Description

This course prepares students for graduate-level coursework in semantics and the syntax/semantics interface. It starts with building a solid foundation in predicate logic and elements of the lambda calculus, and moves on to use Pauline Jacobson's *Compositional Semantics* textbook. Jacobson adopts an approach whereby the syntax builds the expressions while the semantics simultaneously assigns each a model-theoretic interpretation. Alongside this approach, the author also presents a competing view that makes use of an intermediate level, Logical Form. The course will discuss questions and relative clauses, quantifiers in various positions, and binding.

Students are encouraged to engage in research projects on related problems and to discuss them with both instructors. (Office hours by appointment.)

Preparation

See the README file in Resources. Become proficient in the prerequisite material by the beginning of the semester, and email me your solutions for the set theory problem set by January 18.

Textbooks (not all chapters)

  *scanned copies to be posted in Classes*
  *scanned copies to be posted in Classes*
  *acquire your own copy from Bobst, the Book Store, or the Internet*
- Shorter readings of individual articles will supplement the above, to be posted in Classes.

Requirements

- Weekly reading assignments, 30 to 40 pages on average.
- Approximately 9 problem sets or sets of exercises.
- A midterm exam.
- A take-home exam or a short term paper, less than 20 pages.

Weighting

- Active participation in classroom discussion, reflecting reading preparation, 25%
- Logic problem sets and midterm, 25%
- Linguistics problem sets and exercises, 25%
- Take-home exam or term paper, 25%
Calendar

This calendar is truly preliminary. We will slow down, speed up, skip, or digress as appropriate in view of the class participants’ interests. You are invited and encouraged to suggest such modifications.

Propositional logic (1-23, 1-25)

- *Written assignment:* problem set written by instructor / from Gamut I, Ch 2.

Predicate logic (1-30, 2-2, 2-6)

- *Written assignment:* problem set written by instructor / from Gamut I, Ch 3.

The lambda operator, types, combinators (2-8, 2-13, 2-15)

- *Written assignment:* problem set written by instructor / from Gamut II, Ch 4.

Midterm, covering all the above (2-22)

Towards a model theoretic, compositional semantics of English (2-27, 3-1)

  - Ch 2: Semantic Foundations, pp. 27-42.
  - Ch 3: Compositionality, Direct Compositionality, and the syntax/semantics interface, pp. 42-52.

Categories, types, and an apparent mismatch (3-6, 3-8)

- Jacobson, Ch 4: Expanding the fragment: Syntactic categories and semantic types, pp. 52-66.
- *Written assignment:* exercises from textbook or based on literature

Spring break

Categorial Grammar (3-20, 3-22)

- *Written assignment:* exercises from textbook or based on literature

Generalized quantifiers (3-27, 3-29)

- Jacobson, Ch 11: Ordinary NPs and type lifting, pp. 181-198.
- Jacobson, Ch 12: Generalized conjunction, pp. 198-206.
- *Written assignment:* exercises from textbook or based on literature

Comparing approaches, first round (4-3, 4-5)

- Jacobson, Ch 13: Relative clauses: Sketching two accounts, pp. 207-244.
- *Written assignment:* exercises from textbook or based on literature
Comparing approaches, second round (4-10, 4-12)
  Jacobson, Ch 14: Generalized quantifiers in object position: Two approaches, pp. 244-274.
  Written assignment: exercises from textbook or based on literature

Comparing approaches, third round (4-17, 4-19)
  Jacobson, Ch 15: The interpretation of pronouns: Two accounts, pp. 274-323.
  Written assignment: exercises from textbook or based on literature

Questions, from Hamblin to Inquisitive Semantics
  Karttunen (1977), The syntax and semantics of questions. Linguistics & Philosophy.

Continued discussion and summary (4-24, 4-26, 5-1, 5-3, 5-8)
  These class periods are set aside to ensure that we can spend more time on the previous topics if needed.
  We also summarize and discuss the general enterprise.
  Jacobson, Appendices: The full fragment, pp. 323-333.
Advanced Semantics
Szabolcsi

Propositional Logic

Propositional logic, coarse-grained: Every robot walks and Susan talks
Predicate logic, more fine-grained: Every robot walks and Susan talks

Syntax
Propositional variables p, q, r, ...
Your favorite set of connectives &, ~, →, ...
Every propositional variable is a formula.
If α, β are formulas, and # is a 2-place connective,
then (α#β) is a formula.
If α is a formula, and @ is a 1-place connective,
then (@α) is a formula.
Nothing else is a formula.

Semantics
Just two semantic values: t and f.
Each proposition can be characterized with its truth-set: the set of worlds in which it is true.
The truth-set of a tautology is the universal set U.
The truth-set of a contradiction is the empty set Ø.

The connectives ∨, &, and ~ of propositional logic correspond to
the set theoretic operations ∪, ∩, and −− on their truth-sets.

Truth tables

<table>
<thead>
<tr>
<th>p</th>
<th>q</th>
<th>p&amp;q</th>
<th>p</th>
<th>q</th>
<th>p∨q</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>t</td>
<td>t</td>
<td>t</td>
<td>t</td>
<td>t</td>
</tr>
<tr>
<td>2</td>
<td>t</td>
<td>f</td>
<td>f</td>
<td>f</td>
<td>t</td>
</tr>
<tr>
<td>3</td>
<td>f</td>
<td>t</td>
<td>f</td>
<td>t</td>
<td>t</td>
</tr>
<tr>
<td>4</td>
<td>f</td>
<td>f</td>
<td>f</td>
<td>f</td>
<td>f</td>
</tr>
</tbody>
</table>

| p | q | p→q | p | q | ~q | p | q | p|q |
|---|---|-----|---|---|----|---|---|----|
| t | t | t | t | t | f | t | t | f |
| t | f | f | t | f | t | t | f | t |
| f | t | t | f | t | t | f | t | t |
| f | f | t | f | f | t | f | t | t |
Some important tautologies (valid, or always true, propositions), see p.55 of Allwood et al.

Negation is often written as $\neg$, instead of $\sim$.

(i) $p \lor \neg p$ (either $p$ or its negation is true: Excluded Middle)
(ii) $\neg(p \land \neg p)$ ($p$ and its negation can't be true together)
(iii) $p \equiv p$ (everything is identical to itself)
(xiv) $\neg\neg p \equiv p$ (two negations [reversals] cancel out)
(xv) $p \lor p \equiv p$ (the union of $P$ with itself is $P$)
(xvi) $(p \equiv q) \equiv (\neg p \equiv \neg q)$ (negating both sides of an equation preserves the equation)

De Morgan Laws (conjunction and disjunction are duals)

(vi) $\neg(p \lor q) \equiv \neg p \land \neg q$
(vii) $\neg(p \land q) \equiv \neg q \lor \neg p$

Expressing material implication using negation+disjunction or negation+conjunction:

(xiii) $(p \rightarrow q) \equiv (\neg p \lor q)$ (implication true iff antecedent is false or consequent is true)
(x) $\neg(p \rightarrow q) \equiv (p \land \neg q)$ (implication false iff antecedent is true and consequent is false)

Distributivity

$((p \lor q) \land r) \equiv ((p \land r) \lor (q \land r))$
$((p \land q) \lor r) \equiv ((p \lor r) \land (q \lor r))$

Interdefinability, functional completeness

Propositional logic has $2^4 = 16$ distinct connectives. They are all definable in terms of $\land, \neg$ or $\lor, \neg$ or other small sets of primitives. These definitions can be checked (proved) using Venn-diagrams or truth-tables (or by indirect reasoning or the tableau method).

Define $p \land q$ using $\{\lor, \neg\}$

Define $p \lor q$ using $\{\land, \neg\}$

Define $p \rightarrow q$ using $\{\lor, \neg\}$

Define $p \rightarrow q$ using $\{\land, \neg\}$

Define $p \land q$ using $\{|\}$
Truth, entailment, equivalence, tautology/contradiction

\[(p \& q) \rightarrow p\]

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>t</td>
<td>t</td>
<td>t</td>
<td>t</td>
<td>t</td>
</tr>
<tr>
<td>f</td>
<td>f</td>
<td>f</td>
<td>t</td>
<td>t</td>
</tr>
</tbody>
</table>

\[\sim p \rightarrow (p \lor q)\]

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>t</td>
<td>t</td>
<td>f</td>
<td>t</td>
<td>t</td>
</tr>
<tr>
<td>t</td>
<td>f</td>
<td>f</td>
<td>t</td>
<td>t</td>
</tr>
<tr>
<td>f</td>
<td>t</td>
<td>t</td>
<td>t</td>
<td>t</td>
</tr>
<tr>
<td>f</td>
<td>f</td>
<td>f</td>
<td>f</td>
<td>f</td>
</tr>
</tbody>
</table>

Indirect reasoning

<table>
<thead>
<tr>
<th>Level</th>
<th>((p \lor q) &amp; r) \rightarrow p)</th>
<th>Level</th>
<th>(((\sim p \rightarrow (q \lor r)) &amp; (s \rightarrow r)) &amp; s) \rightarrow p)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1)</td>
<td>t</td>
<td>(1)</td>
<td>t</td>
</tr>
<tr>
<td>(2)</td>
<td>t</td>
<td>(2)</td>
<td>t</td>
</tr>
<tr>
<td>(3)</td>
<td>f</td>
<td>(3)</td>
<td>t</td>
</tr>
<tr>
<td>(4)</td>
<td>t</td>
<td>(4)</td>
<td>t</td>
</tr>
<tr>
<td>(5)</td>
<td>t</td>
<td>(5)</td>
<td>f</td>
</tr>
<tr>
<td>(6)</td>
<td>t</td>
<td>(6)</td>
<td>f</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>f*</td>
</tr>
</tbody>
</table>

The assumption of main implication to be false does not lead to a contradiction: the formula is not a tautology.

The assumption of main implication to be false leads to a contradiction: the formula is a tautology.

Analytic tableaux

Can \(((a \lor \neg b) \land \neg a)\) be true?

No, all branches are closed: a contradiction.

Can \(((a \land c) \land (\neg a \lor b))\) be true?

Yes, when \(b, a, c\) are all true.

(The extended tableau method offers a proof theory for predicate logic.)
Your name:

Assignment #1, Propositional Logic

Read the whole of Ch 4 in Allwood et al. (it contains some bits and pieces that we don’t discuss in class), and do the exercises below.

Please type your solutions into this file, observing the PAGE BREAKS, to make it easier for me to look your answers through. Don’t use a font smaller than 11 pt. Please bring a printout on Monday.

Exercises

1. Which of the following expressions is ambiguous and thus ill-formed? Why are the others not ambiguous, in view of semantics and/or conventions?
   
   \( p \land q \land r \)

   \( \neg p \land q \)

   \( p \land q \lor r \)

2. Below you see the truth tables of two propositional connectives, notated as \( \# \) and \( \$ \).

   (a) Insert a shape (say, \( \bigcirc \) ) into those areas of the Venn diagrams where \( p\#q \) and \( p\$q \) are true. (Easier than shading.)
(b) Is $\#$ identical to one of the connectives discussed in Allwood et al. 4.2? If yes, which?

(c) Is $\$$ identical to one of the connectives discussed in 4.2? If yes, which?

(d) The contribution of the new connective can be defined using a combination of familiar connectives. Express the compound proposition involving the new connective using $p$, $q$, two or more of the familiar connectives, and parentheses, as necessary.

3. The following equivalence can be easily explained using set theory (t for 'tautology,' f for 'contradiction'):

(a) \( t \land p \equiv p \)
The conjunction of a tautology $t$ and a contingent proposition $p$ has the same truth value as $p$. Explanation: Let the truth set of $p$ be $P$.
Since $t$ is a tautology, and a tautology is true in every world, $P \subseteq T$.
We know that whenever $A \subseteq B$, $A \cap B = A$.
Hence, $P \cap T = P$.

Explain (b)-(c)-(d) similarly, using set theory:

(b) \( p \lor t \equiv t \)

(c) \( p \land f \equiv f \)
4. Do the exercises at the end of Ch 4 of Allwood et al, and check your solutions against the book's (in a separate PDF). Alert me if your solution doesn't agree with the book's and you don't see why the book is right, but you don't need to write your solutions out for me, with the exception of the last problem:

8. Determine by indirect reasoning whether the following expressions are tautologies.

   (a) \( p \to (q \to (r \to (s \to (t \to p))) \))
   (b) \( ((p \equiv q) \& (q \equiv r)) \to (p \equiv r) \)
   (c) \( (p \& (q \equiv r)) \to ((p \& q) \equiv r) \)

5. Optional (but very much encouraged): Show how \& and \( \lor \) can be expressed using just Sheffer's stroke | (which means 'not both'). Sheffer's stroke is not the only guy that can do this job, and do it elegantly. Can you guess the meaning of the other guy? -- Please write out your answers.

6. Optional. If you want to do a tad more ambitious exercises, see the PDF some propositional logic exercises, Gamut vol 1 in course readings, and the fully worked out solutions to propositional and predicate logic exx, Gamut vol 1. An excellent self-teaching device. You don’t need to hand your answers in.

(d) \( p \lor f \equiv p \)
John walks: \( W(j) \)

John walks and talks: 

John or Mary talks: 

John saw Mary: 

Something barks: \( \exists x[B(x)] \)

Something is a dog and it barks =  
Something that is a dog barks =  
Some dog barks: \( \exists x[D(x) \& B(x)] \)

Some dog barks or growls: 

Some dog does not bark: 

No dog barks: 

John saw some dog: 

Everything changes: \( \forall x[C(x)] \)

Everything changes if it is a rule =  
Everything that is a rule changes =  
Every rule changes: \( \forall x[R(x) \rightarrow C(x)] \)

Every rule changes or gets eliminated: 

Not every rule changes: 

No rule changes: 

Every dog saw John: 

When translating English to predicate logic, we combine \( \exists \) with \& to approximate the content of some, and \( \forall \) with \( \rightarrow \) to approximate the content of every. But in predicate logic qua logic, any quantifier may occur with any connective, and the formulae will mean whatever the combination is supposed to mean, e.g.

\[ \forall x[A(x) \& B(x)] \quad \text{Everything is both an A and a B} \quad \neq \quad \text{Everything that is an A is a B.} \]
Add identity $=$ and non-identity $\neq$ (not part of the Allwood et al. system)

John likes Homer, and
Bill likes somebody else [than Homer]. \[ L(j, h) \land \exists x [x \neq h \land L(b, x)] \]

John likes Homer, and
Bill likes everybody else [than Homer].

John likes himself.

John likes only himself.

How to express numerical quantification?

At least one dog barks. \[ \exists x [D(x) \land B(x)] \]

At least two dogs bark. \[ \exists x \exists y [x \neq y \land D(x) \land D(y) \land B(x) \land B(y)] \]

At least three dogs bark.

Exactly one dog barks.

One dog barks and nothing else does.

The dog barks.

The dog barks and nothing else does.
Free vs. bound variables, duals

- \( \text{blue}(x) \not= \text{blue}(y) \)  
  Why?

Pointing with free variables:

This is blue  'the entity I am pointing to (with my left hand) is blue'  \( \text{blue}(x) \)

That is blue  'the entity I am pointing to (with my right hand) is blue'  \( \text{blue}(y) \)

The free variables \( x \) and \( y \) may accidentally point to the same object, but in general, you cannot assume that they do so. Free variables have their own individual identity.

- \( \forall x[\text{blue}(x)] = \forall y[\text{blue}(y)] \quad \exists x[\text{blue}(x)] = \exists y[\text{blue}(y)] \)  
  Why?

Making general statements with quantifiers using bound variables:

Everything is blue  'Whatever I point to, that entity is blue'

\[ \forall x[\text{blue}(x)] \text{ or } \forall y[\text{blue}(y)] \]

Something is blue  'I can point to an entity that is blue'

\[ \exists x[\text{blue}(x)] \text{ or } \exists y[\text{blue}(y)] \]

A bound variable is a mere placeholder. The sentence says nothing specifically about \( x \) or \( y \).

- **Duals** (= related to each other via negation in a particular way)

\( \exists \) and \( \forall \) are duals.  

Why?

Imagine a universe with 3 elements: \( a, b, c \).

\[ \exists x[\text{blue}(x)] \equiv \text{blue}(a) \lor \text{blue}(b) \lor \text{blue}(c) \]

\[ \forall x[\text{blue}(x)] \equiv \text{blue}(a) \land \text{blue}(b) \land \text{blue}(c) \]

The propositional connectives \( \lor \) and \( \land \) are duals; see the de Morgan laws. \( \exists \) and \( \forall \) inherit duality from \( \lor \) and \( \land \).

\[ \neg(\text{blue}(a) \land \text{blue}(b) \land \text{blue}(c)) \equiv \neg\text{blue}(a) \lor \neg\text{blue}(b) \lor \neg\text{blue}(c) \]

\[ \neg(\text{blue}(a) \lor \text{blue}(b) \lor \text{blue}(c)) \equiv \neg\text{blue}(a) \land \neg\text{blue}(b) \land \neg\text{blue}(c) \]

therefore,

\[ \neg\forall x[\text{blue}(x)] \equiv \exists \neg x[\text{blue}(x)] \quad \text{and} \quad \neg\exists x[\text{blue}(x)] \equiv \forall \neg x[\text{blue}(x)] \]

\[ \neg\forall \neg x[\text{blue}(x)] \equiv \exists x[\text{blue}(x)] \quad \text{and} \quad \neg\exists \neg x[\text{blue}(x)] \equiv \forall x[\text{blue}(x)] \]
Abbreviate predicates with the first letter of the predicate; you don’t need to give keys.

(1)  
(a) Express the meaning of No man flies, using

the quantifier \( \forall \) (and whatever else it takes): ____________________________

the quantifier \( \exists \) (and whatever else it takes): ____________________________

(b) Express the meaning of Not every man flies, using

the quantifier \( \forall \) (and whatever else it takes): ____________________________

the quantifier \( \exists \) (and whatever else it takes): ____________________________

(c) Express the meaning of Some man flies, using

the quantifier \( \forall \) (and whatever else it takes): ____________________________

the quantifier \( \exists \) (and whatever else it takes): ____________________________

(2) Which of these equivalences are tautologies? (I.e. when do the two sides say the same, no matter what?) Briefly explain why (or why not).

Always evaluate the free variables on both sides of an equation with respect to the same pointing (=assignment of values to free variables.) Below, assume everywhere that \( x \Rightarrow \text{Arthur} \) and \( y \Rightarrow \text{Ford} \).

(a) \( \forall x [F(x) \& G(y)] \equiv \forall z [F(z) \& G(y)] \)

(b) \( \forall x [F(x)] \& G(x) \equiv \forall y [F(y)] \& G(y) \)

(c) \( \forall x [F(x) \& G(x)] \equiv \forall y [F(y) \& G(y)] \)

(d) \( \forall x [F(x) \& G(x)] \equiv \forall y [F(y) \& G(x)] \)
The semantics of predicate logic

The universe consists of three dogs and four cats; arrows correspond to barking at something:

- Every dog barked at some cat (on the direct scope reading, Subject > Object)

\( \forall x[D(x) \rightarrow \exists y[C(y) \& B(x, y)]] \) is true, because

for every individual in the universe, if it is a dog, then we find a cat that it barked at, and if it is not a dog, it does not matter whether we find a cat that it barked at:

(i) \( D(a) \) and \( C(d) \) and \( B(a, d) \) and
(ii) \( D(b) \) and \( C(e) \) and \( B(b, e) \) and
(iii) \( D(c) \) and \( C(g) \) and \( B(c, h) \) and
(iv) \( D(d), D(e), D(f) \) and \( D(h) \) are false.

