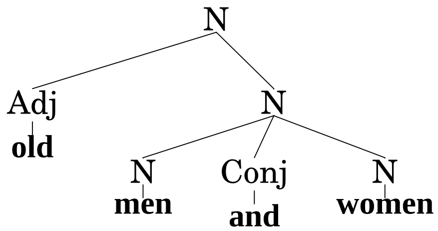
PART 1: Syntax-Semantics Interface **Montague (1970)**

1. Syntax

(1.1) If $\Delta_1,\ldots,\Delta_n$ are deep structures of the corresponding syntactic categories $\kappa_1,\ldots,\kappa_n$, then the result of applying the (n -place) syntactic construction C to $(\Delta_1,\ldots,\Delta_n)$ will be a deep structure of category κ_{n+1}.
Notation: $(C, \kappa_1,\ldots,\kappa_n,\kappa_{n+1})$

(1.2) **old men and women**

(1.3)

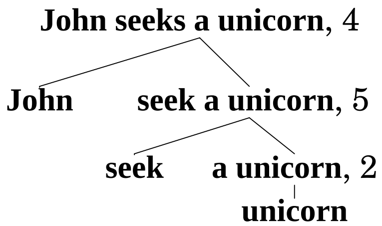


(USC) Uniqueness of Structure Constraint on an algebra $(\Sigma, (C_i)_{i\in I})$:
If $C_i(\Delta_1,\ldots,\Delta_n) = C_j(\Delta'_1,\ldots,\Delta'_m)$, then $i = j$, and $(\Delta_1,\ldots,\Delta_n) = (\Delta'_1,\ldots,\Delta'_m)$ (and hence $C_i = C_j$, $m = n$, $\Delta_1 = \Delta'_1, \ldots,\Delta_n = \Delta'_m$).

(1.4) $(_{\mathrm{N}}\ (_{\mathrm{Adj}}$ **old** $)_{\mathrm{Adj}}\ (_{\mathrm{N}}\ (_{\mathrm{N}}$ **men** $)_{\mathrm{N}}\ (_{\mathrm{Conj}}$ **and**$)_{\mathrm{Conj}}\ (_{\mathrm{N}}$ **women**$)_{\mathrm{N}}\ )_{\mathrm{N}}\ )_{\mathrm{N}}$

(1.5) **≰ Jones (seeks a ∉ horse such** v_7 **that ≰ it** v_7 **speaks ≱ ∌) ≱**

(1.6)



Definition

A *(deep) syntax* is a quintuple $(\Sigma,(C_i)_{i\in I},(L_k)_{k\in K},R,\mathrm{S})$, where

- $(\Sigma, (C_i)_{i\in I})$ satisfies (USC);
- the Lexicon $\bigcup_{k\in K} L_k$ generates the set Σ of (syntactic) structures;
  (i.e. no lecial item is a value of an operation C_i and Σ is the smallest set that contains the lexicon and is closed under all C_i);
- the elements of R[ules] are as in (1.1) – where C is one of the C_i and all $\kappa_j \in K$[ategories];
- S[entence] $\in K$.

If $\Delta\in\Sigma$ and $k^*\in K$, then Δ *is of category* k^* if either $\Delta\in L_{k^*}$ (in which case Δ's *rank* $\rho(\Delta)$ is 0), or $\Delta = C_i(\Delta_1,\ldots,\Delta_n)$ (and thus $\rho(\Delta) = max(\rho(\Delta_1),\ldots,\rho(\Delta_1)) + 1$), $\Delta_1,\ldots,\Delta_n$ are of categories $k_1,\ldots,k_n$, respectively, and $(C_i,k_1,\ldots,k_n,k^*)\in R$.

2. Compositionality

(2.1) The meaning of a complex expression can be determined from the meanings of its parts.

(2.2) If S_1 and S_2 are sentences (of some fixed language), then so is ‘(S_1 **and** S_2)’.

(2.3) $C(\Delta,\Delta')$ = ‘(Δ **and** Δ')’, whenever Δ and Δ' are structures.

(2.4) If $C(\Delta_1,\ldots,\Delta_n)$ is a structure, then its meaning is uniquely determined by the meanings of $\Delta_1,\ldots,\Delta_n$ and C.

(2.5) For each (n-place) syntactic construction C there is a corresponding (n-place) meaning combination M such that the meaning of any structure $C(\Delta_1,\ldots,\Delta_n)$ is $M(b_1,\ldots,b_n)$, where b_1 is Δ_1’s meaning, etc.

(2.6) $\mu(C(\Delta_1,\ldots,\Delta_n)) = M(\mu(\Delta_1),\ldots,\mu(\Delta_n))$

(2.7) **Jones knows that** S, C_0

Jones **knows that** S, C_1

knows **that** S

(2.8) $M_0(\mu(\textbf{Jones}),M_1(\mu(\textbf{love}),\mu(\textbf{Smith})))$

(2.9) μ(**Jones knows that** S), M_0

μ(**Jones**) μ(**knows that** S), M_1

μ(**knows**) μ(**that** S)

(2.10) μ(**Jones knows that** S'), M_0

μ(**Jones**) μ(**knows that** S'), M_1

μ(**knows**) μ(**that** S')

(2.11) $\mu(\textbf{that } S) = \mu(\textbf{that } S') \Rightarrow \mu(\textbf{Jones knows that } S) = \mu(\textbf{Jones knows that } S')$

(2.12) $L_1 = \{\mathbf{0},\ldots,\mathbf{9}\}$; $L_n = \emptyset$ ($n \neq 1$); $C_n(\Delta,\Delta')$ = ‘$\mathbf{[}_n\ \Delta\ \Delta'\ \mathbf{]}$’ $R = \{(C_n,1,n,n+1) \mid n \geq 1\}$.

(2.13) $M_n(x,y) = 10^n x + y$

(2.14) $\mu(C_2(\mathbf{7}, C_1(\mathbf{1},\mathbf{2})))$

$= M_2(\mu(\mathbf{7}),\mu(C_1(\mathbf{1},\mathbf{2})))$

$= M_2(\mu(\mathbf{7}),M_1(\mu(\mathbf{1}),\mu(\mathbf{2})))$

$= M_2(7,M_1(1,2))$

$= 10^2 \times 7 + 10^1 \times 1 + 2$

$= 712$

<u>Definitions</u>

Given a syntax $(\Sigma,(C_i)_{i\in I},(L_k)_{k\in K},R,\mathrm{S})$, a *corresponding semantics* is a triple $(B,\mu_0,(M_i)_{i\in I})$, where B is some non-empty set; $\mu_0 : \bigcup_{k\in K} L_k \to B$; and $(M_i)_{i\in I}$ is similar to $(C_i)_{i\in I}$. Given any $\Delta\in\Sigma$, then either $\Delta\in L_{k^*}$ and its *meaning* (according to $(B,\mu_0,(M_i)_{i\in I})$) is $\mu_0(\Delta)$; or else $\Delta = C_i(\Delta_1,\ldots,\Delta_n)$ and its *meaning* [...] is $\mu(C(\Delta_1,\ldots,\Delta_n)) = M(\mu(\Delta_1),\ldots,\mu(\Delta_n))$, where $\mu(\Delta_1),\ldots,\mu(\Delta_n)$ are the respective meanings [...] of $\Delta_1,\ldots,\Delta_n$.

(2.15) **$(\exists x)$ P(x)**

(2.16) **P(a)**

(2.17) The meaning of a complex expression is determined by the meanings of (certain) less complex expressions.

(2.18) **$(\forall x)$ P(x)**

(2.19) There exists a function f which can be applied to pairs consisting of syntactic constructions and sets of meanings such that the meaning of a complex structure Δ of the form $C(\Delta_1,..,\Delta_n)$ equals the value $f(C,\mathrm{b}[\mathrm{X}])$, where X is some set of structures of ranks less than Δ.

