CS447: Natural Language Processing

http://courses.engr.illinois.edu/cs447

Lecture 16:

More on Compositional Semantics, Verb Semantics

Julia Hockenmaier
juliahmr@illinois.edu
3324 Siebel Center

Admin

Midterm:

Regrade requests for midterm accepted until Nov 9

Points available on Compass. 22 points = 100%

Project/Literature review proposals:

Due at the end of day on Monday on Compass

One page PDF (in LaTeX, not Word) is sufficient

Include your names and NetIDs

Include all references (ideally with hyperlinks)

Explain what you want to do and why.

Include a to-do list

For projects: describe what resources you have or need.

(Use existing datasets, don't annotate your own data)

Combinatory Categorial Grammar (CCG)

CCG categories

Simple (atomic) categories: NP, S, PP

Complex categories (functions):

Return a **result** when combined with an **argument**

VP, intransitive verb	**S\NP**
Transitive verb	**(S\NP)/NP**
Adverb	**(S\NP)\(S\NP)**
Prepositions	**((S\NP)\(S\NP))/NP** **(NP\NP)/NP** **PP/NP**

CCG categories are functions

CCG has **a few atomic categories, e.g**

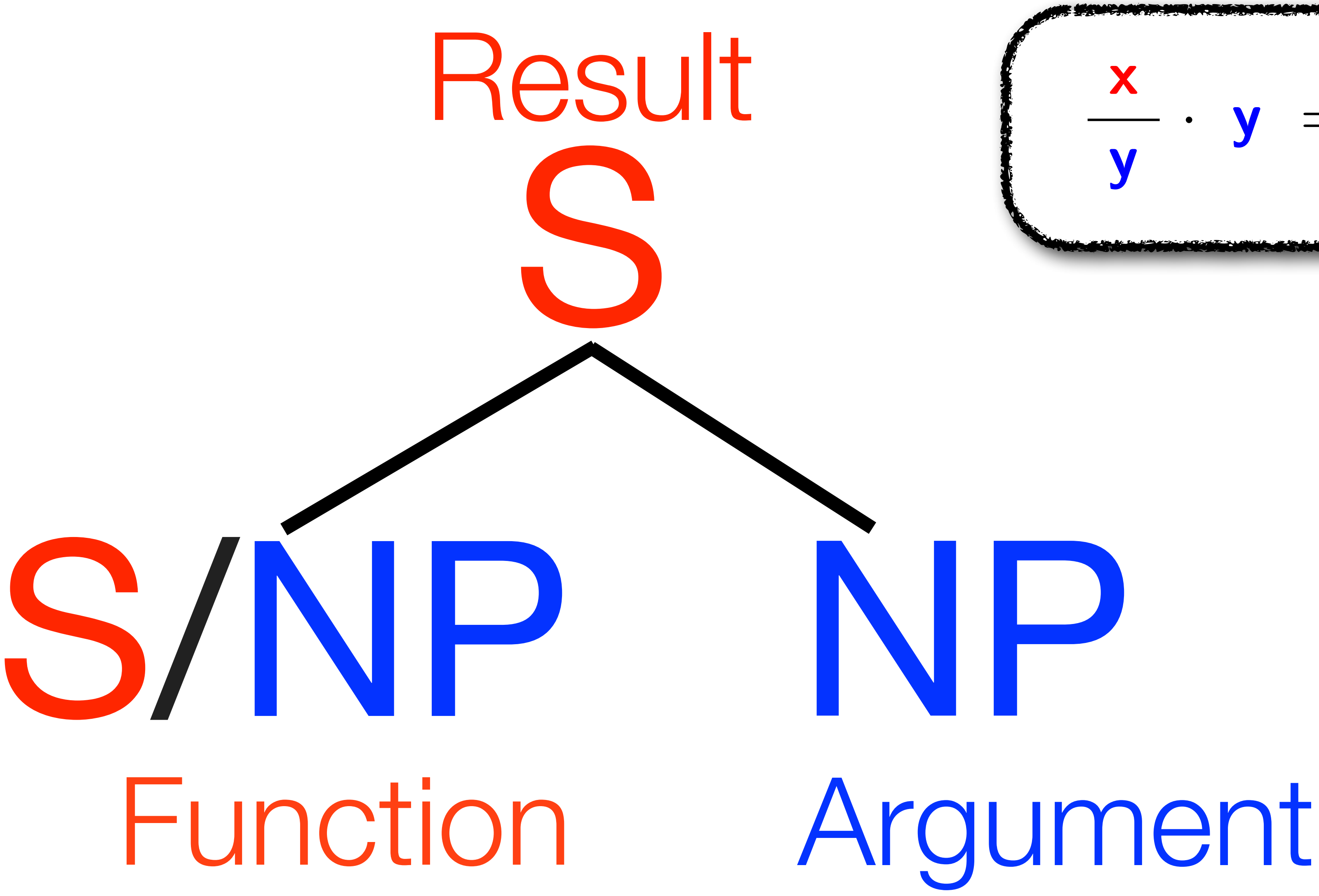
S, NP, PP

All other CCG categories are **functions:**

S / NP

Result Dir. Argument

Rules: Function application



Result

S

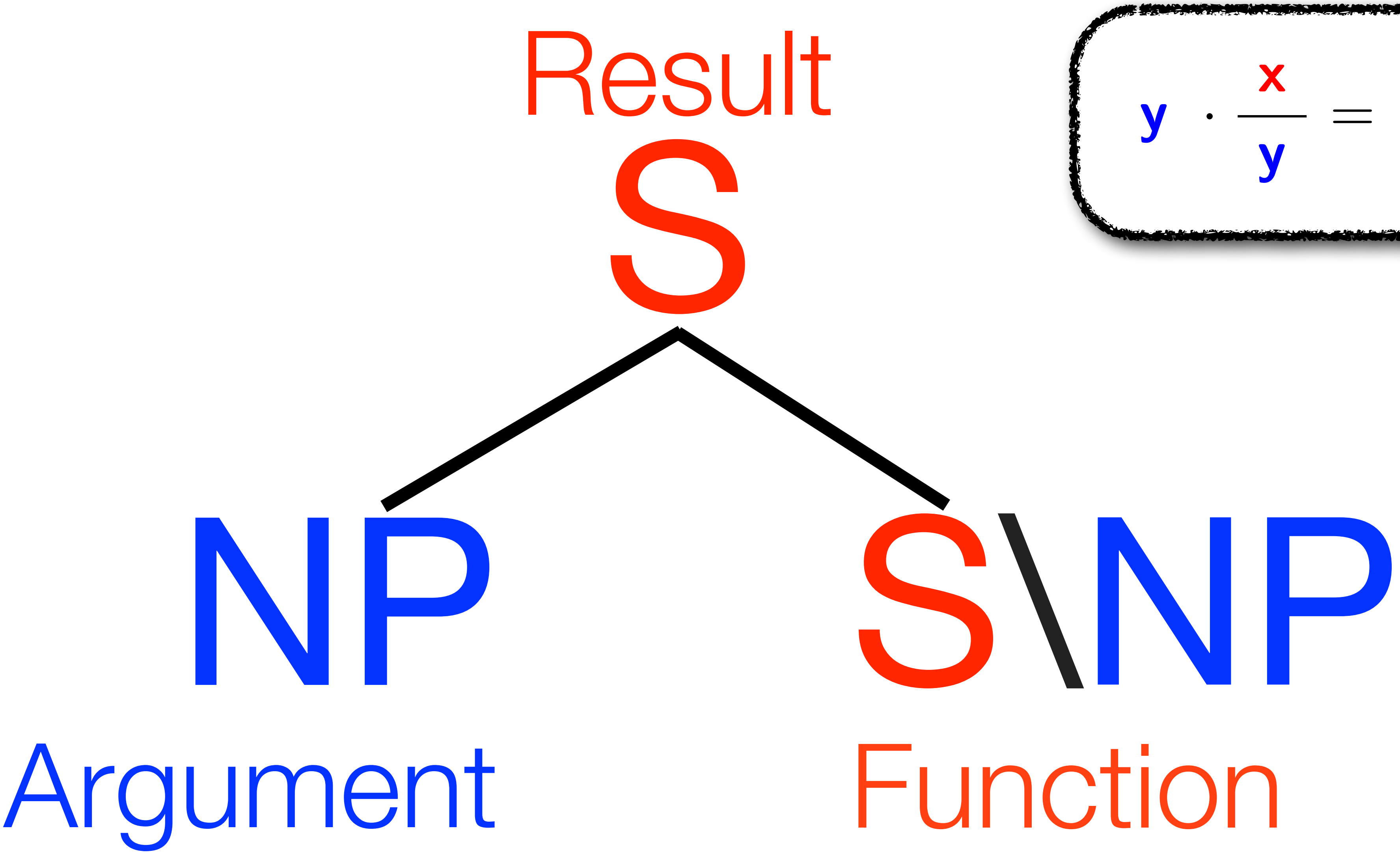
S/NP NP

Function Argument

$$\frac{x}{y} \cdot y = x$$

Rules: Function application

$$y \cdot \frac{x}{y} = x$$



Result

S

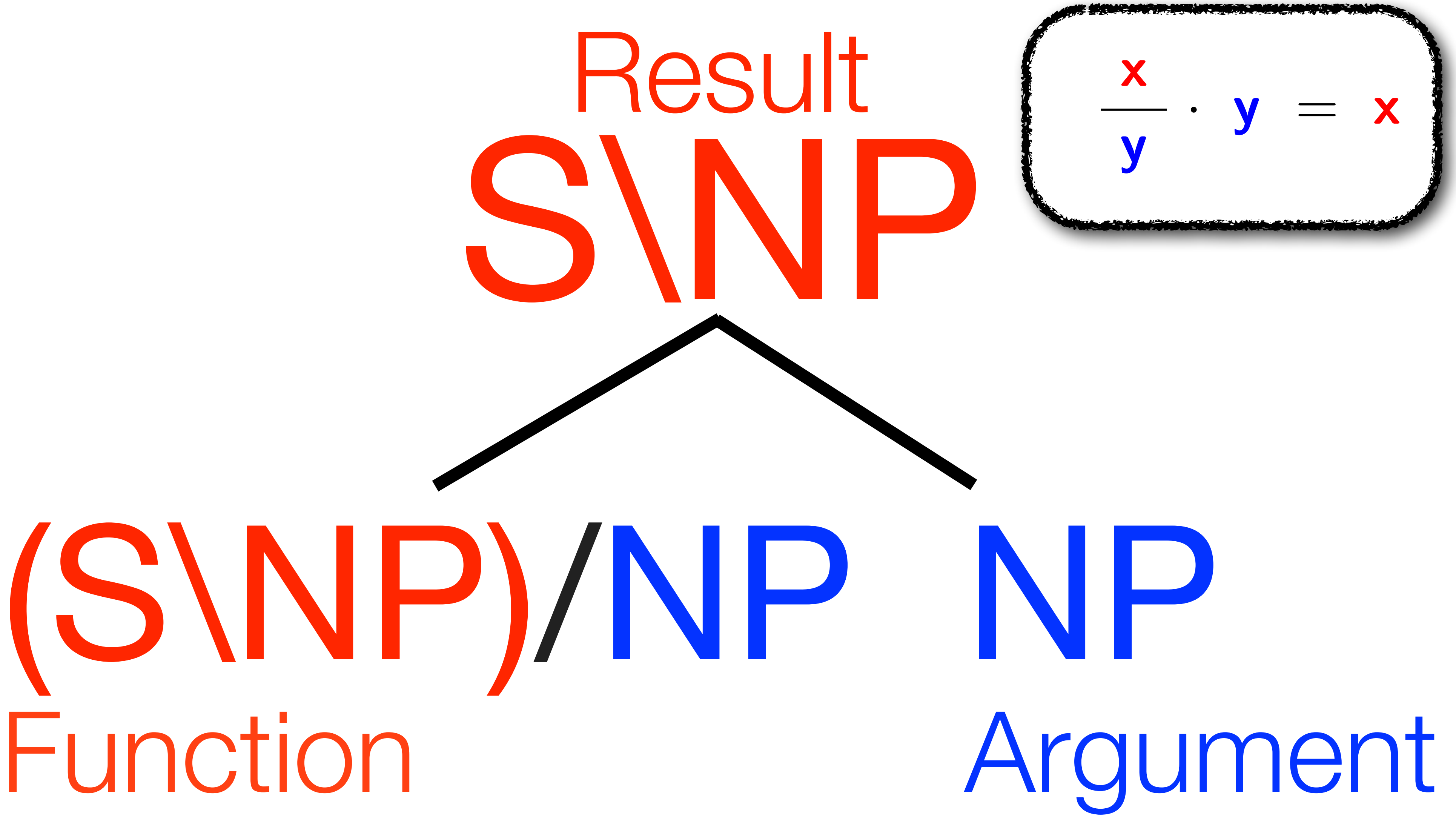