- Every dog barked at some cat (on the inverse scope reading, Object > Subject)

\( \exists y[C(y) \& \forall x[D(x) \rightarrow B(x, y)]] \) is false, because

we do not find any individual that is a cat and every dog barked at it. It would be true if, for example, each of \( a, b, \) and \( c \) had barked at \( f \).

Strategy: Working “from outside in”, cash out each quantifier in terms of individuals.

If the quantifier is universal, check whether its scope holds true for every individual that can be assigned to the variable that the quantifier binds. Cf. \( \forall x[F(x)] = F(a) \& F(b) \& ... \& F(h) \).

If the quantifier is existential, check whether its scope holds for at least one individual that can be assigned to the variable that the quantifier binds. Cf. \( \exists y[G(y)] = G(a) \lor G(b) \lor ... \lor G(h) \).
(3) Work through Exx 14, 15, and 16 using the book’s solutions as a guide. You don’t need to hand them in, but let me know if any of them is not clear.

For Ex 15, note that the book represents the “like” relation as a set of pairs, where the first member of the pair is the liker and the second is the liked one. I don’t know why there are no commas between the pairs enclosed in curly brackets (typo?).

So the formal description of the model M (universe D plus interpretation function I) in Ex 15 corresponds to the following picture. I do not indicate Boy={a,b} and Girl={c,d} so as not to muddle up the drawing.

\[
\begin{align*}
\text{L} &= \{<a,a>, <b,b>, <c,c><a,c>, <a,d>, <b,c>, <c,a>, <d,a>\} \\
\end{align*}
\]

(4) Which of the following are valid (true in all worlds)? Construct models that falsify them, if you can.

(a) \((\forall x[D(x) \rightarrow B(x)] \land D(a)) \rightarrow B(a)\)

(b) \((\exists x[D(x) \land B(x)] \land D(a)) \rightarrow B(a)\)

(c) \((\forall x \forall y[H(x,y)]) \rightarrow (\forall y \forall x[H(x,y)])\)

(d) \((\exists x \exists y[L(x,y)]) \rightarrow (\exists y \exists x[L(x,y)])\)

(e) \((\forall x \exists y[L(x,y)]) \rightarrow (\exists y \forall x[L(x,y)])\)

(f) \((\exists y \forall x[L(x,y)]) \rightarrow (\forall x \exists y[L(x,y)])\)

(g) \((\forall x[A(x) \lor B(x)]) \rightarrow (\forall x[A(x)] \lor \forall x[B(x)])\)

(h) \((\forall x[A(x) \land B(x)]) \rightarrow (\forall x[A(x)] \land \forall x[B(x)])\)
Compositionality

The truth conditions of a formula of predicate logic are uniquely determined by the truth conditions of its constituent parts.

\( G(c) \) is true iff the individual that \( c \) refers to is in the extension of \( G \).

\( \exists y [G(y)] \) is true iff at least one individual that can be assigned to variable \( y \) is in the extension of \( G \).

\( \exists y [G(y) \& H(y) \& K(x)] \) is true iff at least one individual that can be assigned to variable \( y \) is in the extension of \( G \) and in the extension of \( H \), and the individual that is currently assigned to \( x \) (the individual that \( x \) is "pointing at") is in the extension of \( K \).

Two ways of saying the same thing (every/some individual ... = every/some assignment...)

For every individual \( x \), \( F(x) \) is true =
   Every way of assigning an individual to variable \( x \) makes \( F(x) \) true.
For some individual \( y \), \( G(y) \) is true =
   There is at least one way of assigning an individual to variable \( y \) that makes \( G(y) \) true.

Model theoretic semantics:

The truth of each formula is determined with respect to a particular model \( M \) and an assignment \( g \) of individuals to variables.

In predicate logic, a model consists of a set \( D \) of individuals (the universe of discourse, or domain) and a specification \( I \) of what individuals the individual terms refer to, and what sets, relations, and functions are in the extensions of the predicate expressions. This specification \( I \) is called an interpretation function for the constant (non-variable) expressions in the syntax. (Variables are only dealt with by the assignment function \( g \).)

The valid formulas of predicate logic (tautologies) are true in all models, or under all interpretations, irrespective of how big the universe is and what the extensions of the predicates happen to be in it.
A grammar for the language of predicate logic
(propositional logic is subsumed but doesn’t play a role here)

Syntax

Lexicon:  Cat dp = {John}  
           Cat pred = {walks}  
Rule:    If α ∈ Cat dp and β ∈ Cat pred, then αβ ∈ Cat s. 

Semantics

A model M is <D,I>, where D is a set of individuals, and I is an interpretation function from elements of the lexicon to individuals and sets in the domain D. V is the function that specifies how complex expressions are evaluated.

Let D = {j, m, f} and let 

[[John]]M = I(John) = j

[[walks]]M = I(walks) = {m, j}

V^M(αβ) = 1 iff [[α]]M ∈ [[β]]M.

Let g1 = {<x, m>, <y, j>, <z, f>}. Let g2 = {<x, j>, <y, j>, <z, f>}. 

Enriched syntax

Lexicon:  Cat dp = {John, x, y, z, ...}  
           Cat pred = {walks}  
Rules:    If α ∈ Cat dp and β ∈ Cat pred, then αβ ∈ Cat s. 
           If φ ∈ Cat s, ∀x[φ], ∃x[φ] ∈ Cat s.

Enriched semantics

A model M is <D,I>, where D is a set of individuals, and I is an interpretation function from elements of the lexicon to elements of the domain D. g is an assignment of values to variables. g[x/d] is that assignment which differs minimally from g in that it assigns the individual d to the variable x. V is the function that specifies how complex expressions are evaluated.

Let D = {j, m, f}. 

[[John]]^M,g = I(John) = j

[[x]]^M,g = g(x) = m

[[walks]]^M,g = I(walks) = {m, j}

V^M,g(αβ) = 1 iff [[α]]^M,g ∈ [[β]]^M,g.

V^M,g(∀x[φ]) = 1 iff for every d ∈ D, V^M,g[x/d](φ) = 1.

V^M,g(∃x[φ]) = 1 iff for some d ∈ D, V^M,g[x/d](φ) = 1.

Comment on how the formulas compare with the English sentences Everyone walks and Someone walks.
HW 2, Predicate Logic (due in class on Monday)

- You are welcome to use either Ch 5 of Allwod et al. or Ch 3 of Gamut I as a reading. The Gamut chapter is better, but if you find their style difficult, use the other text. (In subsequent lectures we’ll follow the Gamut notation though. We’ll cover some of the content on Monday, as you can see in comparison with the last page of the handout.
- In the Gamut chapter, my experience is that people sometimes find the discussion of the substitution method 3.6.2 confusing. You are entirely free to skip to the assignment method 3.6.3 (as indicated in the PDF).
- In both chapters, the discussion of the properties of relations is an application, from the perspective of predicate logic. It is useful material and a good illustration of how predicate logic can be used to define symmetry, etc., but if these notions are new to you, focus your efforts on predicate logic itself and not on these notions.
- If predicate logic is very familiar to you but linguistics isn’t, use this weekend to catch up with the preparatory materials (slides, Compositionality, Scope). In my “Language” slides, in the Morphology and Syntax parts, focus on the fact that words, phrases, and sentences have a constituent structure, and linguists use particular arguments to establish that structure. Be familiar with both the structures produced by PS/merge rules and transformations (also the category labels). We are going to modify or argue against some of these, but they constitute the point of departure for the Jacobson book. (Also be up on the Semantics slides, which are largely but not fully based on the Compositionality chapter.)

- For the written assignment, I recapped all the problems from the handout. We did some of the problems in class; if this material is new to you, I’d recommend that you do them afresh, rather than copying them over from your class notes. Thinking them through for a second time will be useful. And be absolutely sure to carefully do those items that we didn’t do, including the last batch.
- Be sure to use the correct symbols in Word’s Symbols character set. We’ll continue to use these (and λ) throughout the semester; but feel free to use & and ~ if that makes life much easier. Assign shortcut keys to ∃ , ∀, λ and →.

- In addition to the pages below, do Exx 5 and 11 from Gamut. That’s a big pile, but it’s awfully useful. (If this is too much, let me know and we negotiate.) But, do not specify domains, and always use the first letters of the predicates for abbreviation.
- The book has fully worked-out solutions (uploaded in a separate file). Please check your solutions against the book’s; feel free to check the solutions after every single translation, if you aren’t quite sure how to proceed. If your solution differs from the book’s, determine whether they are equivalent or why the book does something else than you did. Indicate whether you found equivalence or understood what the book was doing, and alert me if neither is the case; then we should talk those points over. I’ll be happy to meet before or after class on Monday (please email on Sunday to make appointments, possibly together with a classmate.)
- The book is a bit inconsistent regarding whether it construes, say, two as at least or as exactly two. (I.e. it folds in implicatures or world knowledge sometimes.) Don’t be confused by that. I personally recommend the “at least” interpretation.
Your name:

John likes Homer, and
Bill likes everybody else [than Homer].

John likes himself.

John likes only himself.

At least three dogs bark.

Exactly one dog barks.

One dog barks and nothing else does.

The dog barks.

The dog barks and nothing else does.

(a) Express the meaning of No man flies, using

the quantifier \( \forall \) (and whatever else it takes):

the quantifier \( \exists \) (and whatever else it takes):

(b) Express the meaning of Not every man flies, using

the quantifier \( \forall \) (and whatever else it takes):

the quantifier \( \exists \) (and whatever else it takes):

(c) Express the meaning of Some man flies, using

the quantifier \( \forall \) (and whatever else it takes):

the quantifier \( \exists \) (and whatever else it takes):
Which of these equivalences are tautologies?

Always evaluate the free variables on both sides with respect to the same pointing (=assignment of values to free variables.) Below, assume everywhere that \(x \rightarrow \text{Arthur}\) and \(y \rightarrow \text{Ford}\).

(a) \(\forall x[F(x) \& G(y)] \equiv \forall z[F(z) \& G(y)]\)

(b) \(\forall x[F(x)] \& G(x) \equiv \forall y[F(y)] \& G(y)\)

(c) \(\forall x[F(x) \& G(x)] \equiv \forall y[F(y) \& G(y)]\)

(d) \(\forall x[F(x) \& G(x)] \equiv \forall y[F(y) \& G(x)]\)

Work through Exx 14, 15, and 16 in Allwood et al., using the book’s solutions as a guide. You don’t need to hand them in, but let me know if any of them is not clear. Use the diagram in the handout for reference.

Which of the following are valid (true in all worlds)? Construct models that falsify them, if you can. If some lesson can be drawn by comparing the members of these pairs: (a)-(b), (c)-(d), (e)-(f), and (g)-(h), say what it is.

(a) \((\forall x[D(x) \rightarrow B(x)] \& D(a)) \rightarrow B(a)\)

(b) \((\exists x[D(x) \& B(x)] \& D(a)) \rightarrow B(a)\)

(c) \(\forall x\forall y[H(x,y)] \rightarrow \forall y\forall x[H(x,y)]\)

(d) \(\exists x\exists y[L(x,y)] \rightarrow \exists y\exists x[L(x,y)]\)

(e) \(\forall x\exists y[L(x,y)] \rightarrow \exists y\forall x[L(x,y)]\)

(f) \(\exists y\forall x[L(x,y)] \rightarrow \forall x\exists y[L(x,y)]\)

(g) \(\forall x[A(x) \lor B(x)] \rightarrow (\forall x[A(x)] \lor \forall x[B(x)])\)

(h) \(\forall x[A(x) \& B(x)] \rightarrow (\forall x[A(x)] \& \forall x[B(x)])\)
Advanced Semantics
Szabolcsi

Lambdas (types, combinators)

We start with some of the basic reasons why semanticists use lambda abstraction and functions that are so conveniently defined using the lambda calculus. Then go on to define the notation, independently of linguistic applications.

(i) Extend the use of sentential connectives to subsentential cases,
(ii) Assign explicit interpretations to arbitrary coherent parts of sentences, as per compositionality,
(iii) Define operations on functions (combinators), e.g. the lifter and the function composer.

(i) run and sing
Bill and Joe
(ii) every man
(iii) lifter
(iv) compositor

* run' \land sing'
* Bill' \land Joe'
* \forall x[man'(x) \rightarrow ...(x)]

\lambda x[run'(x) \land sing'(x)]
\lambda P[P(Bill') \land P(Joe')]
\lambda P\forall x[man'(x) \rightarrow P(x)]
\lambda \forall x\lambda P[P(x)](P(Bill')) = \lambda P[P(Bill')]

Re (i), Step 1: To make the use of the sentential connective \land legitimate, pad out * run' \land sing' with variables: run'(x) \land sing'(x). That is to say, if run' and sing' had argument x, their conjunction would be run'(x) \land sing'(x).

Step 2: But they do not in fact have that argument (run and sing is not a sentence). The assumption of x must be withdrawn. This is indicated by the prefix \lambda x, where \lambda is the abstraction operator.

Re (ii), Step 1: Every man runs would be \forall x[man'(x) \rightarrow run'(x)]. Everything in this formula, save for run', is the contribution of every man. Hence, to represent every man, get rid of run' by replacing it with a predicate variable P.

Step 2: To withdraw the assumption of P, prefix \lambda P.

Definitions

Syntax: If \alpha is a well-formed expression and x a variable, \lambda x[\alpha] is a well-formed expression.
Semantics: \lambda x[\alpha] denotes a function. When this function is applied to some b of the same type as x, the function value is computed by replacing every substitutable occurrence of x in \alpha with b.
This replacement process is called beta reduction.
E.g, \lambda x[x^2] denotes that function which assigns each number its square: \lambda x[x^2](3) = 3^2.

Substitutibility

Case One: Only those x's in \alpha can be replaced that are bound by the relevant lambda operator.
In \lambda x[f(x) \land \forall x[h(x)]], only the x in f(x) can be replaced;
the x in h(x) is bound by the universal.
Notice: \lambda x[f(x) \land \forall x[h(x)]] = \lambda x[f(x) \land \forall y[h(y)].

To prevent mis-applications, it may be useful to reletter the “homonymous” variables that are bound by an operator other than the lambda. (You don’t have to, if you are careful enough, in contrast to Case Two.)
Case Two: If the argument to which the lambda-defined function applies is described by an expression that is, or contains, a free variable, care must be taken to ensure that this variable remains free in the course of computing the function value.

Suppose \( \lambda x[\forall y[f(x) \rightarrow h(y)]] \) is applied to the argument \( y \), a free variable.
Let the current assignment \( g \) of values to variables have \( g(y) = \text{bill} \).
Then, \( \lambda x[\forall y[f(x) \rightarrow h(y)]](y) \) must be the same as \( \lambda x[\forall y[f(x) \rightarrow h(y)]](\text{bill}) \),
i.e. \( \forall y[f(\text{bill}) \rightarrow h(y)] \).
If we had mechanically replaced \( x \) with \( y \), we would have gotten \( \forall y[f(y) \rightarrow h(y)] \),
which is an entirely different thing.
To prevent misapplications, we must reletter those bound variables in \( \alpha \) that happen to be "homonymous" with the free variable in the argument.
That is, \( \lambda x[\forall y[f(x) \rightarrow h(y)]] \) is not applied to \( y \). It is first relettered as \( \lambda x[\forall z[f(x) \rightarrow h(z)]] \).
Note: a free variable can never be relettered.

(Almost) Anything goes! In a good way

Since the lambda operator is an all-purpose device for defining functions, any functions, there are (almost) no restrictions on what its domain and co-domain might be. Above, we assumed for simplicity that \( x \) was an individual variable, but in fact it might be a variable over any domain. Likewise, the \( \alpha \) in \( \lambda x[\alpha] \) may be anything: (a) a truth value, (b) a function, (c) an individual that varies with \( x \), (d) a fixed object; etc.

(a) \( \lambda x[\text{run}'(x)] \) is the characteristic function of the set of runners.
\( \lambda x[\text{run}'(x)] = \text{run}' \) because for every argument \( b \), \( \lambda x[\text{run}'(x)](b) = \text{run}'(b) \)
A characteristic function is a function from some \( D \) to \( \{0,1\} \). It "characterizes" some subset \( C \) of \( D \) by assigning 1 to \( D \)'s that are also in \( C \), and 0 to those that are not.

(b) \( \lambda x[\lambda y[\text{employ'}(x)(y)]] \) is a function from potential employees \( x \) to VP-denotations, viz., functions from potential employers \( y \) to \( \{0,1\} \).

(c) \( \lambda x[\text{mother-of'}(x)] \) is a function from individuals to their mothers.

(d) \( \lambda x[\star] \) is a constant function that maps everything to \( \star \).

Vital conventions on representing argument order:

The notation \( f(a)(b) \) is short for \( (f(a))(b) \). The function \( f \) is first applied to \( a \) and then to \( b \). (This is called left-associativity.) The order of the lambda-prefixes represents the inviolable order of how the function can be applied to arguments:

\[
\lambda x \lambda y \left[ f(x)(y) \right] \text{(a)(b)} = \lambda x \left[ \lambda y \left[ f(x)(y) \right] \right] \text{(a)(b)} = \lambda y \left[ f(a)(y) \right] \text{(b)} = f(a)(b) \quad \text{and never } f(b)(a)!
\]

Naturally, explicit bracketing may indicate that things are otherwise:

\[
\lambda x \left[ \lambda y \left[ f(x)(y) \right] \right] \text{(b)} = \lambda y \left[ f(b)(y) \right] \text{(a)} = f(b)(a)
\]
Types are categories of things in the model. The domain of type e(ntity) is the set of individuals in the universe. The domain of type t(ruth value) is \{0,1\}. If A, B are types, the domain of type \langle A, B \rangle is the set of functions from the domain of A to the domain of B.

How to find the type of a lambda-expression:

\[ \lambda x_e \ [ \lambda P_{(e,t)} \ [ P_{(e,t)} \ (x_e)_{t} ] ] \]

is of type \langle e, \langle e,t \rangle, t \rangle

\( \langle e, \langle e,t \rangle, t \rangle \)

\( \langle \text{arg1}, \langle \text{arg2}, \text{value} \rangle \rangle \)

Different ways of writing the same thing:

\( \lambda x.[\lambda y[f(x)(y)]] = \lambda x.y[f(x)(y)] = \lambda x.y.f(x)(y) = \lambda x.y.f(y,x) \)

Heim–Kratzer and many linguists after them follow the "dot convention" to indicate the scope of the lambda operator. Moreover, they always explicitly indicate the domain of the function:

\( \lambda x : x \in D. f(x) \) or, for short, \( \lambda x \in D. f(x) \)

Can a function be applied to an arbitrary function, including itself?

Some functions applied to themselves yield a perfectly well-behaved and useful new function. But some other self-applications allow us to replicate the Russell paradox. The set theoretic version of the Russell paradox goes as follows:

Assume that for every property P, there is a set \{x : Px\}.

Now ask the reasonable question: What sets are elements of themselves?

Both \{x : x=x\} \in \{x : x=x\} and \( \emptyset \not\in \emptyset \) would make good sense.

But let R be \{x : x \not\in x\}, i.e. the set of those things that are not elements of themselves.

Is R an element of R? If yes, ..., if no, ... Paradox.

R is an element of itself iff it is not an element of itself.

The functional version replicates this (from Curry—Feys 1958):

Let N be negation. We define a function Y such that Y(Y) is N(Y(Y)).

Applying Y to itself is the same as the negation of applying Y to itself. Y is called the paradoxical combinator. How to define a Y that behaves in this paradoxical way?

Let Y be W(B(N)), with W(h)(z) = h(z)(z) and B(f)(g)(x) = f(g(x)).

This will do, because Y(Y) = W(B(N))(Y) = ((B(N))(Y))(Y) = N(Y(Y)).