PART 2: Meaning and Reference

3. Local Perspective

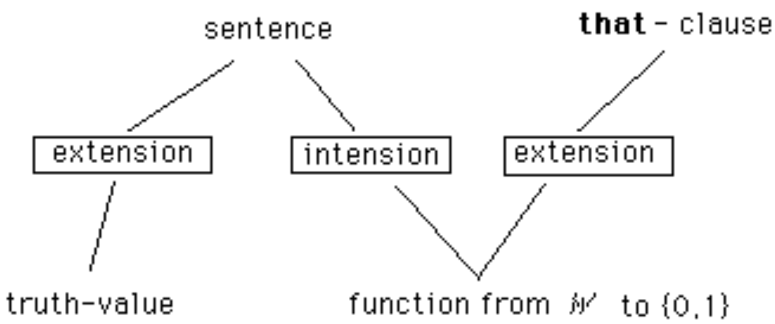
(3.1)

Syntactic Category	*Type of extension*	*Example*	*Extension of example*
proper name	individual (bearer)	**Fritz**	*Fritz Hamm*
definite description	individual (described)	**the fifth-biggest city of France**	*Nice*
count nouns	set (of individuals)	**table**	*set of tables*
intransitive verb	set (of individuals)	**sleep**	*set of sleepers*
transitive verb	set of pairs (of individuals)	**eat**	*set of pairs (eater,food)*
ditransitive verb	set of triples (of individuals)	**give**	*set of triples (giver,recipient, gift)*
sentence	truth value (Ø or {Ø})	**snow is white**	*1*

(3.2) **Jones knows that snow is white.**

(3.3) **Jones knows that grass is green.**

(3.13)



(3.14) (i) $\{\boldsymbol{t},\boldsymbol{e}\} \subseteq \mathbf{IT}$, and $\{(a,b),(\boldsymbol{s},b)\} \subseteq \mathbf{IT}$ if $\{a,b\} \subseteq \mathbf{IT}$. types

(ii) $\mathbf{E}_{\boldsymbol{t}} = \{0,1\}$, $\mathbf{E}_{\boldsymbol{e}} = \mathbf{U}$, $E_{(a,b)} = \mathbf{E}_b{}^{\mathbf{E}_a}$, $\mathbf{E}_{(\boldsymbol{s},b)} = \mathbf{E}_b{}^{\mathbf{W}}$. extensions

(iii) $\mathbf{I}_a = \mathbf{E}_{(\boldsymbol{s},a)}$; $\mathbf{M}_{\boldsymbol{a}} = \mathbf{I}_{\boldsymbol{a}}{}^{\mathbf{C}}$. intensions and meanings

(iv) A *point of reference* is an element of $\mathbf{W} \times \mathbf{C}$.

(v) A *property* is an element of $\mathbf{I}_{\boldsymbol{et}}$.

(vi) P is a *subproperty* of Q iff $P(w) \subseteq Q(w)$, for any $w \in \mathbf{W}$.

<u>Definition</u>

A *local (Fregean) language* is a quintuple $(\Xi, f, (M_i)_{i \in I}, \mu_0, \boldsymbol{\Delta})$, where

- $\Xi = (\Sigma, (C_i)_{i \in I}, (L_k)_{k \in K}, R, \mathrm{S})$ is a syntax;
- $f: K \to \mathbf{IT}$ is a function (type assignment) such that $f(\mathrm{S}) = \boldsymbol{t}$;
- $\mu_0 : \bigcup_{k \in K} L_k \to \bigcup_{a \in \mathbf{IT}} \mathbf{M}_a$ is a function (lexical meaning assignment) such that $\mu_0(\delta) \in \mathbf{M}_a$ whenever $\delta \in L_k$ and $f(k) = a$;
- $(M_i)_{i \in I}$ is a family of meaning operations (similar to $(C_i)_{i \in I}$) such that $M_i(b_1,\ldots,b_n) \in \mathbf{M}_{f(k_0)}$ whenever $b_1 \in \mathbf{M}_{f(k_1)},\ldots,b_n \in \mathbf{M}_{f(k_n)}$ and $(M_i, k_1,\ldots,k_n,k_0) \in R$.
- $\boldsymbol{\Delta}$ is the *diagonal*, i.e. the set of $(w,c) \in \mathbf{W} \times \mathbf{C}$ such that $w = w_c$.

$\Rightarrow$ If Δ is of category k, then its meaning is in $\mathbf{M}_{f(k)}$.

(3.15) $M(b_1,\ldots,b_n)\,(c)\,(w) = M'(b_1(c),\ldots,b_n(c))\,(c)\,(w)$

(3.16) $M(b_1,\ldots,b_n)\,(c) = M'(b_1(c),\ldots,b_n(c))\,(c)$

(3.17) δ is *deictic* iff $\mu(\varphi)(c)(w) = \mu(\varphi)(c)(w')$, for all $w, w' \in \mathbf{W}$.

(3.18) δ is a *hyponym of* δ' (*in* a local language L) iff $\mu_0(\delta)(c)(w)$ is a subproperty of $\mu_0(\delta')(c)(w)$ whenever (c,w) is a point of $\left\{\begin{matrix}\text{reference}\\ \text{utterance}\end{matrix}\right\}$.

(3.19) φ of category S is an *a priori truth* iff $\mu(\varphi)(c)(w) = 1$, for all $(w,c) \in \boldsymbol{\Delta}$.

4. Global Perspective

(4.1) The meaning of **pebble** is that (local) meaning $b$of type $<e,t>$ such that for any point of reference (w,c) and any individual x the following holds:

$$b(c)(w)(x) = \begin{cases} 1, \text{ if } x \text{ is a pebble with respect to the relevant paramters of } <w,c>; \\ 0, \text{ otherwise.} \end{cases}$$

(4.2) The meaning of **stone** is that closed meaning b of type $(\boldsymbol{e},\boldsymbol{t})$ such that for any point of reference (w,c) and any individual x the following holds:

$$b(c)(w)(x) = \begin{cases} 1, \text{ if } x \text{ is a stone with respect to the relevant paramters of } <w,c>; \\ 0, \text{ otherwise.} \end{cases}$$

<u>Definitions</u>

(i) An *ontology* is a pair (E,C) where $C \neq \emptyset$ and there are non-empty sets D and W such that $E = (E_a)_{a\in\mathbf{IT}}$ satisfies the equations (3.14)(ii).

(ii) An *ersatz (Fregean) language* based on an ontology (E,C) is a quintuple that is like a local language except that E and C play the respective rôles of extensions and contexts.

(iii) A *global (Fregean) language* is a class of *ersatz* local languages that share the same syntax and type assignment.

(iv) δ is $\left\{\begin{array}{c} \textit{deictic} \\ \textit{a hyponym of } \delta' \\ \textit{an a priori truth} \end{array}\right\}$ in a global language iff δ is $\left\{\begin{array}{c} \text{deictic} \\ \text{a hyponym of } \delta' \\ \text{an a priori truth} \end{array}\right\}$ in each of its members.

(4.3) φ of category S is *contingent (in* a given [*ersatz*] local language L iff there are points of reference (w,c) and (w',c') such that $\mu(\varphi)(c)(w) = 1$ and $\mu(\varphi)(c')(w') = 0$, where $\mu_L(\varphi)$ is the meaning of $\mu_L(\varphi)$ according to L.

(4.4) φ of category S is *independent of* ψ of category S *(in* a given [*ersatz*] local language) iff $\{(\mu_L(\varphi)(c)(w),\mu_L(\psi)(c)(w)) \mid (w,c) \text{ is a point of reference of } L\}$ has 4 members.

