CHAPTER FIVE: INTENSIONAL TYPE LOGIC

5.1. Intensionality.

Up to now the logical languages we have considered are extensional: we saw that our type logical language satisfies the principle of extensionality: if we substitute in a complex expression φ, α for β, and α and β have the same extension, then φ and φ[β/α] have the same extension.

Frege argued that in natural language the principle of extensionality is not always valid: there are contexts - we call them intensional contexts - where substitution of expressions with the same extension does not preserve truth value.

(1) and (2) do not entail (3):
(1) Mary believes that the author of Ulysses is the author of Ulysses.
(2) The author of Ulysses is the author of Finnegans Wake.
(3) Mary believes that the author of Ulysses is the author of Finnegans Wake.

(4) and (5) do not entail (6):
(4) Mary seeks the author of Ulysses.
(5) The author of Ulysses is the author of Finnegans Wake.
(6) Mary seeks the author of Finnegans Wake.

When we say that (4) and (5) do not entail (6), we mean that there is a reading of (4) and a reading of (6) on which (4) and (5) do not entail (6). At the same time, we have to recognize that there is also a reading of (4) and of (6) on which the inference is perfectly valid. That is, (4) and (6) are ambiguous between a de dicto and a de re reading.

On the de re reading of (4), the interpretation of the description the author of Ulysses comes to the account of the speaker, that is, we interpret this expression as an object that is the author of Ulysses according to the speaker. This reading can be paraphrased by (7):

(7) There is a person, the author of Ulysses, and Mary seeks that person.

The inference from (7) and (5) to (8) is perfectly valid:

(7) There is a person, the author of Ulysses, and Mary seeks that person.
(8) There is a person, the author of Finnegans Wake, and Mary seeks that person.

On the de dicto reading of (4), the interpretation of the description the author of Ulysses comes to the account of Mary. That is, Mary does not necessarily stand in a relation to any actual object at all, or she may stand in relation to the wrong object, an object who she thinks is the author of Ulysses. What plays a role in the interpretation of (4) on this reading is not the person who is the author of Ulysses, but rather the role that that person plays, the role of being the author of Ulysses.

We can think of the difference in terms of when Mary would be satisfied to say that she found the author of Ulysses.

On the de re reading, she would be satisfied if she believed she had found James Joyce, who happens to be the author of Ulysses.
On the *de dicto* reading, she would be satisfied if believed she had found an object which she believed to fill the role of being the author of *Ulysses*. Note that in this formulation of the *de dicto* reading, the intensionality is moved to the expression ‘the role of being the author of *Ulysses*’. The point is: while the author of *Ulysses* happens to be James Joyce, the role of being the author of *Ulysses* is not identical to the role of being James Joyce. In reality these roles are filled by the same object, but that is not necessarily the case.

If the extension of the expression *the role of being the author of Ulysses* is the object that fills the role, we see that the extension of this expression varies across possible situations: in reality this role is filled by James Joyce, but we can think of situation where (God forbid) it is filled by Virginia Woolf instead, or for that matter, we can think of a situation where it is filled by Leopold Bloom.

We see then that *seek* cannot be regarded as operating on the extension of its complement, because on the *de dicto* reading, ***SEEK***(*THE AUTHOR OF ULYSSES*) involves the role of being the author of *Ulysses*, rather than the actual author of *Ulysses*.

Frege, of course, assumes that ***SEEK*** operates on the sense of the expression *the author of Ulysses* rather than its reference (extension).

We have argued that the extension of the expression *the role of being the author of Ulysses* varies across possible situations. The intension of expression *the role of being the author of Ulysses* is the pattern of variation of the extension of *the role of being the author of Ulysses* across possible situations. This is of course nothing but the function that maps every possible situation onto the extension of *the role of being the author of Ulysses* in that situation.

In general:

The intension of expression α is that function which maps every possible situation on the extension of α in that situation.

We call possible situations possible worlds (or worlds for short). In intensional logic, we deal with variation of extension of expressions across possible worlds; we associate with every expression an extension in a possible situation.

This means that we parameterize the notion of $[\alpha]_{M,g}$, the extension of $\alpha$ in $M$ relative to $g$ to possible worlds:

$[\alpha]_{M,w,g}$, the extension of $\alpha$ in $M$ in $w$ relative to $g$.

And we associate with every expression $\alpha$ an intension:

Let $W$ be the set of all possible worlds:

The intension of $\alpha$, $[\alpha]_{M,g}$ is given as follows:

$[\alpha]_{M,g} := \lambda v \in W: [\alpha]_{M,v,g}$

the function that maps every possible world $v$ onto the extension of $\alpha$ in $v$.

Intensionality comes in by assuming that the interpretation of certain expressions, like *believe* or *seek*, or modal operators, create intensional contexts: they are operations that do not operate on the extension of their complement, but that are sensitive to the pattern of
variation of the extension of their complement across possible worlds, i.e. they operate on the intension of their complement.

Thus the notion of intension can be seen as a formalization of Frege's notion of sense.

If we make that assumption - and we will for the purpose of this book - this will mean that in intensional contexts we cannot substitute expressions α and β with the same extension in the world of evaluation and expect the extension of the complex expression to be the same: this is because the intensional operator checks the extension of α and β in worlds other than the world of evaluation, and substitution can fail if the extension of α and β differs in other worlds.

We do make the prediction that if α and β not only have the same extension in the world of evaluation, but in fact have the same extension in all worlds, and this means that they have the same intension, then we can indeed substitute α for β in this complex expression, and the complex expressions will have the same extension, because the intensional operator makes reference to the intension of α and β, which is the same.

This identification of sense and intension is unproblematic in a lot of contexts (like temporal and modal contexts), i.e. a lot of contexts are arguably intensional contexts. There are also contexts where it is actually problematics. Believe is an example: though we may want to admit with Frege that substituting expressions with the same sense preserves truth value in believe sentences, it is not at all clear that substituting expressions with the same intension does.

Another example is given by Gennaro Chierchia (see Chierchia 198?). Look at the predicates being bought and being sold. Clearly every thing that is bought is sold and vice versa. Moreover, this seems to be necessarily the case, given what buy and sell means. Hence in every possible situation everything that is bought is sold.

This means that in every possible world the extension of being bought is the same as the extension of being sold. This, in its turn, means that being bought and being sold have the same intension. Yet it seems that the property of being bought is not the same as the property of being sold. Think of the property of being bought as what it means to be bought. There is a sense in which what it means to be bought is the same as what it means to be sold, because you can't possibly have the one property without the other. Yet, there is also a sense in which what it means to be bought is not the same as what it means to be sold. If the sense of being bought is what it means to be bought, then there is reason to assume that the sense of being bought can be different from the sense of being sold.

We can incorporate this into a type logic. That is, we can set up a theory where we have three relevant types of semantic entities corresponding to an expression α:

the sense of α, which determines:
the intension of α, which determines:
the extension of α in a world w.

Such a type logical language of intensions and senses is developed in Thomason 1980 (see also Chierchia and Turner 1987).
We are here mainly interested in showing the effects of non-extensionality, and in how non-extensionality affects the grammar: introducing both senses and intensions makes the theory too complicated for our limited purposes here, and we will hence identify senses and intension (or rather, only introduce intensions).

5.2. Intensional type logic.

Accommodating intensionality in the type logic goes precisely along the lines of the earlier example where we showed how to accommodate variation of extension across time. Here we want to accommodate variation of extension across worlds. This time, we will however also extend the language with expressions that operate on intensions, and, in fact, we will extend the language with expressions that denote intensions. Still, we follow step by step our earlier example, and indicate what we have to do to our earlier extensional type logic to accommodate intensionality.

THE SYNTAX OF INTENSIONAL LOGIC

We define a new logical language, IL, the language of intensional logic.