Written with lambdas, let W = \( \lambda h.\lambda z[h(z)(z)] \) and B = \( \lambda f.\lambda g.\lambda x[f(g(x))] \). Then

\( Y(Y) = W(B(N))(Y) = \lambda h.\lambda z[h(z)(z)](\lambda f.\lambda g.\lambda x[f(g(x))])(Y) = \lambda h.\lambda z[h(z)(z)](\lambda g.\lambda x[N(g(x))])(Y) = \lambda z[\lambda g.\lambda x[N(g(x))](z)(z)](Y) = \lambda g.\lambda x[N(g(x))](Y)(Y) = N(Y(Y)) \)

The typed versions of the B and W combinators are well-attested even in the grammars of natural languages. It is purely the absence of typing that leads to the paradoxical result.
Pulling things together

Compositionality: the meaning of a complex expression is uniquely determined by (is a function of) the meanings of its parts (immediate constituents) and how they are put together.

The grammar must be able to assign an explicit interpretation to each of every dog, saw Jeremy, every, dog, saw, and Jeremy.

E.g. every dog \( \lambda P \forall x [\text{dog}(x) \rightarrow P(x)] \)

\( \lambda Q \forall x [Q(x) \rightarrow P(x)] \)

- The \( \lambda \) operator is a crucial tool for defining the meanings of subsentential expressions.

Hypothetical reasoning provides an intuition: If every dog were the subject of the predicate \( P \), \( \forall x [\text{dog}(x) \rightarrow P(x)] \) would be a formula. But every dog by itself is not a formula, so we must withdraw the assumption of \( P \) and form a conditional, so to speak, writing \( \lambda P \forall x [\text{dog}(x) \rightarrow P(x)] \).

Is that just a metaphor? No! Implication is to modus ponens as a function is to functional application.

If it’s Tuesday, it’s Belgium AND it’s Tuesday, therefore it’s Belgium.

\( \text{run}_{e,t} \) (jeffe) results in run(jeff) of type t

\( \lambda x_e [\text{run}_{e,t}(x_e)] \) (jeffe) results in run(jeff) of type t

- Curry-Howard (-Lambek-van Benthem) correspondence

<table>
<thead>
<tr>
<th>Logic</th>
<th>Computation</th>
<th>Logic</th>
<th>Computation</th>
</tr>
</thead>
<tbody>
<tr>
<td>implication</td>
<td>function type</td>
<td>( a \rightarrow b )</td>
<td>( \langle a, b \rangle )</td>
</tr>
<tr>
<td>conjunction</td>
<td>product type</td>
<td>( f \land a )</td>
<td>( f \cdot a )</td>
</tr>
<tr>
<td>implication elimination</td>
<td>functional application</td>
<td>( a \rightarrow b \land a )</td>
<td>( c_{\langle a, b \rangle}(d_a) )</td>
</tr>
<tr>
<td>(modus ponens)</td>
<td>functional abstraction</td>
<td>( \lambda d_a [c_{\langle a, b \rangle}(d_a)] )</td>
<td></td>
</tr>
</tbody>
</table>

- A type-free system allows the application of any function to itself and thus to give rise to a non-terminating program, the computational equivalent of the Russell paradox:

\( \lambda x [x(x)](\lambda x [(x(x))]) = \lambda x [x(x)](\lambda x [(x(x))]) = ... \)

When functions are typed, \( f \) cannot apply to \( g \) if their types are the same. Hence a function cannot apply to itself – the Russell paradox is excluded.

\( f : a \rightarrow b \quad (g : a \rightarrow b) \) -- just doesn’t work

- The \( \lambda \) operator is a crucial tool for defining operations on functions (combinators) that will serve to interpret words or rules for assembling words into phrases/sentences (composition, lifting, duplication, possibly connection).
Curry-Howard (-Lambek-van Benthem) correspondence

implication function type \[ a \rightarrow b \quad <a,b> \]
conjunction product type \[ a \land b \quad a \cdot b \]
implication elimination (modus ponens) functional application \[ a \rightarrow b \land a \quad c_{a,b} \rightarrow (d_a) \]
implication introduction (conditionalization) functional abstraction \[ \lambda d_a [ c_{a,b} \rightarrow (d_a) ] \]


And finally, Boolean operations in higher types can be derived from their base meanings in the truth tables. A case in point is the metamorphosis from sentence negation to predicate negation:

\[ \lambda x_{(e,t)} \cdot \lambda y_{e} \cdot \text{NOT}_{(t,t)}(x(y)). \]

There is a system to such changes, as will be seen now.

In fact, type changing is a general phenomenon in natural language that shows many systematic traits (see Chapter 7 of [7], and [10]). We shall outline a few points that will be necessary for our further investigation of logical constants.

Generally speaking, expressions occurring in one type \( a \) can move to another type \( b \), provided that the latter type is derivable from the former in a logical calculus of implication (and perhaps conjunction). The basic analogy operative here is one discovered in the fifties: Functional types \( (a,b) \) behave very much like implications \( a \rightarrow b \). Then, transitions as mentioned above correspond to derivations of valid consequences in implicational logic.

Example (Derivations are displayed in natural deduction trees)

- \( (t,t) = ((e,t),(e,t)) \):

  \[
  \begin{array}{ccc}
  & 1 & 2 \\
  & e & (e,t) \\
  & t & (t,t) \\
  t & \frac{}{\text{withdraw}} 1 \\
  (e,t) & \frac{}{\text{withdraw}} 2 \\
  ((e,t),(e,t)) & \\
  \end{array}
  \]

Moreover, these derivations are not purely syntactic. For they correspond one-to-one with terms from the typed lambda calculus, explaining how denotations in the original type are changed into denotations in the new type. Here is an illustration for Boolean negation:

Example

\[
\begin{array}{ccc}
  & 1 & 2 \\
  & e & (e,t) \\
  t & \frac{}{\text{MP}} \\
  & t & (t,t) \\
  & \frac{}{\text{MP}} \\
  t & \frac{}{C} \\
  (e,t) & \frac{}{C} \\
  ((e,t),(e,t)) & \\
  \end{array}
\]

\[
\begin{array}{ccc}
  & x_e & y_{(e,t)} \\
  & \frac{}{\text{MP}} \\
  y(x) & \frac{}{\text{NOT}_{(t,t)}} \\
  & \frac{}{\text{MP}} \\
  \text{NOT}(y(x)) & \frac{}{\text{MP}} \\
  & \frac{}{\text{MP}} \\
  \lambda x_e \cdot \text{NOT}(y(x)) & \\
  \lambda y_{(e,t)} \cdot \lambda x_e \cdot \text{NOT}(y(x)) & \\
  \end{array}
\]

Note how application encodes modus ponens, and lambda abstraction encodes conditionalization.
Combinators: operations on functions

The $\lambda$ notation allows one to define particular functions, e.g. $\lambda x \lambda y [\text{multiply} \times y]$, but also operations on functions. For example, one can define an operation that turns multiplication into squaring by forcing both arguments to be the same: $W(\text{multiply}) = \lambda x [\text{multiply} \times x] = \lambda x [\text{square} \ x]$. $W$ is a combinator. Combinators correspond to closed $\lambda$-terms (no constants, and all variables bound by $\lambda$s).

The combinators $S$ and $K$ are definitionally independent and together they have the expressive power of the $\lambda$-calculus with vacuous abstraction.

The combinators $W$, $B$, $C$ and $I$ are definitionally independent and together they have the expressive power of the $\lambda$-calculus without vacuous abstraction. (Same for $S$, $B$, $C$ and $I$, as well as other sets.)

Below are non-linguistic examples, but the operations will be shown to have much linguistic significance.

<table>
<thead>
<tr>
<th>$\lambda$-term</th>
<th>Combinator</th>
<th>Axiom</th>
<th>Type</th>
<th>what it does</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\lambda x_{a \to b} \lambda y_{[x(y)]}$</td>
<td>A</td>
<td>$Axy = xy$</td>
<td>$(a \to b) \to (a \to b)$</td>
<td>applicator</td>
</tr>
<tr>
<td>$\lambda x \lambda y \lambda z [x(z)(y(z))]$</td>
<td>$S$</td>
<td>$Sxyz = xz(yz)$</td>
<td>$(a \to (b \to c)) \to ((a \to b) \to (a \to c))$</td>
<td>“connector”</td>
</tr>
<tr>
<td>$\lambda x \lambda y [x]$</td>
<td>$K$</td>
<td>$Kxy = x$</td>
<td>$(a \to (b \to a))$</td>
<td>cancellator</td>
</tr>
<tr>
<td>$\lambda x \lambda y [x(y)(y)]$</td>
<td>$W$</td>
<td>$Wxy = xyy$</td>
<td>$(a \to (a \to b)) \to (a \to b)$</td>
<td>duplicator</td>
</tr>
<tr>
<td>$\lambda x \lambda y [x(y)(y)]$</td>
<td>$C$</td>
<td>$Cxyz = xzy$</td>
<td>$(a \to (b \to c)) \to (b \to (a \to c))$</td>
<td>permutator</td>
</tr>
<tr>
<td>$\lambda x \lambda y \lambda z [x(y)(z)]$</td>
<td>$B$</td>
<td>$Bxyz = x(yz)$</td>
<td>$(a \to b) \to ((c \to a) \to (c \to b))$</td>
<td>compositon</td>
</tr>
<tr>
<td>$\lambda x [x]$</td>
<td>$I$</td>
<td>$Ix = x$</td>
<td>$a \to a$</td>
<td>identificator</td>
</tr>
<tr>
<td>$\lambda y \lambda z [z(y)]$</td>
<td>$Cl$</td>
<td>$Clz = zy$</td>
<td>$b \to ((b \to c) \to c)$</td>
<td>function/argument reverser (lifter)</td>
</tr>
<tr>
<td>$\lambda x \lambda y [x(y)(y)]$</td>
<td>$S(Cl)$</td>
<td>$S(Cl)y = Cl(z)(y) = yzz$</td>
<td></td>
<td>$W$ defined using $S$, $C$, and $I$</td>
</tr>
</tbody>
</table>

$W$ multiply $x = \text{multiply} \times x$ if $m$ is multiplication, $Wm$ is squaring

$C$ follow $x \ y = \text{follow} \ y \ x$ if $f$ is follows, $Cf$ is precedes

$B$ paste $copy \ x = \text{paste}(copy \ x)$ if $p$ is paste and $c$ is copy, $Bpc$ is copy-and-paste

$I$ multiply $\ = \text{multiply}$ if $m$ is multiplication, $Im$ is the same multiplication

$Cl 7 = \text{looks for functions operating on 7}$

$Cl \ 7 \ Wm = \lambda h[h(7)][(\lambda x [\text{multiply} \times x]) = \lambda x [\text{multiply} \times x](7) = 49$
HW3 Lambda basics – due on Monday. We continue with lambdas next week.

Reading

Ch 4.4 (λ-Abstraction) in Gamut 2. At the moment, feel free to skip the 2nd half of p. 111 and anything that talks about types. (Notation preview: \( A^B \) is the set of functions from set B to set A; e is the type of individuals, t of truth values, \(<e,t>\) of functions from individuals to truth values.)

You may go on to read 4.2 (The Theory of Types) in preparation for next week, but this text is way too involved. If you find it rough going, just wait.

Exercises

(A) Is any of these statements ill-formed? Is any of them false? If yes, very briefly say why.
Note that whatever was true about free and bound variables in predicate logic remains true here. You can assume that types match properly (i.e. nothing will be ill-formed due to a type mismatch).

(a) \( \lambda a \forall x[a(x)](\lambda y[f(y)(z)]) = \forall x[f(x)(z)] \)

(b) \( \lambda a \forall x[a(x)](\lambda y[f(y)(x)]) = \forall x[f(x)(x)] \)

(c) \( \lambda a \lambda x[a(x)](\lambda y[f(y)(z)]) = \lambda x[f(x)(z)] \)

(d) \( \lambda a \lambda x[a(x)](\lambda y[f(y)(x)]) = \lambda x[f(x)(x)] \)

(e) \( \lambda x[f(x)] = f \)

(B) Ex 10 on p. 111 of Ch 4 of Gamut 2. You may ignore types if we don’t discuss them on 2/8; the types of functions and arguments match up. Write out your solutions and check them against the book’s solutions. Alert me if there is a divergence and if you aren’t quite sure why, but hand your solutions in anyway. – I recommend rewriting some of the book’s parentheses using square brackets to indicate the scope of quantifiers and lambda operators, as in the handout.
HW4 Lambdas continued -- due in class on W 2-22.

Reading

Use my handout as the main reading, supported by 4.4 and 4.2 of Gamut 2. I also added “Type-logical Semantics” by Muskens to the Background material folder, you may find it helpful and interesting.

Those of you who have an interest in logic and/or computer science should look at the Wikipedia article on Curry-Howard and perhaps “Propositions as types” by Wadler (links in the Background folder). Very important, and even a quick reading will give a sense of what correspondences to look out for.

Written

(C) To practice paying close attention to brackets and parens, work through the lambda conversions (beta reduction) below. You can assume that the types match. I very much recommend doing these using paper and pencil. Copying and pasting almost guarantee mishaps.

(a) $\lambda a \lambda b [\lambda c [d(a) \& d(c)](e)](f)(g)$

(b) Get N(Y(Y)) as the end result. The colors play no role in the conversions. (Apologies, Russell and Curry...)

$Y(Y) = W(B(N))(Y) = \lambda h \lambda z [h(z)(z)] (\lambda f \lambda g \lambda x [f(g(x))] (N)) (Y)$

Should you want more practice, the last section of my “Combinatory grammar and projection from the lexicon” (Background readings folder) has many $\lambda$-conversions written out step-by-step.
(D) The joys of reverse engineering: Give the interpretations of the underlined words/phrases using the \( \lambda \)-notation. Indicate the types of the variables in the quantifier and lambda prefixes, as in problem (F), using not-too-tiny subscripts and our non-abbreviated notation for types.

(a) If The train left is \( \exists x \forall y [(\text{train}(y) \leftrightarrow x=y) \& \left( \text{left}(x) \right)] \), what is the?

(b) If Bill smokes and snores is \( \text{smoke}(b) \& \text{snore}(b) \), what is smokes and snores?

(c) If smokes and snores is as you proposed, what is the pertinent (specific) version of and?

(E) Ex 8 in Ch 4. Write out your solutions as part of this assignment. Check the books’ solutions and alert me if you are puzzled.

(F) Determine the types of the following \( \lambda \).expressions; the method is given in the handout.

(e) \( \lambda \text{a}_{\text{c},\text{t}} \lambda \text{b}_{\text{c},\text{t}} \lambda \text{c}_{\text{e}} [\text{b}(\text{a}(\text{c}))] \)

(f) \( \lambda \text{d}_{\text{e}} \lambda \text{f}_{\text{c},\text{t},\text{t}} \lambda \text{g}_{\text{c},\text{e},\text{t}} [\text{f}(\text{g}(\text{d}))] \)

(G) Define a useful combinator.

(g) If Utah is above Arizona is \( \text{above(\text{arizona})(\text{utah})} \), what is is-above?

Make argument structure explicit using \( \lambda \).

(h) Express is-below in terms of is-above.

(i) Define the combinator that turns is-above into is-below, is-to-the-left-of into is-to-the-right-of, sees into is-seen-by, etc. Write out the function using \( \lambda \)'s and demonstrate its working.
MIDTERM -- not in this file
Semantic foundations (Jacobson Ch 2)

Meaning in model theoretic semantics: truth conditions for declarative sentences; more generally, including non-declaratives and subsentential expressions: denotation conditions.

Truth conditions \(\neq\) truth (1) or falsity (0): a function from possible worlds into \(\{0,1\}\), once contextual parameters are specified, e.g.,
for indexicals
\[ f(x) = I \text{ am here now}. \]
and definedness conditions (presuppositions) are satisfied
\[ f(x) = \text{Only Joseph Conrad was born Polish}. \]

Intension (sense) of a sentence \(S\)
\[ f: W \rightarrow \{0,1\} \text{ that assigns 1 to w if } S \text{ is true in w and 0 otherwise.} \]

Extension (factual value) of a sentence \(S\) in \(w\) is \(f(w)\)

More precisely, sense
\[ f: W \times T \rightarrow \{0,1\} \text{ (a function from world-time pairs to truth values)} \]

Character (Kaplan)
\[ f: C \rightarrow (W \times T \rightarrow \{0,1\}) \text{ (a function from contexts to world-time pairs to truth values)} \]

A function from \(W \times T\) to \(\{0,1\}\) is of type \(<s,t>\).

This course will be mostly concerned with extensional phenomena, but it is important to see why an intensional notion of meaning is needed, and to be aware of a range of phenomena that require figuring in possible worlds / times.

(1) In some world \(w\) at time \(t\), all and only carpenters are violinists. At that \(w,t\),

a. Bill sees every carpenter =?= Bill sees every violinist
b. Bill sees seven carpenters =?= Bill sees seven violinists
c. Bill sees no carpenters =?= Bill sees no violinists
d. Bill is a skillful carpenter =?= Bill is a skillful violinist
e. Bill wants to be a carpenter =?= Bill wants to be a violinist

(2) Does the meaning of There’s a coin in Bill’s pocket change if Bill empties his pocket?

(3) If the US Supreme Court hadn’t stopped the Florida recount in 2000, nuclear power plants would no longer be in use in the US.

(4) If Smith were the murderer, we would see exactly the footprints next to the body that we see now.

(5) Myrtle the Turtle was crossing the street when she got hit by a truck and died.

(6) If I weren’t here now, I’d be outside enjoying the weather.

(7) After Cresswell, Entities and Indices (1990)

Once everyone now happy was going to be miserable. (reference to + quantification over times)
It might have been that everyone actually rich was poor. (reference to + quantification over worlds)
Some notes re: midterm

The exams were generally very good -- correct and thoughtful. Some points to remember as we delve into natural language semantics, over and beyond what you got right. We’ll take up some of these later in this semester (not all of them).

(B) The connective but is not purely truth-functional.
    And doesn’t only connect sentences. We already know that the use of & can be generalized using λs: \( \lambda x [\text{smokes}(x) \land \text{snorers}(x)] \), etc.
    And can form collectives: John and Bill fight with each other. This bread is made of flour and water.
    Is and ambiguous between & and \( \land \)? A topic for much research.
    And is not commutative: John entered and switched on the light vs John switched on the light and entered. But also: John entered and he switched on the light vs. He entered and John switched on the light. A major topic for discourse representation theory / dynamic semantics.

(G) Two functions are the same if they deliver the same values for all arguments, hence \( \lambda x [fx] = \lambda x [gx] \) is equivalent to \( \forall x [fx = gx] \). Very useful to bear in mind.

(H) \( \exists x [Nx \rightarrow \ldots] \) is a very weak statement. It is true as soon as something is not an N, because an implication is true when the antecedent is false.
    \( \exists x [Nx] \rightarrow \forall y [A(x)(y)] \) wouldn’t work, because the underlined \( x \) is not bound by \( \exists \). What makes this formula interesting is that its structure tantalizingly resembles that of If someone is noisy, everybody is annoyed with him. Discourse representation theory / dynamic semantics attempts to bridge the gap.

(I) Most striking is the contrast between Most of the dogs bark vs. \( Mx [dx \rightarrow bx] \). The latter is true as soon as the majority of the things in the universe are not dogs, irrespective of what dogs do or who barks. First order logic quantifies over all elements of the universe, but most (of the) only over its restrictor, here, dogs. This will be a big argument in favor of generalized quantifier theory.