PART 3: Indirect Interpretation

5. Translation

(5.1)

English		Logic	
structure	*category*	$\vert$*structure*$\vert$	*type*
book	N	B	***et***
cheap	Adj	**C**	***et***
$(_{\mathrm{N}}\ (_{\mathrm{Adj}}$**cheap**$), (_{\mathrm{N}}$ **book**$))$ [$=F_{\mathrm{mod}}$(**cheap**,**book**)]	N	$[\lambda x\,[\mathbf{C}(x) \wedge \mathbf{B}(x)]]$ [$= F_\lambda(\boldsymbol{x},F_\wedge(F_{\mathrm{app}}(\mathbf{C},\boldsymbol{x}),F_{\mathrm{app}}(\mathbf{B},\boldsymbol{x})))$]	***et***

(5.2) $\vert F_{\mathrm{mod}}(\Delta,\Delta')\vert = F_\lambda(\boldsymbol{x},F_\wedge(F_{\mathrm{app}}(\Delta,\boldsymbol{x}),F_{\mathrm{app}}(\Delta',\boldsymbol{x})))$ where $\boldsymbol{x}$ is a fixed variable

(5.3) A *syntactic polynomial* (over a given syntax) is a term of the form $\boldsymbol{F(X_1,\ldots,X_n)}$, where $\boldsymbol{F}$ is the (unique) name of an n-place syntactic construction (of that syntax) and each $\boldsymbol{X_i}$ is either itself a syntactic polynomial (…), or a *meta-variable* (standing in for an arbitrary structure), or the (unique) name of a particular structure (…).

A *derived construction* (*on* a syntax) if is an operation on syntactic structures (…) that is denoted by some syntactic polynomial (…) – in the more or less obvious sense.

(5.4) A *translation* from a syntax $\Xi = (\Sigma,(C_i)_{i\in I},(L_k)_{k\in K},R,\mathrm{S})$ to a syntax $\Xi' = (\Sigma',(C'_i)_{i\in I'},(L'_k)_{k\in K'},R',\mathrm{S}')$ is a triple $(g,t,(T_i)_{i\in I},)$ such that:

- $g: K \to K'$ is a function (category assignment) such that $g(\mathrm{S}) = \mathrm{S}'$;
- $t: \bigcup_{k\in K} L_k \to \Sigma'$ is a function (lexical translation) such that $t(\delta)$ is a structure of category $g(k)$ in Ξ' whenever $\delta\in L_k$;
- $(T_i)_{i\in I}$ is a family of derived constructions on Ξ' that is similar to $(C_i)_{i\in I}$;
- if $(C_i,k_1,\ldots,k_n,k^*)\in R$, and $\Delta_1,\ldots,\Delta_n$ are structures of the respective categories $k_1,\ldots,k_n$, then $T_i(\Delta_1,\ldots,\Delta_n)$ is of category $g(k^*)$.

Given any $\Delta\in\Sigma$, then either $\Delta\in L_{k^*}$ and its *translation* $\vert\Delta\vert$ (according to $(g,t,(T_i)_{i\in I},)$) is $t(\Delta)$; or else $\Delta = C_i(\Delta_1,\ldots,\Delta_n)$ and its *translation* […] is $\vert C_i(\Delta_1,\ldots,\Delta_n))\vert = T_i(\vert\Delta_1\vert,\ldots,\vert\Delta_n\vert)$, where $\vert\Delta_1\vert),\ldots,\vert\Delta_n\vert$ are the respective translations […] of $\Delta_1,\ldots,\Delta_n$.

6. Intensional Type Logic (ITL)

(6.1) The *variables* of ITL form a family $(Var_a)_{a\in\mathbf{IT}}$ of pairwise disjoint, infinite sets; the *constants* of ITL form a family $(Con_a)_{a\in\mathbf{IT}}$ of pairwise disjoint sets; the *syncategorematic expressions* form the set $\{\boldsymbol{\lambda},(,),=,\wedge,\vee\}$. No variable is a constant or a syncategorematic expression, etc.

The *syntax of ITL* is a quintuple $(\Sigma,(C_i)_{i\in I},(L_k)_{k\in K},R,\boldsymbol{t})$, where:

- Σ consists of (finite) strings over $\bigcup_{a\in\mathbf{IT}} Con_a \cup \bigcup_{a\in\mathbf{IT}} Var_a \cup \{\boldsymbol{\lambda},(,),=,\wedge,\vee\}$;
- $I = \{\text{app,abs,id,cup,cap}\}$;
- $C_{\text{app}}(\Delta,\Delta') = \Delta\mathbf{(}\Delta'\mathbf{)}$; $C_{\text{abs}}(\Delta,\Delta') = \mathbf{(}\boldsymbol{\lambda}\Delta\ \Delta'\mathbf{)}$; $C_{\text{id}}(\Delta,\Delta') = \mathbf{(}\Delta\ \mathbf{=}\ \Delta'\mathbf{)}$; $C_{\text{cup}}(\Delta) = {}^{\vee}\Delta$; $C_{\text{cap}}(\Delta) = {}^{\wedge}\Delta$;
- $K = \mathbf{IT} \cup \{(\text{VAR},a) \mid a\in\mathbf{IT}\}$;
- $L_k = Var_k \cup Con_k$ if $k\in\mathbf{IT}$; $L_k = Var_a$ if $k = (\text{VAR},a)$;
- $R =$ $\{\{(C_{\text{app}},(a,b),a,b) \mid a,b\in\mathbf{IT}\},\{(C_{\text{abs}},((\text{VAR},a),b,(a,b) \mid a,b\in\mathbf{IT}\},\{(C_{\text{id}},a,a,\boldsymbol{t}) \mid a\in\mathbf{IT}\},$ $\{(C_{\text{cup}},(\boldsymbol{s},a),a) \mid a\in\mathbf{IT}\},\{(C_{\text{cup}},a,(\boldsymbol{s},a)) \mid a\in\mathbf{IT}\}$.

An ITL-ontology is a pair (E,C), where $E = (E_a)_{a\in\mathbf{IT}}$ satisfies the equations (3.14)(ii) and C is the set of *variable assignments*, i.e. the set of functions h: $\bigcup_{a\in\mathbf{IT}} Var_a \to \bigcup_{a\in\mathbf{IT}} E_a$ such that $h(\boldsymbol{x}) \in E_a$ whenever $\boldsymbol{x}\in Var_a$.

A local (*ersatz*) *language of ITL* is a Fregean langugage (E,C) $(\Xi,f,(M_i)_{i\in I},\mu_0,\mathbf{\Delta})$ based on an ITL-ontology where

- Ξ is the syntax of ITL;
- $f(a) = f((\text{VAR},a)) = a$, for any $a\in\mathbf{IT}$;
- for any b, $b'\in \bigcup_{a\in\mathbf{IT}} E_a$, $w,w'\in W$, $h\in C$, and $u\in U$ the following hold:
  $M_{\text{app}}(b,b')\,(h)(w) = b(h)(w)\,(b'(h)(w))$ whenever $b\in \mathbf{M}_{a,b}$ and $b'\in \mathbf{M}_a$;
  $M_{\text{abs}}(\mu_0(\boldsymbol{x}),b)\,(h)(w)\,(u) = b(h[\boldsymbol{x}/u])(w)$ whenever $\boldsymbol{x}Var_a$ and $b\in \mathbf{M}_{a,b}$ and $h[\boldsymbol{x}/u] = (h\setminus\{(\boldsymbol{x},h(\boldsymbol{x}))\}) \cup \{(\boldsymbol{x},u)\}$;
  $M_{\text{id}}(b,b')\,(h)(w) = \{\varnothing \mid b(h)(w) = b'(h)(w)\}$ whenever $b,b'\in \mathbf{M}_a$;
  $M_{\text{cup}}(b)\,(h)(w) = b(h)(w)(w)$ whenever $b\in \mathbf{M}_{\boldsymbol{s}a}$;
  $M_{\text{cap}}(b)\,(h)(w)(w') = b(h)(w')$.
- $\mu_0(\mathbf{c})(h)(w) = \mu_0(\mathbf{c})(h')(w)$ whenever $w\in W$, $h,h'\in C$ and $\mathbf{c}\in \bigcup_{a\in\mathbf{IT}} Con_a$;
  $\mu_0(\boldsymbol{x})(h)(w) = h(\boldsymbol{x})$ whenever $w\in W$, $h\in C$ and $\boldsymbol{x}\in \bigcup_{a\in\mathbf{IT}} Var_a$;
- $\mathbf{\Delta} = W\times C$.

