Montague's Theory of Translation: The Notion of a 'Translation Base' [1]

1. Review of What We Have and Where We Want to Go

(1) **Our Key Ingredients**

a. <u>Two Disambiguated Languages:</u>
Both the languages below are such that every expression in the language is either (i) a basic expression, or (ii) formed in exactly one way from the syntactic operations, but (iii) *not both*.

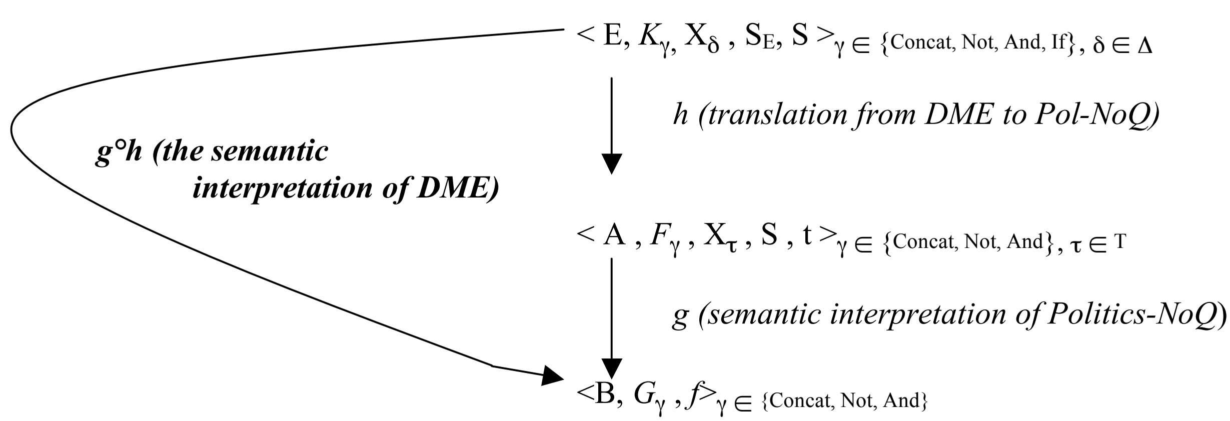
(i) *Politics-NoQ*
The structure $< A \, , F_\gamma \, , X_\tau \, , S \, , t >_{\gamma \in \{\text{Concat, Not, And}\}, \, \tau \in T}$ where the algebra $< A, F_\gamma >_{\gamma \in \{\text{Concat, Not, And}\}}$, and the sets X_τ and S are as before.

(ii) *Disambiguated Mini-English (DME)*
The structure The structure $< E, K_\gamma, X_\delta \, , S_E, S >_{\gamma \in \{\text{Concat, Not, And, If}\}, \, \delta \in \Delta}$, where algebra $<E, K_\gamma>_{\gamma \in \{\text{Concat, Not, And, If}\}}$, and the sets X_δ , S_E as before.

b. <u>An Interpretation for Politics-NoQ</u>
Let the set S = { Michelle, Barack, Mitt }. Let **B** = $<B, G_\gamma \, , f>_{\gamma \in \{\text{Concat, Not, And}\}}$ be the Fregean interpretation based on S, such that *f* is as defined before.

(2) **What We Want to Make from Those Ingredients**
We want to develop a way of homomorphically mapping expressions of DME to expressions of Politics-NoQ (so that we can ultimately get a semantics for English)

<u>Indirect Interpretation in a Picture (Oversimplified):</u>



[1] These notes are based upon material in the following readings: Halvorsen & Ladusaw (1979), Dowty *et al.* (1981) Chapter 8, and Thomason (1974) Chapter 7 (Montague's "Universal Grammar").

Although we've come closer to our goal by restricting our attention to 'disambiguated languages', there are still two key problems facing our project…

(3) **Critical Problem 1:**

- If translation h is to be a homomorphism from DME to a syntactic algebra for P-NoQ, then there must be a syntactic operation *OP* in the latter that 'corresponds' to K_{Concat}.

- Moreover, under this correspondence, it must be that:

  h(K_{Concat}.(*Barack*, *smokes*)) =
  OP (h(*Barack*), h(*smokes*)) =
  OP (**barack'** , **smokes'**) = (**smokes' barack'**)

  h(K_{Concat}.(*loves*, *Barack*)) =
  OP (h(*loves*), h(*Barack*)) =
  OP (**loves'** , **Barack'**) = (**loves' barack'**)

- **But this seems inconsistent!** How can *OP*(**barack'** , **smokes'**) = (**smokes' barack'**), while *OP*(**loves', barack'**) = (**loves' barack'**)???

(4) **Critical Problem 2 (New):**

- Our syntactic algebra for DME contains the operation K_{If}.

- Again, if h is to be a homomorphism from DME to a syntactic algebra for P-NoQ, there must be some syntactic operation in the latter that 'corresponds' to K_{If}

- **But there isn't any!**

<u>The Plan:</u>
We'll go halfway to fixing the problem in (3); at which point, the problems in (3) and (4) will become the same. Then we'll solve that more general problem by introducing a new, central idea of Montague's: **the Translation Base.**

---

2. Prolegomena: A Slight Change to Our Definition of Disambiguated Mini-English

We're going to introduce a slight change to our definition of DME…
It will seem ad hoc for now, but we'll see independent motivation later on (with quantification)…

(5) **Step One: The Category Labels**
The syntactic categories of Disambiguated Mini-English will be just the same as before:

Δ = {NP, IV, TV, S}

(6) **The Basic Expressions**
The basic expressions of DME will be just the same as before, too.

a. X_{NP} = { < *Barack*, ∅>, < *Michelle*, ∅>, < *Mitt*, ∅> }
b. X_{IV} = { < *smokes*, ∅> }
c. X_{TV} = { < *loves*, ∅> }
d. X_S = ∅

(7) **The Syntactic Operations**
Our set of syntactic operations for DME is going to be altered. We're going to split K_{Concat} into two different operations: $K_{Merge\text{-}S}$ and $K_{Merge\text{-}IV}$.