As before in our extensional type logical language, we will define for every expression the extension of that expression in the appropriate domain. But for any expression α of type a, IL will contain an expression β, which will denote the intension of expression α. Since the intension of expression α is the function which assigns to every possible worlds the extension of α in that world, and the extension of α in that world is in D_a, the intension of α is itself in (W → D_a). This means that the expression β, which denotes the intension of α, will be an expression of a type, whose corresponding domain is (W → D_a).

We didn't have such types or domains in our extensional type logical language TL, hence in IL we will have to introduce them.

Thus we start by defining a new set of types:

TYPE_IL is the smallest set such that:
1. e,t ∈ TYPE_IL
2. if a,b ∈ TYPE_IL, then <a,b> ∈ TYPE_IL
3. if α ∈ TYPE_IL, then <s,a> ∈ TYPE_IL

The types <s,a> are intensional types. As we will see, expressions of type <s,a> will be interpreted as functions from possible worlds into D_a.

s here is the type of possible worlds. Note that in IL we do not make type s a basic type. Hence, we do not have expressions of type s in our language. We adopt here, following Montague, the strategy of not expressing the variation across possible worlds in our logical language, but deal with it in the interpretation of our language.

As in the case of variation across time, this is a choice of convenience. We could as well take s to be a basic type and have expressions of that type. This is done in two sorted type logic TY2. I discuss this briefly later.
Some sample new types:
<s,e>, the type of expressions denoting functions from worlds into individuals: **individual concepts.**
<s,t>, the type of expressions denoting functions from worlds into truth values: **propositions.** These functions are characteristic functions of sets of possible worlds. Hence we identify propositions with sets of possible worlds.
<s,<e,t>>, the type of expressions denoting functions from worlds into sets of individuals: **intensional properties.**
<s,<<e,t>,t>>, the type of expressions denoting functions from worlds into generalized quantifiers **intensional generalized quantifiers.**

The language IL has the same logical constants as our extensional type logical language: \( \neg, \land, \lor, \rightarrow, \forall, \exists, =, (,), \lambda \), plus four more:
\( ^\land \) (cap or intension), \( ^\lor \) (cup or extension), \( \Box \) (necessarily), \( \Diamond \) (possibly).

The syntax of IL is the same as before except that we now have constants and variables of the new types as well, and we have clauses for the new logical constants:
From now on TYPE stands for TYPE\(_{IL} \), the same for CON, etc.

**Constants and variables:**
For every \( a \in \text{TYPE} \): \( \text{CON}_a = \{ c^a_1, c^a_2, ... \} \)
\( \) a set of constants of type \( a \) (at most countably many)
For every \( a \in \text{TYPE} \): \( \text{VAR}_a = \{ x^a_1, x^a_2, ... \} \)
\( \) a set of variables of type \( a \) (countably many)

for each \( a \in \text{TYPE} \): \( \text{EXP}_a \) is the smallest set such that:
1. Constants and variables:
\( \text{CON}_a \cup \text{VAR}_a \subseteq \text{EXP}_a \)
2. Functional application:
\( \) If \( \alpha \in \text{EXP}_{s,a,b} \) and \( \beta \in \text{EXP}_a \) then \( (\alpha(\beta)) \in \text{EXP}_b \)
3. Functional abstraction:
\( \) If \( x \in \text{VAR}_a \) and \( \beta \in \text{EXP}_b \) then \( \lambda x \beta \in \text{EXP}_{s,a,b} \)
4. Connectives:
\( \) If \( \varphi, \psi \in \text{EXP}_t \) then \( \neg \varphi, (\varphi \land \psi), (\varphi \lor \psi), (\varphi \rightarrow \psi) \in \text{EXP}_t \)
5. Quantifiers:
\( \) If \( x \in \text{VAR}_a \) and \( \varphi \in \text{EXP}_t \) then \( \forall x \varphi, \exists x \varphi \in \text{EXP}_t \)
6. Identity:
\( \) If \( \alpha \in \text{EXP}_a \) and \( \beta \in \text{EXP}_a \) then \( (\alpha = \beta) \in \text{EXP}_t \)
7. Intension:
\( \) If \( \alpha \in \text{EXP}_a \) then \( ^{\land} \alpha \in \text{EXP}_{s,s,a} \)
8. Extension:
\( \) If \( \alpha \in \text{EXP}_{s,s,a} \) then \( ^{\lor} \alpha \in \text{EXP}_a \)
9. Modals:
\( \) If \( \varphi \in \text{EXP}_t \) then \( \Box \varphi, \Diamond \varphi \in \text{EXP}_t \)

Thus we now have new expressions of the following form:
Let \( \text{JOHN} \in \text{CON}_e \)
\[ \forall \text{JOHN} \in \text{EXP}_{<s,e>} \]
Let \( \text{WALK} \in \text{CON}_{<e,t>} \)
\[ \forall \text{WALK} (\text{JOHN}) \in \text{EXP}_{<s,e>} \]
Let \( P \in \text{EXP}_{<s,e,t>} \)
\[ \forall P \in \text{EXP}_{<e,t>} \]

**SEMANTICS FOR INTENSIONAL TYPE LOGIC**

Before, when we added variation across times to the semantics, we had to add to our models a set of moments of time, and we changed the interpretation function of the constants: instead of interpreting a predicate as a set of individuals, we interpreted a predicate as a function which assigns to every moment of time a set of individuals. We will make exactly the same changes here, but now with respect to variation across possible worlds. We add a set of possible worlds to our model, and instead of interpreting a constant of type \( a \) extensionally as an entity in \( D_a \), we accommodate the possibility that the extension of that expression can vary across possible worlds, by assuming that it is interpreted as a function from possible worlds into entities of domain \( D_a \).

This means, for instance, that a predicate of type \( <e,t> \) is interpreted as a function which assigns to every possible worlds (the characteristic function of) a set of individuals.

We put a special restriction of **rigidity** on our models concerning the interpretation of constants of type \( e \), which will be discussed later.

A **model for IL** is a triple \( M = <D,W,F> \), where:
1. \( D \) is a non-empty set of individuals.
2. \( W \) is a non-empty set of possible worlds.
3. for every \( a \in \text{TYPE} \): \( F : \text{CON}_a \rightarrow (W \rightarrow D_a) \)
4. **Rigidity**: for every \( c \in \text{CON}_e \): for every \( w,v \in W \): \( F(c)(w)=F(c)(v) \)

Let \( M = <D,W,F> \) be a model for IL. for any type \( a \in \text{TYPE} \), we define \( D_a \), the domain of type \( a \):
1. \( D_e = D \)
2. \( D_t = \{0,1\} \)
3. \( D_{<a,b>} = (D_a \rightarrow D_b) \)
4. \( D_{<s,a>} = (W \rightarrow D_a) \)

An assignment (on \( M \)) is any function \( g \) such that:
for every \( a \in \text{TYPE} \), for every \( x \in \text{VAR}_a \): \( g(x) \in D_a \)

Since we incorporate variation of extension across possible worlds, the semantics is parameterized accordingly and we define recursively \( \ll [\alpha] \rr_{M,w,g} \), the extension of \( \alpha \) in \( M \) in \( w \) relative to \( g \).

As before, most of this definition is the same as in the case of extensional type logic, except that we add parameter \( w \) to the relevant clauses.

Since constants of type \( a \) are now interpreted as functions from worlds to entities in \( D_a \), rather than directly as entities in \( D_a \), we have to change the interpretation of the constants:
F(c) ∈ (W → Dₐ)

But [c]ₘₜₜ is the extension of c in M in w relative to g. This means that [x]ₘₜₜ cannot be F(c), but has to be F(c)(w). The extension of constant c in M in world w relative to g is whatever the interpretation of c, F(c), assigns to the world of evaluation w. Again, this is just what we did for variation across times.

The only other thing that changes is that, obviously, we have to interpret the clauses of IL that are new.

