Semantic Typology and Composition

Paul M. Pietroski, University of Maryland
Dept. of Linguistics, Dept. of Philosophy
Human children naturally acquire languages that somehow generate boundlessly many expressions that connect “meanings” with “pronunciations” in accord with certain constraints.

Bingley is eager to please.

(a) Bingley is eager to please relevant parties.
(b) Bingley is eager to be pleased by relevant parties.

Bingley is easy to please.

(a) Bingley can easily please relevant parties.
(b) Bingley can easily be pleased by relevant parties.
Human children naturally acquire languages that somehow generate boundlessly many expressions that connect “meanings” with “pronunciations” in accord with certain constraints.

The senator called the donor from Texas.

(a) The senator called the donor, and the donor was from Texas.

(b) The senator called the donor, and the call was from Texas.

#c The senator called the donor, and the senator was from Texas.
Human Languages

• acquirable by normal children
given ordinary courses of experience

• somehow generate unboundedly many expressions that
  connect “meanings” with “pronunciations”

• what \textit{are} these meanings?
Human Languages

• acquirable by normal children
given ordinary courses of experience

• somehow generate unboundedly many expressions that
  connect “meanings” with “pronunciations”

• what *types* of meaning do human linguistic expressions exhibit?

  (1) Fido  (5) every cat
  (2) chase  (6) chase every cat
  (3) every  (7) Fido chase every cat
  (4) cat    (8) Fido chased every cat.
Human Languages

• acquirable by normal children given ordinary courses of experience

• somehow generate unboundedly many expressions that connect “meanings” with “pronunciations”

• what are the Human Meaning Types?

• standard answer, via Frege’s conception of ideal languages
  
  (i) a basic type <e>, for entity denoters
  
  (ii) a basic type <t>, for truth-value denoters
  
  (iii) if <α> and <β> are types, then so is <α, β>

Fido

Fido chased every cat.
since \( <e> \) and \( <t> \) are types, meanings of type \( <e, t> \) can be “abstracted” from the meanings of sentences (type \( <t> \)) that contain a name (type \( <e> \))

since \( <e> \) and \( <e, t> \) are types, meanings of type \( <e, e, t> \) can be “abstracted” from the meanings of sentences that contain two names
“composition” (reverse abstraction) via Function Application

\[
\begin{align*}
&\text{FELIX}^{\text{e}} \quad \text{IS-A-CAT}(\_)^{\text{e, t}} \\
&\text{FIDO}^{\text{e}} \quad \text{CHASED}(\_ , \_)^{\text{e, e, t}} \\
&\text{RAN}(\_)^{\text{e, t}} \\
&\text{EVERY}(\_ , \_)^{\text{et, et}} \\
&\text{CAT}(\_)^{\text{et, t}}
\end{align*}
\]

...and so on
Human Languages

• acquirable by normal children
given ordinary courses of experience

• somehow generate unboundedly many expressions that
connect “meanings” with “pronunciations”

• what are the Human Meaning Types?

• standard answer, via Frege’s conception of ideal languages

(i) a basic type <e>, for entity denoters
(ii) a basic type <t>, for truth-value denoters
(iii) if <α> and <β> are types, then so is <α, β>

That’s a lot of types
if \( \langle \alpha \rangle \) and \( \langle \beta \rangle \), then \( \langle \alpha, \beta \rangle \)

<table>
<thead>
<tr>
<th>Level</th>
<th>( \langle e \rangle )</th>
<th>( \langle t \rangle )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.</td>
<td>( \langle e \rangle )</td>
<td>( \langle t \rangle )</td>
</tr>
<tr>
<td>1.</td>
<td>( \langle e, e \rangle )</td>
<td>( \langle e, t \rangle )</td>
</tr>
<tr>
<td>2.</td>
<td>eight of ( \langle 0, 1 \rangle )</td>
<td>eight of ( \langle 1, 0 \rangle )</td>
</tr>
<tr>
<td></td>
<td>sixteen of ( \langle 1, 1 \rangle )</td>
<td></td>
</tr>
<tr>
<td>3.</td>
<td>64 of ( \langle 0, 2 \rangle )</td>
<td>64 of ( \langle 2, 0 \rangle )</td>
</tr>
<tr>
<td></td>
<td>128 of ( \langle 1, 2 \rangle )</td>
<td>128 of ( \langle 2, 1 \rangle )</td>
</tr>
<tr>
<td></td>
<td>1024 of ( \langle 2, 2 \rangle )</td>
<td></td>
</tr>
<tr>
<td>4.</td>
<td>2816 of ( \langle 0, 3 \rangle )</td>
<td>2816 of ( \langle 3, 0 \rangle )</td>
</tr>
<tr>
<td></td>
<td>5632 of ( \langle 1, 3 \rangle )</td>
<td>5632 of ( \langle 1, 3 \rangle )</td>
</tr>
<tr>
<td></td>
<td>45,056 of ( \langle 2, 3 \rangle )</td>
<td>45,056 of ( \langle 3, 2 \rangle )</td>
</tr>
<tr>
<td></td>
<td>1,982,464 of ( \langle 3, 3 \rangle )</td>
<td></td>
</tr>
</tbody>
</table>

at Level 5,
more than \( 5 \times 10^{12} \)
Human Languages

• acquirable by normal children given ordinary courses of experience

• somehow generate unboundedly many expressions that connect “meanings” with “pronunciations”

• what are the Human Meaning Types?

