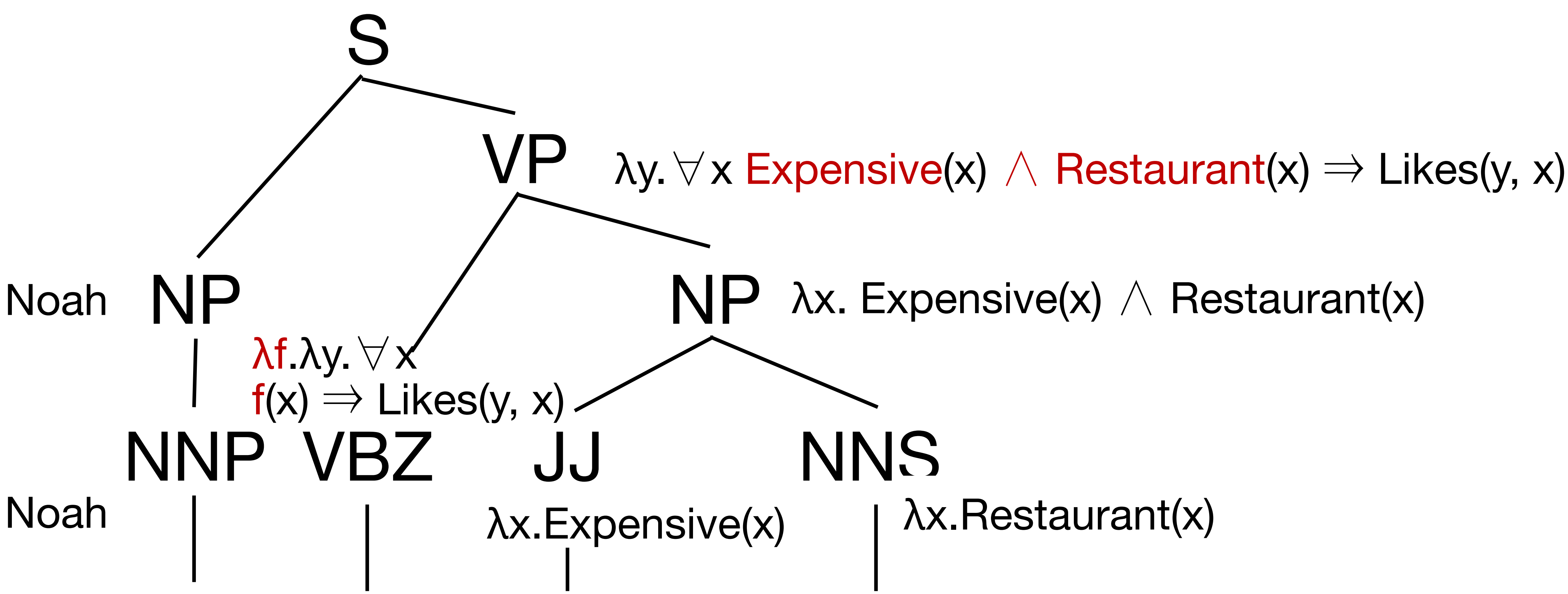
- Noah likes expensive restaurants.
- $\forall x \; Restaurant(x) \wedge Expensive(x) \Rightarrow Likes(Noah, x)$

An Example



- Noah likes expensive restaurants.
- $\forall x$ Restaurant(x) $\wedge$ Expensive(x) $\Rightarrow$ Likes(Noah, x)