If M is a local language of ITL, α is an ITL formula (structure), $\mu(\alpha)$ is α's menaing according to M, $h\in C$, $w\in W$, $[\![\alpha]\!]^{M,h,w}$ is $\mu(\alpha)(h)(w)$.

Given a local ITL-language M, there exists a function F: $\bigcup_{a\in\mathbf{IT}} Con_a \to \bigcup_{a\in\mathbf{IT}} I_a$ such that $F(\mathbf{c}) \in I_a$ whenever $\mathbf{c}\in Con_a$ and such that the following hold:

(i) $[\![\alpha]\!]^{M,g,w} = F(\alpha)(w)$, if $\alpha\in Con_a$;

(ii) $[\![\alpha]\!]^{M,g,w} = g(\alpha)$, if $\alpha\in Var_a$;

(iii) $[\![\alpha]\!]^{M,g,w} = [\![\alpha_1]\!]^{M,g,w}([\![\alpha_2]\!]^{M,g,w})$, if $\alpha = \alpha_1\ (\alpha_2)$;

(iv) $[\![\alpha]\!]^{M,g,w} = \{(u,[\![\alpha_1]\!]^{M,g[x/u],w}) \mid u\in D_b\}$, if $\alpha = (\boldsymbol{\lambda x}\ \alpha_1)$ und $\boldsymbol{x}\in Var_b$;

(v) $[\![\alpha]\!]^{M,g,w} = \{u \mid [u = 0 \text{ and } [\![\alpha_1]\!]^{M,g,w} = [\![\alpha_2]\!]^{M,g,w}\ ]\ \}$, if $\alpha = (\alpha_1 \boldsymbol{=} \alpha_2)$;

(vi) $[\![\alpha]\!]^{M,g,w} = [\![\alpha_1]\!]^{M,g,w}(w)$, if $\alpha = {}^{\vee}\alpha_1$;

(vii) $[\![\alpha]\!]^{M,g,w} = \{(w',[\![\alpha_1]\!]^{M,g,w'}) \mid w'\in W\}$, if $\alpha = {}^{\wedge}\alpha_1$.

(6.2) If α and α' are ITL-formulae of the same category, then α and α' are *logically equivalent* if $[\![\alpha]\!]^{M,g,w} = [\![\alpha']\!]^{M,g,w}$ for any local ITL-languages M, worlds w and assignments g. Notation: $\alpha \equiv \alpha'$.

(6.3) An ITL-formula α is *modally closed* if (i-a) $\alpha\in \bigcup_{a\in\mathbf{IT}} Var_a$; or (i-b) $\alpha = {}^{\wedge}\alpha$ (for some β), or (ii) there are modally closed α_1 and α_2 such that (ii-a) $\alpha = \alpha_1\ (\alpha_2)$, or (ii-b) $\alpha = (\boldsymbol{\lambda}\alpha_1\ \alpha_2)$, or (ii-c) $\alpha = (\alpha_1 \boldsymbol{=} \alpha_2)$.

Down-Up Cancellation

${}^{\vee\wedge}\alpha \equiv \alpha$, for all ITL-formulae α.

Up-Down Cancellation

${}^{\wedge\vee}\alpha \equiv \alpha$, if α is modally closed (and of a category $(\boldsymbol{s},a)$).

Two-sorted Type Theory

2T contains $\boldsymbol{t}$, $\boldsymbol{e}$, and $\boldsymbol{s}$ and all pairs (a,b) such that $a,b\in \mathbf{2T}$.
$(Var_a)_{a\in\mathbf{2T}}$ and $(Con_a)_{a\in\mathbf{2T}}$ are analogous to ITL, but the only syntactic constructions are C_{app} , C_{abs} , and C_{id} .

Gallin's translation ($\boldsymbol{i}$ is a fixed variable in $Var_{\boldsymbol{s}}$).

(i) $\mathbf{c}* =$ '$\mathbf{c}\ (\boldsymbol{i})$', if $\mathbf{c}\in Con_a$;

(ii) $\boldsymbol{x}* = \boldsymbol{x}$, if $\boldsymbol{x} \in Var_a$;

(iii) $\alpha(\beta)* = \alpha*(\beta*)$;

(iv) $(\lambda x\alpha)* = (\boldsymbol{\lambda x}\ \alpha*)$;

(v) $(\alpha = \beta)* = (\alpha* = \beta*)$;

(vi) ${}^{\vee}\alpha* = \alpha*(\boldsymbol{i})$;

(vii) ${}^{\wedge}\alpha'* = (\boldsymbol{\lambda i}\ \alpha*)$.

Restricted **β**-conversion (ITL)

$((\boldsymbol{\lambda x}\ \alpha)\ (\beta)) \equiv \alpha[{}^{x}\!/\!_{\beta}]$, if (i) β does not contain a free variable that would get bound when $\boldsymbol{x}$ in α is replaced by β *and* either (ii-a) no occcurrence of $\boldsymbol{x}$ in α lies within the scope of ^ , or (ii-b) β is modally closed.

<u>$\boldsymbol{\beta}$-conversion (Ty2)</u>
$((\boldsymbol{\lambda x}\ \alpha)\ (\beta)) \equiv \alpha[^{x}/_{\beta}]$, β does not contain a free variable that would get bound when $\boldsymbol{x}$ in α is replaced by β.
[Notation: $((\boldsymbol{\lambda x}\ \alpha)\ (\beta))\ >_{\boldsymbol{\beta}}\ \alpha[^{x}/_{\beta}]$]
NB1: $((\boldsymbol{\lambda x}\ \alpha)\ (\boldsymbol{x}))\ >_{\boldsymbol{\beta}}\ \alpha$;
NB2: $\boldsymbol{\beta}$-contraction may increase length; e.g., if $\boldsymbol{x}\in Var_e$, $\mathbf{R}\in Con_{e(e(et))}$, $\mathbf{f}\in Con_{e(e(ee))}$, $\mathbf{c}\in Con_e$:
$(\boldsymbol{\lambda x}\ \mathbf{R}(\boldsymbol{x})(\boldsymbol{x})(\boldsymbol{x}))(\mathbf{f}(\mathbf{c})(\mathbf{c})(\mathbf{c}))\ >\beta\ \mathbf{R}(\mathbf{f}(\mathbf{c})(\mathbf{c})(\mathbf{c}))\ (\mathbf{f}(\mathbf{c})(\mathbf{c})(\mathbf{c}))\ (\mathbf{f}(\mathbf{c})(\mathbf{c})(\mathbf{c})))$

<u>$\boldsymbol{\eta}$-conversion (Ty2 & ITL)</u>
$(\boldsymbol{\lambda x}\ \beta(\boldsymbol{x})) \equiv \beta$, if $\boldsymbol{x} \notin Fr(\beta)$

<u>$\boldsymbol{\alpha}$-conversion (Ty2 & ITL)</u>
$(\boldsymbol{\lambda x}\ \alpha) \equiv (\boldsymbol{\lambda y}\ \alpha[^{x}/_{y}])$ iff no occurrence of $\boldsymbol{x}$ in α lies within the scope of (some) $\boldsymbol{\lambda y}$ and $\boldsymbol{y} \notin Fr((\boldsymbol{\lambda x}\ \alpha))$.