• standard answer, via Frege’s conception of ideal languages
  (i) a basic type <e>, for entity denoters
  (ii) a basic type <t>, for truth-value denoters
  (iii) if <α> and <β> are types, then so is <α, β>  That’s a lot of types
Human Languages

• acquirable by normal children given ordinary courses of experience

• somehow generate unboundedly many expressions that connect “meanings” with “pronunciations”

• what are the Human Meaning Types?

• another answer
  (i) a type <M>, for monadic predicates horse, brown, run
  (ii) a type <D>, for dyadic predicates on, from, cause
  (iii) all complex expressions are of type <M>
Outline for Rest of the Talk

• say a little more about the suggested alternative to a “Fregean” semantic typology

• then focus on why we need an alternative
  – if $\alpha$ and $\beta$ are types, then so is $\langle \alpha, \beta \rangle$ (way too many)
  – a basic type $\langle e \rangle$, for entity denoters of (unattested)
  – a basic type $\langle t \rangle$, for truth-value denoters (unattested)

• if time permits, brief discussion of quantification
  – serious difficulties for the idea that ‘every cat’ is of type $\langle e_t, t \rangle$
    and ‘every’ is of type $\langle e_t, \langle e_t, t \rangle \rangle$
  – alternatives available if we don’t insist that $\langle e \rangle$ and $\langle t \rangle$ are basic
Outline for Rest of the Talk

• say a little more about the suggested alternative to a “Fregean” semantic typology

• then focus on why we need an alternative
  – if $<\alpha>$ and $<\beta>$ are types, then so is $<\alpha, \beta>$ (way too many)
  – a basic type $<e>$, for entity denoters of (unattested)
  – a basic type $<t>$, for truth-value denoters (unattested)

• if time permits, brief discussion of quantification
  (otherwise, conversations about quantification later)
We can *invent* languages that have boundlessly many expressions that exhibit...

- *no* semantic typology (see Tarski)

- *finitely many* semantic types (2, 3, ..., a million, ...)

- *endlessly many* semantic types (see Frege and Church)

But where are *human* languages located along this dimension of potential variation?

In addressing the empirical questions, it helps to have some “typologically spare” languages as possible models.
We can imagine a language whose expressions are limited to...

(1) finitely many *atomic monadic* predicates: $M_1(\_) \ldots M_k(\_)$

(2) finitely many *atomic dyadic* predicates: $D_1(\_, \_) \ldots D_j(\_, \_)$

(3) boundlessly many *complex monadic* predicates

Monad + Monad $\Rightarrow$ Monad

$\text{BROWN}(\_) + \text{HORSE}(\_) \Rightarrow \text{BROWN}(\_)^{\text{HORSE}(\_)}$

$\text{FAST}(\_) + \text{BROWN}(\_)^{\text{HORSE}(\_)} \Rightarrow \text{FAST}(\_)^{\text{BROWN}(\_)^{\text{HORSE}(\_)}}$
We can imagine a language whose expressions are limited to...

(1) finitely many *atomic monadic* predicates: $M_1(\_)$ ... $M_k(\_)$

(2) finitely many *atomic dyadic* predicates: $D_1(\_, \_)$ ... $D_j(\_, \_)$

(3) boundlessly many *complex monadic* predicates

Monad + Monad $\Rightarrow$ Monad

for each entity:

$\Phi(\_)^\land \Psi(\_)$ applies to it

if and only if

$\Phi(\_)$ applies to it, *and*

$\Psi(\_)$ applies to it
We can imagine a language whose expressions are limited to...

(1) finitely many *atomic monadic* predicates: $M_1(\_)$ ... $M_k(\_)$

(2) finitely many *atomic dyadic* predicates: $D_1(\_,\_)$ ... $D_j(\_,\_)$

(3) boundlessly many *complex monadic* predicates

for each entity:

$\Phi(\_)^\Psi(\_)$ applies to it if and only if $\Phi(\_)$ applies to it, *and* $\Psi(\_)$ applies to it

$\exists [ON(\_,\_)^\text{HORSE}(\_)]$

(thing that is) on a horse
We can imagine a language whose expressions are limited to...

(1) finitely many *atomic monadic* predicates: \(M_1(\_), \ldots, M_k(\_)\)

(2) finitely many *atomic dyadic* predicates: \(D_1(\_, \_), \ldots, D_j(\_, \_)\)

(3) boundlessly many *complex monadic* predicates

for each entity:

\[\Phi(\_)^\wedge \Psi(\_) \text{ applies to it if and only if } \Phi(\_) \text{ applies to it, and } \Psi(\_) \text{ applies to it}\]

\[\exists [ON(\_, \_)^\wedge HORSE(\_)]\]

(thing that is) on a horse # thing that a horse is on
We can imagine a language whose expressions are limited to...