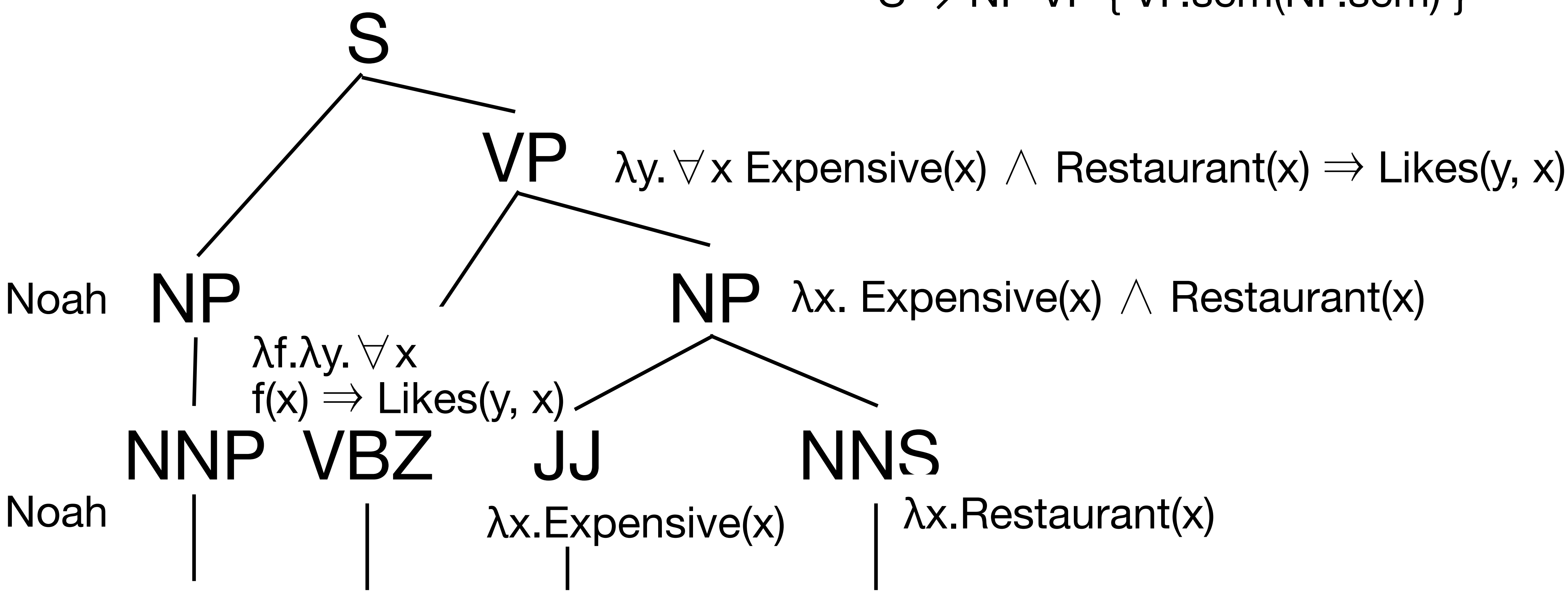
An Example



- Noah likes expensive restaurants.
- $\forall x\ \mathrm{Restaurant}(x) \wedge \mathrm{Expensive}(x) \Rightarrow \mathrm{Likes}(\mathrm{Noah}, x)$