<u>Definition</u>
(a) α is *immediately reducible to* β iff $\alpha = \gamma[^{x}/_{\delta_1}]$, $\beta = \gamma[^{x}/_{\delta_2}]$, and: [$\delta_1 >\alpha\ \delta_2$ or $\delta_1 >\beta\ \delta_2$ or $\delta_1 >\eta\ \delta_2$]
[Notation: $\alpha>\beta$; transitive closure: $\alpha\triangleright\beta$] … for some γ, δ_1, δ_2 and variable x (of the appropriate types)
(b) α is *normal* iff $\alpha\triangleright\beta$ implies $\alpha\triangleright_{\boldsymbol{\alpha}}\beta$.
(c) β is a *normal form of* α iff $\alpha\triangleright\beta$ and β is normal.

<u>Normal Form Theorem (Ty 2)</u>
Every Ty2-formula has a normal form.

<u>Church-Rosser Theorem (Ty2)</u>
If β and β' are normal forms of α, then $\beta \triangleright_{\boldsymbol{\alpha}} \beta'$.

(6.4a) $(\boldsymbol{\lambda x}\ \mathbf{P}((\boldsymbol{\lambda y}\ {}^\wedge\boldsymbol{y})(\boldsymbol{x})))(\mathbf{c})$ (where $\boldsymbol{x}\in Var_{se}$, $\boldsymbol{y}\in Var_e$, $\mathbf{c}\in Con_{se}$, $\mathbf{P}\in Con_{(se)t}$)
(b) $\mathbf{P}((\boldsymbol{\lambda y}\ {}^\wedge\boldsymbol{y})(\mathbf{c}))$
(c) $(\boldsymbol{\lambda x}\ \mathbf{P}({}^\wedge\boldsymbol{x}))(\mathbf{c})$

Friedman & Warren (1980)

Gallin 1975; Zimmermann (1989)

<u>Four observations on $*$:</u>
- $({}^\vee{}^\wedge\alpha)* >_{\boldsymbol{\beta}} \alpha*$.
- An ITL-formula α is modally closed iff $\boldsymbol{i}\notin Fr(\alpha^*)$.
- If α is modally closed, $({}^\wedge{}^\vee\alpha)* >_{\boldsymbol{\eta}} \alpha*$.
- If all constants and free variables of a Ty2-formula α of a type in **2T** \ **IT** are of types in **2T** \ **IT**, then α is logically equivalent to the $*$-image of some ITL-formula.

(6.5a) $(\boldsymbol{\lambda i}\ (\boldsymbol{\lambda j}\ (\boldsymbol{i} = \boldsymbol{j})))$ (where $\boldsymbol{i},\boldsymbol{j}\in Var_s$)
(b) $(\boldsymbol{\lambda i}\ (\boldsymbol{\lambda F}\ (\boldsymbol{\lambda i}\ (\boldsymbol{F} = (\boldsymbol{\lambda p}\ \boldsymbol{p}(\boldsymbol{i}))))))\ ((\boldsymbol{\lambda p}\ \boldsymbol{p}(\boldsymbol{i})))\ )$ (where $F\in Var_{(st)t}$, $p\in Var_{st}$)
(c) ${}^\wedge(\boldsymbol{\lambda F}\ {}^\wedge(\boldsymbol{F} = (\boldsymbol{\lambda p}\ {}^\vee\boldsymbol{p})))\ ((\boldsymbol{\lambda p}\ {}^\vee\boldsymbol{p}))$

(6.6) Abbreviations in ITL and Ty2:

Notation	where	is short for
$\alpha(\beta,\gamma)$	α: $a(ab)$; β,γ: a	$\alpha(\gamma)(\beta)$
T		$(\lambda \boldsymbol{x}_t\, \boldsymbol{x}) = (\lambda \boldsymbol{x}\, \boldsymbol{x})$
$\bot$		$(\lambda \boldsymbol{x}\ \mathrm{T}) = (\lambda \boldsymbol{x}\, \boldsymbol{x})$
$\neg\, \varphi$	φ: $\boldsymbol{t}$	$(\varphi = \bot)$
$(\forall \boldsymbol{x})\ \varphi$	$\boldsymbol{x} \in Var$; φ: $\boldsymbol{t}$	$(\lambda \boldsymbol{x}\ \varphi) = (\lambda \boldsymbol{x}\, \boldsymbol{x})$
$(\exists \boldsymbol{x})\ \varphi$	$\boldsymbol{x} \in Var$; φ: $\boldsymbol{t}$	$\neg\, (\forall \boldsymbol{x})\, \neg\, \varphi$
$[\varphi \leftrightarrow \psi]$	φ,ψ: $\boldsymbol{t}$	$(\varphi = \psi)$
$[\varphi \wedge \psi]$	φ,ψ: $\boldsymbol{t}$	$(\forall \boldsymbol{R}_{t(tt)})\ [\boldsymbol{R}(\varphi,\psi) \leftrightarrow \boldsymbol{R}(\mathrm{T},\mathrm{T})]$ [alternatively: $(\forall \boldsymbol{f}_{tt})\ [\varphi \leftrightarrow [\boldsymbol{f}(\varphi) \leftrightarrow \boldsymbol{f}(\psi)]]$]
$[\varphi \vee \psi]$	φ,ψ: $\boldsymbol{t}$	$[\neg\varphi \wedge \neg\psi]$
$[\varphi \rightarrow \psi]$	φ,ψ: $\boldsymbol{t}$	$[\neg\varphi \vee \psi]$
etc.		

(6.7) Special ITL-conventions:

Notation	where	is short for
$\alpha\{\beta\}$	α: $\boldsymbol{s}(a\boldsymbol{t})$; β: a	${}^{\vee}\alpha(\beta)$
$\alpha\{\beta,\gamma\}$	α: $\boldsymbol{s}(a(a\boldsymbol{t}))$; β,γ: a	${}^{\vee}\alpha(\gamma)(\beta)$
$\Box\ \varphi$	φ: $\boldsymbol{t}$	$({}^{\wedge}\varphi = {}^{\wedge}\mathrm{T})$
$\Diamond\ \varphi$	φ: $\boldsymbol{t}$	$\neg\, \Box\, \neg\, \varphi$

PART 4: Descriptive Montague Grammar

7. Extensional Constructions

(7.1) Simple constructions

Construction	Corresponding Rule(s)	Example
$F_{pred}(\Delta,\Delta') =$ $\Delta\ \Delta'$, *pred* / Δ Δ'	$(F_{pred}$,NP,VP,S)	F_{pred}(**is happy**,**Mary**) = **Mary is happy**, *pred* / **is happy** … **Mary**
$F_{obj}(\Delta,\Delta') =$ $\Delta\ \Delta'$, *obj* / Δ Δ'	$(F_{obj}$,TV,NP,VP)	F_{obj}(**likes**, **the girl**) = **likes the girl**, *obj* / **likes** **the girl** …
$F_{cop}(\Delta) =$ **is** Δ, *cop* / Δ	$(F_{cop}$,Adj,VP)	F_{cop}(**happy**) = **is happy**, *cop* / **happy**
$F_{def}(\Delta) =$ **the** Δ, *def* / Δ	$(F_{def}$,N,NP)	F_{def}(**girl**) = **the girl**, *def* / **girl**

(7.2) Naive type assignment

Category	Example	(Extension) Type
S	**Mary is happy**; **the boy likes the girl**	$\boldsymbol{t}$
NP	**Mary**; **the boy**; **the girl**	$\boldsymbol{e}$
VP	**is happy**; **likes the girl**	$\boldsymbol{et}$
TV	**likes**	$\boldsymbol{e}(\boldsymbol{et})$
Adj	**happy**	$\boldsymbol{et}$

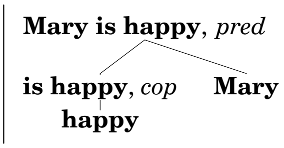
(7.3) Naive lexical translation

Item	ITL	Ty2
Mary	**m** $[\in Con_{\boldsymbol{e}}]$	$\mathbf{m}_{\boldsymbol{i}}$ $[= \mathbf{m}(\boldsymbol{i})]$
boy	**B** $[\in Con_{\boldsymbol{et}}]$	$\mathbf{B}_{\boldsymbol{i}}$
girl	**G** $[\in Con_{\boldsymbol{et}}]$	$\mathbf{G}_{\boldsymbol{i}}$
likes	**L** $[\in Con_{\boldsymbol{e(et)}}]$	$\mathbf{L}_{\boldsymbol{i}}$
happy	**H** $[\in Con_{\boldsymbol{et}}]$	$\mathbf{H}_{\boldsymbol{i}}$