1. finitely many **atomic monadic** predicates: $M_1(_) \ldots M_k(_)$

2. finitely many **atomic dyadic** predicates: $D_1(_, _) \ldots D_j(_, _)$

3. boundlessly many **complex monadic** predicates

for each entity:

- $\Phi(_) \wedge \Psi(_) \text{ applies to it if and only if } \Phi(_) \text{ applies to it, and } \Psi(_) \text{ applies to it}$

for each entity:

- $\exists [\Delta(_, _) \wedge \Psi(_)] \text{ applies to it if and only if it bears } \Delta \text{ to } \textit{something} \text{ that } \Psi(_) \text{ applies to}$
\[
\begin{align*}
\text{FAST}(\_)^{\text{\textregistered}} \text{BROWN}(\_)^{\text{\textregistered}} \text{HORSE}(\_)
\quad \exists[\text{ON}(\_, \_)^{\text{\textregistered}} \text{HORSE}(\_)]

&\textit{is like} \\
\text{FAST}(e) \& \text{BROWN}(e) \& \text{HORSE}(e)
\quad \exists[e [\text{ON}(e', e) \& \text{HORSE}(e)]]
\end{align*}
\]

But ‘\&’ and ‘\(\exists e[\ldots e\ldots]\)’ allow for \textit{much more}.

\[
\begin{align*}
\text{FAST}(e) \& \text{BROWN}(e') \& \text{HORSE}(e'')

&\exists[e [\text{ON}(e, e') \& \text{HORSE}(e)]] \\
&\exists[e [\text{BETWEEN}(e', e, e'') \& \text{SOLD}(e''', e''', e'''''', e)]]
\end{align*}
\]

And human languages are not \textit{this} permissive.
\[\exists [\text{AGENT}(\_ , \_)^\text{HORSE}(\_) ]^\text{EAT}(\_)^\text{FAST}(\_)

is like
\exists e [\text{AGENT}(e', e) & \text{HORSE}(e)] & \text{EAT}(e') & \text{FAST}(e')\]

\[\exists [\text{AGENT}(\_ , \_)^\text{EAT}(\_)^\text{FAST}(\_)^\text{HORSE}(\_) ]^\text{EAT}(\_)

is like
\exists e [\text{AGENT}(e', e) & \text{FAST}(e) & \text{HORSE}(e)] & \text{EAT}(e')]\]

We don’t need variables to capture the meanings of ‘horse eat fast’ and ‘fast horse eat’.
\[ \text{SEE}(\_)^{\exists}[\text{THEME}(\_, \_)^{\exists}\text{HORSE}(\_)] \]

is like

\[ \text{SEE}(e') \land \exists e[\text{THEME}(e', e) \land \text{HORSE}(e)] \]

\[ \text{SEE}(\_)^{\exists}[\text{THEME}(\_, \_)^{\exists}\exists[\text{AGENT}(\_, \_)^{\exists}\text{HORSE}(\_)^{\exists}\text{EAT}(\_)]] \]

is like

\[ \text{SEE}(e'') \land \exists e'[\text{THEME}(e'', e') \land \exists e[\text{AGENT}(e', e)^{\exists}\text{HORSE}(e) \land \text{EAT}(e)]] \]

We don’t need variables to capture the meanings of ‘see a horse’ and ‘see a horse eat’.
What are the Human Meaning Types?

--two basic types, <e> and <t>           --a monadic type <M>
--endlessly many derived types           --a dyadic type <D>, for finitely
  of the form <α, β>                    many atomic expressions
--<α> can combine with                   --<M> + <M> → <M>
  <α, β> to form <β>                    <M> + <D> → <M>
Outline for Rest of the Talk

✔ say a little more about the suggested alternative to a “Fregean” semantic typology

• then focus on why we need an alternative
  – if $<\alpha>$ and $<\beta>$ are types, then so is $<\alpha, \beta>$ (way too many)
  – a basic type $<e>$, for entity denoters of (unattested)
  – a basic type $<t>$, for truth-value denoters (unattested)

• if time permits, brief discussion of quantification
  (otherwise, conversations about quantification later)
if $\alpha$ and $\beta$, then $\alpha, \beta$

0. $\langle e \rangle \langle t \rangle$

1. $\langle e, e \rangle \langle e, t \rangle \langle t, e \rangle \langle t, t \rangle$

2. eight of $\langle 0, 1 \rangle$
   sixteen of $\langle 1, 1 \rangle$

3. 64 of $\langle 0, 2 \rangle$
   128 of $\langle 1, 2 \rangle$
   1024 of $\langle 2, 2 \rangle$

4. 2816 of $\langle 0, 3 \rangle$
   5632 of $\langle 1, 3 \rangle$
   45,056 of $\langle 2, 3 \rangle$
   1,982,464 of $\langle 3, 3 \rangle$

   (2) of $\langle 0, 0 \rangle$

   (4) of $\langle 0, 0 \rangle$

   (32), including
   $\langle e, et \rangle$ and $\langle et, t \rangle$

   (1408),
   including $\langle e, e, et \rangle$
   and $\langle et, et, t \rangle$