(J) We return to the problem of constituency in sentences with three-argument verbs like show soon, cf. (a)-(b). This was just to whet your appetite.
    You tended to retain ingredients of the sentence that do not come from the segment you sought to interpret, see below:
    Licks itself is just \( \lambda x [\text{lick}(x)](x) \). Compare: run is just \( \lambda x [\text{run}(x)] \).
    Itself is just \( \lambda R \lambda x [R(x)(x)] \).
    The quantifier and the connective must come from every cat.

    Which Pat saw may be \( \lambda x [\text{saw}(x)(pat)] \) or, perhaps, \( \lambda Q \lambda x [Q(x) \land \text{saw}(x)(pat)] \), where we stipulate that Q must be a noun, not just an arbitrary <e,t> type property, e.g. walked. Depends on how
much burden you place on the interpretation of the rule that combines the noun with the relative clause. But the quantifier of the noun definitely comes from elsewhere.

(K) I didn’t find it easy to explain the task – apologies! But here’s the general idea (clip from my Quantification book).

\[
\begin{align*}
(i) & \quad T = [\lambda x.x \equiv \lambda x.x] \\
(ii) & \quad F = [\lambda x.x \equiv \lambda x.T] \\
(iii) & \quad \sim = \lambda x[F \equiv x] \\
(iv) & \quad \forall y_{\alpha}.A = [\lambda y_{\alpha}.A \equiv \lambda y_{\alpha}.T]
\end{align*}
\]

Informally, (i) defines Truth, because everything is identical to itself; (ii) defines Falsehood, because it says that every proposition is True; (iii) defines negation as a function that equates its argument with Falsehood. The definition of universal quantification in (iv) is based on the fact that two functions are identical iff they assign the same value to every argument. (iv) says that the function that maps any \( y \) to \( A \) is the same as the one that maps any \( y \) to Truth. Notice that the definitions only use closed typed \( \lambda \)-terms (viz. combinators) and equality.
Compositionality, Direct Compositionality, syntax/semantics interface (Jacobson Ch 3)

Compositionality (Gottlob Frege (1848-1925), Richard Montague (1930-1971))
The meaning of a complex expression is a function of the meanings of its constituent parts and how they are assembled.

Direct Compositionality (Pauline Jacobson)
Each expression is assigned its complete and final interpretation at the point when it is assembled. Syntax and semantics work in tandem. Each syntactic rule is paired with a semantic rule.

Compositionality requires that the meaning of the complex expression be fully traced back to the meanings of its parts in an explicit and mechanical way. Direct compositionality imposes a stricter requirement: whenever syntactic rules assemble an expression, its interpretation is produced and cannot be tampered with later.

A first stab: temporary rules (TR) for very simple structures
Notation: an expression is a triple, < [sound], syntactic category, [[meaning]] >.

TR-1 If α is of the form < [α], S, [[α]] > and β is of the form < [β], S, [[β]] >, then there is a γ of the form < [α-and-β], S, 1 iff [[α]] = 1 and [[β]] = 1 >

TR-2 If α is of the form < [α], S, [[α]] > and β is of the form < [β], S, [[β]] >, then there is a γ of the form < [α-or-β], S, 0 iff [[α]] = 0 and [[β]] = 0 >

TR-3 If α is of the form < [α], S, [[α]] >, then there is a β of the form < [it-is-not-the-case that-α], S, 1 iff [[α]] = 0 >

Some of the challenges that the Direct Compositionality program faces

- Syncategorematic treatment of connectives and quantifiers in propositional and predicate logics -- would a syncategorematic treatment of and, or, not, every, a(n), ... suffice for English?

- Is not a sentential operator in English? Olivia may not leave the room.

- How to assemble, and compute the meanings of, expressions with conflicting evidence for structure?

Mary [vP showed Fido to Snoopy]
?? Mary showed to Snoopy Fido
Mary showed herself to Snoopy—* Herself showed Mary to Snoopy
Mary showed Fido to himself — ?? Mary showed himself to Snoopy
Mary showed Fido to Snoopy and Stan to Pan

- How to compute two scopal readings for sentences with two quantifiers?

Every dog barked at some cat
Exactly two dogs barked at every cat

- How to compute the interpretations of sentences with displaced (moved) constituents?

Pat saw them — ?? Them Pat saw
* (the dog) Pat saw which — (the dog) which Pat saw
* (We know) Pat saw which dog — (We know) which dog Pat saw
Syntactic categories and semantic types (Jacobson Chs %4, %6)

(Slightly outmoded) generative syntax

A phrasal category (NP, PP, VP, S, etc.) is a set of expressions that have the same syntactic distribution and denote things of the same semantic type. E.g.,

NP: Kim, Fido, the old witch are systematically interchangeable in sentences, and denote entities of type e.
VP: arrived, spotted Fido, confided in Fido are systematically interchangeable, and denote sets of entities (or characteristic functions of such sets) of type <e,t>.

A lexical category / part of speech (N, V, P, A, D) is a set of expressions that have the same morphology but not the same syntactic distribution, and often do not denote things of the same semantic type. E.g.,

V: arrive, spot, confide all take suffixes -ed, -s, -ing, but are not interchangeable and have no common type.


arrive: intransitive verb, type <e,t>
spot: transitive verb, type <e, <e, t>>
confide: PP-demanding verb, type <α, <e, t>> (α stands in for whatever type PP-denotations have)

TR-5 If α is an expression of the form <[α], NP, [[α]]> and β is an expression of the form <[β], VP, [[β]]>
then there is a γ of the form

\{ <[α-β], S, 1 iff [[α]] ∈ [[β]], and 0 otherwise > \}
\{ <[α-β], S, [[β]]([[[α]]]) > \} (with [[β]] a set)
\{ <[α-β], S, [[β]](α) > \} (with [[β]] a function)

The correlation between categories, types, and rules seems a bit accidental. Can we do better?

Categorial grammar -- two main ideas

1) Syntactic category labels encode syntactic distribution (“predict what rules can apply”).
2) Syntactic categories are straightforwardly mapped to semantic types:

   f: Cat → Type  f(e)=e,  f(t)=t,  f(A/B) = f(B\A) = <B,A>

A preliminary version with directional Lambek-style categories and just extensional types:

(i) e, t are categories,
(ii) If A, B are categories, A/B and B\A are categories.
(iii) There are no other categories.

Rules:

- If δ is in cat. A/B, and κ is in cat. B, then δκ is in cat. A.
- If κ is in cat. B, and δ is in cat. B\A, then κδ is in cat. A.
- [[δ_{<B,A>}]]([[κ_B]]) is of type A.
- [[δ_{<B,A>}]]([[κ_B]]) is of type A.
More on the semantics of NPs and on intensional types (Jacobson Ch %4)

Jacobson’s book uses NP exclusively as the category of expressions that (can) denote entities. Other noun phrases, e.g. every dog, will have category QP. She doesn’t use the category DP for either.

Expressions of category NP and type e

- Proper names, singular definite descriptions:  
  Kim, the witch
  Denote atomic individuals.
- Definite plurals, conjunctions of names and definites: the witches, Fido and Snoopy, Kim and the witch(es)
  Denote non-atomic individuals (= plural individuals, individual sums) -- Link 1983

\[
\text{fido®snoopy} \quad \text{The dogs refers to the maximal plural individual consisting of just dogs. In this case, that is fido®snoopy.}
\]

- Bare plurals, mass terms:  
  dogs, water
  Denote kinds, another sort of individuals -- Carlson 1977
  The domain of entities is sorted into vanilla individuals, kinds, stages, events, etc.
  Beware: In Dogs are smart and in Pat saw dogs, dogs is NP. But, in the/some dogs, dogs is just N.

  Dogs are smart says that dog-kind has the property smart (individual-level predicate).
  Dogs are barking says that dog-kind has realizations that dog-kind are barking (stage-level predicate).

The intension of an expression of category NP is of type \(<s,e>\), a function from world-time pairs to entities.

- A proper name is a rigid designator (Kripke 1972): a constant function of world-time pairs. The individual it picks out does not vary: if Barack Obama had been born in Kenya, he would not have been POTUS.
- So, the domain of individuals (universe of discourse) is not world-dependent. Santa Claus is in the domain of all worlds. Santa Clause doesn’t exist (in the actual world) is comparable to Santa Claus doesn’t sing.
- Definite descriptions are not rigid designators: their extensions vary with worlds and times.
  Some VPs look at the intension of the subject: The pope has always been a Catholic.
  The temperature is rising.

Intensional types? In Montague, s is not a type, but:

(i) e, t are types.
(ii) If A, B are types, \(\langle B,A \rangle \) is a type.
(iii) If A is a type, \(\langle S,A \rangle \) is a type.
(iv) Nothing else.

Thus, no type \(\langle s,s \rangle\), etc.

In Gallin 1975 and on (two-sorted type theory),

(i) e, t, s are types.
(ii) If A, B are types, \(\langle B,A \rangle \) is a type.
(iii) Nothing else.

Thus, \(\langle s,s \rangle\), etc. are in.

Folding in worlds -- two equivalent notations (similarly for TR-2 and TR-3);

\[
\text{INT-TR-1. If } \alpha \text{ is of the form } \langle [\alpha], S, [[\alpha]] \rangle \text{ and } \beta \text{ is of the form } \langle [\beta], S, [[\beta]] \rangle, \text{ then there is}
\]

\[
\begin{align*}
\text{a } \gamma \text{ of the form } & [\alpha \text{-and-} \beta], \text{ S, for any world } w, [\gamma][w] = 1 \text{ iff } [[\alpha]][w] = 1 \text{ and } [[\beta]][w] = 1 \text{ > } \\
\text{a } \gamma \text{ of the form } & [\alpha \text{-and-} \beta], \text{ S, for any world } w, [\gamma]^w = 1 \text{ iff } [[\alpha]]^w = 1 \text{ and } [[\beta]]^w = 1 \text{ > }
\end{align*}
\]
Resolving an apparent syntax/semantics mismatch: Currying and Wrap (Jacobson Ch 5)

- In predicate logic, transitive verbs denote sets of ordered pairs of individuals.
  
  feed may be \{<kim, fido>, <pat, snoopy>, <joe, snoopy>\} \subseteq e \times e  
  
  (Cartesian product)

Suppose we insist that that is the correct semantics, and add product types to our grammar. Transitive verbs are now of type \(e \times e, t\). Then Lee fed the pig has a flat structure in which there is no VP constituent.

Problem: VP-conjunctions involving intransitive verbs: Lee meditated and fed the pig

Lots of flat rules? Sentential conjunction plus deletion? Abandon direct compositionality?
Interpret trees as wholes?
The recursive and-rule then forces infinitely many tree-interpreting rules.

Obtain the problematic sentences by (classical) Conjunction Reduction?

Some pig chased the wolf and grunted =?= Some pig chased the wolf and some pig grunted
Every pig chased the wolf or grunted =?= Every pig chased the wolf or every pig grunted

[but see Schein, forthcoming, “And: Conjunction Reduction Redux;” with heavy event semantics]

- The standard solution is to move from set-world to function-world: Currying (a.k.a. Schoenfinkelization)

The characteristic function of a set of pairs \(A \times B \subseteq C \times D\), viz.

\(f: \{C \times D\} \rightarrow \{0,1\}\)

is the same as

\(f1: C \rightarrow (D \rightarrow \{0,1\})\) or \(f2: D \rightarrow (C \rightarrow \{0,1\})\),

of type \(e, \langle e, t\rangle\)

Recall Curry-Howard and compare

\((p \& q) \rightarrow r\)
\(p \rightarrow (q \rightarrow r)\)
\(q \rightarrow (p \rightarrow r)\)

The difference is just syntactic. Based on syntax, we choose \(V_{trans}: \text{object} \rightarrow (\text{subject} \rightarrow \text{sentence})\).
But syntactic choices in natural language have consequences for both form and interpretation, because some interpretive rules are sensitive to hierarchy and constituency.

- Similarly, 3-place verbs \(A \times B \times C \rightarrow \{0,1\}\) can be Curried into some \(C \rightarrow (B \rightarrow (A \rightarrow \{0,1\}))\).
  What is the desirable version? Recall the reflexivization facts, among others.

The literature has considered four constituent structures for VPs like give the toy to Mitka:

(1) \([VP \ V \ NP \ PP]\)
(2) \([VP \ V \ [NP \ PP]]\)
(3) \([VP \ [V \ NP] \ PP]\)
(4) \(V \) and \(PP\) form a discontinuous constituent -- how?

(4a) \(\text{PP is extrapoosed to the right}\)

(4b) \(V \) moves to the left

(4c) \(\text{NP is directly infixed between } V \text{ and } PP\) by \text{WRAP}\n
Bach 1979, Pollard 1984, Jacobson 1987
HW5  Rises and changes -- a research problem -- you are encouraged to collaborate (due on Monday, 3-20)

Be sure to read all of Chs 2-3-4-5 of Jacobson, so we can continue with Ch 6, and ...

Read Lasersohn 2005, The temperature paradox as evidence for a presuppositional analysis of definite descriptions (in the Course Readings folder). The article is altogether 8 pages, but you should focus on the segments copied below, and then go on to consider the following examples involving the verb change.

(a) The temperature has changed. The temperature is 90F. 90F has changed.
(b) These days you wouldn’t get such a contract. The mayor has changed.
(c) I visited the Smiths. Everything is as before, but the dog has changed.
(d) A bald man is sitting at the window, eating a candy bar. The guy has changed.
(e) A bald man is sitting at the window, eating a candy bar. The guy is changing.

In each of (a)-(e), determine whether the last sentence makes sense/can be true in the context of the preceding text; what the last sentence means, including whether it is ambiguous. Write out these briefly but clearly. What descriptive generalizations emerge from (a)-(e)?

Now explore whether and how Lasersohn’s proposal extends to (a)-(e). Are any innovations needed? Add further examples if your argument so requires. Be sure to explain your proposal informally. You may go on to formalize it, but only after/alongside a clear informal explication.

Montague’s (1973) analysis of the temperature paradox is well known. Sentences (1), (2), and (3) are assigned logical translations equivalent to (4), (5), and (6), respectively.

(1) The temperature rises.
(2) The temperature is ninety.
(3) Ninety rises.

\[
\begin{align*}
(4) & \exists x \forall y [\text{temperature}(y) \rightarrow x = y] \land \text{rise}(x) \\
(5) & \exists x \forall y [\text{temperature}(y) \rightarrow x = y] \land \forall x = n \\
(6) & \text{rise}('n')
\end{align*}
\]

Here, \( x \) and \( y \) are variables of type \( (s, e) \), ranging over “individual concepts” (functions from indices to individuals); \text{temperature} and \text{rise} are predicates of type \( (s, e, t) \), taking individual concepts as arguments; \( n \) is a constant of type \( e \), denoting an individual (presumably the number 90); \( \forall x \) is an expression of type \( e \), denoting the individual yielded by \( x \) for the index of evaluation; and \( \forall x = n \) is an expression of type \( (s, e) \), denoting the constant function that yields at every index the individual denoted by \( n \).\footnote{In this way, sentence (1) is analyzed as asserting that the unique temperature function rises; (2) is analyzed as asserting that the value of this function for the index of evaluation is 90; and (3) is analyzed as asserting that the function that picks out 90 at all indices rises. Because the temperature function might yield 90 at the index of evaluation, without being identical to the function that yields 90 at all indices rises.} In this way, sentence (1) is analyzed as asserting that the unique temperature function rises; (2) is analyzed as asserting that the value of this function for the index of evaluation is 90; and (3) is analyzed as asserting that the function that picks out 90 at all indices rises. Because the temperature function might yield 90 at the index of evaluation, without being identical to the function that yields 90 at all indices, sentences (1) and (2) may be true even while (3) is false, resolving the paradox.

This analysis has been criticized on a number of grounds. In some cases, however, it is possible to reconstruct the temperature paradox using examples to which these criticisms do not apply.

In this terminology, an index is a name or variable for times or worlds. The symbols may be \( i \), \( j \), or \( w \).

Often, the actual time / world is notated as \( i^* \) or \( w^* \) or as \( @ \).
Now consider the argument with (15) and (16) as premises and (17) as its conclusion.

(15) Necessarily, the temperature is the price.
(16) The temperature rises.
(17) The price rises.

In fact, I think there is a preferable solution to the problem that becomes obvious once we consider what the motivation is for analyzing nouns like temperature and price as predicates of individual concepts. In considering this issue, we should be careful to distinguish the motivation for treating rise in this way from the motivation for treating temperature and price in this way, because the reasons are quite different in the two cases.

Rise is treated as taking individual concepts as its arguments because it is impossible to determine whether a function \( f \) is rising at index \( i \) simply by examining the value that \( f \) yields for \( i \). Instead, we must also know what values \( f \) yields for neighboring indices, to see if the earlier ones yield lower values and the later ones yield higher values. Put somewhat differently, we cannot determine whether the temperature is rising at a given moment by examining a snapshot of a thermometer taken at that moment—we need more than one snapshot, taken at different times. This sort of consideration makes it seem quite reasonable to regard rise as creating an authentic temporally intensional context, and to analyze it as taking individual concepts, rather than individuals, as its arguments.

For temperature and price, the situation is quite different. To know that a particular value is the temperature at a given moment, a single snapshot suffices, and snapshots taken at other times are essentially irrelevant. Nor do we need to know anything about the price history of a product to know its current price. Temperature and price do not intuitively require consideration of multiple indices to determine their extensions, and taken purely on their own ground, they do not provide any reason for an analysis in which they take individual concepts rather than individuals as their arguments.

Why, then, did Montague give an analysis in which their arguments were individual concepts? Only because of this: in formula (4), repeated here as (23), the variable \( y \), with which the predicate temperature' combines, must be of the same type as the variable \( x \), with which rise' combines; otherwise, the clause "\( x = y \)" will not be well formed, or make any sense. But the variable with which rise' combines must be of type \( \langle s, e \rangle \), and so the variable with which temperature' combines will have to be of type \( \langle s, e \rangle \) as well.

(23) \( \exists x [\forall y [\text{temperature}'(y) \leftrightarrow x = y] \land \text{rise}'(x)] \)
Why do we need these variables? Only because Montague assumes a Russellian analysis of the definite determiner, in which it expresses unique existential quantification. If we were to adopt a presuppositional analysis of definites, the need for the variables disappears, and with it the requirement that the variables must match in type. This removes the motivation for treating temperature and price as being of type \(\langle s, e, t \rangle\) and allows an analysis in which the argument in (15) through (17) comes out valid without the extra stipulation of the meaning postulate in (21).

To see this, add to the syntax and semantics of Montague’s Intensional Logic the following rules:

(24) If \(\phi \in \text{ME}\) and \(u\) is a variable of type \(a\), then \(\iota \phi \in \text{ME}_a\).

(25) \([\iota \phi]^{\iota, \iota} \) is the unique object \(d \in \text{D}_{u, A, L, S}\) such that \([\phi]^{\iota, \iota} \) \(d = 1\) (where \(g'\) is the \(\forall\)-assignment like \(g\) except for the possible difference that \(g'(u) = d\)) if such an object exists; undefined otherwise.

These rules sometimes produce expressions with an undefined semantic value, and this effect will propagate up the syntactic tree, yielding a truth value gap. We take such cases to be examples of presupposition failure: an expression of the form \(\iota \phi\) carries a presupposition that there is a unique object satisfying \(\phi\). Readers who wish to limit upward propagation of the truth value gap are invited to adapt the present rules to their favorite system of three-valued logic or supervaluations.

We now stipulate that common nouns receive translations of type \(\langle e, t \rangle\) rather than \(\langle s, e, t \rangle\).