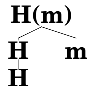
(7.4) Naive meaning combinations

Construction	Corresponding Polynomial
F_{pred}	$G_{pred}(\alpha,\beta) = C_{\mathrm{app}}(\alpha,\beta)$ [$= \alpha(\beta)$]
F_{obj}	$G_{pred}(\alpha,\beta) = C_{\mathrm{app}}(\beta,\alpha)$ [$= \beta(\alpha)$]
F_{cop}	$G_{cop}(\alpha) = \alpha$
F_{def}	$G_{def}(\alpha) = C_{\mathrm{app}}(\boldsymbol{\iota},\alpha)$ (where $\boldsymbol{\iota} \in Con_{(\boldsymbol{et})\boldsymbol{e}}$) [$= \boldsymbol{\iota}(\alpha)$]

(7.5a)



(a')



$= \quad \lvert F_{pred}(F_{cop}(\textbf{happy}),\textbf{Mary})\rvert$

$= \quad G_{pred}(\lvert F_{cop}(\textbf{happy})\rvert, \lvert\textbf{Mary}\rvert)$

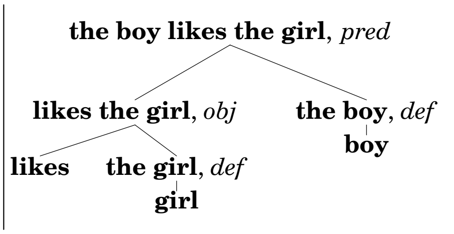
$= \quad G_{pred}(G_{cop}(\lvert\textbf{happy}\rvert), \lvert\textbf{Mary}\rvert)$

$= \quad G_{pred}(G_{cop}(\mathbf{H}),\mathbf{m})$

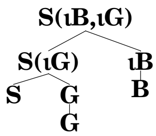
$= \quad G_{pred}(\mathbf{H},\mathbf{m})$

$= \quad \mathbf{H(m)}$

(b)



(b')



$= \quad \lvert F_{pred}(F_{def}(\textbf{boy})),F_{obj}(\textbf{likes},F_{def}(\textbf{girl}))\rvert$

$= \quad G_{pred}(\lvert\,\lvert F_{def}(\textbf{boy})\rvert,F_{obj}(\textbf{likes},F_{def}(\textbf{girl}))\rvert)$

$= \quad G_{pred}(G_{def}(\lvert\textbf{boy}\rvert),G_{obj}(\lvert\textbf{likes}\rvert,G_{def}(\lvert\textbf{girl}\rvert)))$

$= \quad G_{pred}(G_{def}(\mathbf{B}),G_{obj}(\mathbf{L},G_{def}(\mathbf{G})))$

$= \quad G_{pred}(\boldsymbol{\iota}(\mathbf{B}),G_{obj}(\mathbf{L},\boldsymbol{\iota}(\mathbf{G})),)$

$= \quad G_{pred}(\boldsymbol{\iota}(\mathbf{B}),\mathbf{L}(\boldsymbol{\iota}(\mathbf{G})))$

$= \quad \mathbf{L}(\boldsymbol{\iota}(\mathbf{G}))\ (\boldsymbol{\iota}(\mathbf{B}))$

$= \quad \mathbf{L}(\boldsymbol{\iota}(\mathbf{B}),\boldsymbol{\iota}(\mathbf{G}))$

(7.6) **Every boy likes Mary.**

<u>Context Principle</u>

The reconstructed extension ρ_α of α (in $F(\alpha,–)$) is a function f that assigns to the extension of any (relevant) γ the extension of $F(\alpha,\gamma)$:

- $\rho_\alpha(\mu(\beta)\ (c)\ (w)) = \mu(F(\alpha,\beta))\ (c)\ (w)$ — direct version
- $\lvert\alpha\rvert(\lvert\beta\rvert) \equiv \lvert F(\alpha,\beta)\rvert$ — indirect version
- $\lvert\alpha\rvert = (\boldsymbol{\lambda x}\ \lvert \mathrm{F}(\alpha,\boldsymbol{x})\rvert)$ — abstract version

(7.7a) $|\mathbf{every\ boy}|(|\mathbf{likes\ Mary}|) \equiv (\forall x)\ [\mathrm{B}(x) \rightarrow |\mathbf{likes\ Mary}|(x)]$

$|\mathbf{every\ boy}|(|\mathbf{is\ happy}|) \equiv (\forall x)\ [\mathrm{B}(x) \rightarrow |\mathbf{is\ happy}|(x)]$

…

$|\mathbf{every\ boy}|(|\beta|) \equiv (\forall x)\ [\mathrm{B}(x) \rightarrow |\beta|(x)]$

(7.7b) $|\mathbf{every\ boy}| = (\lambda Q_{et}\ (\forall x)\ [\mathrm{B}(x) \rightarrow P(x)])$ type $((et)t)$ [=: q]

(7.8a) $|\mathbf{every}|(|\mathbf{boy}|) \equiv (\lambda Q_{et}\ (\forall x)\ [\,|\mathbf{boy}|(x) \rightarrow Q(x)])$

$|\mathbf{every}|(|\mathbf{girl}|) \equiv (\lambda Q_{et}\ (\forall x)\ [\,|\mathbf{girl}|(x) \rightarrow Q(x)])$

…

$|\mathbf{every}|(|\beta|) \equiv (\lambda Q_{et}\ (\forall x)\ [\,|\beta|(x) \rightarrow Q(x)])$

(7.8b) $|\mathbf{every}| = (\lambda P_{et}\ (\lambda Q_{et}\ (\forall x)\ [P(x) \rightarrow Q(x)]))$ type $(et)q$

(7.9a) $|\mathbf{every\ boy\ or\ every\ girl}|(|\mathbf{likes\ Mary}|)$

$\equiv$ $[(\forall x)\ [\mathrm{B}(x) \rightarrow |\mathbf{likes\ Mary}|(x)] \vee (\forall x)\ [\mathrm{G}(x) \rightarrow |\mathbf{likes\ Mary}|(x)]]$

$|\mathbf{every\ boy\ or\ every\ girl}|(|\mathbf{is\ happy}|)$

$\equiv$ $[(\forall x)\ [\mathrm{B}(x) \rightarrow |\mathbf{is\ happy}|(x)] \vee (\forall x)\ [\mathrm{G}(x) \rightarrow |\mathbf{is\ happy}|(x)]]$

…

$|\mathbf{every\ boy}|(|\beta|)$

$\equiv$ $[(\forall x)\ [\mathrm{B}(x) \rightarrow |\beta|(x)] \vee (\forall x)\ [\mathrm{G}(x) \rightarrow |\beta|(x)]]$

(7.9b) $|\mathbf{every\ boy\ or\ every\ girl}|$

$=$ $(\lambda Q_{et}[(\forall x)\ [\mathrm{B}(x) \rightarrow Q(x)] \vee (\forall x)\ [\mathrm{G}(x) \rightarrow Q(x)]])$ type q

(7.10a) $|\mathbf{or}|(|\mathbf{every\ boy}|)\ (|\mathbf{every\ girl}|)$

$\equiv$ $(\lambda Q_{et}[(\forall x)\ [\mathrm{B}(x) \rightarrow Q(x)] \vee (\forall x)\ [\mathrm{G}(x) \rightarrow Q(x)]])$

$\equiv$ $(\lambda Q_{et}[\,|\mathbf{every\ boy}|(Q)] \vee |\mathbf{every\ girl}|(Q)])$