   (2,089,472), including
   $\langle e, e, e, et \rangle$
But even below Level Five...

\[
\lambda y.\lambda x.\text{Predecessor}(x, y) \quad \text{Level Two function of type } <e, et>
\]

\[
\lambda y.\lambda x.\text{Precedes}(x, y) \quad \text{another function of the same type}
\]

\[
\lambda R.\text{TRANSITIVE}[R] \quad \text{Level Three function of type } <<<e, et>, t>, \text{ with values } 3, 2, 0
\]

\[
\lambda y.\lambda x.\text{Precedes}(x, y) = \text{ANCESTRAL}[\lambda y.\lambda x.\text{Predecessor}(x, y)]
\]

\[
\lambda R.\text{ANCESTRAL}[R] \quad \text{Level Three function of type } <<<e, et>, <e, et>>
\]
But even below Level Five...

$$\lambda y.\lambda x.\text{Predecessor}(x, y)$$  Level Two function of type $<e, et>$

$$\lambda y.\lambda x.\text{Precedes}(x, y) = \text{ANCESTRAL}[\lambda y.\lambda x.\text{Predecessor}(x, y)]$$

$$\lambda R.\text{ANCESTRAL}[R]$$  Level Three function of type $<<e, et>, <e, et>>$

$$\text{ANCESTRAL-OF}[\lambda y.\lambda x.\text{Precedes}(x, y), \lambda y.\lambda x.\text{Predecessor}(x, y)]$$

$$\lambda R.\lambda R'.\text{ANCESTRAL-OF}[R', R]$$  Level Four function of type $<<e, et>, <<e, et>, t> \\
| | | \\
4 2 3$
Frege had to *invent* a language that supported abstraction on *relations*

The plate outweighs the knife.
The plate is something *which outweighs the knife*.
The knife is something *which the plate outweighs*.
*Outweighs is some relat* *which the plate the knife*.

Three precedes four.
Three is something *that precedes four*. λx. Precedes(x, 4)
Four is something *that three precedes*. λx. Precedes(3, x)
*Precedes is some relat* *that three four*. λR.R(3, 4)
<e>    <t>
if <α> and <β>, then <α, β>

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0.</td>
<td>&lt;e&gt;</td>
<td>&lt;t&gt;</td>
</tr>
<tr>
<td>1.</td>
<td>&lt;e, e&gt;</td>
<td>&lt;e, t&gt;</td>
</tr>
<tr>
<td>2.</td>
<td>&lt;e, &lt;e, e&gt;&gt;</td>
<td>...</td>
</tr>
<tr>
<td>3.</td>
<td>&lt;e, &lt;e, &lt;e, e&gt;&gt;&gt;</td>
<td>...</td>
</tr>
<tr>
<td>4.</td>
<td>&lt;e, &lt;e, &lt;e, &lt;e, e&gt;&gt;&gt;&gt;</td>
<td>...</td>
</tr>
</tbody>
</table>
Can Human Lexical Items have “Level Four Meanings”? 

whatever the **order** of arguments,  
the concept SOLD, which differs from GAVE,  
is plausibly (at least) **tetradic**
Can Human Lexical Items have “Level Four Meanings”? 

So why not…

\[ \lambda y. \lambda z. \lambda w. \lambda x. x \text{ sold } y \text{ to } z \text{ for } w \]
Can Human Lexical Items have “Level Four Meanings”? 

\[ \lambda Z. \lambda Y. \lambda X. \text{GLONK}(X, Y, Z) \]
\[ \forall x [X(x) \lor Y(x) \lor Z(x)] \]
\[ \exists x [X(x) \land Y(x)] \land \exists x [Y(x) \land Z(x)] \]

Glonk  cat  friendly  dog
<e>    <t>

if <α> and <β>, then <α, β>

0. <e>    <t>  (2)
1. <e, e> <e, t> <t, e> <t, t>  (4)
2.               (32), including <e, et> and <et, t>
3.               (1408), including <e, <e, et>> and <<e, et>, <e, et>>
4. maybe some kind of “resource limitation” keeps us from going beyond Level Three
    (2,089,472), including <e, <e, <e, et>>> and <<e, et>, <<e, et>, t> and <et, <et, <et, t>>>
Can Human Lexical Items have Level Three Meanings?

Diagram:

```
<e, t>
/     \
FIDO<e>  CHASED(_, _)<e, e, t>  GARFIELD<e>
/     \
<e, et>
/     \
ROMEO<e>  GAVE(_, _)<e, e, e, t>  GARFIELD<e>
```

```
<e, t>
/     \
<e, t>
/     \
JULIET<e>
```
but double-object *constructions* do not show that verbs can have Level Three Meanings

```
Romeo  gave  it  to  Juliet
```

```
Romeo  kicked  the rock  to  Juliet
Romeo  kicked  Juliet  the rock
```
a thief jimmied a lock with a knife
The concept JIMMIED is plausibly (at least) triadic. So why *isn’t* the verb of type $<e, <e, <et>>$?
Why not…

\[
\text{‘between’ } \rightarrow \lambda z. \lambda y. \lambda x. x \text{ is between } y \text{ and } z
\]
Still, one might think that many verbs do have Level Three Meanings...

\[
\begin{align*}
&<t> \\
&\quad -\text{ED}(\_)<_{et}, t> \\
&\quad \text{FIDO}_{e} \\
&\quad \text{BARK}(\_, \_)<_{e, et}> \\
&<_{et}> \\
&\quad \text{FIDO}_{e} \\
&\quad \text{CHASE}(\_, \_)<_{e, e, et}> \\
&\quad \text{GARFIELD}_{e}
\end{align*}
\]
Can Human Lexical Items have Level Three Meanings?