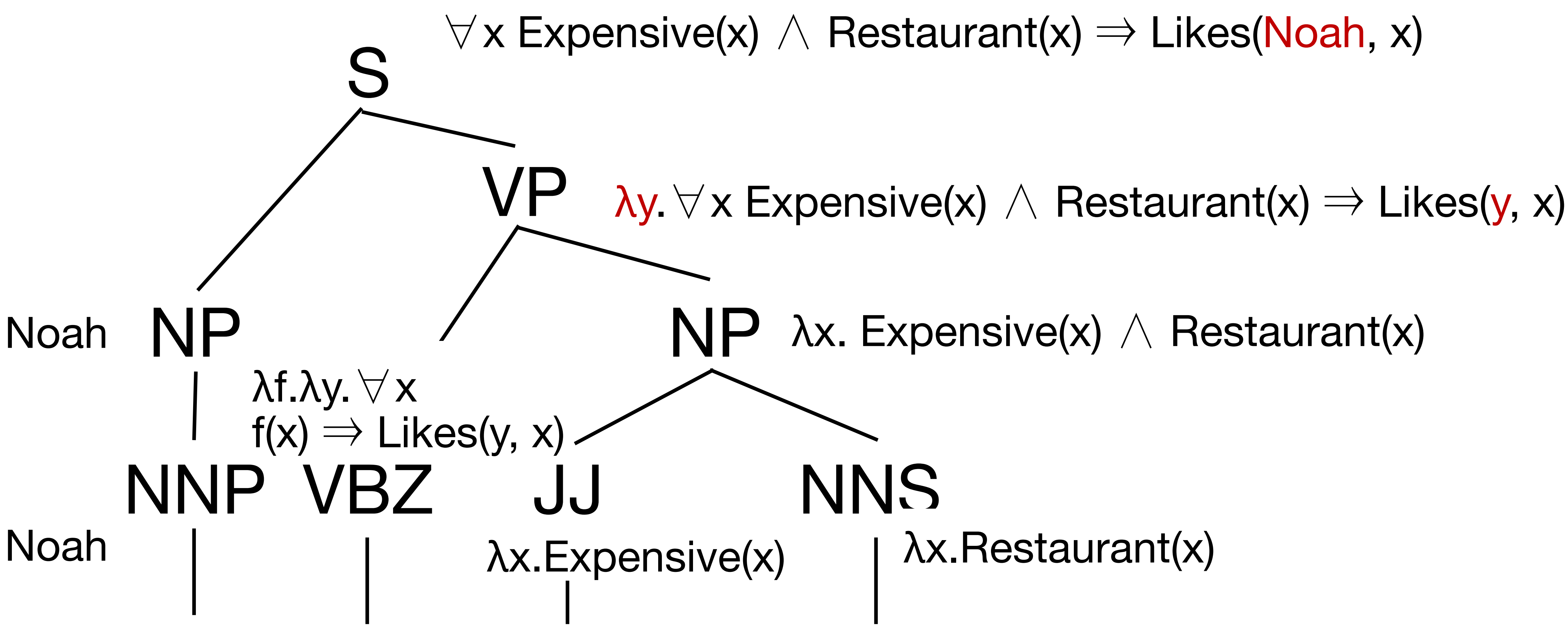
An Example

S → NP VP { VP.sem(NP.sem) }



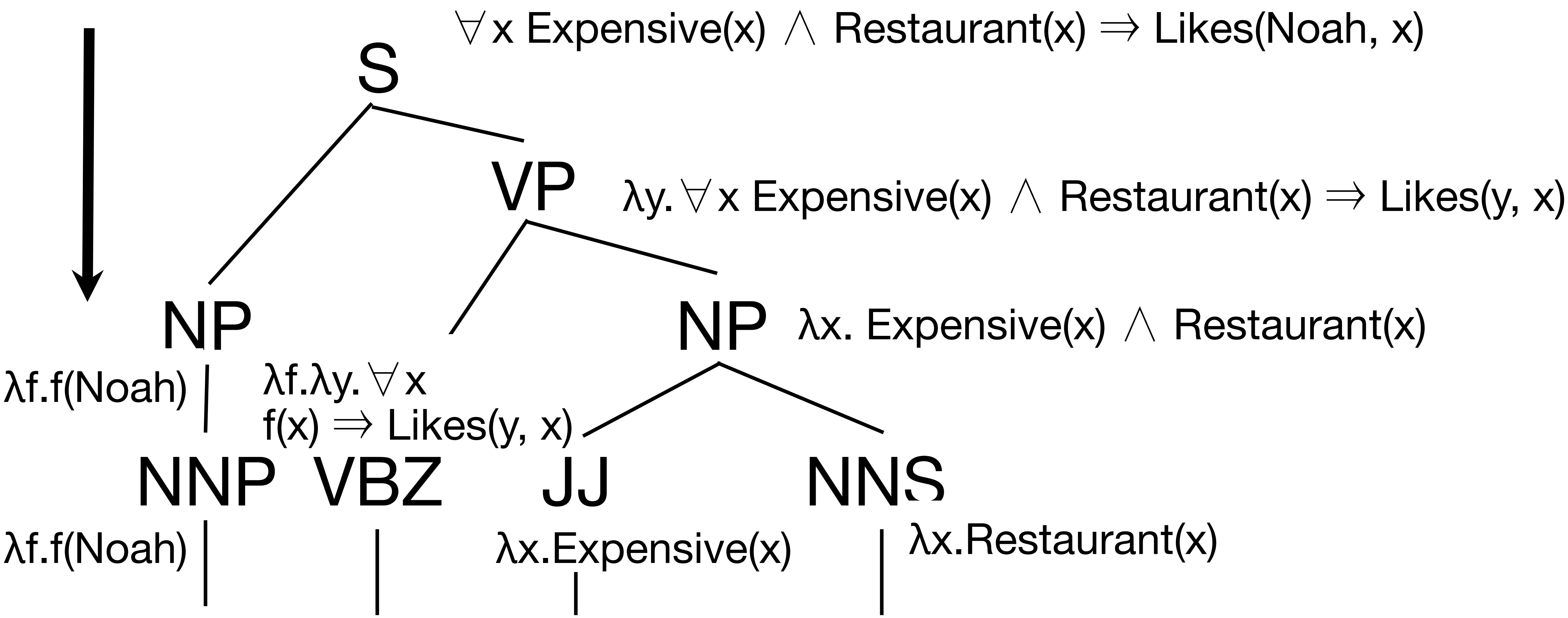
- Noah likes expensive restaurants.
- $\forall x$ Restaurant(x) $\wedge$ Expensive(x) $\Rightarrow$ Likes(Noah, x)