- In the definitions below, α and β are trees whose root nodes are ordered pairs. In addition α' and β' are the first members of the root nodes of α and β (respectively).

a. $K_{Merge\text{-}S}(\alpha, \beta)$ = < α' β' , Merge-S >
   α β

b. $K_{Merge\text{-}IV}(\alpha, \beta)$ = < α' β' , Merge-IV >
   α β

c. $K_{Not}(\alpha)$ = < *it is not the case that* α' , Not >
   |
   α

d. $K_{And}(\alpha, \beta)$ = < α' and β' , And >
   α β

e. $K_{If}(\alpha, \beta)$ = < If α' then β' , If >
   α β

Right now, the only difference between $K_{Merge\text{-}S}$ and $K_{Merge\text{-}IV}$ is the index on the root of the output. ***Again, later on these operations will become more substantively different.***

(8) **The Syntactic Algebra**
E is the smallest set such that:
a. For all $\delta \in \Delta$, $X_\delta \subseteq E$.
b. E is closed under the operations $K_{Merge\text{-}S}$, $K_{Merge\text{-}IV}$, K_{Not}, K_{And}, and K_{If}

With the changes to our syntactic operations in (7) come some concomitant changes to our syntactic rules…

(9) **The Syntactic Rules**
We can retain much the same set of syntactic rules S_E that we had before:

a. $< K_{\text{Merge-IV}}, < \text{TV}, \text{NP} >, \text{IV} >$
b. $< K_{\text{Merge-S}}, < \text{NP}, \text{IV} >, \text{S} >$
c. $< K_{\text{And}} , <\text{S} , \text{S}> , \text{S} >$
d. $< K_{\text{If}} , <\text{S} , \text{S}> , \text{S} >$
e. $< K_{\text{Not}}, <\text{S}>, \text{S} >$

(10) **New Definition for Disambiguated Mini-English**
The structure $< \text{E}, K_\gamma, \text{X}_\delta , \text{S}_\text{E}, \text{S} >_{\gamma \in \{\text{Merge-S, Merge-IV, Not, And, If}\}, \delta \in \Delta}$ where E, K_γ, X_δ , S_E, and Δ are as defined in (38)-(42).

Illustrative Structure:

< *Barack loves Michelle*, Merge-S >
- < *Barack* , ∅ >
- < *loves Michelle*, Merge-IV >
  - < *loves* ∅ >
  - < *Michelle*, ∅ >

(11) **Half of Problem (3) Solved**
Now we don't need to find a single operation in Politics-NoQ that corresponds to K_{Concat}

- In particular, we can now assume without problem that $K_{\text{Merge-IV}}$ corresponds to F_{Concat}
  h($K_{\text{Merge-IV}}$(*loves*, *Michelle*)) =
  F_{Concat} (h(*loves*), h(*Michelle*)) =
  F_{Concat} (**loves'** , **michelle'**) = (**loves' michelle'**)

(12) **Remaining Problem**

- Now, however, we have this additional operation $K_{\text{Merge-S}}$. **And, there doesn't seem to be an operation in Politics-NoQ that corresponds to it.**

- **And, we still need to find an operation in Politics-NoQ that corresponds to K_{If}!**

(13) **Key Idea Behind the Solution to (12)**
Because of K_{If} and $K_{\text{Merge-S}}$, there won't be a homomorphism between the syntactic algebras $< \text{E}, K_\gamma>_{\gamma \in \{\text{Merge-S, Merge-IV, Not, And, If}\}}$ and $< \text{A} , F_\gamma>_{\gamma \in \{\text{Concat, Not, And}\}}$

- **But maybe we can make a new algebra from $< \text{A} , F_\gamma>_{\gamma \in \{\text{Concat, Not, And}\}}$ that *will* fit the bill!!!**

3. Polynomial Operations and Derived Syntactic Rules

(14) **Road Map for This Section**
In this section, we will see how to do the following:

a. From an algebra <A, F_γ>$_{\gamma \in \Gamma}$, create new complex operations from the operations $\{ F_\gamma \}_{\gamma \in \Gamma}$.

b. From a (disambiguated) language <A, F_γ , X_δ , S, δ_0>$_{\gamma \in \Gamma , \delta \in \Delta}$, create new complex syntactic rules from the rules S.

This will provide us with the key tools for introducing Montague's notion of a 'translation base'.

Advisory:
In my own view, the technical concept of a 'translation base' is the least *a priori* intuitive ingredient of Montague's theory…

- Once you see what the thing does – and how it does what it does – it becomes easier to get your mind around…
- But, it's hard to build up piece-by-piece in a completely intuitive way…
- **Thus, please have faith that we're going somewhere interesting with all this…**

3.1 Polynomial Operations Over an Algebra

In this section, we'll cover the goal in (14a). The principle means of creating a complex operation from simpler operations is *function composition* (Handout 1), repeated below.