Saying that expressions of type \( <e, \_et> \) can be modified by expressions of type \( \_et \) is like positing a covert Level 4 element.

And why does the modifier skip over the thing chased, applying instead to the chase?
if the meaning of ‘chase’ is at Level Three, then a “passivizer” would also be at Level Four: 
\[\langle\langle e, e, \text{et}\rangle, \langle e, \text{et}\rangle\rangle\]

Kratzer and others “sever” agent-variables from verb meanings:

\[\langle e, \text{et}\rangle\]

Garfield was chased
\[\langle e, \langle e, \text{et}\rangle\rangle\]

\[\langle e, \text{et}\rangle\]

Garfield was chased
\[\langle e, \langle e, \text{et}\rangle\rangle\]

\[\langle e\rangle\]

\[\langle e, \text{et}\rangle\]

\[\lambda y. \lambda e. e \text{ is a chase of } y\]
But if the posited verb meaning is below Level Three, do we really need the covert Level Three element?
Are names really expressions of type <e>?
CHASE(_ , _) INTO-A-BARN

FIDO AGENT

CHASE(_ , _) GARFIELD
Do Human i-Languages have expressions of type <e>?

Initially tempting hypothesis: proper names are of type <e>

Robin_{e} \rightarrow \lambda x . x = \mathcal{R} \\
\lambda x . \text{Robinizes}(x)

But alternatives have been considered.

Robin_{et} \rightarrow \lambda P : \mathcal{R} (\lambda x . \text{Robinizes}(x)).P(x)

\[\text{Indexes}(1, x) \land \text{Called}(x, \text{‘Robin’)}\]
Do Human i-Languages have expressions of type \(<e>\)?

Initially tempting hypothesis: proper names are of type \(<e>\)

\[
\begin{aligned}
\text{Robin}_{<e>} & \rightarrow \quad \circ \quad -- | -- \\
& \quad / \quad \backslash
\end{aligned}
\]

But alternatives have been considered for good reasons.

- Neptune is a planet, but Vulcan isn’t.
- Every Tyler at the party was tall, and every philosopher was a Tyler. That Tyler stayed late, and so did this one. There were three Tylers at the party, and Tylers are clever.
- We sat next to (that nice) Professor Tyler Burge.
Do Human i-Languages have expressions of type $<e>$?

Another tempting hypothesis: deictic pronouns are of type $<e>$

\[
\begin{align*}
\text{that/she/him}_{<e>} \rightarrow & \quad \begin{array}{c}
\text{\textglue} \\
\text{\textglue}
\end{array} \\
\begin{array}{c}
\text{\textglue} \\
\text{\textglue}
\end{array}
\end{align*}
\]

But we need alternatives.

\[
\begin{align*}
\text{That planet is bright.} & \quad \begin{array}{c}
\text{\textglue} \\
\text{\textglue}
\end{array} \\
\begin{array}{c}
\text{\textglue} \\
\text{\textglue}
\end{array}
\end{align*}
\]

\[
\begin{align*}
\text{This}_1 \text{ trumps that}_2. & \quad \begin{array}{c}
<e> \\
<et>
\end{array} \quad \begin{array}{c}
<et> \\
<et>
\end{array} \quad \begin{array}{c}
<et> \\
<et>
\end{array}
\end{align*}
\]

\[
\begin{align*}
\text{That planet} & \quad \begin{array}{c}
<et> \\
<et>
\end{array} \quad \begin{array}{c}
<et> \\
<et>
\end{array} \quad \begin{array}{c}
<et> \\
<et>
\end{array}
\end{align*}
\]

\[
\begin{align*}
\text{planet/woman/…} & \quad \begin{array}{c}
<et> \\
<et>
\end{array} \quad \begin{array}{c}
<et> \\
<et>
\end{array}
\end{align*}
\]

\[
\begin{align*}
\text{This}_1 & \quad \begin{array}{c}
<et> \\
<et>
\end{array} \quad \begin{array}{c}
<et> \\
<et>
\end{array}
\end{align*}
\]

\[
\begin{align*}
\text{that}_2 & \quad \begin{array}{c}
<et> \\
<et>
\end{array} \quad \begin{array}{c}
<et> \\
<et>
\end{array}
\end{align*}
\]

\[
\begin{align*}
\text{This}_1 & \quad \begin{array}{c}
<et> \\
<et>
\end{array} \quad \begin{array}{c}
<et> \\
<et>
\end{array}
\end{align*}
\]
Do Human i-Languages have expressions of type $<t>$?