$|\mathbf{or}|(|\mathbf{every\ boy}|)\ (|\mathbf{some\ girl}|)$

$\equiv$ $(\lambda Q_{et}[(\forall x)\ [\mathrm{B}(x) \rightarrow Q(x)] \vee (\exists x)\ [\mathrm{G}(x) \wedge Q(x)]])$

$\equiv$ $(\lambda Q_{et}[\,|\mathbf{every\ boy}|(Q)] \vee |\mathbf{some\ girl}|(Q)])$

…

$|\mathbf{or}|(|\beta|)\ (|\gamma|) \equiv (\lambda Q_{et}[\,|\beta|(Q)] \vee |\gamma|(Q)])$

(7.10b) $|\mathbf{or}| = (\lambda \mathfrak{A}_{(et)t}\ (\lambda \mathfrak{B}_{(et)t}(\lambda Q_{et}[\mathfrak{B}(Q)] \vee \mathfrak{A}(Q)])))$ type $q(qq)$

(7.10) $|\mathbf{Mary\ or\ every\ boy}|$

$\equiv$ $(\lambda Q_{et}[Q(\mathrm{m})] \vee (\forall x)\ [\mathrm{B}(x) \rightarrow Q(x)]])$

$\neq$ $(\lambda Q_{et}[\,|\mathbf{Mary}|(Q)] \vee |\mathbf{every\ girl}|(Q)])$ wrong type (e vs. q)

(7.11a) $|\mathbf{Mary}|(|\mathbf{likes\ Mary}|) \equiv |\mathbf{likes\ Mary}|(\mathbf{m})$

$|\mathbf{Mary}|(|\mathbf{is\ happy}|) \equiv |\mathbf{is\ happy}|(\mathbf{m})$

…

$|\mathbf{Mary}|(|\beta|) \equiv |\beta|(\mathbf{m})$

(7.11b) $|\mathbf{Mary}| = (\lambda Q_{et}\ Q(\mathbf{m}))$ type q

(7.12) Revised rules and constructions

Construction	Corresponding Rule(s)	Example
$F_{pred}(\Delta,\Delta') =$ Δ Δ', *pred* Δ Δ'	$(F_{pred}$,NP,VP,S)	F_{pred}(**Mary** ,**is happy**) = **Mary is happy**, *pred* **Mary** **is happy** …
	$(F_{pred}$,Det,N,NP)	F_{pred}(**the**,**girl**) = **the girl**, *pred* **the** **girl**
$F_{coord}(\Delta,\Delta',\Delta'')$ = Δ Δ Δ'', *coord* Δ Δ' Δ''	$(F_{coord}$,NP,Conj,NP,NP)	F_{pred}(**every boy**,**or**,**Mary**) = **every boy or Mary**, *coord* **every boy** **or** **Mary** …
	$(F_{coord}$,VP,Conj,VP,VP)	F_{pred}(**likes Mary**,**or**, **is happy**) = **is happy or likes Mary**, *coord* **is happy** **or** **likes Mary** …
$F_{cop}(\Delta) =$ **is** Δ, *cop* Δ	$(F_{cop}$,Adj,VP)	F_{cop}(**happy**) = **is happy**, *cop* **happy**
$F_{obj}(\Delta,\Delta') =$ Δ Δ', *obj* Δ	$(F_{obj}$,TV,NP,VP)	F_{obj}(**likes**, **the girl**) = **likes the girl**, *obj* **likes** **the girl** …

(7.13) Revised (and expanded) type assignment

Category	(Extension) Type
S	***t***
NP	***q***
VP	***et***
TV	***e***(***et***)
Adj	***et***
Conj	***q***(***qq***)

(7.14) Lexical translation: revisions and additions

Item	ITL
Mary	$(\lambda Q_{et}\ Q(\mathbf{m}))$
or	$(\lambda \mathfrak{A}_{(et)t}\ (\lambda \mathfrak{B}_{(et)t}(\lambda Q_{et}[\mathfrak{B}(Q)] \vee \mathfrak{A}(Q)])))$
every	$(\lambda P_{et}\ \ (\lambda Q_{et}\ (\forall x)\ [P(x) \rightarrow Q(x)]))$
some	$(\lambda P_{et}\ \ (\lambda Q_{et}\ (\exists x)\ [P(x) \wedge Q(x)]))$
the	$(\lambda P_{et}\ \ (\lambda Q_{et}\ (\exists x)\ (\forall y)\ [[P(y) \leftrightarrow (x = y)] \wedge Q(x)]))$

(7.15) New meaning combinations

Construction	Corresponding Polynomial
F_{pred}	$G_{pred}(\alpha,\beta) = C_{\text{app}}(\alpha,\beta)$ [= $\alpha(\beta)$]
F_{coord}	$G_{coord}(\alpha,\beta,\gamma) = C_{\text{app}}(C_{\text{app}}(\beta,\gamma),\alpha)$ [= $\beta(\gamma)(\alpha)$]
F_{cop}	$G_{cop}(\alpha) = \alpha$
F_{obj}	$C_{\text{abs}}(x, C_{\text{app}}(\beta, C_{\text{abs}}(y, C_{\text{app}}(C_{\text{app}}(\alpha,y),x))))$ [= $(\lambda x\ \beta(\lambda y\ \alpha(x,y)))$]

8. Intensional Constructions

Attitude verbs

(8.1a) **John <u>thinks</u> that Mary is happy.**

(8.1b) **Mary is happy.**

(8.1c) **Every boy likes Mary.**

(8.1d) **John <u>thinks</u> that every boy likes Mary.**

Opaque verbs

(8.2a) **John <u>is looking for</u> a book on Clinton.**

(8.2b) **Every book on Clinton is a book by Clinton.**

(8.2c) **John <u>is looking for</u> a book by Clinton.**

Core-intensional verbs

Montague (1973)

(8.3a) **The temperature <u>is rising</u>.**

(8.3b) **The temperature is ninety.**

(8.3c) **Ninety <u>is rising</u>.**

(8.4) Attitude reports: type assignment

Category	(Extension) Type
AttV	$(\boldsymbol{st})(\boldsymbol{et})$
Prop	$\boldsymbol{st}$

(8.5) Attitude reports: additional rules and constructions

Construction	Corresponding Rule(s)	Example
$F_{pred}(\Delta,\Delta')$ = $\Delta\ \Delta'$, *pred* [Δ Δ']	(F_{pred},AttV,Prop,VP)	F_{pred}(**thinks,that Mary is happy**) = **thinks that Mary is happy**, *att* [**thinks** **that Mary is happy** …]
$F_{that}(\Delta)$ = **that** Δ, *that* [Δ]	(F_{that},S,Prop)	F_{that}(**Mary is happy**) = **that** Δ, *that* [Δ]

(8.6) Attitude verbs: lexical translation

Item	ITL	Ty2
thinks	**T** [$\in Con_{(\boldsymbol{st})(\boldsymbol{et})}$]	$\mathbf{T}_{\boldsymbol{i}}$
believes	$(\lambda \boldsymbol{p}\ (\lambda \boldsymbol{x}\ (\lambda \boldsymbol{q}\ \Box[{}^{\vee}\boldsymbol{q} \rightarrow {}^{\vee}\boldsymbol{p}])(\mathbf{B}(\boldsymbol{x})))$ [$\mathbf{B}\in Con_{(\boldsymbol{est})}$]	$(\lambda \boldsymbol{p}\ (\lambda \boldsymbol{x}\ (\forall \boldsymbol{j})\ [\mathbf{B}_{\boldsymbol{i}}(\boldsymbol{x})(\boldsymbol{j}) \rightarrow \boldsymbol{p}_{\boldsymbol{j}}]$

(8.7) Attitude verbs: additional meaning combination

Construction	Corresponding ITL-Polynomial	Corresponding Ty2-Polynomial
F_{that}	$G_{pred}(\alpha) = C_{\mathrm{cap}}(\alpha)$ [= ^α]	$G_{pred}(\alpha) = C_{\mathrm{abs}}(\boldsymbol{i},\alpha)$ [= $(\lambda \boldsymbol{i}\ \alpha)$]