(15) **(Generalized) Function Composition**
Let g be an n-ary function, and let f_1, …, f_n be a series of n m-ary functions. The *composition of g and f_1, …, f_n* is the m-ary function defined as follows:

g<f_1, …, f_n> $=_{def}$ the m-ary function such that for any m-ary sequence a_1, … a_m
g<f_1, …, f_n>(<a_1, … a_m>) =
g(f_1(<a_1, … a_m>), …, f_n(<a_1, … a_m>))

Illustration:

Let g = { <<x,y>,z> : z = x + y }, f = { <x,y> : y = x – 1 }, h = { <x,y> : y = x + 2 }

Then: g<f,h>(2) = g(f(2),h(2)) = g(1,4) = 5

g<f,h> = { <x,y> : y = (x-1) + (x+2) }

In addition to function composition, we'll also make use of the special functions in (16) and (17).

(16) **Identity Functions**

The function $Id_{n,m}$ takes as argument an m-tuple α and returns the nth member of α

a.	$Id_{1,1}(a)$	=	a			
	$Id_{1,1}(b)$	=	b			
b.	$Id_{1,2}(a,b)$	=	a	$Id_{1,2}(c,d)$	=	c
	$Id_{2,2}(a,b)$	=	b	$Id_{2,2}(c,d)$	=	d
c.	$Id_{1,3}(a,b,c)$	=	a	$Id_{1,3}(c,d,e)$	=	c
	$Id_{2,3}(a,b,c)$	=	b	$Id_{2,3}(c,d,e)$	=	d
	$Id_{3,3}(a,b,c)$	=	c	$Id_{3,3}(c,d,e)$	=	e

(17) **Constant Functions**

The function $C_{\alpha, m}$ takes as argument an m-tuple β, and for any such m-tuple β, returns α

a.	$C_{a,1}(a)$	=	a	$C_{b,1}(a)$	=	b
	$C_{a,1}(b)$	=	a	$C_{b,1}(b)$	=	b
b.	$C_{a,2}(a,b)$	=	a	$C_{b,2}(a,b)$	=	b
	$C_{a,2}(c,d)$	=	a	$C_{b,2}(c,d)$	=	b
c.	$C_{a,3}(a,b,c)$	=	a	$C_{b,3}(a,b,c)$	=	b
	$C_{a,3}(c,d,e)$	=	a	$C_{b,3}(c,d,e)$	=	b

With these ingredients in place, we can introduce the key concept in (18).

(18) **The Polynominal Operations Over an Algebra**

Let $\mathbf{A} = \langle A, F_\gamma \rangle_{\gamma \in \Gamma}$ be an algebra. The class of polynomial operations over **A** is the smallest class K such that the following all hold.

a. $F_\gamma \in K$ for all $\gamma \in \Gamma$
<u>Note:</u> The polynomial operations over **A** include all the operations in **A**

b. $Id_{n,m} \in K$, for all n, m $\in \mathbb{N}$
<u>Note:</u> Every possible identity function is also a polynomial operation over **A**.

c. $C_{a,m} \in K$, for all $a \in A$ and $m \in \mathbb{N}$
<u>Note:</u> Every possible constant function (to A) is also a polynomial op. over **A.**

d. If G is an n-ary function in K, and $F_1, \ldots F_n$ are n m-ary functions in K, then $G\langle F_1, \ldots F_n \rangle \in K$
<u>Note:</u> The polynomial operations over **A** are closed under function composition.

(19) **Remark**
So, in other words, F is a polynomial operation over $\mathbf{A}$ (= $<A, F_\gamma>_{\gamma \in \Gamma}$) if any of the following hold:

a. F is one of the operations $\{ F_\gamma \}_{\gamma \in \Gamma}$.

b. F is an identity function.

c. F is a constant function (to A).

d. You can obtain F via iterated function composition from either (a), (b), or (c).

We'll now illustrate the key concept in (18) by looking at some polynomial operations over the syntactic algebra for Politics-NoQ, $<A, F_\gamma>_{\gamma \in \{\text{Concat, Not, And}\}}$.

(20) $\mathbf{C_{(smokes'\ barack'), 1}}$
This function takes any expression in A and returns the expression (**smokes' barack'**)

$C_{(smokes'\ barack'), 1}$ ((**loves' michelle'**)) = (**smokes' barack'**)
$C_{(smokes'\ barack'), 1}$ (**mitt'**) = (**smokes' barack'**)

(21) $\mathbf{Id_{2,3}}$ This function takes any triple in A and returns the second member.

(22) $\mathbf{Id_{1.2}}$ This function takes any pair in A and returns the first member.

(23) $\mathbf{Id_{2.2}}$ This function takes any pair in A and returns the second member.

(24) $\boldsymbol{F_{Not}<F_{Not}>}$
This function takes any expression in A and returns its double negation.

$F_{Not}<F_{Not}>$((**smokes' barack'**)) =
F_{Not}(F_{Not}((**smokes' barack'**))) =
~~(**smokes' barack'**)

(25) $\boldsymbol{F_{And}< C_{(smokes'\ barack'), 1}, Id_{1,1} >}$
This function takes any expression α in A and returns the conjunction of (**smokes' barack'**) with α.

$F_{And}< C_{(smokes'\ barack'), 1}, Id_{1,1} >$((**smokes' mitt'**)) =
F_{And}($C_{(smokes'\ barack'), 1}$((**smokes' mitt'**)), $Id_{1,1}$((**smokes' mitt'**))) =
F_{And}((**smokes' barack'**)), (**smokes' mitt'**)) =
((**smokes' barack'**) & (**smokes' mitt'**))

(26) $\boldsymbol{F_{Not}}$**<**$\boldsymbol{F_{And}}$**<Id**$_{\mathbf{1,2}}$**,** $\boldsymbol{F_{Not}}$**<Id**$_{\mathbf{2,2}}$**>>>**
This function takes any pair of expressions α, β in A and returns ~(α & ~β).