The semantic interpretation of IL: \[ [\alpha]_{M,w,g} \]

For any a, b ∈ TYPE:
1. Constants and variables.
   - If c ∈ CON, then \[ [c]_{M,w,g} = F(c)(w) \]
   - If x ∈ VAR, then \[ [x]_{M,w,g} = g(x) \]
2. Functional application.
   - If α ∈ EXP and β ∈ EXP, then:
     \[ [(\alpha(\beta))]_{M,w,g} = [\alpha]_{M,w,g}([\beta]_{M,w,g}) \]
3. Functional abstraction.
   - If x ∈ VAR and β ∈ EXP, then:
     \[ [\lambda x \beta]_{M,w,g} = h, \]
     where h is the unique function in (Dₐ → Dₙ) such that:
     for every d ∈ Dₐ: h(d) = \[ [\beta]_{M,w,g,d} \]
   - If φ, ψ ∈ EXP, then:
     \[ [\neg \varphi]_{M,w,g} = \neg([\varphi]_{M,w,g}) \]
     \[ [(\varphi \land \psi)]_{M,w,g} = \land([\varphi]_{M,w,g},[\psi]_{M,w,g}) \]
     \[ [(\varphi \lor \psi)]_{M,w,g} = \lor([\varphi]_{M,w,g},[\psi]_{M,w,g}) \]
     \[ [(\varphi \rightarrow \psi)]_{M,w,g} = \rightarrow([\varphi]_{M,w,g},[\psi]_{M,w,g}) \]

As before on the right hand side we find the following truth functions:
\[ \neg = \{ <0,1>, <1,0> \} \]
\[ \land = \{ <1,1>, <1,0>, <0,1>, <0,0> \} \]
\[ \lor = \{ <1,1>, <1,0>, <0,1>, <0,0> \} \]
\[ \rightarrow = \{ <1,1>, <1,0>, <0,1>, <0,0> \} \]
5. Quantifiers.
   - If x ∈ VAR and φ ∈ EXP, then:
     \[ [\forall x \varphi]_{M,w,g} = 1 \text{ iff for every } d \in Dₐ: [\varphi]_{M,w,g,d} = 1; 0 \text{ otherwise} \]
     \[ [\exists x \varphi]_{M,w,g} = 1 \text{ iff for some } d \in Dₐ: [\varphi]_{M,w,g,d} = 1; 0 \text{ otherwise} \]
6. Identity.
   - If α, β ∈ EXP, then:
     \[ [\alpha = \beta]_{M,w,g} = 1 \text{ iff } [\alpha]_{M,w,g} = [\beta]_{M,w,g} \]
7. Intension.
   - If α ∈ EXP, then:
     \[ [\tilde{\alpha}]_{M,w,g} = h, \]
     where h is the unique function in (W → Dₐ) such that:
     for every v ∈ W: h(v) = \[ [\alpha]_{M,v,g} \]
8. Extension.
   If $\alpha \in \text{EXP}_{s,s,a}$ then:
   $$\llbracket \alpha \rrbracket_{M,w,g} = \llbracket \alpha \rrbracket_{M,w}(w)$$

   If $\varphi \in \text{EXP}_t$ then:
   $$\llbracket \square \varphi \rrbracket_{M,w,g} = 1 \text{ iff for every } v \in W: \llbracket \varphi \rrbracket_{M,v,g} = 1; 0 \text{ otherwise}
   \llbracket \Diamond \varphi \rrbracket_{M,w,g} = 1 \text{ iff for some } v \in W: \llbracket \varphi \rrbracket_{M,v,g} = 1; 0 \text{ otherwise}$$

When we compare pairwise the clauses 7 and 2, 8 and 3, 9 and 5, then we see that the intensionalizing operator $^\wedge$ is just like the $\lambda$-operator, except that it abstracts over possible worlds. Since we do not have variable of type $s$, the type of possible worlds, we cannot express $\lambda$-abstraction over possible worlds explicitly in IL, but we do express it implicitly through the operator $^\wedge$.

Similarly, the extensionalization operator $^\vee$ is just like the operation of functional application, as applied to possible worlds. Again, we don’t have such application explicitly in our language, but implicitly through $^\vee$.

Similarly, the operations $\square$ and $\Diamond$ are just the universal and the existential quantifier, except that they quantify implicitly over possible worlds. For normal linguistic applications, we assume that these quantifiers over worlds are restricted by an accessibility relation between worlds, determining the nature of the modality. We will not bother about that here, though.

As we will see, since the language TY2 will contain a type of possible worlds, it has variables of type $s$, hence the normal operations of $\lambda$-abstraction, functional application, and the quantifiers work in TY2 with variables of type $s$ as well. In TY2 we do not need the operators $^\wedge$, $^\vee$, $\square$, $\Diamond$, because we have explicit abstraction, application, and quantification over possible worlds, hence we can define the IL operators (at the cost of representing the possible world variables explicitly in our representation language).

A sentence is a formula without free variables.
Let $X$ be a set of sentences and $\varphi$ a sentence.

$$\llbracket \varphi \rrbracket_{M,w} = 1, \varphi \text{ is true in } M \text{ in } w, \text{ iff for every } g: \llbracket \varphi \rrbracket_{M,w,g} = 1$$
$$\llbracket \varphi \rrbracket_{M,w} = 0, \varphi \text{ is false in } M \text{ in } w, \text{ iff for every } g: \llbracket \varphi \rrbracket_{M,w,g} = 0$$

$$X \models \varphi, X \text{ entails } \varphi, \text{ iff for every model } M = <D,W,F>:$$
$$\text{for every } w \in W:$$
$$\text{if for every } \delta \in X: \llbracket \delta \rrbracket_{M,w} = 1 \text{ then } \llbracket \varphi \rrbracket_{M,w} = 1$$

Some examples:

Let $\text{WALK} \in \text{CON}_{s,e,t}$ and $j \in \text{CON}_e$. Then:

$$\llbracket \text{WALK}(j) \rrbracket_{M,w,g} = 1 \text{ iff } \llbracket \text{WALK} \rrbracket_{M,w,g}(\llbracket j \rrbracket_{M,w,g}) = 1 \text{ iff}$$
$$\llbracket \text{F(WALK)(w)}(\text{F(j)(w)}) \rrbracket = 1 \text{ iff } \text{F(j)(w)} \in \text{F(WALK)(w)}$$

Thus $\text{WALK}(j)$ is true in world $w$ iff the interpretation of $j$ in $w$ is in the set of walkers in $w$. Since the set of walkers can vary across possible worlds, we have indeed incorporated variation across possible worlds in our semantics.
\[ \langle \langle \text{WALK}(j) \rangle \rangle \rangle_{M,w,g} = h, \]
where for every \( v \in W \): \( h(v) = \langle \langle \text{WALK}(j) \rangle \rangle_{M,v,g} \)

= \( h, \) where for every \( v \in W \): \( h(v) = [\langle \langle \text{WALK} \rangle \rangle_{M,v,g}]j \rangle_{M,v,g} \)

= \( h, \) where for every \( v \in W \): \( h(v) = [F(\text{WALK})(v)](F(j)(v)) \)

= \( h, \) where for every \( v \in W \): \( h(v) = 1 \) iff \( F(j)(v) \in F(\text{WALK})(v) \)

\[ \{ v \in W : F(j)(v) \in F(\text{WALK})(v) \} \]

Thus, \( \langle \langle \text{WALK}(j) \rangle \rangle \) denotes the proposition that John walks, the set of all possible worlds where John is in the set of individuals that walk in that world.

\[ \langle \langle \text{WALK}(j) \rangle \rangle \rangle_{M,w,g} = 1 \) iff for every \( v \in W \): \( \langle \langle \text{WALK}(j) \rangle \rangle_{M,v,g} = 1 \) iff

for every \( v \in W \): \( F(j)(v) \in F(\text{WALK})(v) \)

Hence \( \langle \langle \text{WALK}(j) \rangle \rangle \) is true in a possible world iff for every world, the interpretation of \( j \) in that world is in the set of walkers in that world.