An Example



- Noah likes expensive restaurants.
- $\forall x$ Restaurant(x) $\wedge$ Expensive(x) $\Rightarrow$ Likes(Noah, x)

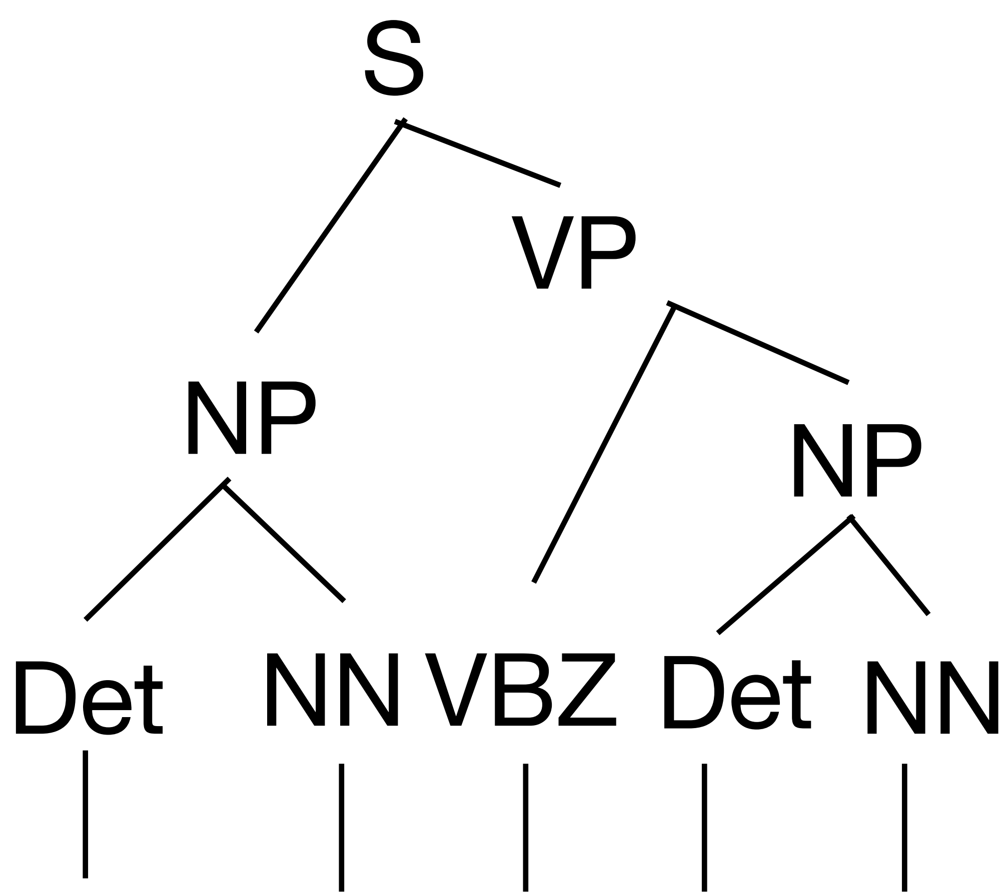
Alternative (Following *SLP*)



- Noah likes expensive restaurants.
- $\forall x\ Restaurant(x) \land Expensive(x) \Rightarrow Likes(Noah, x)$

S → NP VP { NP.sem(VP.sem) }

Quantifier Scope Ambiguity



$S \rightarrow NP\ VP\ \{\ NP.sem(VP.sem)\ \}$

$NP \rightarrow Det\ NN\ \{\ Det.sem(NN.sem)\ \}$

$VP \rightarrow VBZ\ NP\ \{\ VBZ.sem(NP.sem)\ \}$

$Det \rightarrow every\ \{\ \lambda f.\lambda g.\forall u\ f(u) \Rightarrow g(u)\ \}$

$Det \rightarrow a\ \{\ \lambda m.\lambda n.\exists x\ m(x) \wedge n(x)\ \}$

$NN \rightarrow man\ \{\ \lambda v.Man(v)\ \}$

$NN \rightarrow woman\ \{\ \lambda y.Woman(y)\ \}$

$VBZ \rightarrow loves\ \{\ \lambda h.\lambda k.h(\lambda w.\ Loves(k, w))\ \}$

- Every man loves a woman.