F_{Not}<F_{And}<Id$_{1,2}$, F_{Not}<Id$_{2,2}$>>>((**smokes' barack'**), (**smokes' mitt'**)) =

F_{Not}(F_{And}(Id$_{1,2}$((**smokes' barack'**), (**smokes' mitt'**))
F_{Not}(Id$_{2,2}$((**smokes' barack'**), (**smokes' mitt'**))) …) =

F_{Not}(F_{And}((**smokes' barack'**), F_{Not}((**smokes' mitt'**)) …) =

F_{Not}(F_{And}((**smokes' barack'**), ~(**smokes' mitt'**)) …) =

F_{Not}(((**smokes' barack'**) & ~(**smokes' mitt'**))) =

~((**smokes' barack'**) & ~(**smokes' mitt'**))

(27) **Key Observation**
Recall that the formulae (φ → ψ) and ~(φ & ~ψ) are logically equivalent.

- Consequently, we are viewing (φ → ψ) as a special 'abbreviation' for ~(φ & ~ψ)
- **Thus, the operation** $\boldsymbol{F_{Not}}$**<**$\boldsymbol{F_{And}}$**<Id**$_{\mathbf{1,2}}$**,** $\boldsymbol{F_{Not}}$**<Id**$_{\mathbf{2,2}}$**>>> would seem to be a good 'translational correspondent' of** $\boldsymbol{K_{\mathrm{If}}}$ **in Disambiguated Mini-English**

h(K_{If} (α, β)) = F_{Not}<F_{And}<Id$_{1,2}$, F_{Not}<Id$_{2,2}$>>>(h(α), h(β)) = ~(h(α) & ~h(β)) = (h(α) → h(β))

(28) $\boldsymbol{F_{\mathrm{Concat}}}$**<Id**$_{\mathbf{2,2}}$**, Id**$_{\mathbf{1,2}}$**>**
This function takes any pair of expressions α, β in A and returns (β α)

F_{Concat}<Id$_{2,2}$, Id$_{1,2}$>(**barack'**, **smokes'**) =

F_{Concat}(Id$_{2,2}$(**barack'**, **smokes'**) , Id$_{1,2}$(**barack'**, **smokes'**)) =

F_{Concat}(**smokes'** , **barack'**) =

(**smokes' barack'**)

(29) **Key Observation**
It seems like F_{Concat}<Id$_{2,2}$, Id$_{1,2}$> would be a good 'translational correspondent' of $K_{\mathrm{Merge\text{-}S}}$ in Disambiguated Mini-English.

h($K_{\mathrm{Merge\text{-}S}}$ (*Barack*, *smokes*)) = F_{Concat}<Id$_{2,2}$, Id$_{1,2}$>(h(*Barack*), h(*smokes*)) =

F_{Concat}<Id$_{2,2}$, Id$_{1,2}$>(**barack'**, **smokes'**) = (**smokes' barack'**)

(30) **Summary Observation**

- If we look to the **polynomial operations over** the syntactic algebra for Politics-NoQ, $<A, F_\gamma>_{\gamma \in \{\text{Concat, Not, And}\}}$, we will find syntactic operations over A that could viably correspond to the syntactic operations K_{If} and $K_{\text{Merge-S}}$ over E
- *How, though, does this help us in our quest for a homomorphism from E to A?...*

(31) **Polynomial Operations and Algebras**

a. Key Fact:
Let **A** be an algebra $<A, F_\gamma>_{\gamma \in \Gamma}$. If the set $\{ H_{\gamma'} \}_{\gamma' \in \Gamma'}$ consists of polynomial operations over **A**, then A is closed under $\{ H_{\gamma'} \}_{\gamma' \in \Gamma'}$.

(proof left as an exercise to the student)

b. Key Consequence:
Let **A** be an algebra $<A, F_\gamma>_{\gamma \in \Gamma}$, and let $\{ H_{\gamma'} \}_{\gamma' \in \Gamma'}$ consist of polynomial operations over **A**. The structure $<A, H_\gamma>_{\gamma' \in \Gamma'}$, is an algebra.

Thus, given (31b), if $<A, F_\gamma>_{\gamma \in \{\text{Concat, Not, And}\}}$ is the syntactic algebra for Politics-NoQ, then the following is also an algebra: $<A, H_\gamma>_{\gamma \in \{\text{Merge-S, Merge-IV, Not, And, If}\}}$, where

- H_{Not} and H_{And} = F_{Not} and F_{And}, respectively
- $H_{\text{Merge-IV}}$ = F_{Concat}
- H_{If} = $F_{Not}<F_{And}<\text{Id}_{1,2}, F_{Not}<\text{Id}_{2,2}>>>$
- $H_{Merge\text{-}S}$ = $F_{\text{Concat}}<\text{Id}_{2,2}, \text{Id}_{1,2}>$

And, it seems like it might be possible to have a homomorphism from the syntactic algebra for DME $<E, K_\gamma>_{\gamma \in \{\text{Merge-S, Merge-IV, Not, And, If}\}}$ to $<A, H_\gamma>_{\gamma \in \{\text{Merge-S, Merge-IV, Not, And, If}\}}$

Before we can use all of this to lay out a theory of homomorphic translation between languages, we also need to introduce a way of constructing complex syntactic rules (14b)*...*