NP

S\NP

Argument

Function

Rules: Function application



Result

S\NP

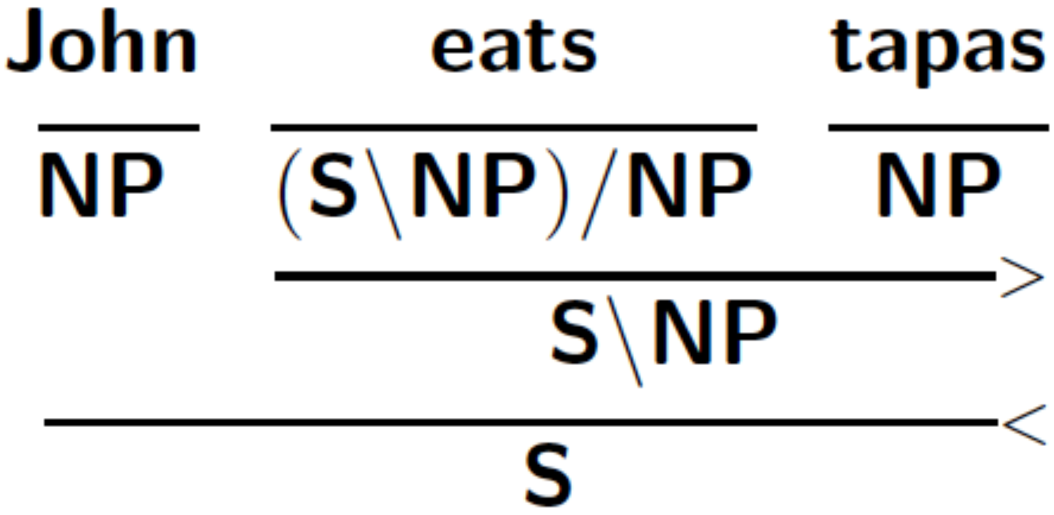
(S\NP)/NP NP

Function

Argument

$$\frac{x}{y} \cdot y = x$$

A (C)CG derivation



John eats tapas

$NP \quad (S\backslash NP)/NP \quad NP$

$>$

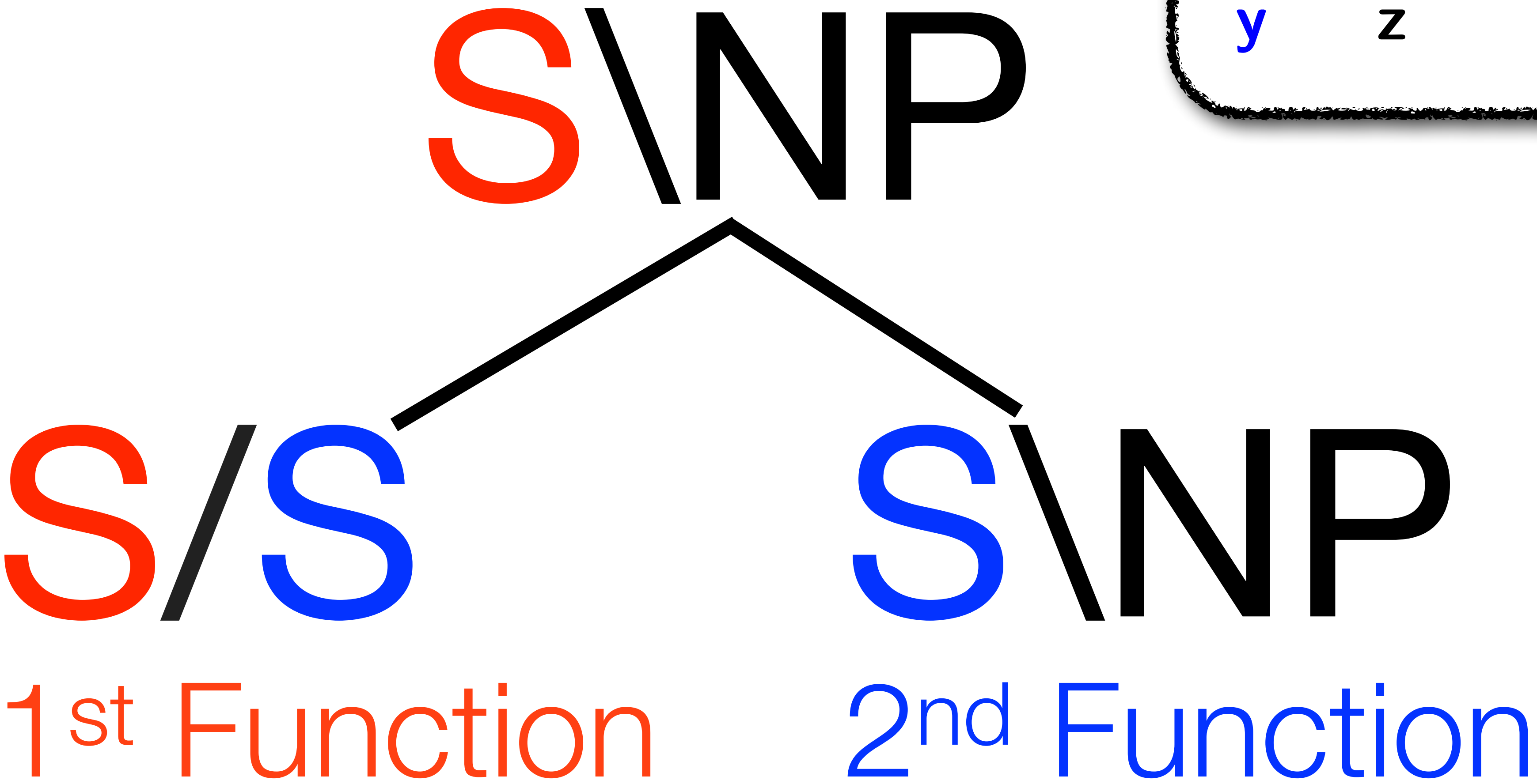
$S\backslash NP$

$<$

S

Rules: Function Composition

$$\frac{x}{y} \cdot \frac{y}{z} = \frac{x}{z}$$



S\NP

S/S

S\NP

1st Function

2nd Function

Rules: Type-Raising

S/(S\NP)