$S \Rightarrow NP \; aux \; VP$

Why think **tensed** phrases denote truth values?

$$\begin{array}{c}
\text{T(P)} \\
\downarrow \\
\text{T} \\
\downarrow \\
\text{past} \\
\downarrow \\
\text{D(P)}
\end{array}$$

$\Rightarrow \; \lambda e . e$ is (tenselessly) a John-see-Mary event

$\begin{array}{c}
\text{V(P)} \\
\downarrow \\
\text{V} \\
\downarrow \\
\text{see} \\
\downarrow \\
\text{D(P)}
\end{array}$

Why think the **tense** morpheme

is of type $<et, t>$

$\lambda E . \exists e [\text{Past}(e) \; \& \; E(e)]$

as opposed to $<et>$

$\lambda e . \text{Past}(e)$
Do Human i-Languages have expressions of type <t>?

Why think the *tense* morpheme is of type <et, t>?

\[
\begin{array}{c}
T(P) \\
/ \ \\
T \ V(P) \\
past \\
/ \ \\
D(P) \ V(P) \\
John \\
/ \ \\
V \ D(P) \\
see \ Mary
\end{array}
\]

\[
\lambda e \ . \ e \text{ is (tenselessly) a John-see-Mary event}
\]

\[
\exists e [\text{Past}(e) \ & \ E(e)]
\]

...that is also a conjunctive adjunct to V?
Do Human i-Languages have expressions of type $<t>$?

?? $\Rightarrow$ Longer Story$^1$

\[
\begin{array}{c}
? \\
? \\
? \\
\end{array}
\]

\[
\begin{array}{c}
T(P) \Rightarrow \lambda e. \text{Past}(e) \land e \text{ is a John-see-Mary event} \\
\end{array}
\]

\[
\begin{array}{c}
past \\
\end{array}
\]

\[
\begin{array}{c}
T \\
V(P) \Rightarrow \lambda e. e \text{ is (tenselessly) a John-see-Mary event} \\
\end{array}
\]

\[
\begin{array}{c}
D(P) \\
V(P) \\
John \\
V \\
D(P) \\
see \\
Mary \\
\end{array}
\]

$\lambda e. \text{Past}(e)$

1. Everybody needs some version of the following Tarskian idea:
   a predicate that is satisfied by some but not all things can be
   “polarized” into a predicate that is satisfied by everything or nothing.
Do Human i-Languages have expressions of type <t>?

\[
\text{Pol}(P) \xrightarrow{+} + [\text{Past}(\_)^\text{John-see-Mary}(\_)] \quad \text{applies to e if and only if there was an event of John seeing Mary}
\]

\[
\begin{array}{c|c}
\text{T}(P) & \text{Past}(\_)^\text{John-see-Mary}(\_) \\
\text{V}(P) & \text{John-see-Mary}(\_) \\
\end{array}
\]

\[
\begin{array}{c|c|c}
\text{D}(P) & \text{V}(P) & \text{John} \\
\text{V} & \text{D}(P) & \text{see Mary} \\
\end{array}
\]

suppose that a monadic predicate \( M \) can be “polarized” into a monadic predicate \( +M \) that applies to e iff \( M \) applies to something

‘such that John saw Mary’ is sentential; but it is a sentential \textit{predicate}, not a truth-value denoter
Do Human i-Languages have expressions of type \(<t>\)?

\[
\text{Pol}(P) \Rightarrow \neg [\text{Past}(\_)^\wedge \text{John-see-Mary}(\_)] \quad \ldots \ldots \text{applies to } e \\
\text{past} / \backslash
\neg \quad \text{T}(P) \Rightarrow \text{Past}(\_)^\wedge \text{John-see-Mary}(\_) \quad \text{there was no event of John seeing Mary}
\]

\[
\text{suppose that a monadic predicate } M \text{ can be “polarized” into a monadic predicate } +M \text{ that applies to } e \text{ iff } M \text{ applies to } \text{something} \\
\text{or a monadic predicate } -M \text{ that applies to } e \text{ iff } M \text{ applies to } \text{nothing}
\]
Outline for Rest of the Talk

✔ say a little more about the suggested alternative to a “Fregean” semantic typology

✔ then focus on why we need an alternative
  – if $\langle \alpha \rangle$ and $\langle \beta \rangle$ are types, then so is $\langle \alpha, \beta \rangle$ (way too many)
  – a basic type $\langle e \rangle$, for entity denoters of (unattested)
  – a basic type $\langle t \rangle$, for truth-value denoters (unattested)

• if time permits, brief discussion of quantification
  – serious difficulties for the idea that ‘every cat’ is of type $\langle et, t \rangle$
    and ‘every’ is of type $\langle et, \langle et, t \rangle \rangle$
  – alternatives available if we don’t insist that $\langle e \rangle$ and $\langle t \rangle$ are basic
...Carstairs-McCarthy argues that the apparently universal distinction in human languages between sentences and noun phrases cannot be assumed to be inevitable....His work suggests...that there is also no conceptual necessity for the distinction between basic types e and t.... If I am asked why we take e and t as the two basic semantic types, I am ready to acknowledge that it is in part because of tradition, and in part because doing so has worked well....