3.2 The Derived Syntactic Rules of a Language L

(32) **Background Motivation**

- Intuitively, a translation from one language *L* to another language *L'* should always map the well-formed expressions of *L* to well-formed expressions of *L'*
- As we'll soon see, one way of ensuring this appeals to the notion of a 'derived syntactic rule', defined in this section.
- **For reasons that will also be clear shortly, this definition is going to closely mirror the definition for the polynomial operations over an algebra**

(33) **Derived Syntactic Rules of a Language**

Let **L** be a language $<< A, F_\gamma, X_\delta, S, \delta_0>_{\gamma \in \Gamma, \delta \in \Delta}, R>$. The derived syntactic rules of **L** is the smallest class K such that the following hold:

a. $S \subseteq K$
Note: Thus all the syntactic rules of **L** are also 'derived syntactic rules'

b. For all n, m $\in \mathbb{N}$ such that $n \leq m$, if $<\delta_1, \ldots, \delta_n, \ldots, \delta_m>$ is an m-tuple of elements from Δ, then $< Id_{n,m}, <\delta_1, \ldots, \delta_n, \ldots, \delta_m>, \delta_n> \in K$

Note:
This means that all the logically possible syntactic rules of the form below are also 'derived syntactic rules':

$< Id_{2,4}, < e, t, <e,<e,t>>, <e,t> >, t >$
'The result of applying $Id_{2,4}$ to a quadruple consisting of an expression of type e, one of type t, one type $<e,<e,t>>$ and one of type $<e,t>$ is an expression of type t.'

c. For all $n \in \mathbb{N}$, if $a \in C_\delta$, and $<\delta_1, \ldots, \delta_n>$ is an n-tuple of elements from Δ, then the triple $< C_{a,n}, <\delta_1, \ldots, \delta_n>, \delta > \in K$.

Note:
This means that all the logically possible syntactic rules of the form below are also 'derived syntactic rules'.

$< C_{(\text{smokes' barack'}), 2}, < <e,t>, <e,t> >, t >$
'The result of applying $C_{(\text{smokes' barack'}), 2}$ to a pair consisting of an expression of type $<e,t>$ and an expression of type $<e,t>$ is an expression of type t.'

The fourth and final condition on the derived syntactic rules basically amounts to them being closed under 'composition'… it is rather complex to state formally, however…

d. If $\langle F, \langle \delta_1, \ldots \delta_n \rangle, \delta \rangle \in K$, F is an n-ary operation, and each of $G_1, \ldots G_n$ are an m-ary operation such that $\langle G_j, \langle \delta'_1, \ldots \delta'_m \rangle, \delta_j \rangle \in K$, then the following is also a member of K:

$$\langle F\langle G_1, \ldots G_n \rangle, \langle \delta'_1, \ldots \delta'_m \rangle, \delta \rangle$$

<u>Note:</u> To get a sense of how this 'composition' operation on rules works, consider that (i) and (ii) are rules in our language Politics-NoQ.

(i) $\langle F_{\text{Not}}, \langle t \rangle, t \rangle$
(ii) $\langle F_{\text{Concat}}, \langle \langle e,t \rangle, e \rangle, t \rangle$

Thus, definition (33) would entail that the following is a 'derived rule' of Politics-NoQ.

(iii) $\langle F_{\text{Not}}\langle F_{\text{Concat}} \rangle, \langle \langle e,t \rangle, e \rangle, t \rangle$
'The result of applying $F_{\text{Not}}\langle F_{\text{Concat}} \rangle$ to a pair consisting of an expression of type $\langle e,t \rangle$ and an expression of type e is an expression of type t.'

Note, too, that this derived rule would intuitively 'be true of' for Politics-NoQ.

- This raises the following key generalization…

(34) **Derived Syntactic Rules and the Syntactic Categories of a Language**
Let **L** be a language $\langle\langle A, F_\gamma, X_\delta, S, \delta_0 \rangle_{\gamma \in \Gamma, \delta \in \Delta}, R \rangle$, and let $\langle H, \langle \delta_1, \ldots \delta_n \rangle, \delta \rangle$ be a derived syntactic rule of **L**.

a. <u>Claim:</u> If $\varphi_1, \ldots \varphi_n$ are such that each $\varphi_i \in C_{\delta i}$, then $H(\varphi_1, \ldots \varphi_n) \in C_\delta$
(That is, the derived syntactic rules will only ever generate 'meaningful expressions' of a language.)

b. <u>Proof:</u> (left as an exercise to the student)

4. The Concept of a Translation Base

In the previous section, we developed the tools below. We also developed them so that they mirror one another.

(35) **Polynomial Operations (Over an Algebra)**
A way of taking algebra $\langle A, F_\gamma \rangle_{\gamma \in \Gamma}$ and creating new complex operations $\{H_{\gamma'}\}_{\gamma' \in \Gamma'}$ from the operations $\{F_\gamma\}_{\gamma \in \Gamma}$ such that $\langle A, H_{\gamma'} \rangle_{\gamma' \in \Gamma'}$ is also an algebra.

(36) **Derived Syntactic Rules**

A way of taking a language $<A, F_\gamma , X_\delta , S, \delta_0>_{\gamma \in \Gamma , \delta \in \Delta}$, and creating new complex syntactic rules S' *from the polynomial operations over* $<A , F_\gamma>_{\gamma \in \Gamma}$.