Next, revise the rule for translating definite noun phrases into Intensional Logic (given as part of \(T_3\) in Montague 1973) as follows,

7 I revert here to Montague’s original system, in which interpretation is relativized to pairs of indices. I depart slightly from Montague’s notation in using the now-standard double brackets to indicate semantic values.

where \(u\) should be understood as a variable of type \(e\), and \(P\) as a variable of type \(s, \langle s, e, t \rangle\):

(26) If \(z \in P\) and \(z\) translates as \(z'\), then the \(z\) translates as \(\lambda P[\forall P(\iota \phi z'(u))].\)

The presuppositional analysis comes from Strawson 1950 (On referring). That’s in fact the standard analysis these days; important to know about. The formalization involves the (upside down) iota-operator, as in (25).

\[\text{ME} = \text{the set of meaningful expressions. } \iota \phi \text{ has the same type as the variable } u \text{ (here it will be } e). \text{ E.g. } [[\iota \phi . \text{ dog}(u)]] \text{ is the unique dog if such a unique dog exists, and is undefined otherwise.}\]

\[\text{P}_{CN} \text{ is the category of common noun phrases (e.g. (small) dog).}\]

\[\forall P \text{ is of type } \langle s, e, t \rangle \text{ and } \forall (\iota \phi u . \text{ Z}(u)) \text{ is type } \langle s, e \rangle: \text{ it is the function that picks out the } Z \text{ at every index. See (30) below.}\]

\[\lambda P[\forall P(\iota \phi z'(u))]] \text{ is the set of properties } P \text{ that the } Z \text{ has.}\]

****

The fact that the letter P occurs in two roles is a pure accident!
With these rules in place, sentences (15) through (17) receive translations equivalent to (27) through (29), respectively.

\[(27) \Box \text{utemperature}'(u) = \text{uprice}'(u)\]
\[(28) \text{rise}'(\land \text{utemperature}'(u))\]
\[(29) \text{rise}'(\land \text{uprice}'(u))\]

Note that \text{rise}' continues to take a type \(\langle s, e \rangle\) argument, just as before. But instead of a variable ranging over the full set of type \(\langle s, e \rangle\) functions, the argument in this case is formed by prefixing the intensional operator \(\land\) to the type \(e\) expression \(\text{utemperature}'(u)\). This operator is defined in the standard way.

\[(30) \langle \land \alpha \rangle^\text{a,i,j,k} \text{ is that function } f \text{ with domain } I \times J \text{ such that for all } i' \in I, j' \in J: f(i', j') = \langle \alpha \rangle^\text{a,i,j,k} \]

Note that this definition guarantees that \(\land \alpha\) will always denote the same function, regardless of choice of indices of evaluation; \(\langle \land \alpha \rangle^\text{a,i,j,k} = \langle \land \alpha \rangle^\text{a,i',j,k}\), for all \(i, i', j, j'\).

In particular, \(\land \text{utemperature}'(u)\) will always denote the same function, which we may regard as the function that picks out the temperature at each index. Likewise, \(\land \text{uprice}'(u)\) will always denote the same function, picking out the price at each index. Since neither of these expressions varies from index to index in which function it picks out, we cannot obtain a situation like that presented in (13) and (14), with a different temperature function at each index and a different price function at each index; instead, we have a single temperature function for all indices and a single price function for all indices.

Suppose (27) is true. Then \(\text{utemperature}'(u) = \text{uprice}'(u)\) is true at all indices; that is, the denotation of \(\text{utemperature}'(u)\) at any given index is identical to the denotation of \(\text{uprice}'(u)\) at that index. Therefore, the function that picks out the denotation of \(\text{utemperature}'(u)\) at any given index must be identical to the function that picks out the denotation of \(\text{uprice}'(u)\) at any given index. In other words, \(\land \text{utemperature}'(u)\) must denote the same function as \(\land \text{uprice}'(u)\).

But then, if (28) is true, (29) must be true as well. That is, the argument from (27) and (28) to (29) is valid. Since these formulas are the translations of (15), (16), and (17), the argument from (15) and (16) to (17) is valid as well.

We thus obtain intuitively correct results for these examples, without the arbitrary lexical stipulation of the meaning postulate in (21). These results are made possible because we are treating common nouns, including \textit{price} and \textit{temperature}, at their intuitively correct type of \(\langle e, t \rangle\), instead of the higher type \(\langle (s, e), t \rangle\) proposed by Montague. The use of this higher type was necessitated by Montague's assumption...
of a Russelian, quantificational analysis of definite noun phrases (combined with the treatment of intensional verbs like rise as being of type \((s, e, t)\)); once we drop the Russelian analysis in favor of a presuppositional analysis, the simpler type assignment becomes possible, and the validity of the argument in (15) through (17) falls out, eliminating a significant problem in Montague’s treatment of the temperature paradox. Because the presuppositional analysis makes possible an improved analysis of the temperature paradox, we may regard the temperature paradox as providing evidence in favor of the presuppositional account of definites.

Optional comments on the text:

Montague’s operators \(\downarrow\) (down, wedge), \(\uparrow\) (up, caret) and \(\Box\) (box) do not bind variables. Their action is encoded in a semantic rule.

If expression \(a\) has an intensional type, \(<s,A>\) and thus denotes a function from indices to \(A\)-type things, then \(\uparrow a\) is the value of that function at the index of evaluation. It is easy to rewrite \(\uparrow a\) with lambda-abstraction over indices as \(\lambda i[a(i)][i^*]\), using the same \(a\) of type \(<s,A>\). \((i^*)\) is the actual world or time.

If \(b\) is an expression of any type \(B\), \(\uparrow b\) is a function from indices to \(B\)-type things that picks out the reference of expression \(b\) at each such index. The content of \(\uparrow b\) can be written as \(\lambda i[\beta(i)]\). However, here \(\beta\) is of type \(<s,B>\) and it is distinct from \(b\), which is of type \(B\).

Importantly, \(\uparrow^\uparrow a = a\) is valid. But \(\uparrow^\uparrow a =/\alpha\), i.e., they are not guaranteed to be the same (even if \(\uparrow^\uparrow a\) is type-wise well-formed).

In current literature neither \(\uparrow\) nor \(\uparrow\) is used any more (except in order to pay homage to ‘Seventies work).

The box \(\Box\) is the necessity operator of modal logic. \(\Box\)hesperus=phosphorus says that necessarily, the two names refer to the same heavenly body (Venus). This can be written using explicit universal quantification over indices, but in the spirit of the above, using the \(<s,e>\) type symbols \(H\) and \(P\): \(\forall i[H(i) = P(i)]\), which is equivalent to \(H=P\). Likewise, for Necessarily, Socrates is mortal, \(\Box\)mortal(socrates) one can write, \(\forall i[M(i)(S(i))]\) -- at every index, the individual called Socrates at that index is in the set of those who are mortals at that index \(i\). This is what linguists do these days. Modal logicians usually use the box.
Some comments on the vicissitudes of the temperature paradox
My own judgments of the examples (you often had the same judgments):

(a) The temperature has changed

“Our the value of the temperature-function at t =/= the value of the temperature-function at t’ “
[not the same as “The temperature-function has changed”, cf. Gupta -- this is clearer with change than
with rise (what does it mean for a function to rise, really?)]

The temperature is 90F “The value of the temperature-function at t is 90F”

90F has changed: 90F is either an individual (type e) or a constant function (of type <s,e>), so it either has
no values, or those values do not vary, and cannot change in that sense. (For some reason, 90F has
changed also cannot mean something like, “90F used to be unpleasant but now it is pleasant”.)

(b) The mayor has changed

(i) “Some property of mayor Bill at t =/= that property of mayor Bill at t’ “

or “Mayor Bill has changed clothes” (and suppose that reflects a change in policies)

(ii) “The value of the mayor-function at t (Bill) =/= the value of the mayor-function at t’ (Tim) “

Both (a) and (b) assume that the temperature and the mayor can denote functions from times to individuals
(type <s,e>). Not surprising; temperature and mayor are called functional nouns. We can keep the time fixed (as
a moment or as a longish period) and we get a fixed individual as a value. Or we can look at values at different
times, and we get different individuals. In the latter case, we still have two options:

The mayor is the fire chief (always / now)

(i) “The mayor-function is the fire chief function (their values co-vary: pairwise identical)”

Makes the analog of Gupta’s syllogism valid: The mayor is the fire chief; the mayor has changed (from
Bill to Tim); the fire chief has changed.

(ii) “At t, the value of the mayor-function happens to be the same as that of the fire chief function (Bill)”

Cf. The mayor is Bill; The temperature is ninety. Makes the analog of Gupta’s syllogism invalid; if Bill is
no longer mayor, he may remain fire chief.

(c) The dog has changed.

Ambiguous like (b), but this is surprising: dog is not a functional noun! But we could turn it into one by
adding a silent relative clause: dog that the family keeps (at time t). This seems natural. The relative
clause may be interpreted in a temporally stable manner, or in a temporally variable, episodic manner.

(i) “Some property of the dog that the family keeps in a temporally stable manner (Fido) has changed”

(ii) “The value of the function [the dog that the family keeps] at t =/= the value of the function [the dog
that the family keeps] at t’ (used to be Fido, but now it’s Snoopy)”

(d) A bald man is sitting at the window, eating a candy bar. The guy has changed.

To my mind (some of you disagreed), the guy can refer back to the bald man, and so the second sen-
tence could mean, “The bald man didn’t use to eat candy bars but now he does (habit change).” But I
believe the guy (a so-called epithet, a bleached anaphoric definite) can only refer back to an individual (type e). It cannot be construed as a function that picks out different men at different times, so there’s no ambiguity.

Likewise “I visited the Smiths. The dog has changed. The beast used to be a Lab and now it’s a terrier”. To me this can only mean, infelicitously, that Fido changed species. Interestingly, “I visited the Smiths. The dog has changed. It used to be a Lab and now it’s a terrier” seems felicitous, indicating that it can refer back to the temporally variable family dog. He/she also has a limited ability to do this: “The pope is the head of the Catholic church. Naturally, he has always been a Catholic” where he can be the temporally variable pope.

(e) A bald man is sitting at the window, eating a candy bar. The guy is changing.

To my mind, this can mean that the bald man is turning green from the candy bar (property change). Again, unambiguous.

The property-change meaning of change should have the same vanilla type <e,t> as the clothes-change meaning. The verb just looks at the individual in subject position. The lexical semantics of property-change may be an explicitly second-order one that quantifies over properties P, e.g.

\[ \text{change}^1 = \lambda x \lambda t \exists \exists t' [t=/>t' \& P(x)(t) \& \neg P(x)(t')] \]

where t, t’ are times that the tense operator of the sentence uses. (Note that it requires more work to figure out what properties P are relevant; e.g. if Bill goes from the kitchen to the living room, there is a P (being in the kitchen) that no longer holds of him, but we wouldn’t say that Bill has changed.) But property-talk doesn’t seep into the semantic type of change! Change doesn’t take properties as arguments. In The mayor has changed, or Bill has changed, the subject is just an individual, not a property. Mayor is a functional noun, to be this reading we must sync the tense argument of the mayor-function with the reference time of the sentence, i.e. t.

The other meaning of change is similar to rise, discussed by Montague and Lasersohn. We want it to be sensitive to the subject NP’s denotation being temporally variable, so it should be of type <<s,e>,t>. We define,

\[ \text{change}^2 = \lambda x \lambda s \lambda e \lambda t \exists t' [t=/>t' \& X(t)=/>X(t')] \]

This will make sense if the intension of the subject is not a constant function of time. This should work with both the mayor and with the dog (that the family keeps).

Lasersohn would formalize this as, change#2('the_mayor'), where the_mayor denotes the unique individual who is the mayor at the reference time t of the sentence, etc. So far as I can see, if Lasersohn was correct about the treatment of the temperature paradox and the valid inference suggested by Gupta, then his solution carries over, if we offer a theory of when a non-functional noun can be turned into a functional one by adding silent material.

But Lasersohn’s formalism isn’t really suited to manipulate (explicitly quantify over) times, which seems crucial. Furthermore, determiners other than the also work with change#2. Consider,

(f) You wouldn’t get this contract these days. Two town officials has changed. The mayor has been replaced and the controller has been replaced.

(g) In the living room every piece of furniture has changed. The sofa has been replaced with a desk, the coffee table has been replaced with a filing cabinet, ...

See Maribel Romero (The Temperature Paradox and Temporal Interpretation, Linguistic Inquiry 39: 655-667, 2008) for some relevant insights. She also shows that with explicit temporal sensitivity, the correct formalization
is independent of whether we choose the Russellian or the Frege/Strawsonian approach to definiteness. Also Kiyomi Kusumoto, On the quantification over times in nat. lang. (Natural Language Semantics 13 (2005).

Maribel Romero (The Temperature Paradox and Temporal Interpretation, Linguistic Inquiry 39: 655-667, 2008) observes two main things. One is that according to Lasersohn, the definite article that produces an individual (type e) is crucial to saving Montague from Gupta’s problem. But, Romero notes, Gupta’s valid syllogism can be reproduced with different determiners, so the analysis of the must be orthogonal to the main issue:

(22) The prices in supermarket A are (the very same as) the prices in supermarket B.
(23) One/Three/Most price(s) in supermarket A is(are) rising.
(24) One/Three/Most price(s) in supermarket B is(are) rising.

Two, the validity of the inference from (22)-(23) to (24) depends on whether (22) is meant as “always” or as “at time t”. Cf. The mayor is the fire chief, above. In the former case the inference is valid, in the latter case it is not valid. So, Romero says, Lasersohn gets his examples right, but he focuses on the wrong thing (Russell vs. Frege/Strawson) and his solution won’t capture temporal dependence.

Romero offers an analysis using Montague’s original semantics (formalism updated using abstraction and quantification with variables over times, instead of ^, “ and □). This doesn’t mean that using the iota-operator semantics for couldn’t work, she just wants to demonstrate that the Russellian the is not an obstacle.

(1) The temperature in Chicago is (the very same as) the temperature in St. Louis.
(2) The temperature in Chicago is rising.
(3) The temperature in St. Louis is rising.

INVALID:

(38) Translations under the episodic “now” reading
   a. \( \exists x(s,e) \forall y \{ \text{temp-Chicago}(y,i) \mapsto x = y \} \land \exists z(s,e) \forall v \{ \text{temp-StLouis}(v,i) \mapsto z = v \} \land x(i) = z(i) \} \]

   b. \( \forall i' \in \text{Acc}(i) \exists x(s,e) \forall y \{ \text{temp-Chicago}(y,i') \mapsto x = y \} \land \exists z(s,e) \forall v \{ \text{temp-StLouis}(v,i') \mapsto z = v \} \land x(i') = z(i') \} \]

(39) \( \exists x(s,e) \forall y \{ \text{temp-Chicago}(y,i) \mapsto x = y \} \land \text{rise}(i) \]
(40) \( \exists z(s,e) \forall v \{ \text{temp-StLouis}(v,i) \mapsto z = v \} \land \text{rise}(z,i) \]

VALID:

(41) Translations under the habitual “always” reading
   a. \( \exists x(s,e) \forall y \{ \text{temp-Barcelona}(y,i) \mapsto x = y \} \land \exists z(s,e) \forall v \{ \text{temp-LA}(v,i) \mapsto z = v \} \land \forall i'' \leq i \{ x(i'') = z(i'') \} \}

   b. \( \forall i' \in \text{Acc}(i) \exists x(s,e) \forall y \{ \text{temp-Chicago}(y,i') \mapsto x = y \} \land \exists z(s,e) \forall v \{ \text{temp-StLouis}(v,i') \mapsto z = v \} \land \forall i'' \leq i' \{ x(i'') = z(i'') \} \}

(72) The prices in supermarket A are (the very same as) the prices in supermarket B.
(73) One/Three/Most price(s) in supermarket A is(are) rising.
(74) One/Three/Most price(s) in supermarket B is(are) rising.
Note that in (b)-(c), it also mattered whether the mayor is a stable individual or we are talking about different values of the mayor-function, and similarly for the dog that the family keeps.

* 

Greg Carlson (A unified analysis of the English bare plural, Linguistics and Philosophy 1: 413-457, 1977), and his subsequent PhD dissertation, Reference to Kinds in English) provides a semantics for individuals and stages and for individual-level (habitual) and episodic (stage-level) predication.

Jake = a temporally stable individual, type e

its realizations, also type e, are temporal slices, or stages, of the individual

Jake runs (habitual, individual-level) \[ \lambda P[P[jake']] (run') = run'(jake') \]

Jake is running (episodic, stage-level) \[ \lambda P[P[jake']] (\lambda x \exists y[Real(x)(y) & run'(y)]) = \exists y[Real(jake')(y) & run'(y)] \]

Dogs = dog-kind, type e

its realizations, also type e, are individual dogs/dog stages

Dogs bark (kind predication, individual level) = \[ \lambda P[P[dogs']] (bark') = bark'(dogs') \]

Dog are barking in the yard (episodic, stage level) = \[ \lambda P[P[dogs']] ((\lambda x \exists y[Real(x)(y) & bark-in-yard'(y)]) = \exists y[Real(dogs')(y) & bark-in-yard'(y)]) \]

The realizations (temporal slices, stages) of Jake the individual and Dogs the kind are very similar to the values of the temperature/mayor/family dog functions. But the semantic effect is achieved by the Real(ization) relation, not by making Jake and Dogs functions. So type theory is not everything!
Yet another set of related phenomena and another way to formalize the individual/stage distinction (Gupta - Krifka - Doetjes & Honcoop - Barker):

Four thousand ships passed through the lock last year.

(i) “4,000 different ships”

(ii) “4,000 lock traversals by ships, possibly many fewer different ships”

In (i), four thousand ships means 4,000 objects.

In (ii), four thousand ships means 4,000 <event, object> pairs.

The <event,object> pairs are pretty much the same things as the values of the temperature, etc. function (Montague, Romero) and the realizations of the individuals (Carlson).

Haoze Li’s work on event-related measure readings belongs in this same family. See Haoze’s paper in the Short and clear... folder for references.
Advanced Semantics
Szabolcsi

Ecumenical Categorial Grammar
These notes aim to bridge between some of the different versions of categorial grammar
(but not including Type Logical Grammar that uses the sequent calculus)

Categorial grammars share these two main ideas:
(1) Syntactic categories are nothing but explicit encodings of what combinations are possible. E.g. to say that walks belongs to category e\(\backslash t\) is to say that it can combine with a member of category e that occurs to its left (its specifier) and the result will be a member of category t. On this approach walks cannot belong to the same category as knows, since the first takes one and the second two arguments.
(2) Syntactic categories are mapped to semantic types in a trivially simple way. An expression of category e\(\backslash t\) is a function of type \(<e,t>\), one that applies to an entity-type thing and gives a truth-value-type thing (true or false) as a value. This makes a strong commitment to syntax and semantics being parallel.