(8.8) **John is-trying-for-it-to-be-the-case that John finds a book on Clinton.**

(8.9a) |**seeks**|(|**a book**|)

≡ $(\lambda \boldsymbol{x}\ |\mathbf{tries}|(\boldsymbol{x},{}^\wedge(\exists \boldsymbol{y})\ [\mathbf{B}(\boldsymbol{y}) \wedge |\mathbf{finds}|(\boldsymbol{x},\boldsymbol{y})])$

≡ $(\lambda \boldsymbol{x}\ |\mathbf{tries}|(\boldsymbol{x},{}^\wedge\,{}^\wedge|\mathbf{a\ book}|\{\lambda \boldsymbol{y}\ |\mathbf{finds}|(\boldsymbol{x},\boldsymbol{y}))\})$

…

$|\mathbf{seeks}|(|\beta|) \equiv (\lambda \boldsymbol{x}\ |\mathbf{tries}|(\boldsymbol{x},{}^\wedge\,{}^\wedge|\beta|\{\lambda \boldsymbol{y}\ |\mathbf{finds}|(\boldsymbol{x},\boldsymbol{y}))\})$

(8.9b) $|\ \mathbf{seeks}\ | = (\lambda \mathfrak{A}_{\boldsymbol{s}((\boldsymbol{et})\boldsymbol{t})}\ (\lambda \boldsymbol{x}\ |\mathbf{tries}|(\boldsymbol{x},{}^\wedge \mathfrak{A}\{\lambda \boldsymbol{y}\ |\mathbf{finds}|(\boldsymbol{x},\boldsymbol{y}))\})$

(8.10) Opaque verbs: revised type assignment

Category	(Extension) Type
TV	$(\boldsymbol{sq})(\boldsymbol{et})$

(8.11) Opaque verbs: revised meaning combination

Construction	Corresponding Polynomial (ITL)	Corresponding Polynomial (Ty2)
F_{obj}	$G_{obj}(\alpha,\beta) = C_{\text{app}}(\alpha, C_{\text{cap}}(\beta))$ $[= \alpha({}^\wedge\beta)]$	$G_{obj}(\alpha,\beta) = C_{\text{app}}(\alpha, C_{\text{abs}}(\boldsymbol{i},\beta))$ $[= \alpha(\boldsymbol{\lambda i}\ \beta)]$

(8.11) Transitive verbs: revised lexical translation

Item	ITL	Ty2
seeks	$(\lambda\mathfrak{A}\ (\boldsymbol{\lambda x}\ \vert\textbf{tries}\vert(\boldsymbol{x},{}^\wedge\mathfrak{A}\{\boldsymbol{\lambda y}\ \mathbf{F}(\boldsymbol{x},\boldsymbol{y})\}\})$ $(\mathbf{F}\in Con_{e(et)})$	$(\lambda\mathfrak{A}\ (\boldsymbol{\lambda x}\ \mathbf{T}_{\boldsymbol{i}}(\boldsymbol{x}, \boldsymbol{\lambda j}\ \mathfrak{A}_{\boldsymbol{j}}(\boldsymbol{\lambda y}\ \mathbf{F}(\boldsymbol{x},\boldsymbol{y})))))$
finds	$(\lambda\mathfrak{A}\ (\boldsymbol{\lambda x}\ \mathfrak{A}\{\boldsymbol{\lambda y}\ \mathbf{F}(\boldsymbol{x},\boldsymbol{y})\}))$	$(\lambda\mathfrak{A}\ (\boldsymbol{\lambda x}\ \mathfrak{A}_{\boldsymbol{i}}(\boldsymbol{\lambda y}\ \mathbf{F}_{\boldsymbol{i}}(\boldsymbol{x},\boldsymbol{y}))))$
likes	$(\lambda\mathfrak{A}\ (\boldsymbol{\lambda x}\ \mathfrak{A}\{\boldsymbol{\lambda y}\ \mathbf{L}(\boldsymbol{x},\boldsymbol{y})\}))$	$(\lambda\mathfrak{A}\ (\boldsymbol{\lambda x}\ \mathfrak{A}_{\boldsymbol{i}}(\boldsymbol{\lambda y}\ \mathbf{L}_{\boldsymbol{i}}(\boldsymbol{x},\boldsymbol{y}))))$
is	$(\lambda\mathfrak{A}\ (\boldsymbol{\lambda x}\ \mathfrak{A}\{\boldsymbol{\lambda y}\ (\boldsymbol{x} = \boldsymbol{y})\}))$	$(\lambda\mathfrak{A}\ (\boldsymbol{\lambda x}\ \mathfrak{A}_{\boldsymbol{i}}(\boldsymbol{\lambda y}\ (\boldsymbol{x} = \boldsymbol{y}))))$

(8.9a) **Mary is looking for a** [certain] **book on Clinton.**

(b) $(\exists\boldsymbol{x})\ [\ \vert\textbf{book on Clinton}\vert(\boldsymbol{x}) \wedge \vert\textbf{tries}\vert(\mathbf{m},{}^\wedge\mathbf{F}(\mathbf{m},\boldsymbol{x}))]$

(8.10) Scope construction

Construction	Rule(s)	Example
$F_{scope,\boldsymbol{x}}(\Delta,\Delta') =$ $\Delta'[x/\Delta]$, *scope*,***x*** (Δ Δ') $(\boldsymbol{x}\in Var_{\boldsymbol{e}})$	$(F_{pred,\boldsymbol{x}}, \text{NP,S,S})$	$F_{scope,\boldsymbol{x}}$(**a book**,**Mary seeks** ***x***) = **Mary seks a book**, *scope*,***x*** (**a book** … **Mary seeks** ***x*** …)

(8.10) Variables in the lexcion

Item	ITL	Ty2
$\boldsymbol{x}$	$(\boldsymbol{\lambda P}\ \boldsymbol{P}(\boldsymbol{x}))$	$(\boldsymbol{\lambda P}\ \boldsymbol{P}(\boldsymbol{x}))$

(8.11) Scope: meaning combination

Construction	Corresponding Polynomial (ITL and Ty2)
$F_{scope,\boldsymbol{x}}$	$G_{scope,\boldsymbol{x}}(\alpha,\beta) = C_{\text{app}}(\alpha, C_{\text{abs}}(\boldsymbol{x},\beta))$ $[= \alpha(\boldsymbol{\lambda x}\ \beta)]$

(8.10) $(\lambda\mathfrak{A}\ [\ (\exists\boldsymbol{j})\ [\boldsymbol{j}<\boldsymbol{i} \wedge (\forall\boldsymbol{k})\ (\forall\boldsymbol{k}')\ [\boldsymbol{j}\leq \boldsymbol{k}\leq \boldsymbol{k}'\leq \boldsymbol{i} \rightarrow \mathfrak{A}_{\boldsymbol{k}}(\boldsymbol{\lambda x}\ \mathfrak{A}_{\boldsymbol{k}'}(\boldsymbol{\lambda y}\ \boldsymbol{x}\leq \boldsymbol{y}))]]$

$\wedge \quad (\exists\boldsymbol{j})\ [\boldsymbol{j}>\boldsymbol{i} \wedge (\forall\boldsymbol{k})\ (\forall\boldsymbol{k}')\ [\ \boldsymbol{i}\leq \boldsymbol{k}\leq \boldsymbol{k}'\leq \boldsymbol{j} \rightarrow \mathfrak{A}_{\boldsymbol{k}}(\boldsymbol{\lambda x}\ \mathfrak{A}_{\boldsymbol{k}}(\boldsymbol{\lambda y}\ \boldsymbol{x}\leq \boldsymbol{y}))]]]$

(8.11) **The temperature is ninety and <u>it</u> is rising.**