Let us now come to the rigidity condition. We could also impose this condition by a meaning postulate:

For every \( c \in \text{CON}_e \) and every model for our language the following sentence is true in every world in that model:

\[ \text{MP: } \exists x[\langle \langle x = c \rangle \rangle] \]

The constants of type \( e \) in our language are the interpretations of the proper names. This meaning postulate tells us that proper names are interpreted as rigid designators: proper names do not vary their extension from world to world: a proper name refers to the same individual in every possible world. This is a difference between proper names and definite descriptions.

Kripke 1972 argues that proper names are rigid in the context of modals and counterfactual conditionals. Look at the definite description \textit{the president} in sentence (1):

\[ (1) \quad \text{If the president had not been a democrat, there would have been a republican administration.} \]

Sentence (1) is ambiguous. On the one reading, it means that if the current president, Bill Clinton, had not been a democrat he would have been a republican. The semantics for counterfactuals instructs us here to take the current president from our world to close worlds where he is not a democrat: it says that he's a republican there.

On the other reading, the sentence expresses some thing like: if the expression \textit{the president} had referred to somebody who isn't a democrat, it would have referred to somebody who is a republican. This instructs us to look at the reference of \textit{the president} in different worlds: go
to close worlds where whoever is the president there is not a democrat, that person is a republican there.

Now look at the proper name Bill Clinton in sentence (2):

(2) If Bill Clinton had not been a democrat he would have been a republican.

Sentence (2) is not ambiguous. It only has a reading corresponding to the first reading (1) discussed above:
look at close worlds where Clinton is not a democrat, he’s a republican there.
Sentence (2) does not mean: look at close worlds where the name Clinton does not refer to somebody who is a democrat: it refers there to a republican.

Kripke argues (and Montague follows him in this) that this means that while definite descriptions can refer to different individuals in different worlds, proper names cannot, proper names are rigid designators, they refer to the same individual in every world where they refer at all. This is what the condition of rigidity (or the meaning postulate) guarantees.

5.3. two sorted type logic TY2.

In two-sorted type we follow the other alternative for accommodating variation of extension across possible worlds, namely to make the reference to possible worlds explicit in the logical language. Two sorted type logic is just a simple variant of the extensional type logic given before.

\[ \text{TYPE}_{TY2} \text{ is the smallest set such that:} \]
1. e,t,s \( \in \text{TYPE}_{TY2} \)
2. if a,b \( \in \text{TYPE}_{TY2} \) then \(<a,t> \in \text{TYPE}_{TY2}\)

\( \text{TY2} \) has the same logical constants as extensional type logic: \( \neg,\land,\lor,\rightarrow,\forall,\exists,=,(,),\lambda \)

**Constants and variables:**
For every a \( \in \text{TYPE} \): \( \text{CON}_a = \{c^a_1,c^a_2,...\} \)
a set of constants of type a (at most countably many)
For every a \( \in \text{TYPE} \): \( \text{VAR}_a = \{x^a_1,x^a_2,...\} \)
a set of variables of type a (countably many)

As before, we now write \( \text{TYPE} \) for \( \text{TYPE}_{TY2} \).
For each a \( \in \text{TYPE} \), \( \text{EXP}_a \) is the smallest set such that:
1. Constants and variables:
   \( \text{CON}_a \cup \text{VAR}_a \subseteq \text{EXP}_a \)
2. Functional application:
   If \( \alpha \in \text{EXP}_{a,b} \) and \( \beta \in \text{EXP}_a \) then \((\alpha(\beta)) \in \text{EXP}_b \)
3. Functional abstraction:
   If x \( \in \text{VAR}_a \) and \( \beta \in \text{EXP}_b \) then \( \lambda x \beta \in \text{EXP}_{a,b} \)
4. Connectives:
   If \( \phi,\psi \in \text{EXP}_t \) then \( \neg \phi, (\phi \land \psi), (\phi \lor \psi), (\phi \rightarrow \psi) \in \text{EXP}_t \)
5. Quantifiers:
   If x ∈ VAR_a and φ ∈ EXP, then ∀xφ, ∃xφ ∈ EXP.
6. Identity:
   If α ∈ EXP_a and β ∈ EXP_a then (α=β) ∈ EXP.

A model for TY2 is a triple M = <D,W,F>, where:
1. D is a non-empty set of individuals.
2. W is a non-empty set of possible worlds.
3. For every a ∈ TYPE: F(a) ∈ D_a

Let M = <D,F> be a model for L. For any type a ∈ TYPE, we define D_a, the domain of type a:
1. D_a = D
2. D_t = {0,1}
3. D_s = W
4. D_<a,b> = (D_a → D_b)

An assignment (on M) is any function g such that:
   For every a ∈ TYPE for every x ∈ VAR_a: g(x) ∈ D_a

The semantic interpretation is exactly the same as that for extensional type logic. We define [[α]]_M,g, the extension of α in M relative to g. I will not give the semantic interpretation here.

How do we express in TY2 the variation across worlds? By changing our representations. Before, we assumed a representation WALK(j) for john walks where WALK ∈ CON_{e,t} and j ∈ EXP_e. But walk is an expression whose extension varies with respect to possible worlds (though, given rigidity, john is not). We express this directly in the language. We do not choose a constant WALK ∈ CON_{e,t}, as the representation of walk, but a constant WALK ∈ CON_{s,<e,t,>}. Type <s,<e,t,>> is the type of relations between individuals and possible worlds. We choose a special designated variable w ∈ VAR_s to indicate the world of evaluation.

And we represent walk as the complex expression:
   WALK(w) ∈ EXP_{s,e,t,>}

John does not vary its extension with possible worlds, hence we can continue to represent john as j ∈ CON_e.

John walks then gets represented as:
   [WALK(w)](j)

[[WALK(w)](j)]_M,g = 1 iff [[WALK(w)]]_M,g(F(j)) = 1 iff
   [F(WALK)(g(w))](F(j)) = 1 iff <F(j),g(w)> ∈ F(WALK)

Varying the assignment to variable w, will then deal with variation of extension across possible worlds.

We can find a translation from IL into TY2 which translates every IL expression into a TY2 expression with the same meaning. As before, the advantage of TY2 over IL is that all the
abstraction and quantification that goes on is explicit in the expression. The disadvantage is that because of that the expressions tend to get longer.

Some comparisons:

<table>
<thead>
<tr>
<th>IL</th>
<th>TY2</th>
</tr>
</thead>
<tbody>
<tr>
<td>WALK(j) → WALK(j,w)</td>
<td>WALK(j,w)</td>
</tr>
<tr>
<td>^WALK(j) → λw.WALK(j,w)</td>
<td></td>
</tr>
<tr>
<td>□WALK(j) → ∀w[WALK(j,w)]</td>
<td></td>
</tr>
<tr>
<td>◊WALK(j) → ∃w[WALK(j,w)]</td>
<td></td>
</tr>
</tbody>
</table>

Groenendijk and Stokhof 1982 make fruitful use of TY2 in representing the meanings of questions. Another fruitful use of TY2 is that it makes certain logical peculiarities of IL completely perspicuous. We will see that in the next section.

3.4. Logical properties of IL and TY2.

When we look at the semantics for TY2, we see that it is exactly the same as the semantics for extensional type theory. In fact, the only difference with extensional type theory is that we have one more sort of individuals (called possible worlds) that we can quantify over. But these individuals are treated in exactly the same way as normal individuals are, there is no difference in the semantics between the two theories, hence TY2 is semantically just the same theory as extensional type logic. This means that TY2 has exactly the same logical properties as extensional type logic: the principles of alphabetic variants, extensionality and λ-conversion and the function-identity principle hold for TY2 just as much as they hold for extensional type logic.

Things are different for IL. The principle of alphabetic variants holds as much for IL as for the other theories, because this has only to do with the nature of variable binding operations. But the principles of extensionality and λ-conversion do not hold in the unrestricted forms in which they hold in extensional type logic.