--Barbara Partee, “Do We Need Two Basic Types?”

Next Question: Why take <e> or <t> as basic types?

• little or no independent support for the claim that Human Languages (as opposed to certain languages of thought) generate denoters

• treating names and deictic pronouns as predicates isn’t hard

• Tarski showed us how to treat sentences as predicates, and

• the grammatical status of sentences is unclear
if $\alpha$ and $\beta$, then $\alpha, \beta$

0. $\langle e \rangle$ $\langle t \rangle$ 

1. $\langle e, e \rangle$ $\langle e, t \rangle$ $\langle t, e \rangle$ $\langle t, t \rangle$ 

2. eight of $\langle 0, 1 \rangle$ eight of $\langle 1, 0 \rangle$ 
   sixteen of $\langle 1, 1 \rangle$ 

3. 64 of $\langle 0, 2 \rangle$ 64 of $\langle 2, 0 \rangle$ 
   128 of $\langle 1, 2 \rangle$ 128 of $\langle 2, 1 \rangle$ 
   1024 of $\langle 2, 2 \rangle$ 

4. 2816 of $\langle 0, 3 \rangle$ 2816 of $\langle 3, 0 \rangle$ 
   5632 of $\langle 1, 3 \rangle$ 5632 of $\langle 1, 3 \rangle$ 
   45,056 of $\langle 2, 3 \rangle$ 45,056 of $\langle 3, 2 \rangle$ 
   1,982,464 of $\langle 3, 3 \rangle$ 

(2) 

(4) of $\langle 0, 0 \rangle$ 

(32), including $\langle e, et \rangle$ and $\langle et, t \rangle$ 

(1408), including $\langle e, e, et \rangle$ and $\langle et, et, t \rangle$ 

(2,089,472), including $\langle e, e, e, et \rangle$
1. <M>

2. <D>
Semantic Typology for Human I-Languages

THANKS!
Extra Slides Follow
Suppose that a monadic predicate $M$ can be “polarized” into

- a monadic predicate $\mathbf{+}M$ that applies to $e$
  
  if and only if $M$ applies to $\textit{something}$

\[
\text{BROWN}(\_)^\text{HORSE}(\_)
\]

applies to $e$ if and only if $e$ is both brown and a horse

\[
\mathbf{+[BROWN(\_)^{HORSE}(\_)]}
\]

applies to $e$ if and only if $\textit{something}$ is both brown and a horse
Suppose that a monadic predicate $M$ can be “polarized” into

- a monadic predicate $+M$ that applies to $e$ if and only if $M$ applies to \textit{something}
- a monadic predicate $-M$ that applies to $e$ if and only if $M$ applies to \textit{nothing}

$\text{BROWN(\_)}^{\text{HORSE(\_)}}$ applies to $e$ if and only if $e$ is both brown and a horse

$\neg[\text{BROWN(\_)}^{\text{HORSE(\_)}}]$ applies to $e$ if and only if \textit{nothing} is both brown and a horse
Suppose that a monadic predicate $M$ can be “polarized” into

- a monadic predicate $+M$ that applies to $e$ if and only if $M$ applies to \textit{something}
- a monadic predicate $-M$ that applies to $e$ if and only if $M$ applies to \textit{nothing}

$+[\text{BROWN}(\_)^\text{HORSE}(\_)]$ applies to $e$ if and only if (e is such that) there is a brown horse
\[\text{cp. } \exists x[\text{Brown}(x) \& \text{Horse}(x)]\]

$-[\text{BROWN}(\_)^\text{HORSE}(\_)]$ applies to $e$ if and only if (e is such that) there is no brown horse
\[\text{cp. } \neg \exists x[\text{Brown}(x) \& \text{Horse}(x)]\]
We can try assigning a higher type to ‘chased’.

But then (I claim): \textit{adjuncts} like ‘into the barn’ and \textit{passive} constructions will lead us to posit Level Four meanings.
But then how do we explain the *un*ambiguity of strings like

‘every cat which Fido chased’

#Every cat is one which Fido chased.
Maybe the external argument of a raised quantifier like ‘every cat’ is a polarized predicate, while a relative clause is a more ordinary ("unsentential") predicate that can apply to some things without applying to all things.
Technical Details Remain (see Conjoining Meanings)

But as a refinable first approximation, suppose that...

• ‘every’ is a *plural* Dyad:
  
  \[ \text{EVERY}( _, _) \text{ applies to } \langle \text{the Xs, the Ys} \rangle \text{ if and only if the Xs include the Ys} \]

• ‘every cat’ is a *plural* Monad:
  
  \[ \exists \left[ \text{EVERY}( _, _) \wedge \text{THE-CATS}( _) \right] \]
  
  applies to the Xs if and only if the Xs include *the cats*

• ‘Fido chased every cat’ is a *plural* Monad:
  
  \[ \exists \left[ \text{EVERY}( _, _) \wedge \text{THE-CATS}( _) \wedge \text{FIDO-CHASED-THEM}( _) \right] \]
  
  applies to the Xs if and only if the Xs include *the cats*, and *Fido chased* the Xs
Technical Details Remain (see *Conjoining Meanings*)

but relative to each assignment $A$ of values to variables...