NP

$$y = \frac{x}{x} \cdot y = \frac{x}{\left(\frac{x}{y}\right)}$$

Type-raising and composition

Type-raising: X → T/(T\X)

Turns an argument into a function.

NP → S/(S\NP) (subject)
NP → (S\NP)\((S\NP)/NP) (object)

Harmonic composition: X/Y Y/Z → X/Z

Composes two functions (complex categories)

(S\NP)/PP PP/NP → (S\NP)/NP
S/(S\NP) (S\NP)/NP → S/NP

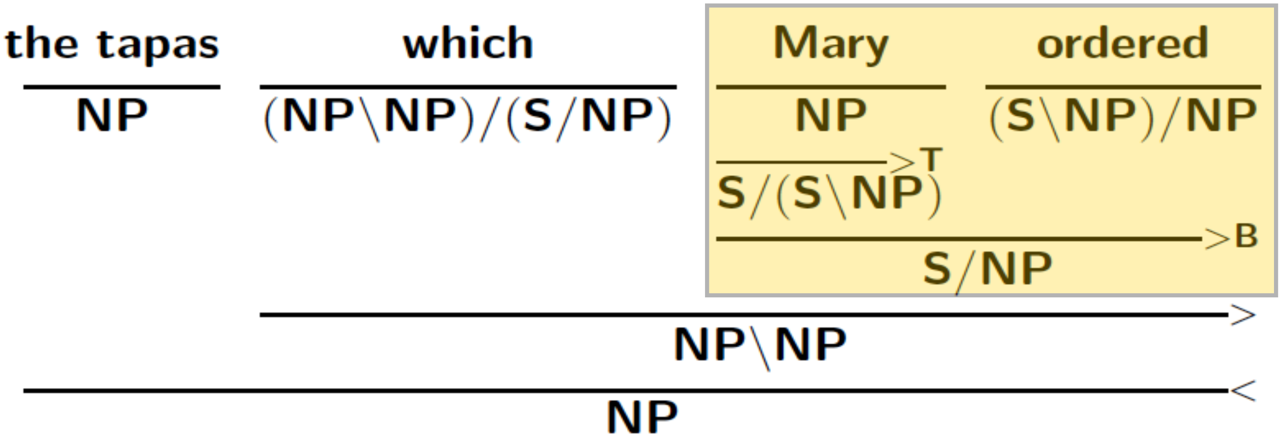
Crossing function composition: X/Y Y\Z → X\Z

Composes two functions (complex categories)