This is, of course, to be expected, since we made IL in order to allow for contexts in which extensionality fails. Let us look at some examples.

To make the examples simple, let us add the description-operator σ to the logical language:

**Descriptions.**

If P ∈ EXP_<e,t> then σ(P) ∈ EXP_ε

\[ [\sigma(P)]_{M,w,g} = \text{the unique object } d \in D_\varepsilon \text{ such that } [P]_{M_{w,g}}(d) = 1 \]

if there is such a unique object; undefined otherwise.

Let BELIEVE ∈ CON_<<s,t>,<e,t>>>, m ∈ CON_ε

Let AU (author of Ulysses) and AF (author of Finnegans Wake) be constants of type <e,t>.

Look at the inference from (1) and (2) to (3):
Mary believes that the author of Ulysses is the author of Ulysses.
The author of Ulysses is the author of Finnegans Wake.
Mary believes that the author of Ulysses is the author of Finnegans Wake.

We can represent this inference as follows:

1. \( \text{BELIEVE}(m, \sigma_{AU} = \sigma_{AU}) \)
2. \( \sigma_{AU} = \sigma_{AF} \)
3. \( \text{BELIEVE}(m, \sigma_{AU} = \sigma_{AF}) \)

We show that this inference is not valid by constructing a model \( M \) and a world \( w \) where premises (1') and (2') are true, but conclusion (3') is false.

In the following example, I will, for simplicity take characterizing sets (of individuals or possible worlds) wherever readability makes that useful. Let \( M = \langle D, W, F \rangle \) be a model for IL, let \( w \in W \), \( m, jj, vw \in D \).

Let \( F(m)(w) = m \)
Let \( F(AU)(w) = F(AF)(w) = \{ jj \} \).

Now let us make the following assumption. We associate with \( m \) in \( w \) a set of worlds \( B_{m,w} \), the worlds compatible with what Mary believes in \( w \). And we assume the following constraint on the model \( M \):

for every proposition \( p \subseteq W \): \( F(\text{BELIEVE})(w)(m,p) = 1 \) iff \( B_{m,w} \subseteq p \)

This means that we assume that in \( M \) in \( w \), Mary's believes follows the standard Hintikka style possible world analysis (for a strong exposition and defense of this, see Stalnaker 1987).

We now make the following assumptions:
1. for every \( v \in B_{m,w} \): \( F(AU)(v) = \{ jj \} \)
   This means that in all the worlds compatible with what Mary believes, the predicate Author of Ulysses denotes James Joyce.
2. for every \( c \in B_{m,w} \): \( F(FW)(v) = \{ vw \} \)
   In all worlds compatible with what Mary believes, the predicate Author of Finnegans Wake denotes Virginia Woolf. In other words, Mary is under the delusion that Virginia Woolf wrote Finnegans Wake.

Now, we can argue as follows.

(2') is true in \( M \) in \( w \), because \( F(AU)(w) = F(AF)(w) \).

(1') is true iff \( F(\text{BELIEVE})(w)(m, \sigma_{AU} = \sigma_{AU}) = 1 \) iff
(by the constraint: \( B_{m,w} \subseteq \{ v \in W : \sigma_{AU} = \sigma_{AU} \} = W \).

Hence (1') is true iff \( M_{m,w} \subseteq Q \), which is of course the case.

So indeed (1') is true.

We see then that indeed both premises (1') and (2') are true in \( M \) in \( w \).

(3') is true in \( M \) in \( w \) iff \( F(\text{BELIEVE})(w)(m, \sigma_{AU} = \sigma_{AF}) = 1 \) iff
\( B_{m,w} \subseteq \{ v \in W : \sigma_{AU} = \sigma_{AF} \} \)

128
But given the assumptions about Mary's believe, this is not true, in fact, what is true is:

\[ B_{m,w} \cap [\tau(\sigma_{AU}=\sigma_{AF})]_{M,w,g} = \emptyset \]

Thus (3') is false in M in w.

Thus model M and world w provide a counterexample to extensionality: the principle of extensionality is not valid in IL.

\( \sigma_{AU} \) and \( \sigma_{AF} \) have the same extension in M in w, but \( BELIEVE(m,\tau(\sigma_{AU}=\sigma_{AF})) \) and \( BELIEVE(m,\tau(\sigma_{AU}=\sigma_{AF})) [\sigma_{AU}/\sigma_{AF}] \) do not have the same truth value in M and w.

I have given an example in a natural setting, but the failure of extensionality holds also if we don't impose the Hintikka semantics on \textit{believe}:

Though \( \sigma_{AU} \) and \( \sigma_{AF} \) happen to have the same extension in w, this is not the case in other worlds. Hence the proposition denoted by \( \tau(\sigma_{AU}=\sigma_{AU}) \) and \( \tau(\sigma_{AU}=\sigma_{AF}) \) are not the same: the first denotes the trivial proposition, the set of all possible worlds, the second denotes a non-trivial proposition. Since these are different propositions and believe expresses a relation between individuals and propositions, one can stand in the believe relation to the one proposition without standing in the believe relation to the other. This is the reason that substitution fails.

What about TY2? I said before that we can translate IL expressions into TY2 while preserving meaning. That means that also in TY2 the inference should not be valid. But TY2 \textit{does} satisfy extensionality. How is that possible?

The following are the TY2 translations:

1. \( BELIEVE(m,\lambda w.\sigma[AU(w)]=\sigma[AU(w)],w) \)
2. \( \sigma[AU(w)]=\sigma[AF(w)] \)
3. \( BELIEVE(m,\lambda w.\sigma[AU(w)]=\sigma[AF(w)],w) \)

This is indeed also in TY2 an invalid inference. But, as we can see, it is not an instance of extensionality. If we try to substitute \( \sigma[AF(w)] \) for \( \sigma[AU(w)] \), based in (2), we violate the condition of variable binding: variable \( w \) is free in \( \sigma[AU(w)] \) in (2), and would get bound by the \( \lambda \)-operator if we substitute it in (1). Only if we replace (2) by the \textit{modal} statement (2') does the argument become valid:

\( (2') \forall w[\sigma[AU(w)]=\sigma[AF(w)]] \)

But this, of course, doesn't hold in general. In particular, it doesn't hold in the model we have given.

When is substitution valid in IL?
Clearly the failure of substitution arises when we substitute an expression whose extension can vary across possible worlds in an intensional context. If we substitute such expression for an expression with the same extension at the relevant world in an extensional context, substitution is valid (i.e. substituting \( \sigma(AU) \) for \( \sigma(AF) \) in (2) preserves truth value, though these expressions vary their extension across possible world): extensional contexts are contexts where the extension in other worlds is not relevant. Similarly, if the expressions not only have the same extension in the world of evaluation, but in fact in every world, then again substitution is valid, because they have the same intension. Thus:
Principle of substitution for IL:
Let \( \phi \) be an expression containing an occurrence of an expression \( \alpha \), let \( \phi[\beta/\alpha] \) be the result of substituting \( \beta \) for that occurrence of \( \alpha \). Let \( x_1, \ldots, x_n \) be the free variables in \( \alpha \) and \( \beta \). Then:

\[
\forall x_1 \ldots \forall x_n (\alpha = \beta) \text{ entails } \phi = \phi[\beta/\alpha]
\]
if the occurrence of \( \alpha \) is not in the scope of an intensional operator \(^\land, \Box \) or \( \Diamond \).

Further more:

\[
\forall x_1 \ldots \forall x_n (\alpha = \beta) \text{ entails } \phi = \phi[\beta/\alpha]
\]

We have similar problems with \( \lambda \)-conversion:

Let us now look at the following model:
AF stands for ‘being the author of Finnegans Wake’.
AL stands for ‘being the author of To The Lighthouse’.