---

applies to entity $e$ if and only if ($e$ is such that)

Fido chased whatever whatever $A$ assigns to $t_1$

so relative to any assignment that assigns

e to the variable,

the polarized predicate

applies to $e$ if and only if

Fido chased $e$
and it’s not (too) hard to formulate
Tarski-style composition principles according to which...

Every cat past Fido chase t₁

- applies to the Xs if and only if
  - (i) the Xs include the cats, and
  - (ii) Fido chased the Xs

(i.e., for each of the Xs, the polarized predicate applies to that entity when that entity is the value of the variable)

Technical Details Remain (see Conjoining Meanings)
Technical Details Remain (see *Conjoining Meanings*)

and it’s not (too) hard to formulate
*Tarski-style composition principles according to which...*

\[
\begin{align*}
&<+M> \Rightarrow \text{applies to } e \text{ if and only if } \\
&\quad \text{*Fido chased the cats*} \\
&<M> \Rightarrow \text{applies to the Xs if and only if} \\
&\quad (i) \text{ the Xs include *the cats*, and} \\
&\quad (ii) \text{*Fido chased* the Xs}
\end{align*}
\]
• function in intension
  a procedure that pairs inputs with outputs in a certain way

• function in extension
  a set of ordered pairs (with no \(<x, y>\) and \(<x, z>\) where \(y \neq z\))

\[ |x - 1| + \sqrt{(x^2 - 2x + 1)} \]
\[ \{..., (-2, 3), (-1, 2), (0, 1), (1, 0), (2, 1), ...\} \]
\[ \lambda x . |x - 1| \neq \lambda x . +\sqrt{(x^2 - 2x + 1)} \]
\[ \lambda x . |x - 1| = \lambda x . +\sqrt{(x^2 - 2x + 1)} \]
Extension[\(\lambda x . |x - 1|\)] = Extension[\(\lambda x . +\sqrt{(x^2 - 2x + 1)}\)]
i before e

• i-Language

a procedure that connects meanings with pronunciations in a certain way, thereby respecting certain constraints.

Bingley is eager to please.

(a) Bingley is eager to please relevant parties.

#(b) Bingley is eager to be pleased by relevant parties.

Bingley is easy to please.

#a) Bingley can easily please relevant parties.

(b) Bingley can easily be pleased by relevant parties.
• i-Language
  a *procedure* that connects meanings with pronunciations in a certain way, thereby respecting certain *constraints*

‘i’ connotes: **intensional**/procedural;
  **implementable**, and hence constrained;
  internal/individual, rather than external/public

• e-language
  a language in any other sense—e.g.,
  a suitable *set* of meaning-pronunciation pairs
i before e

- i-Language
  a *procedure* that connects meanings with pronunciations in a certain way, thereby respecting certain *constraints*

<table>
<thead>
<tr>
<th>Finite i-Languages</th>
<th>Human i-Languages</th>
<th>Less Permissive i-Languages</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gruesome i-Languages</td>
<td></td>
<td>More Permissive i-Languages</td>
</tr>
<tr>
<td></td>
<td>unbounded “sane” i-Languages</td>
<td></td>
</tr>
</tbody>
</table>
and without quantifier raising, severing external quantifiers still requires high types
every cat ran every dog chase every cat
if T is (semantically) the verb’s 3rd argument, then why not...

Tense may be needed (in matrix clauses). But does it do two semantic jobs: adding time information via the ‘e’-variable, like the adjunct ‘yesterday’; and closing the ‘e’ variable, as if tense is the 3rd argument of a verb that can’t take a 3rd argument?
\[ T(P) \rightarrow \exists e [PAST(e) \& John-\text{see}-Mary(e)] \]

\[ T \quad V(P) \rightarrow \lambda e . John-\text{see}-Mary(e) \]

\[ past \quad D(P) \quad V(P) \]

\[ John \quad D(P) \]

\[ V \quad see \quad Mary \]

\[ \langle\langle e, t>, t\rangle \quad \lambda E . \exists e [PAST(e) \& E(e)] \]

\[ (e < \text{RefTime}) \& (\text{RefTime} = \text{SpeechTime}) \]

“God likes Fregean Semantics” theory of tense
Fregean i-Languages: expressions of the types: $<e>$, $<t>$, and if $<\alpha>$ and $<\beta>$ are types, so is $<\alpha, \beta>$

Level-$n$ Fregean i-Languages: expressions of the types: $<e>$, $<t>$, and the nonbasic types up to Level $n$

Psuedo-Fregean Languages: expressions of the types: $<e>$, $<t>$, and some nonbasic types
A Pseudo-Fregean Language Might Have a Dozen Types

\[
\begin{align*}
<&e> & <t> \\
<&et> & <t, t> \\
<&e, et> & <t, <tt> & <et, t> \\
<&e, e, et> & <t, <t, tt>> & <e, et>, t> & <<et>, <et, t>> & <<e, et>, <et, t>>> \\
\end{align*}
\]
Mary saw John \\
and John saw Mary

Mary saw John \\
And
before John saw Mary
And