Basic syntax:

Categories in Lambek notation

\[
e, t \in \text{Cat};
\]
\[
\text{if } A \text{ and } B \in \text{Cat}, A/B \text{ and } B\backslash A \in \text{Cat};
\]
\[
\text{nothing else is in Cat.}
\]

Lexicon:

\[
\begin{align*}
\text{Cat } e &= \{\text{John1, Mary1}\} \\
\text{Cat } t &= \varnothing \\
\text{Cat } e\backslash t &= \{\text{walks}\} \\
\text{Cat } ((e\backslash t)/e) &= \{\text{sees}\} \\
\text{Cat } (t/(e\backslash t)) &= \\
&\quad \{\text{John2, Mary2, everyone, something}\} \\
&(\text{as subjects, i.e. NOM})
\end{align*}
\]

In Jacobson’s notation, Ch 6

\[
S, \text{NP}, \text{PP}, N \in \text{Cat}; \\
\text{if } A \text{ and } B \in \text{Cat}, A/LB, A/RB, \text{and } A/WB \in \text{Cat};
\]
\[
\text{nothing else is in Cat (just yet).}
\]

But most L, R, W are defaults, not lexically marked.

\[
\begin{align*}
\text{Cat } &\text{NP} = \{\text{John1, Mary1}\} \\
\text{Cat } S &= \varnothing \\
\text{Cat } S/L\text{NP} &= \{\text{walks}\} \\
\text{Cat } (S/\text{NP})/R\text{NP} &= \{\text{sees}\} \\
\text{Cat } S/R(S/\text{NP}) &= \\
&\quad \{\text{John2, Mary2, everyone, something}\} \\
\text{Cat } ((S/\text{NP})/W\text{NP})/P[TO] &= \{\text{introduce}\}
\end{align*}
\]

Defaults:

\[
\begin{align*}
S/X &\text{ is } S/X, \quad (\text{subjects}) \\
X/X &\text{ is } X/X, \quad (\text{adjuncts}) \\
X/Y &\text{ is } X/RY \text{ elsewhere.}
\end{align*}
\]

[SG], [PL], [NOM], [ACC], [TO] on arguments.

Rules:

If \(\alpha \in \text{Cat } A/B \text{ and } \beta \in \text{Cat } B\), then \(\alpha\beta \in \text{Cat } A\).
If \(\beta \in \text{Cat } B \text{ and } \alpha \in \text{Cat } B\backslash A\), then \(\beta\alpha \in \text{Cat } A\).

If \(\alpha \in \text{Cat } A/\backslash B \text{ and } \beta \in \text{Cat } B\), then \(\alpha-\beta \in \text{Cat } A\).
If \(\beta \in \text{Cat } B \text{ and } \alpha \in \text{Cat } A/\backslash B\), then \(\beta-\alpha \in \text{Cat } A\).
If \(\alpha \in \text{Cat } A/\backslash B \text{ or } \text{Cat } A/\backslash B\), and \(\alpha\) has an infixation point | so that its form is \(x\backslash y\), then the result of concatenation inherits |.
If \(\alpha \in \text{Cat } A/WB \text{ and } \alpha\) has the form \(x\backslash \beta\), and \(\beta \in \text{Cat } B\), then \(x\backslash \beta\gamma \in \text{Cat } A\).
Semantics:

The universe is typed. Types are “categories of things in the model”. The tangible members of the universe of discourse are individuals, but functions of all sorts can be defined based on individuals and truth values. E.g., functions of type \(<e,e>\) (from entities to entities) or \(<e,t>\) (from entities to truth values).

A model \(M\) is \(<D,I>\), where \(D\) is a set of individuals. \(I\) is an interpretation function from expressions to things in the universe. The \(\text{Dom}(\text{ain})\) of a type with respect to a universe \(D\) is the set of things in \(D\) that belong to that type: entities, or truth values, or various kinds of functions.

Types: \(e,t \in \text{Type};\)

\[
\begin{align*}
\text{Dom}_{e,D} &= D, \\
\text{Dom}_{t,D} &= \{0,1\} \\
\text{Dom}_{<B,A>,D} &= \{f: \text{Dom}_{B,D} \to \text{Dom}_{A,D}\}
\end{align*}
\]

if \(B, A \in \text{Type}\), \(<B,A> \in \text{Type};\)

nothing else is in \(\text{Type}\).

Syntactic categories are related to semantic types by the following simple map \(f\). Expressions of category \(e\) are mapped to things of category \(e(n\text{ntity})\). Expressions of category \(t\) are mapped to things of category \(t(\text{truthvalue})\). Function categories are mapped to appropriate functions. Syntactic directionality becomes irrelevant (likewise for Jacobson).

\[
f(e) = e \quad f(t) = t \quad f(A/B) = f(B \setminus A) = <f(B), f(A)>
\]

We ignore intensional types. Different authors add \(s\) differently. For Montague, all function categories are mapped to type \(<<s, (fB)>, f(A)>\). For Muskens 2011, only \(t\) is intensionalized: \(f(t) = <s, t>\).

Let \(D = \{j, m, f\}\) and let \(I\) work as follows:

\[
\begin{align*}
[[\text{Mary1}]]^M &= m \\
[[\text{walks}]]^M &= \text{that function from individuals to truthvalues which assigns 1 to an individual iff it walks, viz. the characteristic function of the set of walkers (in our M, let it be \(\{m, j\}\))} \\
[[\text{sees}]]^M &= \text{that function from individuals to [a function from individuals to truthvalues] which assigns 1 to a pair of individuals if the first sees the second} \\
[[\text{Mary2}]]^M &= \text{that function from sets of individuals to truthvalues which assigns 1 to a set P iff m\(\in\)P, viz. the characteristic function of the set of those sets that contain m} \\
[[\text{everyone}]]^M &= \text{that function from sets of individuals to truthvalues which assigns 1 to a set P iff human\(\subset\)P, viz. the characteristic function of the set of those sets that contain everyone} \\
[[\text{something}]]^M &= \text{that function from sets of individuals to truthvalues which assigns 1 to a set P iff thing\(\cap\)P\(\neq\)\(\emptyset\), viz. the characteristic function of the set of those sets that contain something}
\end{align*}
\]

Concatenation and wrap are interpreted as functional application.

If \(\alpha\) is a function expression of category \(A/RB, A/LB,\) or \(A/WB,\)
and \(\beta\) is of category \(B,\)
then the result of their combination is invariably \([[\alpha]]^M ( [[\beta]]^M ).\)
Some further specifics

Common nouns vs. intransitive verbs. Like walks, boy is interpreted as a function of type <e,t> (the characteristic function of the set of boys), but boy does not combine directly with subjects in English syntax. Syntax makes more distinctions here than semantics. To deal with common nouns, one may introduce a new primitive category N, which is mapped to <e,t> by fiat. Or, one may add a syntactically “inert” category, let’s say S//NP, mapped to type <e,t> as expected. S//NP is syntactically inert if there is no syntactic rule that combines X//Y with Y, but there are functions that look for a X//Y.

\[
\text{the boy walks, } S \\
\text{the, } NP/_{R}(S//NP) \\
\text{boy, } S//NP \\
\text{walks, } S/_{L}NP
\]

Different notations for the directionality of combination. Only in categories! Semantic types are always written as Dom_{<A,B>,D} = \{f: Dom_{A,D} \rightarrow Dom_{B,D}\}, viz. <A,B> is the type of functions from A-type things to B-type things (relative to the universe of discourse D).

In Montague 1974 (=PTQ), the category of walks is notated as t/e, and the (extensional) type of walks as <e,t>. Montague’s notation is not meant to encode left-right in the categories, and in mapping categories to types you simply invert the order: (value-cat / argument-cat) maps to <argument-type, value-type>.

In a directional Lambek grammar, walks is e\ t and knows is (e\ t)/e, to encode that the direct object is expected from the right and the subject from the left. This notation reflects the division/multiplication analogy for functions and concatenation: value-cat ≈ numerator, argument-cat ≈ denominator, e . e\ t = t, t/e . e = t. To perform the category-type mapping, you need to find the ultimate value category above the main slash and the corresponding argument category under the main slash, etc. This notation is often used in present-day literature.

Another notation is used by Steedman and in my own earlier papers. Here only the leaning of the slash, but not the placement of the argument category, indicates whether the argument is expected from the left or the right. Thus, Lambek’s (subject\ t)/object is Steedman’s (t\ subject)/object. This has a perceptual advantage with multi-layered categories: it is easy to see at once who is the argument and who is the function value. But in all these cases, the type is written as <object, <subject, t>>.

Jacobson uses a similar strategy as Steedman (value category / argument category), but indicates left-right orientation with a subscript L or R (and wrap by subscript W) on the slash. So, a transitive verb has the category \(S/_{L}NP[NOM]\ ) /_{R} NP[ACC].
Hands-on categorial grammar (cf. Jacobson Chs %6, 7, 8)

When a function $\alpha$ is a characteristic function of some set, the application of $\alpha$ to some argument $\beta$ is equivalent to saying that $\beta \in \alpha$. That is, in our model $\text{[walk]}([\text{[John]}]) = j \in \{m,j\}$.

When a function $\alpha$ is a characteristic function of some set, the application of $\alpha$ to some argument $\beta$ is equivalent to saying that $\beta \in \alpha$. That is, in our model $\text{[walk]}([\text{[John]}]) = j \in \{m,j\}$.

$\lambda.P \forall x[h'(x) \rightarrow P(x)](\text{walk'}) = \forall x[h'(x) \rightarrow \text{walk'}(x)]$

Alternatively, John walks might be built in analogy to Everyone walks, using $\text{John2}$. John walks will come out interpreted exactly the same both ways, in one derivation with $\text{John1}$ as the argument and walks the function, and in the other, with $\text{John2}$ as the function and walks the argument.

$\lambda.P[P(j)](\text{walk'}) = \text{walk'}(j)$

$\lambda.P[P(j)] : <<e,t>,t> \quad \text{walk'} : <e,t>$

$\text{John1}$ and $\text{John2}$ should not be independent: $\text{John2}$ will be the lifted version of $\text{John1}$. (See next page.)

In view of the defaults, we know that $S/(S/NP[\text{NOM}])$ is $S/(S/LNP)[\text{NOM}]$. We’ll need to ensure that $S/(S/NP[\text{NOM}])$ is $S/(S/LNP)[\text{NOM}]$, irrespective of whether it is the only category of the expression (cf. everyone) or it is obtained by lifting (cf. $\text{John2}$).

Another, standard way of writing these in the categorial literature:

$\text{John1}$ walks $\text{Everyone}$ walks

NP $S/LNP$ $S/R(S/LNP)$ $S/LNP$

S

Do the categorial derivations of the following sentences a la Jacobson, Chs 6-7-8, possibly with some innovations. The first task is to assign lexical categories to each word. Postulate as little category ambiguity as possible and be consistent. Watch out – one of these is a trick question.

(a) Kim and every cat jumped.    (g) Kim saw the proud cat.
(b) Kim hails from Ohio.         (h) Kim saw the cat proud of Pat.
(c) Kim sneezed and kicked Pat.  (i) Kim may hail from Ohio.
(d) Kim saw and kicked Pat.      (j) Kim doesn’t hail from Ohio.
(e) Kim or Pat sneezed.          (k) Kim saw and Pat heard Chris.
(f) Kim showed Pat to Chris.     (l) Kim showed Pat to Chris and Fido to Fluffy.
Looking forward: CG with more than just functional application. \([a] \) will be written as \(a'\). 

Concatenation interpreted as functional application. The basic operation. No annotation. The result of concatenating \(a\) and \(b\) is interpreted as \(a'(b')\).

\[
\begin{array}{ccc}
  a & b & b & a \\
  x/y & y & y & y\backslash x \\
  x & x
\end{array}
\]

Category lifting, a unary operation. Notated as \(T\). Reverses function/argument relations between two “sister nodes”. If expression \(a\) has syntactic category \(x\), then its lifted version will have the syntactic category of a function that looks for functions that themselves would look for things of category \(x\). \(y\) can be any category. The facing / \ orientation of the slashes in the lifted category preserves the original order of combination. With lambdas, lifted \(a\) is interpreted as \(\lambda f[f(a')]\).

\[
\begin{array}{ccc}
  a & x & a \\
  x & x \\
  y/(x\backslash y) & (y/x)\backslash y
\end{array}
\]

Concatenation interpreted as function composition. Notated as \(B\). If \(b\) looks for an expression of category \(z\) to return an expression of category \(y\), and \(a\) looks for \(y\) to return \(x\), then their composition looks for \(z\) to return \(x\). Here \(a\) is called the principal (main) functor and \(b\) the minor functor. The result of composing \(a\) with \(b\) is interpreted as \(\lambda v[a'(b'(v))]\) (also written as \(a' \circ b'\)).

The Geach rule (a.k.a. division) is unary composition: \(B\) applied to one argument: \(\lambda f\lambda v[a'(f(v))]\).

\[
\begin{array}{ccc}
  a & b & b & a \\
  x/y & y/z & z\backslash y & y\backslash x \\
  x/z & z\backslash x & x/z
\end{array}
\]

\[
\begin{array}{cccc}
  z & b & y & a \\
  & & & x
\end{array}
\]

\(B(a)(b)\) where the range of \(b\) is a subset of the domain of \(a\)

In syntax, composition allows one to bracket things together that are by default not bracketed together. It will be useful in coordination, in handling extraction without traces, and also in assigning different constituent structures to a simple sentence that correspond to different intonation patterns, e.g. \(JOHN\ saw\ Bill\) versus \(John\ saw\ BILL\).

The disharmonic \(y/z.y\backslash x=x/z\) version of composition \((B/\backslash)\) gives a limited amount of freedom of constituent order. If we also included \(x/y\). \(z\backslash y = z\backslash x\), order would be entirely free (“permutation closure”). Lifting and harmonic composition are theorems of the Lambek calculus; disharmonic composition is not. Moortgat (1999) supplements the LC with structural rules (permutation, associativity) to achieve similar effects without disharmonic composition.
Some readings re: categorial grammar with combinators and the Lambek calculus

http://semanticsarchive.net/Archive/TFmMjVkZ/bernardi_szabolcsi_optionality_scope/licensing_2007_nov.pdf


Steedman, Mark (1996), Surface Structure and Interpretation. The MIT Press.


Szabolcsi, Anna (1992), Combinatory grammar and projection from the lexicon. In Sag and Szabolcsi, eds., Lexical Matters. CSLI. (in Resources)
2.2 Generalized quantifiers and their elements: operators and their scopes

In many logics, operators are introduced syncategorematically. They are not expressions of the logical language; the syntax only specifies how they combine with expressions to yield new expressions, and the semantics specifies what their effect is:

1) If $\phi$ is a formula, $\forall x[\phi]$ is a formula.
   $\forall x[\phi]$ is true if and only if every assignment of values to the variable $x$ makes $\phi$ true.

The quantifier prefix $\forall x$ functions like a diacritic in the phonetic alphabet: ' is not a character of the IPA but attaching it to a consonant symbol indicates that the sound is palatal (e.g. [t']). In line with most of the linguistic literature we are going to assume that operators embodied by morphemes or phrases are never syncategorematic. But if every and every dragon are ordinary expressions that belong to some syntactic category, then, by the principle of compositionality, they must have their own self-contained interpretations. This contrasts with the situation in predicate logic. In (2) the contributions of every and every dragon are scattered all over the formula without being subexpressions of it. Everything in (2) other than guard treasure' comes from every dragon, and everything other than guard treasure' and dragon' comes from every.

2) Every dragon guards treasure.
   $\forall x[\text{dragon}'(x) \rightarrow \text{guard treasure}'(x)]$

Not only would we like to assign a self-contained interpretation to every dragon, we would also like to assign it one that resembles, in significant respects, the kind of interpretations we assign to Smaug and more than three dragons. The reason why these are all categorized as DPs in syntax is that they exhibit very similar syntactic behavior. It is then natural to expect them to have in some respects similar semantics. If they did not, then the syntactic operations involving DPs (e.g. merging DP with a head, in current terminology) could not be given uniform interpretations. To a certain point it is easy to see how that interpretation would go. Assume that the DP Smaug refers to the individual $s$ and the predicate (TP, a projection of Tense) guards treasure to the set of individuals that guard treasure. Interpreting the DP–TP relation as the set theoretical element-of relation, Smaug guards treasure will be interpreted as $s \in \text{guard treasure}'$. Now consider Every dragon guards treasure. The DP every dragon does not denote an individual, but we can associate with it a unique set of individuals, the set of dragons. Reinterpreting DP–TP using the subset relation, Every dragon guards treasure is compositionally
interpreted as \( \text{dragon}' \subseteq \text{guard treasure}' \). To achieve uniformity, we can go back and recast \( s \in \text{guard treasure}' \) as \( \{s\} \subseteq \text{guard treasure}' \), with \( \{s\} \) the singleton set that contains just Smaug. But indefinite DPs like \textit{more than three dragons} still cannot be accommodated, because there is no unique set of individuals they could be associated with. In a universe of just 5 dragons, sets of more than three dragons can be picked in various different ways.

One of Montague’s (1974a) most important innovations was to provide a self-contained and uniform kind of denotation for all DPs in the form of generalized quantifiers, introduced mathematically in Mostowski (1957) based on Frege’s fundamental idea. The name is due to the fact we generalize from the first order logical \( \forall \) and \( \exists \) and their direct descendants \textit{every dragon} and \textit{some dragon} to the whole gamut, \textit{less than five dragons}, \textit{at least one dragon}, \textit{more dragons than serpents}, \textit{the dragon}, etc., even including proper names like \textit{Smaug}.

A generalized quantifier is a set of properties. In the examples below the generalized quantifiers are defined using English and, equivalently, in the language of set theory and in a simplified Montagovian notation, to highlight the fact that they do not have an inherent connection to any particular logical notation. The main simplification is that we present denotations extensionally. Thus each property is traded for the set of individuals that have the property (rather than the intensional analogue, a function from worlds to such sets of individuals), but the term “property” is retained, as customary, to evoke the relevant intuition. This approach fits all three of our examples equally well:

\[
(3) \quad \begin{align*}
& \text{a. } \text{Smaug denotes the set of properties that Smaug has. If Smaug is hungry, then the property of being hungry is an element of this set.} \\
& \text{b. } \text{Smaug denotes } \{P : s \in P\}. \text{ If Smaug is hungry, then} \\
& \{a : a \in \text{hungry}'\} \in \{P : s \in P\}. \\
& \text{c. } \text{Smaug denotes } \lambda P[P(s)]. \text{ If Smaug is hungry, then} \\
& \lambda P[P(s)](\text{hungry}') \text{ yields the value True.}
\end{align*}
\]

\[
(4) \quad \begin{align*}
& \text{a. } \text{Every dragon denotes the set of properties that every dragon has. If every dragon is hungry, then the property of being hungry is an element of this set.} \\
& \text{b. } \text{Every dragon denotes } \{P : \text{dragon}' \subseteq P\}. \text{ If every dragon is hungry, then} \\
& \{a : a \in \text{hungry}'\} \in \{P : \text{dragon}' \subseteq P\}. \\
& \text{c. } \text{Every dragon denotes } \lambda P\forall x[\text{dragon}'(x) \rightarrow P(x)]. \text{ If every} \\
& \text{dragon is hungry, then } \lambda P\forall x[\text{dragon}'(x) \rightarrow P(x)](\text{hungry}') \text{ yields the value True.}
\end{align*}
\]
To visualize a generalized quantifier we draw the Hasse-diagram of the powerset of the universe. The lines represent the subset relation, thus \{a\} is below \{a, b\} and \{a, b\} below \{a, b, c\}, because \{a\} \subseteq \{a, b\} \subseteq \{a, b, c\}. Each generalized quantifier is represented as an area (a subset) in this diagram. If Smaug is the individual \(a\), and the set of dragons is \{a, b, c\}, the generalized quantifiers denoted by the DPs Smaug, every dragon, and more than one dragon are the shaded areas in Figures 2.1, 2.2, and 2.3, respectively. Such diagrams will be used over and over in Chapter 4.