We now assume that in every world \( v \in W: \llbracket \text{AF} \rrbracket_{M,v,g} = \{jj\} \)
And we assume that in the real world \( w: \llbracket \text{AL} \rrbracket_{M,v,g} = \{vw\}, \)
but in every other world \( v: \llbracket \text{AL} \rrbracket_{M,v,g} = \{jj\}. \)

Thus in this example, \( vw \) wrote To The Lighthouse as a miracle.
(The assumptions made are, of course, not very natural, but that is only due to the simplification we have made to let the modal operators be unrestricted.)

Now look at the following statements (1) and (2):

(1) \( \lambda x. \Diamond (x = \sigma \text{AF}) (\sigma \text{AL}) \)
(2) \( \Diamond (\sigma \text{AL} = \sigma \text{AF}) \)

(2) is true in world \( w \) if there is some world \( v \) such that \( \llbracket \sigma \text{AL} = \sigma \text{AF} \rrbracket_{M,v,g} = 1 \). This is actually the case in every world except for the real world, because in every world, except for the real world \( \sigma \text{AL} \) and \( \sigma \text{AF} \) have the same extension \( jj \). Hence (2) is true.

(1) is true in world \( w \) iff \( \llbracket \lambda x. \Diamond (x = \sigma \text{AF}) (\sigma \text{AL}) \rrbracket_{M,w,g} = 1 \).

\[ [\sigma \text{AL}]_{M,w,g} \in \{ d \in D: \text{for some } v \in W: d = [\sigma \text{AF}]_{M,v,g} \} \]

for some \( v \in W: \llbracket \sigma \text{AF} \rrbracket_{M,v,g} = vw \).

There is no such world, since in no world did \( vw \) write Finnegans Wake, hence (1) is false.

The conclusion is that \( \lambda \)-conversion is not always valid in IL. And this is so, because the expressions in (1) and (2) do not have the same meaning. (1) expresses that the actual author of To The Lighthouse could have written Finnegans Wake. (2) expresses that both Finnegans Wake and To The Lighthouse could have been written by the same author. One may well believe that the first is preposterous, because clearly Virginia Woolf couldn't have written Finnegans Wake, while at the same time believing that the second is true, because Joyce could have written To The Lighthouse as well.

When is \( \lambda \)-conversion valid in IL? We're looking at \( [\lambda x. \beta](\alpha) \) and \( [\beta](\alpha/x) \). One case where \( \lambda \)-conversion is obviously valid is when no occurrence of \( x \) in \( \beta \) is in an intensional context in \( \beta \). In case an occurrence of \( x \) is in an intensional context there is still another case where \( \lambda \)-conversion is valid, and that is the case where the extension of \( \beta \) doesn't vary its extension with possible worlds, i.e. if \( \beta \) is rigid.
β is rigid iff for every model $M = <D,W,F>$ and assignment $g$ and for every $w,v \in W$: $\llbracket \beta \rrbracket_{M,w,g} = \llbracket \beta \rrbracket_{M,v,g}$

In that case it doesn't matter for the interpretation which world $β$ gets evaluated relative to, and hence it has the same extension relative to the outside world of evaluation and the world where it will be evaluated relative in the scope of an intensional operator.

As an example, let $jj \in CONe$.
We assumed that constants of type $e$ are rigid.

Let's look at:

$$\llbracket [\lambda x. \circ (AL(x))] \rrbracket_{M,w,g} = 1 \text{ iff } F(jj)(w) \in \{d \in D: \text{for some } v \in W: d \in F(AL)(v) \} \text{ iff for some } v \in W: F(jj)(w) \in F(AL)(v)$$

$$\llbracket \circ (AL(jj)) \rrbracket_{M,w,g} = 1 \text{ iff for some } v \in W: F(jj)(v) \in F(AL)(v)$$

Given that for every $w$ and $v$: $F(jj)(w)=F(jj)(v)$, clearly:

$$\llbracket [\lambda x. \circ (AL(x))] \rrbracket_{M,w,g} = 1 \text{ iff } \llbracket \circ (AL(jj)) \rrbracket_{M,w,g} = 1$$

hence:

$$[\lambda x. \circ (AL(x))](jj) = \circ (AL(jj))$$

In other words:

It could have been the case that James Joyce had written To The Lighthouse.

iff

James Joyce has the property that he could have written To The Lighthouse.

We can define syntactically which IL-expressions are rigid:

1. constants of type $e$ are rigid.
2. for any type $a$: variables of type $a$ are rigid.
3. if $\alpha \in EXP_a$, then $\hat{\alpha}$ is rigid.
4. if $\phi \in EXP_t$ then $\Box \phi$, $\Diamond \phi$ are rigid.
5. if $\alpha \in EXP_{a,b}$ and $\beta \in EXP_a$ and $\alpha$ and $\beta$ are rigid, then $(\alpha(\beta))$ is rigid.
6. if $x \in VAR_a$ and $\beta \in EXP_b$ and $\beta$ is rigid then $\lambda x. \beta$ is rigid.
7. if $\varphi, \psi \in EXP_t$ and $\varphi, \psi$ are rigid then $(\varphi \land \psi)$, $(\varphi \lor \psi)$, $(\varphi \rightarrow \psi)$ are rigid.
8. if $\varphi, \psi \in EXP_t$ and $\varphi$ is rigid then $\forall x \varphi$, $\exists x \varphi$ are rigid.
9. if $\alpha, \beta \in EXP_a$ and $\alpha, \beta$ are rigid then $(\alpha=\beta)$ is rigid.

As an example, we show that $\Diamond WALK(j)$ is rigid:

$$\llbracket \Diamond \alpha \rrbracket_{M,w,g} = h,$$
where for every $v \in W$: $h(v) = \llbracket \alpha \rrbracket_{M,v,g}$
\[\alpha_{M,v,g} = k,\]
where for every \(v \in W: k(v) = \alpha_{M,v,g}\)
Obviously, \(h = k\).

Now we can formulate the principle of \(\lambda\)-conversion for IL:

**Principle of \(\lambda\)-conversion for IL:**
Let \(x \in VAR_a\) and \(\beta \in EXP_b\) and \(\alpha \in EXP_a\).
Let \(\beta[\alpha/x]\) be the result of substituting \(\alpha\) for every occurrence of \(x\) that is free in \(\beta\). Then:
\[
[\lambda x. \beta](\alpha) = \beta[\alpha/x]
\]
1. if no variable free in \(\beta\) gets bound in \(\beta[\alpha/x]\) and
   either 2. no occurrence of \(x\) is in the scope of \(\wedge, \Box\) or \(\Diamond\) in \(\beta\).
or 3. \(\beta\) is rigid.

This means that when we want to do a \(\lambda\)-conversion into an intensional context, we have to make sure that the expression to be converted in is modally closed. We will see in the grammar that this will effect the formulation of the semantic interpretations of our grammar: we will make certain translations more complicated to avoid the problem of converting a non-rigid expression into an intensional context.

We will see that when we think about the semantic interpretation of *seek*. We have before argued that the type of a relation like *kiss* is \(<<e,t>,t>,<e,t>>\); we assumed a constant \(KISS \in CON<<e,<e,t>>\) and we argued for the following translation of *kiss*:

\[
\lambda T \lambda x. T(\lambda y. \text{KISS}(x,y))
\]
This will ensure that sentence (1) gets interpreted as (2):

(1) Mary kisses a girl.
(2) \(\exists x[\text{GIRL}(x) \land \text{KISS}(m,x)]\)

If we make the same assumptions for *seek*, we interpret (3) as (4):

(3) Mary seeks a girl.
(4) \(\exists x[\text{GIRL}(x) \land \text{SEEK}(m,x)]\)

This is ok for one of the readings of (3), the **de re** reading, but (3) also has a **de dicto** reading, on which it does not commit you to a relation of seeking holding between Mary and any actual girl.

The question becomes: what should the type of SEEK be to accommodate the **de dicto** reading.