- $\forall u\ Man(u) \Rightarrow \exists x\ Woman(x) \wedge Loves(u, x)$

This Isn't Quite Right!

- "Every man loves a woman" really is ambiguous.
  - $\forall$u Man(u) $\Rightarrow$ $\exists$x Woman(x) $\wedge$ Loves(u, x)
  - $\exists$x Woman(x) $\wedge$ $\forall$u Man(u) $\Rightarrow$ Loves(u, x)

- This gives only one of the two meanings.
  - Extra ambiguity on top of syntactic ambiguity
- One approach is to delay the quantifier processing until the end, then permit any ordering.

Quantifier Scope

- A seat was available for every customer.
- A toll-free number was available for every customer.

- A secretary called each director.
- A letter was sent to each customer.

- Every man loves a woman who works at the candy store.
- Every 5 minutes a man gets knocked down

  and he’s not too happy about it.

What Else?

- Chapter 18 discusses how you can get this to work for other parts of English (e.g., prepositional phrases).
- Remember attribute-value structures for parsing with more complex things than simple symbols?
  - You can extend those with semantics as well.
- No time for ...
  - Statistical models for semantics
  - Parsing algorithms augmented with semantics
  - Handling idioms

Generalized Quantifiers

- In FOL, we only have universal and existential quantifiers
- One formal extension is type-restriction of the quantified variable: *Everyone likes Udipi*:

  $\forall x \; Person(x) \Rightarrow Likes(x, Udipi)$ becomes

  $\forall x \mid Person(x).Likes(x, Udipi)$
- English and other languages have a much larger set of quantifiers: *all, some, most, many, a few, the, …*
- These have the same *form* as the original FOL quantifiers with type restrictions:

  <quant><var>|<restriction>.<body>

Generalized Quantifier examples

- *Most dogs bark*

  Most x | Dog(x) . Barks(x)

- *Most barking things are dogs*

  Most x | Barks(x) . Dog(x)

- *The dog barks*

  The x | Dog(x) . Barks(x)

- *The happy dog barks*

  The x | (Happy(x) $\wedge$ Dog(x)) . Barks(x)

- ***Interpretation*** and ***inference*** using these are harder…

Speech Acts

- ***Mood*** of a sentence indicates relation between speaker and the concept (proposition) defined by the LF
- There can be operators that represent these relations:
  - ASSERT: the proposition is proposed as a fact
  - YN-QUERY: the truth of the proposition is queried
  - COMMAND: the proposition describes a requested action
  - WH-QUERY: the proposition describes an object to be identified

ASSERT (Declarative mood)

- *The man eats a peach*

  ASSERT(The x | Man(x) . (A y | Peach(y) . Eat(x,y)))

YN-QUERY (Interrogative mood)

- *Does the man eat a peach?*

  YN-QUERY(The x | Man(x) . (A y | Peach(y) . Eat(x,y)))

COMMAND (Imperative mood)

- *Eat a peach, (man).*

    COMMAND(A y | Peach(y) . Eat(*HEARER*,y))

WH-QUERY

- *What did the man eat?*

  WH-QUERY(The x | Man(x) . (WH y | Thing(y) . Eat(x,y)))

- One of a whole set of new quantifiers for wh-questions:
  - *What*: WH x | Thing(x)
  - *Which dog*: WH x | Dog(x)
  - *Who*: WH x | Person(x)
  - *How many men*: HOW-MANY x | Man(x)

Other complications

- Relative clauses are propositions embedded in an NP
  - Restrictive versus non-restrictive: *the dog that barked all night* vs. *the dog, which barked all night*
- Modal verbs: non-transparency for truth of subordinate clause: *Sue thinks that John loves Sandy*
- Tense/Aspect
- Plurality
- Etc.