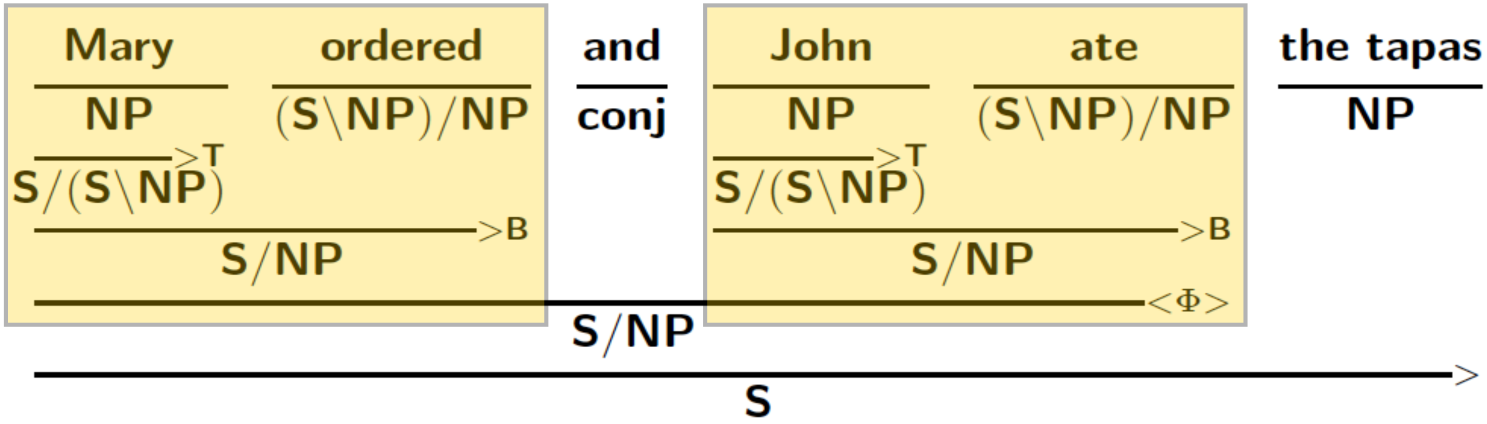
(S\NP)/S S\NP → (S\NP)\NP

Type-raising and composition

Wh-movement (relative clause):



Right-node raising:



Using Combinatory Categorial Grammar (CCG) to map sentences to predicate logic

λ-Expressions

We often use **λ-expressions** to construct complex logical formulas:

- $\lambda x.\varphi(..x...)$ is a **function** where x is a variable, and φ some FOL expression.

- **β-reduction** (called λ-reduction in textbook):
  Apply $\lambda x.\varphi(..x...)$ to some argument a:
  $(\lambda x.\varphi(..\mathrm{x}...)\ a) \Rightarrow \varphi(..\mathrm{a}...)$
  Replace all occurrences of x in $\varphi(..\mathrm{x}...)$ with a

- **n-ary functions** contain embedded λ-expressions:
  $\lambda x.\lambda y.\lambda z.give(\mathrm{x,y,z})$

CCG semantics

Every syntactic constituent has a semantic interpretation:

Every **lexical entry** maps a word to a syntactic category and a corresponding semantic type:

John=(**NP**, john') Mary= (**NP**, mary')
loves: $((\mathbf{S\backslash NP})\mathbf{/NP}\ \lambda x.\lambda y.\mathrm{loves}(x,y))$

Every **combinatory rule** has a syntactic and a semantic part:
Function application: $\mathbf{X/Y}:\lambda x.f(x)\ \ \mathbf{Y}:a \rightarrow \mathbf{X}:f(a)$
Function composition: $\mathbf{X/Y}:\lambda x.f(x)\ \ \mathbf{Y/Z}:\lambda y.g(y) \rightarrow \mathbf{X/Z}:\lambda z.f(\lambda y.g(y).z)$
Type raising: $\mathbf{X}:a \rightarrow \mathbf{T/(T\backslash X)}\ \lambda f.f(a)$

An example with semantics

$$
\begin{array}{cccl}
John & sees & Mary & \\
\cline{1-1}\cline{2-2}\cline{3-3}
\mathbf{NP} : John & (\mathbf{S}\backslash\mathbf{NP})/\mathbf{NP} : \lambda x.\lambda y.sees(x,y) & \mathbf{NP} : Mary & \\
\cline{2-3}
 & \multicolumn{2}{c}{\mathbf{S}\backslash\mathbf{NP} : \lambda y.sees(Mary,y)} & > \\
\cline{1-3}
\multicolumn{3}{c}{\mathbf{S} : sees(Mary,John)} & <
\end{array}
$$

Supplementary material: quantifier scope ambiguities in CCG

Quantifier scope ambiguity

“Every chef cooks a meal”

- **Interpretation A:**
  For every chef, there is a meal which he cooks.