Let us add a second piece of data: (5) and (6) are not equivalent (on their **de dicto** readings):

(5) Mary seeks a griffin.
(6) Mary seeks a centaur.

In the real world there are no griffins, nor are there centaurs.
The non-equivalence of (5) and (6) shows very clearly that the type of the object of SEEK needs to be intensional.

In the real world \( w: F(\text{GRIFFIN})(w) = F(\text{CENTAUR})(w) = \emptyset \).

Any extensional translation of the object of seek, resp. a griffin and a centaur, will be compositionally built from the extensions in \( w \) of GRIFFIN resp. CENTAUR. Since these are the same in \( w \), these extensional translations of a griffin and a centaur will also be the same, and (5) and (6) will become equivalent.

Thus we cannot translate a griffin and a centaur as resp. GRIFFIN \( \in \text{CON}_{e,t} \) and CENTAUR \( \in \text{CON}_{e,t} \) and let SEEK be of type \( <<e,t,<e,t>> \), because it makes (7) and (8) equivalent:

\[
\begin{align*}
(7) & \text{SEEK}(m, \text{GRIFFIN}) \\
(8) & \text{SEEK}(m, \text{CENTAUR})
\end{align*}
\]

Nor can we assume that SEEK is of type \( <<e,t,t>,<e,t>> \) and let it operate just on the normal generalized quantifier meanings as in (9) and (10), because also (9) and (10) are equivalent.

\[
\begin{align*}
(9) & \text{SEEK}(m, \lambda P. \exists x [\text{GRIFFIN}(x) \land P(x)]) \\
(10) & \text{SEEK}(m, \lambda P. \exists x [\text{CENTAUR}(x) \land P(x)])
\end{align*}
\]

Hence the type for the object of SEEK needs to be intensional.

There are in the literature two proposals: Zimmerman 1992 assumes that SEEK is of type \( <<s,<e,t,t>,<e,t>> \), hence the type of the object of SEEK is an intensional property of type \( <s,<e,t,t>> \). Montague 1974 assumes that SEEK is of type \( <<s,<<e,t>,t>>,e,t>> \), hence the type of the object of SEEK is an intensional generalized quantifier of type \( <s,<<e,t>,t>> \).

Let GRIFFIN, CENTAUR \( \in \text{CON}_{e,t} \).

On the first approach we represent (5) and (6) as:

\[
\begin{align*}
(11) & \text{SEEK}(m, \check{\text{GRIFFIN}}) \\
(12) & \text{SEEK}(m, \check{\text{CENTAUR}})
\end{align*}
\]

On the second approach we represent (5) and (6) as:

\[
\begin{align*}
(13) & \text{SEEK}(m, \check{\lambda P. \exists x [\text{GRIFFIN}(x) \land P(x)]}) \\
(14) & \text{SEEK}(m, \check{\lambda P. \exists x [\text{CENTAUR}(x) \land P(x)]})
\end{align*}
\]

All these cases involve a relation SEEK to an intensional entity, and it doesn't follow that if Mary stands in this relation to a function from worlds into extensions, that she has to stand in an extensional relation of seeking to any individual. Hence the de dicto reading of (3), (5), (6) does not entail the de re reading, as it should be.

Though the set of griffins in \( w \) is the same as the set of centaurs in \( w \), the function which maps every world on the set of griffins in that world is not the same as the function which maps every world on the set of centaurs in that world.

Nor is the function which maps every world onto the set of all properties that some griffin has in that world the same as the function which maps every world on the set of all
properties that some centaur has in that world. Hence, indeed on either analysis (5) and (6) are not equivalent.

The difference between these approaches has to do with other NPs. Zimmerman argues that seek only has a de dicto reading with object NPs that can have a predicative reading (definite and indefinite NPs, but not quantificational NPs like every boy or most girls). Zimmermann assumes that predicative NPs can get an interpretation at the type of intensional properties <s,<e,t>>. Something else is done to get the proper analysis of cases with quantificational NPs as the object of seek.

On Montague's analysis seek has a de dicto reading for all object NPs. Thus the difference of opinion concerns the analysis of (15): Zimmerman generates only a de re reading, roughly (17), Montague generates (17) and the de dicto reading (18):

\[
(16) \text{Mary seeks every griffin.} \\
(17) \forall x[\text{GRiffin}(x) \rightarrow \text{SEEK}(m,x)] \quad \text{SEEK} \in \text{CON}_{<e,<e,t>} \\
(18) \text{SEEK}(m,\lambda P.\forall x[\text{GRiffin}(x) \rightarrow P(x)]) \quad \text{SEEK} \in \text{CON}_{<<s,<<e,t>,<e,t>>,<e,t>>}
\]

A situation to test the analyses on is the following. Suppose Mary believes there are griffins, and moreover she believes that she has caught all of them the previous weeks, but she believes that in the meantime they have escapes. She wants to recapture them and hence she starts seeking. Of course, there are no griffins.

If you feel that in this situation you can felicitously and truthfully say (16), Montague's analysis might be the one for you. If you can't, you might want to adopt Zimmerman's.

Since Montague's analysis is useful and instructive in building the grammar, I will adopt it for the time being. I will discuss Zimmerman's approach in a later chapter.

So, we assume that SEEK \in \text{CON}_{<<s,<<e,t>,<e,t>>,<e,t>>}

We have interpreted kiss as \lambda T \lambda x.T(\lambda y.\text{KISS}(x,y)) of type <<e,t>,<<e,t>>. If we interpret seek as SEEK, that means that the types of the translations of kiss and seek are not the same.

We might feel that this is unfortunate and wonder: couldn't we, using the relation SEEK as we have it, define the meaning of seek at type <<e,t>,<<e,t>> as follows?

\[
\lambda T \lambda x.\text{SEEK}(x,^T)
\]

Then both seek and kiss would translate as expressions of the same type. Then, just as we get the meaning (21) of (19) through applying the translation of kiss to \lambda P.\exists x[\text{UNICORN}(x) \land P(x)] as in (20) and do \lambda -conversion, we get the meaning (24) of (22) though applying the translation of seek to \lambda P.\exists x[\text{UNICORN}(x) \land P(x)], as in (23), and do \lambda -conversion:
(19) John kisses a unicorn.
(20) \[[\lambda T. \lambda x. T(\lambda y. \text{KISS}(x,y))] (\lambda P. \exists x [\text{UNICORN}(x) \land P(x)])(j)\]
(21) \[\exists x [\text{UNICORN}(x) \land \text{KISS}(j,x)]\]
(22) John seeks a unicorn.
(23) \[[\lambda T. \lambda x. \text{SEEK}(x,^c T)] (\lambda P. \exists x [\text{UNICORN}(x) \land P(x)])(j)\]
(24) \[\text{SEEK}(j,^c, \lambda P. \exists x [\text{UNICORN}(x) \land P(x)]\]

This is where the principle of $\lambda$-conversion of IL is relevant. That would be nice, but this analysis is impossible.

\(\lambda P. \exists x [\text{UNICORN}(x) \land P(x)]\) is an expression which is not rigid. Through $\lambda$-conversion in (23) it would end up in the scope of $^c$, hence $\lambda$-conversion is not allowed: (23) and (24) do not have the same meaning.

(Again, there is no such problem in (20), because there the expression does not end up under the scope of an intensional operator.)

Thus, we need to reject this suggestion, and we will have to develop different ideas about the different types that the noun phrase complements of transitive verbs seem to take.

Let us now come to interactions of the cap and the cup operator. These operators, I have already indicated, correspond to abstraction and application over possible worlds. As a consequence of that, we have the following principles regulating their interaction, which are useful in simplifying expressions:

**Principle of cup-cap elimination:**
Let \(\alpha \in \text{EXP}_a\):
\[\forall^c \alpha = \alpha\]

This is easy to prove:
\[\llbracket \forall^c \alpha \rrbracket_{M,w,g} = [\llbracket ^c \alpha \rrbracket_{M,w,g}](w)\]
\[\llbracket \forall^c \alpha \rrbracket_{M,w,g} = h,\]
where \(h\) is that function such that: for every \(v \in W\): \(h(v) = [\llbracket \alpha \rrbracket_{M,v,g}\]

Hence \[\llbracket \forall^c \alpha \rrbracket_{M,w,g} = h(w) = [\llbracket \alpha \rrbracket_{M,w,g}\]

This means that wherever we see \(\forall^c \alpha\) in an expression, we can simplify that expression through cup-cap elimination: replace \(\forall^c \alpha\) by \(\alpha\).