We're now going to use these tools to construct Montague's general theory of translation…

To do this, let's first consider some ideal properties of a translation function h from one language $<A , F_\gamma , X_\delta , S, \delta_0>_{\gamma \in \Gamma , \delta \in \Delta}$ to another language $< A' , F'_{\gamma'} , X'_{\delta'} , S', \delta'_0>_{\gamma' \in \Gamma' , \delta' \in \Delta'}$

(37) **Correspondence Between the Syntactic Categories**

In Montague's theory of translation, there must be a consistent mapping from the syntactic categories of *L* to the syntactic categories of *L'*.

- That is, if $\delta \in \Delta$, then there must be a corresponding $\delta' \in \Delta'$ such that if $\varphi \in C_\delta$, then $h(\varphi) \in C_{\delta'}$
- For example, thinking of our languages DME and Politics-NoQ, such a mapping of the categories would be as follows:

  NP → e
  TV → <e,<e,t>>
  IV → <et>
  S → t

- The reason why such a mapping is needed is ultimately tied to Montague's (final) definition of a 'Fregean Interpretation'… *Just go with it for now…*

Consequence: A translation from *L* to *L'* will need to specify a function g from Δ to Δ'

(38) **Polynomial Operations**

If we want the translation h: A → A' from *L* to *L'* to be a homomorphism, then we're going to need to find some operations F*'* to correspond with the syntactic operations of *L*.

- We've already seen that in the general case the basic operations $\{ F'_{\gamma'} \}_{\gamma' \in \Gamma'}$ of *L'* are not going to be sufficient.
- We've also already seen that the polynomial operations over the syntactic algebra for *L'*, $< A' , F'_{\gamma'} >_{\gamma' \in \Gamma'}$ can supply us with such operations.

Consequence:

A translation from *L* to *L'* must identify some polynomial operations $\{ H_\gamma \}_{\gamma \in \Gamma}$ over the syntactic algebra for *L'* $< A' , F'_{\gamma'} >_{\gamma' \in \Gamma'}$

(39) **Derived Syntactic Rules**
A translation h from *L* to *L'* should always map the 'meaningful expressions' of *L* to meaningful expressions in *L'*.

- Now, recall that we're going to want h to be a homomorphism, where every syntactic operation F_γ in *L* corresponds with some polynomial operation H_γ over the syntactic algebra of *L'*.

$$h(\ F_\gamma(\alpha_1, \dots \alpha_n)\ ) \qquad = \qquad H_\gamma(\ h(\alpha_1), \dots h(\alpha_n)\ )$$

- Consequently, we will want it to be that if $F_\gamma(\alpha_1, \dots \alpha_n)$ is a meaningful expression of *L*, then $H_\gamma(\ h(\alpha_1), \dots h(\alpha_n)\ )$ is a meaningful expression of *L'* too.

<u>Consequence:</u>
Under a translation h from *L* to *L'*, if F_γ in *L* corresponds with H_γ (a polynomial operation over *L'*), then if (a) is a **syntactic rule** of *L*, then (b) is a **derived syntactic rule** of *L'*

a. $< F_\gamma , <\delta_1 , \dots \delta_n>, \delta >$
b. $< H_\gamma , <g(\delta_1), \dots g(\delta_n)>, g(\delta) >$

- To see how the condition above works, recall the general result in (34): if the tuple $<H_\gamma , <g(\delta_1), \dots g(\delta_n)>, g(\delta) >$ is a derived syntactic rule, and $\varphi_1, \dots \varphi_n$ are such that each $\varphi_i \in C_{g(\delta i)}$, then $H(\varphi_1, \dots \varphi_n) \in C_{g(\delta)}$

  - Now, suppose that α is a meaningful expression of *L*, and $F_\gamma(\alpha_1, \dots \alpha_n) = \alpha$, where $\alpha_1 \in \delta_1, \dots, \alpha_n \in \delta_n$

  - Thus, the translation $h(\alpha) = h(F_\gamma(\alpha_1, \dots \alpha_n)) = H_\gamma(h(\alpha_1), \dots h(\alpha_n))$

  - Now, given our category correspondence (37), it follows that $h(\alpha_1) \in C_{g(\delta 1)}$, …, $h(\alpha_n) \in C_{g(\delta n)}$

  - Therefore, from our general result in (34) – and the fact that (b) is a derived rule of *L'* – it follows that $H_\gamma(h(\alpha_1), \dots h(\alpha_n)) \in C_{g(\delta)}$

  - **Thus, we have it that $h(\alpha)$ is also a meaningful expression of *L'* !**

With all of these ingredients on the table, we can now provide Montague's general definition of a 'translation base'…

(40) **Translation Base from Language L to Language L'**
Let **L** be a language <*L*, R> and **L'** be a language <*L'*, R'>, where *L* is the disambiguated language $<A, F_\gamma, X_\delta, S, \delta_0>_{\gamma \in \Gamma, \delta \in \Delta}$ and $L' = <A', F'_{\gamma'}, X'_{\delta'}, S', \delta'_0>_{\gamma' \in \Gamma', \delta' \in \Delta'}$

A **translation base from L to L'** is a structure $<g, H_\gamma, j>_{\gamma \in \Gamma}$ such that:

a. g is a function from Δ to Δ' (37)

b. For all $\gamma \in \Gamma$, H_γ is a polynomial operation over the algebra $<A', F'_{\gamma'}>_{\gamma' \in \Gamma'}$ sharing the same arity as F_γ (38)

c. If $<F_\gamma, <\delta_1, \dots \delta_n>, \delta> \in S$, then the following is a derived syntactic rule for **L'**: $<H_\gamma, <g(\delta_1), \dots g(\delta_n)>, g(\delta)>$ (39)

d. j is a function whose domain is $\cup_{\delta \in \Delta} X_\delta$, and whenever $\varphi \in X_\delta$, $j(\varphi) \in C'_{g(\delta)}$.