Fig. 2.1 The set of properties Smaug has: all the sets that have \(a\) as an element

Fig. 2.2 The set of properties every dragon has: all the sets that have \{a, b, c\} as a subset

Fig. 2.3 The set of properties more than one dragon has: all the sets whose intersection with \{a, b, c\} has more than one element
Boolean operations: intersection, union, complement

Fig. 4.1 The intersection of two generalized quantifiers

\[ \text{every dragon}' = \{\{a, b\}, \{a, b, c\}, \{a, b, d\}, \{a, b, c, d\}\} \]

\[ \text{at least one serpent}' = \{\{c\}, \{d\}, \{a, c\}, \{a, d\}, \{b, c\}, \{b, d\}, \{c, d\}, \{a, b, c\}, \{a, b, d\}, \{a, c, d\}, \{b, c, d\}, \{a, b, c, d\}\} \]

\[ \text{every dragon and at least one serpent}' = \{\{a, b, c\}, \{a, b, d\}, \{a, b, c, d\}\} \]

(15) a. A function \( f \) is monotonically increasing (with respect to a particular argument) iff it preserves the partial ordering in its domain. That is, if \( X, Y \) are in the domain of \( f \) and \( X \leq Y \), then \( f(X) \leq f(Y) \).

b. A function \( f \) is monotonically decreasing (with respect to a particular argument) iff it reverses the partial ordering in its domain. That is, if \( X, Y \) are in the domain of \( f \) and \( X \leq Y \), then \( f(X) \geq f(Y) \).

c. A function \( f \) is non-monotonic (with respect to a particular argument) iff it obliterates the partial ordering in its domain. That is, if \( X, Y \) are in the domain of \( f \) and \( X \leq Y \), then neither \( f(X) \leq f(Y) \) nor \( f(X) \geq f(Y) \) is guaranteed.

Fig. 4.6 A decreasing generalized quantifier

\[ \text{at most two robots}' = \{\{b\}, \{a\}, \{\emptyset\}, \{c\}, \{d\}, \{a, b\}, \{a, c\}, \{a, d\}, \{b, c\}, \{b, d\}, \{c, b\}, \{a, b, d\}, \{a, c, d\}, \{b, c, d\}\} \]

Fig. 4.7 A non-monotonic generalized quantifier

\[ \text{exactly two robots}' = \{\{a, b\}, \{a, c\}, \{b, c\}, \{a, b, d\}, \{a, c, d\}, \{b, c, d\}\} \]
**Semantic determiners (DET)**
(Szabolcsi 2010, Ch 4)

**Terminological caveat:** DETs are studied as part of GQ theory but they are not GQs, in careful parlance. DETs are Lindström’s <1,1> quantifiers, GQs are Lindström’s <1> quantifiers.

**Determiners as relations between two sets**

\[
\begin{align*}
\text{DET is symmetrical} & \quad \text{iff} \quad \text{DET}(N)(P) = \text{DET}(P)(N) \\
\text{DET is intersective} & \quad \text{iff} \quad \text{DET}(N)(P) = \text{DET}(N \cap P)(P) \\
\text{DET is co-intersective} & \quad \text{iff} \quad \text{DET}(N)(P) = \text{DET}(N \setminus P)(P) \\
\text{DET is proportional} & \quad \text{iff} \quad \text{DET}(N)(P) \quad \text{iff} \quad \frac{n}{m} \text{ of } N \text{ are in } P
\end{align*}
\]

- e.g. a(n), two, no
- e.g. a(n), two, no
- e.g. every
- e.g. most of, few of

**Determiners as functions from restrictor-denotations to GQs -- Currying**

\[
\begin{align*}
\text{Restrictedness} & = \text{Conservativity} + \text{Extension} \\
\text{DET is conservative} & \quad \text{iff} \quad \text{DET}(N)(P) = \text{DET}(N \cap P) \\
\text{DET has extension} & \quad \text{iff} \quad \text{for all universes } E_1, E_2, \\
& \quad \text{if} \quad N_{E1} = N_{E2} \text{ and } P_{E1} = P_{E2}, \\
& \quad \text{then} \quad \text{DET}(N_{E1})(P_{E1}) = \text{DET}(N_{E2})(P_{E2}).
\end{align*}
\]

- area (iii) can be ignored
- area (iv) can be ignored

**Empirical claim:** Natural language determiners are (overwhelmingly) restricted.

**Significance:** syntax–semantics match
- computation, learnability
- inherently sortal DETs are possible
**DET is sortally reducible** if $\text{DET}(N)(P)$ can be expressed in terms of $\text{DET}$ quantifying over the whole universe $E$. $\text{DET}$ is inherently sortal if must be defined as quantifying over the restrictor $N$.

$\text{DET}$ is sortally reducible \iff there is a function $\#$ into \{0,1\} expressible in terms of $\land, \lor, \neg$ such that for all $N,P \subseteq E$, $\text{DET}(N)(P) = \text{DET}(E)(\#(N,P))$.

If there is no such $\#$, $\text{DET}$ is inherently sortal.

You have already convinced yourselves that there is no Boolean connective $\#$ such that $\text{Most dragons are asleep}$ \iff $M_x[\text{dragon}(x) \# \text{asleep}(x)]$, where $M_x[\varphi]$ is “most things in $E$ are $\varphi$.”

(Sortally reducible $\neq$ definable in first order logic. The DET *most* is not sortally reducible and not first-order definable, because it critically depends on the $N$-set. It is not first-order definable, because it talks about arbitrary cardinalities. *(In)finately many* is sortally reducible, while not definable in first order logic (cardinality). *Neither, both, and the ten* are first-order definable, while not sortally reducible, because they impose presuppositions specifically on the $N$-set.)

* 

**Not only DETs are restricted.** Adverb-meanings like [[always]], [[for the most part]] and [[only]] are too, although their restrictors may be more difficult to compute. The restrictor set may be given by an if/when clause, but topic-focus relations, presuppositions, event-structure, etc. also play a role. (See “Cross-categorial parallelisms” in the Background folder.)

**Possibly unrestricted DETs?** This election could have two winners. ‘there are two individuals who could win this election’

**Covert restrictions on the domain of quantification (=restrictor of DET)**

Every *bottle* is empty. 

In the whole universe?

The *syntacticians* lobbied for the *syntacticians*. 

Consider disjoint sets: organizers, submittors.

More than one child devoured every *cookie*. 

How many times was each cookie devoured?

* 

**Scalar implicatures in determiner interpretation**

Some students love formal semantics. 

=\implies \text{Not all students love formal semantics.}
HW6 Due on March 27

Reading:
- Jacobson Chs 6, 7, and 8, in full (= whether or not we discussed every issue the chapters bring up).
- Ch 9 is lambdas and predicate logic; we’ve done that, so skip to Ch 10, Generalized quantifiers.
- Be sure to have read the Compositionality and Scope chapters from the Preparation folder before Monday (just in case you haven’t yet).

Organization:
We should start/continue to discuss your term paper plans. I’ll send an email re: appointments.

Written:
Do the categorial grammar derivations of those examples below that you already can, based on (i) the toy grammar in the handout and (ii) Jacobson’s Chs 6-7-8 (so read before you start). It is your task to determine which of the examples are already within your reach, based on (i)-(ii) plus a little bit of creativity.

Write the derivations in the linear format. For better readability, keep the interpretations out of the derivations. Instead, write out the \( \lambda \)-expressions separately below. Be sure to make the semantic types of the interpretations exactly what the syntactic categories in the derivation predict.

<table>
<thead>
<tr>
<th>S/( R(\text{S/}\text{NP}) )</th>
<th>S/\text{NP}</th>
</tr>
</thead>
<tbody>
<tr>
<td>\text{everyone} walks</td>
<td>\text{S}</td>
</tr>
<tr>
<td>\text{everyone} = \lambda \text{P} \forall x [h(x) \rightarrow P(x)]</td>
<td></td>
</tr>
<tr>
<td>\text{walks} = \text{walk'}</td>
<td></td>
</tr>
<tr>
<td>everyone \text{walks} = \lambda \text{P} \forall x <a href="%5Ctext%7Bwalk'%7D">h(x) \rightarrow P(x)</a> = \forall x [h(x) \rightarrow \text{walk'}(x)]</td>
<td></td>
</tr>
</tbody>
</table>

Some of the examples require the use of lifting or functional composition, defined on the penultimate page of the handout. If you want, you can try to do those exx too, but then use lifting and composition strictly as defined in the handout. (We’ll cover these in due course.) The main thing though is to do an excellent job on the exx that you can officially handle already. The first task is to assign a lexical category to each word. Postulate as little category ambiguity as possible across examples, and be consistent. Watch out – one of the exx is a trick question.

(a) Kim and every cat jumped.        (g) Kim saw the proud cat.
(b) Kim hails from Ohio.             (h) Kim saw the cat proud of Pat.
(c) Kim sneezed and kicked Pat.      (i) Kim may hail from Ohio.
(d) Kim saw and kicked Pat.          (j) Kim doesn’t hail from Ohio.
(e) Kim or Pat sneezed.              (k) Kim saw and Pat heard Chris.
(f) Kim showed Pat to Chris.         (l) Kim showed Pat to Chris and Fido to Fluffy.
Reading – this is important and requires time

(i) Related to (but somewhat going beyond) the material covered this week, pp. 5-12 (of Ch 2) and pp. 45-62 (of Ch 4) of Quantification [in the Background folder] The endnotes should be read as part of the text -- lots of good stuff there.

The notion of a witness set (4.1.3) is good to know about, but we won’t explicitly work with it.

(ii) In preparation for next week, Determiner denotations (pp. 60-70, of Ch 4). Much of this will be familiar from Determiners and nouns (=section 7.3.3 of Compositionality).

We will come back to the treatment of scope in Chapter 2 after this.

Written

Use the blank Hasse-diagrams [in Resources] for this assignment, both (1) and (2).

Do the shadings clearly, using color pencils or highlighters, so that both you and I can see at a glance what’s going on in each diagram. If comments are required, write them next to the diagram.

(1) "Quantification" contains 5 Hasse-diagrams in Chapter 4 where the membership of the sets is given in the caption, but the shading is left to the gentle reader. Take your own diagrams and do the shading for each of these. Reason out how the shading has to go, but check your results against the caption in the book, and make corrections as needed.

(2) Let the humans in the universe be \{Andy, Belinda, Carl\}. Using altogether 4 Hasse-diagrams,

(i) Shade each of the GQs \{P: P(a)\}, \{P: P(b)\}, and \{P: P(c)\}. In Keenan’s terminology, these are the “Montagovian individuals”.

(ii) Shade the GQ [[everyone]], and concisely say by what operation(s) it is obtained from the Montagovian individuals.

(iii) Shade the GQ [[someone]], and concisely say by what operation(s) it is obtained from the Montagovian individuals.

(iv) Shade the GQ [[no one]], and concisely say by what operation(s) it is obtained from the Montagovian individuals.

(v) You don’t need to do the shading for this, but say how you’d obtain the GQ [[two people]].
Quantifier scope -- a preview

The scope of a quantifier phrase is that part of the sentence which denotes a property asserted to be an element of the generalized quantifier denoted by the quantifier phrase (on the given analysis of the sentence).

If another operator (connective, modal, or another quantifier) is buried in the definition of that property, then the generalized quantifier scopes over the operator.

(7')  [[every dragon]] > [[or]]
\[ \lambda P[\text{every dragon}'(P)] (\lambda x[\text{flies}'(x) \text{ or lumbers}'(x)]) \]

(8')  [[more than one dragon]] > [[every adventurer]]
\[ \lambda P[\text{more than one dragon}'(P)] (\lambda y[\text{spotted every adventurer}'(y)]) = \]
\[ \lambda P[\text{more than one dragon}'(P)] (\lambda y [\lambda P[\text{every adventurer}'(P)] (\lambda z[\text{spotted}'(z)(y)]) ) ) = \]
\[ \text{GQ1} (\lambda y [\text{GQ2} (\lambda z [\text{verb}'(z)(y)]) ] ) ] \]

(9')  [[every adventurer]] > [[more than one dragon]]
\[ \lambda P[\text{every adventurer}'(P)] (\lambda z[\text{more than one dragon spotted}'(z)]) \]
\[ \lambda P[\text{every adventurer}'(P)] (\lambda z [\lambda P[\text{more than one dragon}'(P)] (\lambda y[\text{spotted}'(z)(y)]) ) ) = \]
\[ \text{GQ2} (\lambda z [ GQ1 (\lambda y [\text{verb}'(z)(y)]) ] ) ] \]

The semantic constituent structures corresponding to (7')-(8')-(9') are,

(12)  ((Every dragon) ((flies) or (lumbers)))

(13)  ((More than one dragon) ((spotted) (every adventurer)))

(14)  ((More than one dragon) (spotted) (every adventurer))

This is the prize we keep our eyes on.

The task is to figure out how to design syntactic derivations and matching compositional interpretations that deliver the above, in a linguistically motivated way.

Accomplishing the task will involve multiple steps: introducing type lifting, generalized conjunction, and functional composition, with a rich set of data pertaining to each, and with comparisons between alternative approaches in the literature.
Readings for Monday, April 10

As I said, there is no written assignment for this weekend, but there are substantial readings.

(i) Complete reading Szabolcsi Ch 4 (Some empirically significant properties of quantifiers and determiners) that we have discussed. Optionally, I very much recommend Ch 5 (Potential challenges for generalized quantifiers).

(ii) Corresponding to today's "preview", be sure to read Szabolcsi Ch 2, up to 2.3.2 now. (FYI, we are going to cover 2.3.2 (Montague), 2.3.4 (May), and 2.3.5 (Hendriks), which are all referenced and relied on in Jacobson's book as well.)

(iii) Next week we'll discuss Jacobson Chs 11-12 (type lifting and generalized conjunction). These are interesting in their own right, but Jacobson also uses them to lead up to the treatment of quantifier scope. We did the "preview" today so you recognize where we are heading in those chapters. It'd be great if you read them in advance. Moreover, I ask you to bring the text with you. I posted them in Resources. You might bring your laptops, or print the chapters out, or bring the book itself, whichever you think is easiest for you to use in classroom discussion. I won't be bringing handouts.
Advanced Semantics
Szabolcsi

Generalized conjunction and functional composition

- Be sure to read Jacobson’s Chs 11 and 12 in full.

- For generalized conjunction, I very much recommend at least the first few pages of Partee & Rooth (1983) in the “Backgrounds” folder.

- The last page of my “Categorial Grammar” handout (posted on March 19) defines lifting and composition in simple and precise terms.

- Jacobson discusses composition in Ch 13.4.1 and the Geach rule (division, composition applied to just one argument) in 13.4.4.

- For derivations and discussion of “right-node raising” (John likes, and Mary hates, all of these cakes), “non-constituent coordination” (I gave the book to Mary and the record to Sue), and their relation to prosody, see Steedman (1991) in the “Backgrounds” folder.

  Notice that (unlike Jacobson) Steedman assumes that the first argument of give is the NP immediately to its right, so in (25) he lifts differently from what we did in class. This is immaterial in the big scheme of things. You first determine, based on whatever evidence you see fit, what the order of the arguments of ditransitive give is, and then ensure that George a book (or a book to George) forms a constituent that looks for give of that category to its left.

  Recall that Jacobson’s (S/NP)/SNP is Steedman’s (S/NP)/NP.

- Be assured that everything important that Jacobson, Steedman, and my handouts say is entirely consistent. The differences are only in the slash notation and in a linguistic decision regarding ditransitive verbs.

Practice problems

After you studied Jacobson and, say, the first 15 pages of Steedman, try to derive the following examples:

1. John brought ten cups and saucers. (5 / 10 cups with matching saucers)
2. John visited, but Susan didn’t visit, Mary.
3. John visited Ted on Tuesday and Will on Wednesday.
4. John introduced Ettie to Elsa and Penny to Petra.

How about these? See what you can do, and comment.

5. John saw two, and Susan heard three, sopranos.
6. The Japanese won and French lost. (say, in a Japan--France soccer game)
The grammar of quantifier scope

Readings corresponding to what we covered on 4-17 and 4-19

(a) Jacobson Ch 14, Generalized quantifiers in object position: Two approaches
[on p. 247, g-sl comes from 13.4.4.2]

(b) Szabolcsi 2010, 2.3.2 (Montague), 2.3.4 (May), 2.3.5 (Hendriks)

(c) Szabolcsi 2010, 6.1 Different quantifiers, different scopes

(d) Recommended: Szabolcsi, Ch 11, Clause-internal scopal diversity

Written assignment, due Monday April 24

(1) Summarize the contents of Jacobson’s Ch 14 in 1/2 to 1 single-spaced page. What is the problem the chapter focuses on? What are the two approaches in the literature that are contrasted? Which approach does Jacobson advocate, and why?

(2) We have considered or and not in earlier assignments. Just in case your solution was not perfect, recall that John or Sue does not denote an individual, so it cannot be of type e; it must be a generalized quantifier. Also that didn’t has semantic type <<e,t>,<e,t>>, with interpretation \(\lambda P \lambda x [\neg P(x)]\). Using these ingredients, build the following sentences on the indicated interpretations. The task is to assign the correct and syntax and semantics to John or Sue and to determine how its interaction with negation can be handled using the Montagovian quantifier scoping apparatus. Write out both full derivations, syntax and semantics.

You may use either Montague’s method or Hendriks’s method as a point of departure -- you choose.

Start out with a brief statement that explains how the interpretations specified below are matters of scope: what is scoping over what in each case. Go on to build the derivations only when you see these clearly.

(a) Mary didn’t visit John or Sue ["she visited neither"]

(b) Mary didn’t visit John or Sue ["I’m not sure which she didn’t visit"]

(3) Do either the collective readings task on the next page or the practice problems from “Generalized conjunction and functional composition” (you may do both if you wish).
Read 5.3 “Collective readings” as a background. This exercise details the way Winter 2001 obtains collective readings without using Link’s non-Boolean and interpreted as $\oplus$. Winter’s solution is also a popular one.

Winter starts from *John and Mary* interpreted as $\lambda P(j) \land P(m)$ and interprets *John and Mary collided* ultimately as $\text{collided}'((j,m))$. Here is how. Names, definites and their conjunctions as generalized quantifiers are principal filters: they have a unique minimal element (the generator set). Winter’s MIN is a so-called type-shifter that identifies the set of minimal elements of the GQ, and his Existential Raising, $E$ turns that into a quantifier. Existential Raising is basically a generalization of the existential determiner, generalized to any type $\theta$.

Inspect the MIN shifter. Apply it to $\lambda P(j) \land P(m)$, and explain how it picks out $\{(j,m)\}$, the set of minimal elements of this GQ. Then apply $E$ and derive *John and Mary collided*.

\[
\text{MIN}_{((0,t),(0,t))} = \lambda Q_{(0,t)} [\lambda A_0 [Q(A) \land \forall B \in Q [B \subseteq A \rightarrow B = A]]]
\]

\[
E_{((0,t),((0,t),t))} = \lambda R_{(0,t)} [\lambda P_{(0,t)} [\exists X_0 [R(X) \land P(X)]]]
\]

The goals of this exercise are two-fold: to see that collective readings can be handled within a fully Boolean system (if we that is what we want), and to practice deciphering formalization, in preparation for reading the literature.
Advanced Semantics
Szabolcsi

**Restrictive relative clauses and questions**

Wh-movement (see next page) vs. **functional composition**

\[
\{\text{Mary} / \text{every girl}\} \text{ saw } \quad \text{and } \quad \{\text{Bill} / \text{every boy}\} \text{ heard } \quad \text{the famous soprano.}
\]

\[
\begin{align*}
S/\pi(S/\pi\text{NP}) & \quad (S/\pi\text{NP})/\pi\text{NP} \\
\lambda x[\text{saw}(x)(\text{mary})] & \quad \lambda x\forall y[\text{saw}(x)(y)] \\
\end{align*}
\]

---

**Predicate modification**

\[
\begin{align*}
\lambda z[\text{soprano}(z)] \quad \lambda \text{P}[\text{P}] \\
\lambda x[\text{saw}(x)(\text{mary})] \\
\lambda \text{P}[\text{P}(\lambda x[\text{saw}(x)(\text{mary})])] = \lambda x[\text{saw}(x)(\text{mary})] \\
\lambda \text{v}[\text{soprano}(v) \& \text{saw}(v)(\text{mary})] \\
\lambda \text{P}[\text{iota } x[\text{P}(x)]](\lambda \text{v}[\text{soprano}(v) \& \text{saw}(v)(\text{mary})]) = \\
\text{iota } x[\text{soprano}(x) \& \text{saw}(x)(\text{mary})] \\
\end{align*}
\]

**Restrictive:** the soprano which Mary saw  
**Non-restrictive:** the soprano, which Mary saw

* every soprano which Mary saw

**Does the semantics of questions raise new issues as compared to relative clauses?**

**Question--answer pairs**

**Conjunctions of relatives clauses and questions**

a soprano who sings and who dances  
Who sings and who dances?

a soprano who sings and dances  
Who sings and dances?
Movement transformations

The dog will drink water.
Will the dog drink water?