$$\forall x[chef(x) \rightarrow \exists y[meal(y) \wedge cooks(y,x)]]$$

- **Interpretation B:**
  There is some meal which every chef cooks.

$$\exists y[meal(y) \wedge \forall x[chef(x) \rightarrow cooks(y,x)]]$$

Interpretation A

$$
\begin{array}{cccccl}
\text{Every} & \text{chef} & \text{cooks} & \text{a} & \text{meal} & \\
\hline
(\mathbf{S}/(\mathbf{S}\backslash\mathbf{NP}))/\mathbf{N} & \mathbf{N} & (\mathbf{S}\backslash\mathbf{NP})/\mathbf{NP} & ((\mathbf{S}\backslash\mathbf{NP})\backslash((\mathbf{S}\backslash\mathbf{NP})/\mathbf{NP}))/\mathbf{N} & \mathbf{N} & \\
\lambda P\lambda Q.\forall x[Px \rightarrow Qx] & \lambda z.chef(z) & \lambda u.\lambda v.cooks(u,v) & \lambda P\lambda Q\exists y[Py \wedge Qy] & \lambda z.meal(z) & \\
\cline{1-2}\cline{4-5}
\multicolumn{2}{c}{\mathbf{S}/(\mathbf{S}\backslash\mathbf{NP})} & & \multicolumn{2}{c}{(\mathbf{S}\backslash\mathbf{NP})\backslash((\mathbf{S}\backslash\mathbf{NP})/\mathbf{NP})} & > \\
\multicolumn{2}{c}{\lambda Q.\forall x[\lambda z.chef(z)x \rightarrow Qx]} & & \multicolumn{2}{c}{\lambda Q\exists y[\lambda z.meal(z)y \wedge Qy]} & \\
\multicolumn{2}{c}{\equiv \lambda Q.\forall x[chef(x) \rightarrow Qx]} & & \multicolumn{2}{c}{\equiv \lambda Q\lambda w.\exists y[meal(y) \wedge Qyw]} & \\
\cline{3-5}
 & & \multicolumn{3}{c}{\mathbf{S}\backslash\mathbf{NP}} & < \\
 & & \multicolumn{3}{c}{\lambda w.\exists y[meal(y) \wedge \lambda u\lambda v.cooks(u,v)yw]} & \\
 & & \multicolumn{3}{c}{\equiv \lambda w.\exists y[meal(y) \wedge cooks(y,w)]} & \\
\hline
\multicolumn{5}{c}{\mathbf{S} : \forall x[chef(x) \rightarrow \lambda w.\exists y[meal(y) \wedge cooks(y,w)]x]} & > \\
\multicolumn{5}{c}{\equiv \forall x[chef(x) \rightarrow \exists y[meal(y) \wedge cooks(y,x)]]} &
\end{array}
$$

Interpretation B

Every	chef	cooks	a	meal
$(\mathbf{S}/(\mathbf{S}\backslash\mathbf{NP}))/\mathbf{N}$	$\mathbf{N}$	$(\mathbf{S}\backslash\mathbf{NP})/\mathbf{NP}$	$(\mathbf{S}\backslash(\mathbf{S}/\mathbf{NP}))/\mathbf{N}$	$\mathbf{N}$
$\lambda P\lambda Q.\forall x[Px \rightarrow Qx]$	$\lambda z.chef(z)$	$\lambda u.\lambda v.cooks(u,v)$	$\lambda P\lambda Q\exists y[Py \wedge Qy]$	$\lambda z.meal(z)$

$$\overline{\hspace{8cm}}{>}$$

$$\mathbf{S}/(\mathbf{S}\backslash\mathbf{NP})$$
$$\lambda Q\forall x[\lambda z.chef(z)x \rightarrow Qx]$$
$$\equiv \lambda Q\forall x[chef(x) \rightarrow Qx]$$

$$\overline{\hspace{8cm}}{>}$$

$$\mathbf{S}\backslash(\mathbf{S}/\mathbf{NP})$$
$$\lambda Q\exists y[\lambda z.meal(z)y \wedge Qy]$$
$$\equiv \lambda Q\exists y[meal(y) \wedge Qy]$$

$$\overline{\hspace{12cm}}{>}\mathbf{B}$$

$$\mathbf{S}/\mathbf{NP}$$
$$\lambda w.\forall x[chef(x) \rightarrow \lambda u\lambda v.cooks(u,v)wx]$$
$$\equiv \lambda w.\forall x[chef(x) \rightarrow cooks(w,x)]$$

$$\overline{\hspace{16cm}}{<}$$

$$\mathbf{S}\exists y[meal(y) \wedge \lambda w.\forall x[chef(x) \rightarrow cooks(y,w)]x]$$
$$\equiv \exists y[meal(y) \wedge \forall x[chef(x) \rightarrow cooks(y,x)]]$$

To summarize…

Understanding sentences

“Every chef cooks a meal”

$$\forall x[chef(x) \rightarrow \exists y[meal(y) \wedge cooks(y,x)]]$$

$$\exists y[meal(y) \wedge \forall x[chef(x) \rightarrow cooks(y,x)]]$$

We translate sentences into (first-order) predicate logic.