<u>Note:</u> j is a function that maps the basic expressions of **L** of category δ to some meaningful expressions of **L'** of the corresponding category g(δ).

With the notion of a translation base, we can construct the following definition of a translation function…

(41) **Translation Function from Language L to Language L'**
Let **L** be a language <*L*, R> and **L'** be a language <*L'*, R'>, where *L* is the disambiguated language $<A, F_\gamma, X_\delta, S, \delta_0>_{\gamma \in \Gamma, \delta \in \Delta}$ and $L' = <A', F'_{\gamma'}, X'_{\delta'}, S', \delta'_0>_{\gamma' \in \Gamma', \delta' \in \Delta'}$
Let **T** be a translation base $<g, H_\gamma, j>_{\gamma \in \Gamma}$ from **L** to **L'**

The **translation function determined by T** is the unique homomorphism k from the algebra $<A, F_\gamma>_{\gamma \in \Gamma}$ to the algebra $<A', H_\gamma>_{\gamma \in \Gamma}$ such that $j \subseteq k$.

(42) **Remarks**

- That any translation base **T** determines such a homomorphism k is essentially guaranteed by the conditions we placed on the polynomial operations $\{ H_\gamma \}_{\gamma \in \Gamma}$
- If k is a translation function from **L** to **L'**, then k is not *necessarily* a homomorphism from the syntactic algebra of **L** to the syntactic algebra of **L'**
  - Rather, it's a homomorphism to an algebra we define on the basis of **L'**
  - This allows translation to be a homomorphism even if two language algebras are not themselves homomorphic!

(43) **Definition of Translation**
Let **L** be a language $<L, R>$ and **L'** be a language $<L', R'>$, where L is the disambiguated language $<A , F_\gamma , X_\delta , S, \delta_0>_{\gamma \in \Gamma , \delta \in \Delta}$ and $L' = <A' , F'_{\gamma'} , X'_{\delta'} , S', \delta'_0>_{\gamma' \in \Gamma' , \delta' \in \Delta'}$
Let **T** be a translation base from **L** to **L'**, and let k be the translation function determined by **T**.

If α is an expression of **L'** and β is an expression of **L**, then α **is a translation of** β if there are $\alpha' \in A'$ and $\beta' \in A$ such that:
(i) $\alpha'R'\alpha$ and $\beta'R\beta$
(ii) α' is a meaningful expression of L' and β' is a meaningful expression of L
(iii) $k(\beta') = \alpha'$

---

5. Translating from Mini-English to Politics-NoQ

To round out these notes, we'll use all these tools to spell out a translation base and (homomorphic) translation function from Mini-English to Politics-NoQ

(44) **Mini-English**
Mini-English is the structure $< < E, K_\gamma, X_\delta , S_E, S >_{\gamma \in \{\text{Merge-S, Merge-IV, Not, And, If}\}, \delta \in \Delta}, R>$, where

a. The structure $<E, K_\gamma, X_\delta , S_E, S>_{\gamma \in \{\text{Merge-S, Merge-IV, Not, And, If}\}, \delta \in \Delta}$ is Disambiguated Mini-English, as defined in (10).

b. R is a function which takes as input a tree T in E, and returns as output the first member of the root node of T.

(45) **Politics-NoQ**
Politics-NoQ is the structure $<<A , F_\gamma , X_\tau , S , t >_{\gamma \in \{\text{Concat, Not, And}\}, \tau \in T} , R'>$, where:

a. The structure $<A , F_\gamma , X_\tau , S , t >_{\gamma \in \{\text{Concat, Not, And}\}, \tau \in T}$ is the disambiguated language Politics-NoQ, as defined previously.

b. R is the identity function.

We'll now lay out each of the three main ingredients for a translation base from Mini-English to Politics-NoQ…

(46) **Correspondence Between the Syntactic Categories**
Given our discussion in (37), let us define the function g: $\Delta \rightarrow$ T as follows:

$g(NP) = e$ $g(TV) = <e, <e,t>>$ $g(IV) = <e,t>$ $g(S) = t$

(47) **The Polynomial Operations**
Given our discussion below (31), we will want to build our translation base from the following polynomial operations over the algebra $<A, F_\gamma>_{\gamma \in \{\text{Concat, Not, And}\}}$.

a. H_{Not} and H_{And} = F_{Not} and F_{And}, respectively

b. $H_{\text{Merge-IV}}$ = F_{Concat}

c. H_{If} = $F_{Not}<F_{And}<\text{Id}_{1,2}, F_{Not}<\text{Id}_{2,2}>>>$

d. $H_{\text{Merge-S}}$ = $F_{\text{Concat}}<\text{Id}_{2,2}, \text{Id}_{1,2}>$

(48) **Crucial Step We Will Usually Leave Implicit**
If we build our translation base on the polynomial operations above, then condition (39)/(40c) – combined with our category correspondence in (46) – requires the following:

a. $< H_{\text{Not}}, <t>, t >$ is a derived rule of Politics-NoQ

b. $< H_{\text{And}}, <t, t>, t >$ is a derived rule of Politics-NoQ

c. $< H_{\text{Merge-IV}}, < <e,<e,t>>, e >, <e,t> >$ is a derived rule of Politics-NoQ

d. $< H_{\text{If}}, <t, t>, t>$ is a derived rule of Politics-NoQ

e. $< H_{\text{Merge-S}}, <e, <et>>, t>$ is a derived rule of Politics-NoQ

Showing that (48a-e) hold will be left as an exercise for the student. Note that (48a,b,c) are trivial; only (48d,e) require some calculating out…