You know (that) Mary will read this book.
You know which book Mary will read.
Mary will read this book.
Which book will Mary read?

She will never do such a thing.
Never will she do such a thing.

Which book will Mary read?

Our system of rules has to enable which book to occur in the “first position” and will in the “second position”, AND prevent them from also occurring in their traditional positions, in the same sentence.

* Which book will Mary will read which book?

Add movement transformations to the set of structure-building (= P5 or merge) rules.

Movement = copying + deletion

Complement questions and relative clauses

I know which book Mary will read
know
which book
CP
C will
NP
Mary
Aux
will
V
read
NP
which book

I read every book which/that/Ø Mary will read
book
which
CP
C will
NP
which book

Syntactic details that we will not attend to

Overtly fronted wh-phrases vs wh-in-situ
I know which book you gave to which student

Languages with or without overt wh-fronting
Languages with or without multiple wh-fronting
Languages with or without overt C in compl. questions

Which vs. that vs. Ø in English relative clauses
Differences btw interrogative and relative pronouns
Head-internal relative clauses, correlatives, etc.
Different analyses of relative clauses (e.g. Kayne 1994)
Theories of questions in a nutshell

Hamblin 1973 (main-clause): the set of propositions that are possible answers

Who smokes? \{ \text{smoke}(w)(x) : x \in D_e \} = \lambda p \exists x \in D_e [p = \lambda w.\text{smoke}(w)(x)]

Karttunen 1977 (complement): the set of propositions that are true answers

[Mary knows/wonders] who smokes \lambda p \exists x \in D_e [p(w*) & p = \lambda w.\text{smoke}(w)(x)]

\[
\begin{array}{c}
\text{CP} \\
\lambda P \exists x [\text{hum}(x) & P(x)] (\lambda y [\lambda p[p(w*) & p = \lambda w.\text{smoke}(w)(y))])
\end{array}
\]

C

\[
\begin{array}{c}
\lambda P \exists x [\text{hum}(x) & P(x)] \\
\lambda P[p(w*) & p = \lambda w.\text{smoke}(w)(x)]
\end{array}
\]

C

\[
\begin{array}{c}
\lambda q \lambda p[p(w*) & p = q] \\
\lambda w.\text{smoke}(w)(y)
\end{array}
\]

Who smokes? \lambda x [\text{smoke}(w)(x)]

An elliptical answer has a context-variable whose type matches the question: \Gamma(John).

Groenendijk & Stokhof 1984 (main, complement): a partition of the set of worlds

[[Who smokes]] \lambda w' \lambda w [ \lambda x [\text{smoke}(w)(x)] = \lambda x [\text{smoke}(w')(x)] ]

- all the worlds in which just Mary smokes
- all the worlds in which just Bill smokes
- all the worlds in which both Mary and Bill smoke
- all the worlds in which no one smokes

The wh-complement of wonder is the question itself. The wh-complement of know is the proposition that is a complete and exhaustive answer in w*. It is equivalent to a that-complement.

[[Who smokes]](w*) \lambda w [ \lambda x [\text{smoke}(w)(x)] = \lambda x [\text{smoke}(w*)(x)] ]

If just Mary smokes in w*, then [[Who smokes]](w*) \lambda w [ \forall x [\text{smoke}(w)(x) \leftrightarrow x = m] ]

Inquisitive Semantics (Ciardelli, Groenendijk, Mascarenhas, Roelofsen 2008--): a cover of the set of ws

\[
\begin{array}{c|c|c}
\text{Bill and Mary} & \text{Mary} & \lambda x [\text{smoke}(w)(x)] = \lambda x [\text{smoke}(w')(x)] \\
\text{Bill} & \text{no one} & \lambda w' \lambda w [ \lambda x [\text{smoke}(w)(x)] = \lambda x [\text{smoke}(w')(x)] ]
\end{array}
\]

\[
\begin{array}{c|c|c}
\text{Bill and Mary} & \text{Mary} & [[\text{Who smokes}]] = \lambda w' \lambda w [ \lambda x [\text{smoke}(w)(x)] = \lambda x [\text{smoke}(w')(x)] ] \\
\text{Bill} & \text{no one} & [[\text{Someone smokes}]] = \lambda w' \lambda w [ \lambda x [\text{smoke}(w)(x)] = \lambda x [\text{smoke}(w')(x)] ]
\end{array}
\]
In reading the paper, note the conventions used by Karttunen (from Montague's PTQ):

- Half-circle above variable $\tilde{x} = \lambda x$
- Circumflex above variable $\hat{x} = ^\forall x$

$P(x) = ^\forall P(x)$, where $^\forall A$ is well-formed if $A$ denotes an intension. Then $^\forall A$ is the value of $A$ in the real world, i.e., the extension of $A$ in the real world. ($^\forall$ = extension sign)

For the time being, ignore intensions and extensions. Also ignore subscripted asterisks on verb interpretations; they indicate the extendedized versions of verbs.

- Big 'and' ($\land$) = $\forall$
- Big 'or' ($\lor$) = $\exists$

Why the 'and' and 'or' notation, common to mathematical logicians, at least at the time? Universal quantification is expressible as a conjunction of propositions over a finite universe. If the universe is $\{a, b, c\}$, then everyone talks = $a$ talks and $b$ talks and $c$ talks. Existential quantification is a disjunction of propositions over a finite universe: someone talks = $a$ talks or $b$ talks or $c$ talks.

Below the original notation is replaced by contemporary notation. The syntactification is from Higginbotham (1993) via Cresti (1993). See also Lahiri (2002:508).

These pages only cover parts of the paper, to aid your preparatory reading. We are going to talk over both the linguistic claims and the formalism carefully in class. The class discussion should be relaxed, but not rambling. Read the paper closely, more than once. If you get stuck, feel free to email.

Indirect questions denote the set of their true answers.
Almost the same as Hamblin, adding that $p$ is true: Who came? is $\lambda p \exists x[p \land p = ^\forall \text{came}(x)]$.
Assumes hidden performatives ([I want to know, etc.] in matrix questions.

Proto-question rule:
If $\alpha$ is a sentence (= IP) interpreted as $\alpha$, then the proto-question (= CP), interpreted as $\lambda p [p \land p = ^\forall \alpha]$.

Yes-no question rule:
If $\alpha$ is a proto-question (= CP) interpreted as $\beta$, whether $\alpha$ is a question (= CP), interpreted as $\lambda p [\beta(p) \lor \exists q [\beta(q) \land p = ^\forall \exists q [\beta(q)]]]$.

Wh-phrase rule:
If $n$ is a noun, which $n$ is a wh-phrase, interpreted as $\lambda p \exists x[N(x) \land P(x)]$.

Wh-quantification rule (skeleton):
If $d$ is a wh-phrase interpreted as $\delta$, and $\epsilon$ a proto-question (= CP) containing a designated pronoun (= variable), interpreted as $\beta$, then replacing $\epsilon$ by $\beta$ and deleting the pronoun we get a question (= CP) that is interpreted as $\lambda p [\delta(\lambda x [\beta(p)])]$. [d interpreted as delta]
The $\exists$ above does not mean that the question somehow asserts the existence of entities with the property $N$. The idea is that *Who walks?* denotes the set of all propositions of the form *So-and-so walks*, where *So-and-so* is the name of a human. The $\exists$ ensures that for any person there is, one of the propositions mentions him/her as a walker.

If the proto-question did not contain a designated pronoun (empty category interpreted as a variable), adding Spec,CP would amount to vacuous quantification: *Which man Mary is here.*

If wh-quantification applied to *whether a, instead of ?a, John knows who came* would require John to know about everyone whether or not (s)he came. This would require that John have direct knowledge about all inhabitants of the universe.

A more realistic requirement is that if John knows who came, (i) he should know about everyone who came that he/she came, or (ii) he should know about everyone who came that he/she came and that no one else came. -- Which of the two do you prefer? Which of the two does Karttunen's grammar yield?
Two sample derivations:

*Mary cooks* is a proto-question (=C': \( \lambda p[p & p=\text{cook}(m)] \))

whether *Mary cooks* is interpreted as

\[ \lambda p(\lambda p'[p & p=\text{cook}(m)])(p) \land \neg \exists q(\lambda p'[p & p=\text{cook}(m)])(q) \land p=\neg \exists q(\lambda p'[p & p=\text{cook}(m)])(q)] \]

\[ \lambda p([p & p=\text{cook}(m)] \land \neg \exists q([t'[q & q=\text{cook}(m)] \land p=\neg \exists q(t'[q & q=\text{cook}(m)])]) \]

This is equivalent to \( ^\text{\text{\text{-}}} \text{cook}(m) \) or \( ^\text{\text{\text{-}}} \text{cook}(m) \), whichever is true.

*which man* is a wh-phrase: \( \lambda P \exists x[\text{man}(x) \land P(x)] \).

*Which man came* is derived from *which man* and *HE came*, by replacing *HE* with *which man* and deleting *HE*.

*HE came* is interpreted as \( \lambda p[p & p=\text{came}(y)] \).

*Which man came* is interpreted as

\[ \lambda p(\lambda P \exists x[\text{man}(x) \land P(x)])(\lambda y(\lambda p'[p & p=\text{came}(y)])(p)]) \]

\[ \lambda P \exists x[\text{man}(x) \land \neg (p \land p=\text{came}(x))] \]

Read 2.8 through 2.11 lightly,
but take 2.12 (wh/quantifier scope ambiguities) seriously,
and go light on the rest.
A question is interpreted as a set of [strictly linguistic] answers.
Main-clause questions and interrogative complement clauses (sloppily, complement questions) are assumed to be alike.

Hamblin 1973 (main-clause): the set of propositions that are possible answers

Who smokes? \{\text{smoke}(w)(x) : x \in \text{De}\}

Karttunen 1977 (complement): the set of propositions that are true answers

[\text{Mary knows/wonders} who smokes \{p : \exists x \in \text{De}. \ p=\{w: \text{smoke}(w)(x)\} \& p(w^*)\}]

Hausser 1978 (main-clause): the set of constituent answer denotations

Who smokes? \{x: \text{smoke}(w^*)(x)\}

Elliptical answer has a context-variable whose type matches the question: \(\Gamma(\text{John})\).

Szabolcsi 1981 (main-clause): on topic and focus, not especially questions. Adopts Hausser, but caters to different kinds of elliptical answers.

Who smokes? + JOHN. = JOHN smokes. FOC(John)(\text{smoke})
Who smokes? + John/\ DOES. = John/\ smokes. ContrTop(John)(\text{smoke})
Who smokes? + Most men. = Most men smoke. most\_men(\text{smoke})

Szabolcsi 1981, 1983 on exhaustive focus in Hungarian:

If \(\Omega\) denotes a GQ and P, R properties, then (omitting intension/extension signs),

\[\text{FOC}(\Omega)(P) = \forall z [ (P(z) \land \Omega(P)) \leftrightarrow \forall R[\Omega (\lambda x [P(x) \land R(x)]) \rightarrow R(z)] ]\]

A MAN smokes ‘For any individual z, z smokes when a man smokes iff z has every property R that a man who smokes has’

Ensures that just one man smokes and that no one else does. If Tim is a man who smokes, take R to be \(\{x: x=t\}\). (Note \(\text{FOC}[\text{only}]\); not presuppositional, not mirative.)

All of the above assume that questions have uniformly undemanding interpretations.
Actual answers however introduce differing degrees of completeness or exhaustivity.
Groenendijk & Stokhof 1982, 1984, 1989 criticize Hamblin, Karttunen, and Hausser on account of problems with (i) de dicto readings, (ii) conjunction of questions and (iii) quantification into questions.

G&S treat elliptical answers as Szabolcsi 1981,1983; their EXH = FOC:

If $\Omega$ denotes a GQ, P, R properties, and $a$ is the actual world, then

$$\text{EXH}(\Omega)(P) = \Omega(a)(P) \land \neg \exists R[\Omega(a)(R) \land P(a) \neq \neg R(a) \land \forall x[R(a)(x) \rightarrow P(a)(x)]]$$

G&S criticize Karttunen for not getting the right entailments under know, and for not distinguishing wonder from know.

If just Mary smokes, and John knows who smokes, then

John knows that just Mary smokes. (No such entailment with wonder)

The wh/that complement of know is the proposition that truthfully, completely, and exhaustively answers the question. $[[\text{who smokes}]] = [[\text{EXH(so-and-so smokes}]]$.

The complement of wonder is the question itself, the intension of that proposition.

$$[[\text{Who smokes}]](a) = \lambda i[ \lambda x[\text{smoke}(i)(x)] = \lambda x[\text{smoke}(a)(x)] ]$$

$$[[\text{Who smokes}]] = \lambda j \lambda i[ \lambda x[\text{smoke}(i)(x)] = \lambda x[\text{smoke}(j)(x)] ]$$

$[[\text{Who smokes}]]$ denotes a partition of the set of worlds (=the logical space):

- worlds in which just Mary smokes
- worlds in which just Bill smokes
- worlds in which both Mary and Bill smoke
- worlds in which no one smokes

John knows who smokes iff he knows in which cell the actual world $a$ is.

John wonders who smokes iff he wonders (about) the partition.

As a main-clause question, Who smokes? semantically speaking requests an exhaustive answer, but pragmatically speaking it accepts partial and mention-some answers too.

Heim 1994: G&S’s theory works well for know. But not all verbs are like know (or wonder). Karttunen’s theory is richer (=more flexible). It allows to define different answer relations, and different embedding verbs can make different demands.

$$\text{ANS-1} (Q, w) = \bigcap [[Q]]_{\text{Karttunen}}(w)$$

not definable in G&S

$$\text{ANS-2} (Q, w) = \lambda w' [\text{ANS-1}(Q, w') = \text{ANS-1}(Q, w)]$$

equivalent to G&S

Know favors ANS-2, but tell, divulge, and perhaps be surprised favor ANS-1.
This is the state of the art until fairly recently. Strongly exhaustive answers joined the race to feature in the one and only correct interpretation of questions. Different embedding verbs aren’t investigated much. Heim 1994 is little known.

But EXH comes in through the back door: strengthened meanings in the calculation of scalar implicatures. E.g. John or Bill smokes implicates ‘not both’.

Neo-Gricean, globalist view (Horn, Kadmon, Sauerland, Russell, Geurts):
post-compositional strengthening

Grammatical, localist view (Landman, Chierchia, Fox, C-F-Spector):
strengthening is part of the recursive semantics

Rooth’s focus alternatives and only: Mary only [vp introduced BILL to Sue]. Focus alternatives C = {^intro Bill to Sue, ^intro Mary to Sue, ^intro Ken to Sue, ...} only(C)(VP)(mary) : ∀P[(P ∈ C & P(mary)) → P=VP]
`For every property P, where P comes from the set C and holds of Mary, P is [[VP]]’

Implicatures: [[O_ALT(S)]w = 1 iff [[S]]w and every true scalar alt. is entailed by [[S]]. Alternatives stronger than [[S]] are negated (to be refined with innocent exclusion).

Fox 2007, Spector 2009: possibly recursive exhaustification; multiple readings due to varying placement (see Szabolcsi 2010: 149-50 on focus sensitivity for O)

Exh(C’)[Exh(C)[You may eat the cake or the icecream]] free choice
where C={MAY(p v q)}, C’={Exh(C)(p): p ∈ C}

Exh(C’)[Exh(C)[There’s beer in the fridge or the ice-bucket]] beer at both places
where C={∃x.Px∧Qx}, C’={Exh(C)∃x.Px, ∃xh(C)∃x.Qx}

Every student has to solve O(two problems) ‘Every st has to solve exactly two’

Every student O(has to solve two problems) ‘Every st has to solve at least two, is allowed to solve more, and no st has to solve more than two’

O(Every student has to solve two problems) ‘Every st must solve at least two, and not every st has to solve more than two’

The two lines connect! An analysis like Heim’s can be produced by placing EXH into different positions within the sentence that contains the complement question.
Klinedinst & Rothschild 2011

Non-veridical versions: do not imply correctness: *John told me/predicted who sang.*

Examine veridical Vs: weak/intermed/strong exh: *John told me/predicted who sang.*

Fact: Frank and Emilio were the only people who sang.

(6) If John says, “Frank and Emilio sang,” then he has, in a sense, told me/predicted who sang, even if he doesn’t say that no one else sang.

(8) John told me/predicted who sang, but he didn’t tell me/predict who didn’t sing.

(9-10) John tells me/predicts that Frank, Emilio, and Ted sang. Here John told me/predicted the Karttunen-proposition, but it’s not clear/not true that he told me/predicted who sang.

(6)-(8): Karttunen (weak) suffices, G&S (strong) too demanding.

(9)-(10): Karttunen + no false proposition of the form *x sang* (=intermediate).

---

**Intermediate:**

```
  [EXH x VP]
  [V x VP]
  [Q V]
```

- *x predicted all p ∈ Q*
- *x didn’t predict any p, Q ⊈ p* (F+)
- *(not scalar impl, persists in ↓)*

**Strong, G&S:**

```
  [V x VP]
  [Q EXH]
```

- *x predicted [p ↔ p ∈ Q]*

---

Factive *know*: strong ok, intermediate=strong, so it is perceived as non-existing

\[
[[\text{EXH}(\text{J knows who sang})]] = [[\text{J knows who sang}]] \land (\cap p \in [\text{J knows who sang}]^{F+} \cdot \neg p),
\]

because anything stronger than K-answer is false, and you can’t *know* a falsehood.

Remaining issue: *foretell* and *deviner* are factual, but Spector 2006 says they have intermediate readings. That requires lexical decomposition of *foretell/deviner*.

(Szabolcsi 1997: *John discovered {which criminals sang/what every criminal wanted}* don’t entail that Joh discovered that those persons were criminals.)
Cremers & Chemla 2014: *know*-complements also have intermediate readings, and don’t differ much from *predict*-complements.

https://philpapers.org/rec/CREAPS

http://semanticsarchive.net/Archive/GRhZmM4N/Cremers-Chemla-ExpEmotiveFactuals.pdf

More readings than ever before!
Derived from weak, Hamblin/Karttunen-style Qs plus EXH operators on Q or higher.
Need to be correctly matched up with embedding Vs.


*  

1980s links
What is Dynamic Semantics? See Gamut Vol 2, Ch 7.4, Discourse Representation Theory

What is variable-free semantics?

Combinatory logic (Curry and Feys 1958) is a “variable-free” alternative to the lambda calculus. The two have the same expressive power but build their expressions differently. “Variable-free” semantics is, more precisely, “free of variable binding”: it has no operation like abstraction that turns a free variable into a bound one; it uses combinators—operations on functions—instead. For the general linguistic motivation of this approach, see the works of Steedman, Szabolcsi, and Jacobson, among others.

Do these two combine? See Szabolcsi 2003, Binding on the fly: cross-sentential anaphora in variable-free semantics. In Resources.