Every (declarative) sentence corresponds to a proposition, which can be true or false.

But…

… what can we do with these representations?

Being able to translate a sentence into predicate logic is not enough, unless we also know what these predicates mean.

Semantics joke (B. Partee): The meaning of life is *life*'

Compositional formal semantics tells us how to fit together pieces of meaning, but doesn't have much to say about the meaning of the basic pieces (i.e. lexical semantics)

… how do we put together meaning representations of multiple sentences?

We need to consider discourse (there are approaches within formal semantics, e.g. Discourse Representation Theory)

… Do we really need a *complete* analysis of each sentence?

This is pretty brittle (it's easy to make a parsing mistake)

Can we get a more shallow analysis?

Semantic Role Labeling/ Verb Semantics

What do verbs mean?

Verbs describe events or states (‘eventualities’):

Tom broke the window with a rock.
The window broke.
The window was broken by Tom/by a rock.

We want to translate verbs to predicates.
But: a naive translation (e.g. subject = first argument, object = second argument, etc.) does not capture the differences in meaning

```
break(Tom, window, rock)
break(window)
break(window, Tom)
break(window, rock)
```

Semantic/Thematic roles

Verbs describe events or states ('eventualities'):

Tom broke the window with a rock.
The window broke.
The window was broken by Tom/by a rock.

Thematic roles refer to participants of these events:

Agent (who performed the action): Tom
Patient (who was the action performed on): window
Tool/Instrument (what was used to perform the action): rock

Semantic/thematic roles (agent, patient) are different from grammatical roles (subject or object).

The inventory of thematic roles

We need to define an inventory of thematic roles

To create systems that can identify thematic roles automatically, we need to create labeled training data.

It is difficult to give a formal definition of thematic roles that generalizes across all verbs.

PropBank and FrameNet

Proposition Bank (PropBank):
Very coarse argument roles (arg0, arg1,…),
used for all verbs (but interpretation depends on the specific verb)

Arg0 = proto-agent
Arg1 = proto-patient
Arg2...: specific to each verb
ArgM-TMP/LOC/...: temporal/locative/... modifiers

FrameNet:

Verbs fall into classes that define different kinds of frames
(`change-position-on-a-scale` frame: rise, increase,...).
Each frame has its own set of “frame elements” (thematic roles)

PropBank

agree.01 Arg0: Agreer Arg1: Proposition
Arg2: Other entity agreeing
[$_{\text{Arg0}}$ The group] agreed [$_{\text{Arg1}}$ it wouldn't make an offer]
[$_{\text{Arg0}}$ John] agrees with [$_{\text{Arg2}}$ Mary]

fall.01 Arg1: patient/thing falling Arg2: extent/amount fallen
Arg3: start point Arg4: end point
[$_{\text{Arg1}}$ Sales] fell [$_{\text{Arg4}}$ to $251 million]
[$_{\text{Arg1}}$ Junk bonds] fell [$_{\text{Arg2}}$ by 5%]

Semantic role labeling: Recover the semantic roles of verbs (nowadays typically PropBank-style)

Machine learning; trained on PropBank
Syntactic parses provide useful information

Diathesis Alternations

Active/passive alternation:

Tom **broke** the window with a rock. (active voice)

The window **was broken** by Tom/by a rock. (passive voice)

Causative alternation:

Tom **broke** the window. ('causative'; active voice)

The window **broke**. ('anticausative'/'inchoative'; active voice)

Dative alternation

Tom **gave** the gift to Mary.

Tom **gave** Mary the gift.

Locative alternation:

Jessica **loaded** boxes into the wagon.

Jessica **loaded** the wagon with boxes.

Verb classes

Verbs with similar meanings undergo the same syntactic alternations, and have the same set of thematic roles (Beth Levin, 1993)

VerbNet (verbs.colorado.edu; Kipper et al., 2008)
A large database of verbs, their thematic roles and their alternations