(49) **The (Lexical Translation) Function j**
Given the category correspondence in (46) – and what we obviously want to achieve – let us define the function j as follows:

a. j(< *Barack* , ∅ >) = **barack'**
b. j(< *Michelle* , ∅ >) = **michelle'**
c. j(< *Mitt* , ∅ >) = **mitt'**
d. j(< *smokes* , ∅ >) = **smokes'**
e. j(< *loves* , ∅ >) = **loves'**

(50) **Putting It All Together: The Translation Base from Mini-English to Politics-NoQ**

Let **T** be the structure $< g , H_\gamma , j >_{\gamma \in \{\text{Merge-S, Merge-IV, Not, And, If}\}}$, where g, H_γ , and j are as defined in (46)-(49). **T** is a translation base from Mini-English to Politics-NoQ.

*We can now use the translation function k determined by **T** to homomorphically map expressions of mini-English to expressions of Politics-NoQ.*

(51) **Translating from Mini-English to Politics-NoQ**

Let k be the translation function determined by **T**, as defined in (50). Let *T* be the tree in (10) above.

a. k(*T*) = (by definition of DME)

b. k($K_{\text{Merge-S}}$(< *Barack* , ∅ >, $K_{\text{Merge-IV}}$(< *loves* ∅ >, < *Michelle*, ∅ >))) = (by homomorphism property of k)

c. $H_{\text{Merge-S}}$(k(< *Barack* , ∅ >) , k($K_{\text{Merge-IV}}$(< *loves* ∅ >, < *Michelle*, ∅ >))) = (by homomorphism property of k)

d. $H_{\text{Merge-S}}$(k(< *Barack* , ∅ >) , $H_{\text{Merge-IV}}$(k(< *loves* ∅ >), k(< *Michelle*, ∅ >))) = (by definition of k and j)

e. $H_{\text{Merge-S}}$(j(< *Barack* , ∅ >) , $H_{\text{Merge-IV}}$(j(< *loves* ∅ >), j(< *Michelle*, ∅ >))) = (by definition of j)

f. $H_{\text{Merge-S}}$(**barack’** , $H_{\text{Merge-IV}}$(**loves’**, **michelle’**)) = (by definition of $H_{\text{Merge-IV}}$)

g. $H_{\text{Merge-S}}$(**barack’** , (**loves’ michelle’**)) = (by definition of $H_{\text{Merge-S}}$)

h. ((**loves’ michelle’**) **barack’**)

(52) **Remark**

Under the translation function k determined by **T**, and given the definition in (43), it follows that there are *two* different formulae in Politics-NoQ that are translations of the Mini-English sentence *It is not the case that Barack smokes and Mitt smokes.*

(exercise for the student!)

(53) **What We Wanted**

We wanted to develop a way of homomorphically mapping expressions of DME to expressions of Politics-NoQ (so that we can ultimately get a semantics for English)

$<E, K_\gamma, X_\delta, S_E, S>_{\gamma \in \{\text{Merge-S, Merge-IV, Not, And, If}\},\ \delta \in \Delta}$

$\downarrow$ *k (translation from DME to Pol-NoQ)*

$< A, F_\gamma, X_\tau, S, t >_{\gamma \in \{\text{Concat, Not, And}\},\ \tau \in T}$

$\downarrow$ *g (semantic interpretation of Politics-NoQ)*

$<B, G_\gamma, f>_{\gamma \in \{\text{Concat, Not, And}\}}$

g°k (the semantic interpretation of DME)

(54) **What We Have Now**

$< E, K_\gamma >_{\gamma \in \{\text{Merge-S, Merge-IV, Not, And, If}\}}$ $\downarrow$ *k* $<A, H_\gamma>_{\gamma \in \{\text{Merge-S, Merge-IV, Not, And, If}\}}$

$< A, F_\gamma >_{\gamma \in \{\text{Concat, Not, And}\}}$ $\downarrow$ *g* $< B, G_\gamma >_{\gamma \in \{\text{Concat, Not, And}\}}$

- We have a way of homomorphically mapping expressions of Mini-English to expressions of Politics-NoQ.

- **However, our translation homomorphism doesn't hold between the syntactic algebra of Mini-English and the syntactic algebra of Politics-NoQ.**
  - Rather, it holds between the syntactic algebra of Mini-English and *another* syntactic algebra that we construct on the basis of $< A, F_\gamma>_{\gamma \in \{\text{Concat, Not, And}\}}$

- **But, our interpretation homomorphism holds between the syntactic algebra of Politics-NoQ and the interpretation $<B, G_\gamma, f>_{\gamma \in \{\text{Concat, Not, And}\}}$**
  - Our interpretation function g is not a homomorphism from our derived syntactic algebra $<A, H_\gamma>_{\gamma \in \{\text{Merge-S, Merge-IV, Not, And, If}\}}$

- **Therefore, the composition of k and g is *not* a homomorphism from our syntactic algebra for Mini-English to the interpretation $<B, G_\gamma, f>_{\gamma \in \{\text{Concat, Not, And}\}}$**
  - ***So how do we get what we want, a homomorphism from the syntactic algebra of Mini-English to an interpretation structure?....***

Tune in next time for Part 3 of 'Montague's Theory of Translation'…