When we look at the translations of these expressions in TY2, we see the reason behind this principle:

\(\forall^c \alpha\) translates as \([\lambda w. \alpha](w)\), which through $\lambda$-conversion reduces to \(\alpha\).

In a real example:
\(\text{WALK}(j)\) translates as: \(\text{WALK}(j,w)\)
\(\forall^c \text{WALK}(j)\) translates as: \([\lambda w. \text{WALK}(j,w)](w)\)
\(\lambda\)-conversion reduces the second to the first.

The inverse of this principle does not hold:

\[ \wedge \alpha = \alpha \text{ is not valid.} \]

We can give a counterexample (with a bit of use-mention blurring):

Let \(MI \in \text{CON}_{<s,<s,e>}\) the Most Impopular office
Let \(P \in \text{CON}_{<s,e>}\) the Prime minister
Let \(I \in \text{CON}_{<s,e>}\) the minister of the Interior

Let:

\[
\begin{align*}
P(w_0) &= n & I(w_0) &= m & MI(w_0) &= P \\
P(w_1) &= m & I(w_1) &= n & MI(w_1) &= I \\
\end{align*}
\]

\[
\begin{align*}
\llbracket MI \rrbracket_{w_0} &= MI(w_0) = P \\
&= \{<w_0, n>, <w_1, m>\} \\
\end{align*}
\]

\[
\begin{align*}
\llbracket \wedge MI \rrbracket_{w_0} &= \lambda v. \llbracket MI \rrbracket_v(v) \\
&= \lambda x. (MI(v))(v) \\
&= \{<w_0, P(w_0)>, <w_1, I(w_1)>\} \\
&= \{<w_0, n>, <w_1, n>\}
\end{align*}
\]

When does cap-cup elimination hold? As usual, when \(\alpha\) is rigid:

**Principle of cap-cup elimination:**

Let \(\alpha \in \text{EXP}_a\)

\[ \wedge \alpha = \alpha \]

if \(\alpha\) is rigid

Again, we can see these facts easily by considering their translations in TY2.

Our constant \(P\) in IL is not rigid, hence in TY2 we choose a constant \(P\) of type \(<s,<s,<s,e>>\) and translate \(P\) as \(P(w)\) of type \(<s,<s,e>>\).

\(\wedge P\) translates as \(\lambda w.[[P(w)](w)]\) of type \(<s,<s,e>>\).

In \(\lambda w.[[P(w)](w)]\) both occurrences of variable \(w\) are bound by \(\lambda w\), in particular, the variable \(w\) in \(P(w)\) is bound. On the other hand in \(P(w)\) itself \(w\) is free, hence clearly these expressions don't need to have the same interpretation.

On the other hand, if \(\alpha\) is an IL expression that is rigid, its translation doesn't contain a free variable \(w\). In that case \(\alpha = \lambda w.\alpha(w)\).

This follows from the function-identity principle of extensional type logic.

Let me finally discuss the cross categorial definitions that we have given in extensional type logic in the context of IL. Partee and Rooth [1983] extend these definitions to IL in the following way:

In IL, the set of boolean types is defined as follows:
BOOL is the smallest set such that:
1. \( t \in BOOL \)
2. if \( a \in TYPE \) and \( b \in BOOL \) then \( <a,b> \in BOOL \)
3. if \( a \in BOOL \) then \( <s,a> \in BOOL \)

The cross categorial definitions that we have given before carry over to IL, but we have to add definitions of the connectives for the new boolean types:

Let \( a \in BOOL \), \( p,q \in VAR_{<s,a>} \):

\[
\begin{align*}
\not_{[<s,a>]} &= \lambda p. \not_{[a]}(\gamma p) \\
\and_{[<s,a>]} &= \lambda q \lambda p. \and_{[a]}(\gamma q)(\gamma p) \\
\or_{[<s,a>]} &= \lambda q \lambda p. \or_{[a]}(\gamma q)(\gamma p)
\end{align*}
\]

As before, in TY2 the definition of the connectives as we gave them before applies to the new boolean types as well, we get from the standard definition:

Let \( p,q \in VAR_{<s,t>}, w \in VAR_s \):

\[
\begin{align*}
\not_{[<s,t>]} &= \lambda p \lambda w. \not_{[a]}(p(w)) \\
\and_{[<s,t>]} &= \lambda q \lambda p \lambda w. \and_{[a]}(q(w))(p(w)) \\
\or_{[<s,t>]} &= \lambda q \lambda p \lambda w. \or_{[a]}(q(w))(p(w))
\end{align*}
\]

I give this definition, because it is standardly assumed in the literature. However, I will not incorporate it here. Rather, I will keep the definition of Boolean types as it was before in extensional type theory, hence all intensional types of the form \( <s,a> \) are non-boolean types, and that the generalized connectives are not defined for them. This means that, as for type \( e \), if we want to express conjunction, negation, disjunction for expressions of, say, type \( <s,t> \), we have to do it in another way.

The reason for differing from Partee and Rooth here is the following. If we assume that that the cross-categorial definition of the connectives is the one that gives us the standard interpretation of the connectives at different types, Partee and Rooth predict that, on their most standard interpretation (1) is equivalent to (2) rather than to (3):

(1) Mary believes that John walks or that John doesn’t walk.
(2) Mary believes that John walks or John doesn’t walk.
(3) Mary believes that John walks or Mary believes that John doesn’t walk.

(1) gets represented as (1’):

\[
\begin{align*}
\text{(1')} \ \& \ BELIEVE(m,[\not \WALK(j) \or_{[<s,t>]} \not\WALK(j)])
\end{align*}
\]

\[
\begin{align*}
\not \WALK(j) \or_{[<s,t>]} \not\WALK(j)] &= \not (\WALK(j) \or \not\WALK(j)] = \\
\WALK(j) \or \not\WALK(j)] &= \WALK(j) \or \not\WALK(j)]
\end{align*}
\]

Hence (1’) is equivalent to (2’):

\[
\begin{align*}
\text{(2')} \ \& \ BELIEVE(m,\WALK(j) \or \not\WALK(j)]
\end{align*}
\]
If we do not allow generalized disjunction at type \( <s,t> \), the grammar we will give later will generate *that John walks or that John doesn't walk* as \( \lambda P. P(\texttt{WALK}(j)) \lor P(\texttt{¬WALK}(j)) \), which will produce (3'):

\[
(3') \ \text{BELIEVE}(m, \texttt{WALK}(j)) \lor \text{BELIEVE}(m, \texttt{¬WALK}(j))
\]

Now, I am not saying that sentence (1) can never have a reading equivalent to (2). But I think this reading is marked and should be treated as a special interpretation procedure, rather than as the standard case. In the grammar that I will set up, Partee and Rooth's extension makes it unavoidably the standard case.

In sum: I will adopt for IL exactly the same definition of the boolean types (but for the IL types): \( t \in \text{BOOL} \), if \( a \in \text{TYPE} \) and \( b \in \text{BOOL} \) then \( <a,b> \in \text{BOOL} \). The definitions of the connectives are just as before. This means that we *are* extending the definitions to IL. For instance, we now have and\([<<s,t>,t>,t>]\) because \( <<<s,t>,t>,t> \in \text{BOOL} \). But we do not have and\([<s,t>]\), because in IL, \( <s,t> \) is not in \( \text{BOOL} \). The reason is that in IL \( s \) is not a type.