Classical Montague Grammar (Zimmermann)

(This course covers essentially the content of [Montague1970] – with some small changes and additions.)

Part I
Syntax-Semantics Interface

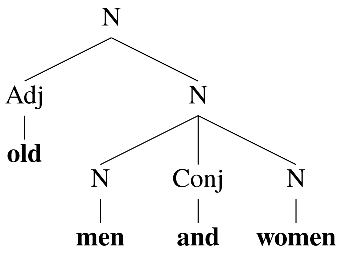
1 Syntax

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

Notation: $(C,\ \kappa_1, ..., \kappa_n, \kappa_{n+1})$

(1.2) **old men and old women**

(1.3)

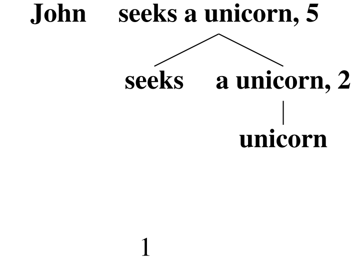


(USC) Uniqueness of Structure Constraint on an algebra $(\Sigma, (C_i)_{i \in I})$:
If $C_i(\Delta_1, ..., \Delta_n) = C_j(\Delta'_1, ..., \Delta'_m)$, then $i = j$, and $(\Delta_1, ..., \Delta_n) = (\Delta'_1, ..., \Delta'_m)$ (and hence $C_i = C_j$, $m = n$, $\Delta_1 = \Delta'_1$,..., $\Delta_n =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

(1.4) $({}_N({}_{Adj}\ \textbf{old})_{Adj}\ ({}_N({}_N\ \textbf{men})_N\ ({}_{Conj}\ \textbf{and})_{Conj}\ ({}_N\ \textbf{women})_N)_N)_N$

(1.5) **⋹John ∈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

- $(\Sigma, (C_i)_{i\in I})$ statisfies (USC);
- the Lexikon $\bigcup_{k\in K} L_k$ generates the set Σ of (syntactic) structures; (i.e. no lexical 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 $k_j \in K$[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 $\Delta = C_i(\Delta_1, ..., \Delta_n)$ (and thus $\rho(\Delta) = max(\rho(\Delta_1), ..., \rho(\Delta_n)) + 1$), $\Delta_1, ..., \Delta_n$ are of categories $k_1, .., k_n$, respectively, and $(C_i, k_1, ..., k_n, k^*) \in R$.

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)'.

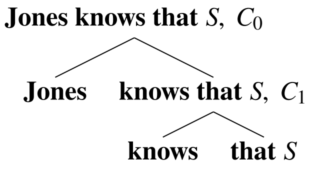
(2.3) $C(\Delta, \Delta') =$ '(Δ **and** Δ')', whenever Δ and Δ' are structures

(2.4) If $C(\Delta_1, ..., \Delta_n)$ is a structure, then its meaning is uniquely determined by the meanings of $\Delta_1, ..., \Delta_n$ 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 $C(\Delta_1, ..., \Delta_n)$ is $M(b_1, ..., b_n)$, where b_1 is Δ_1's meaning, etc.

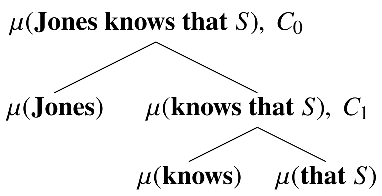
(2.6) $\mu(C(\Delta_1, ..., \Delta_n)) = M(\mu(\Delta_1), ..., \mu(\Delta_n))$

(2.7)

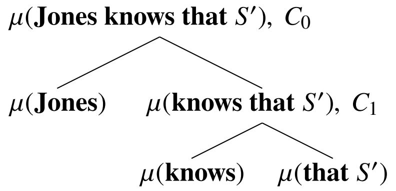


(2.8) $M_0(\mu(\textbf{Jones}), M_1(\mu(\textbf{know}), \mu(\textbf{that}\ S)))$

(2.9)


μ(**Jones knows that** *S*), C_0
μ(**Jones**)
μ(**knows that** *S*), C_1
μ(**knows**)
μ(**that** *S*)

(2.10)


μ(**Jones knows that** S'), C_0
μ(**Jones**)
μ(**knows that** S'), C_1
μ(**knows**)
μ(**that** S')

(2.11) μ(**that** S) = μ(**that** S')
$\Rightarrow$ μ(**Jones knows that** S') = μ(**Jones knows that** S')

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

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

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

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

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

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

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

$= 712$

Definitions

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; $\mu_0 : \bigcup\limits_{k \in K} L_k \to B$; 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 $\Delta = C_i(\Delta_1, ..., \Delta_n)$ and its *meaning* [...] is $\mu(C(\Delta_1, ..., \Delta_n)) = M(\mu(\Delta_1), ..., \mu(\Delta_n))$, where $\mu(\Delta_1), ..., \mu(\Delta_n)$ are the respective meanings [...] of $\Delta_1, ..., \Delta_n$.

(2.15) $(\exists \boldsymbol{x})\mathbf{P}(\boldsymbol{x})$

(2.16) $\mathbf{P}(\boldsymbol{a})$

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

(2.18) $(\forall \boldsymbol{x})\mathbf{P}(\boldsymbol{x})$

(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 $f(C,$b[X]$)$, where X is some set of structures of ranks less than Δ.

Part II
Meaning and Reference

3 Local Perspective

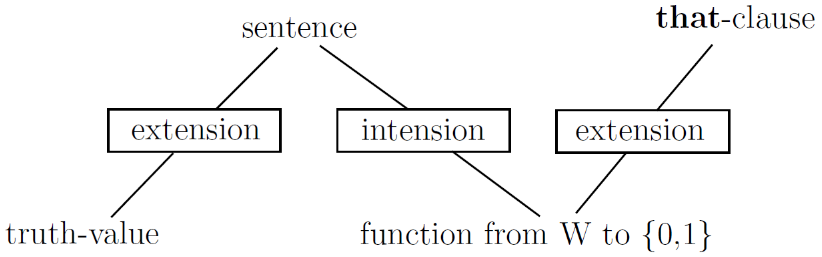
(3.1)

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

(iii) $\mathbf{I}_a = \mathbf{E}_{(s,a)}$; $\mathbf{M}_{\boldsymbol{a}} = \mathbf{I}_{\boldsymbol{a}}{}^{\mathbf{C}}$. 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 $w \in \mathbf{W}$.

Definition

A *local (Fregean) language* is a quintuple $(\Xi, f, (M_i)_{i\in I}, \mu_0, \Delta)$, 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 $F(\mathrm{S}) = \mathbf{t}$;
- $\mu_0 : \bigcup_{k\in K} L_k \to \bigcup_{a\in\mathbf{IT}} \mathbf{M}_a$ 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 $M_i(b_1, ..., b_n) \in \mathbf{M}_{f(k_0)}$ whenever $b_1 \in \mathbf{M}_{f(k_1)}, ..., b_n \in \mathbf{M}_{f(k_n)}$ and $(M_i, k_1, ..., k_n, k_0) \in R$.
- $\mathbf{\Delta} \subseteq \mathbf{W} \times \mathbf{C}$ is the *diagonal* or set of *utterance points*.

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

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

(3.16) $M(b_1, ..., b_n)(c) = M'(b_1(c), ..., b_n(c))(c)$

(3.17) δ is *deictic* iff $\mu(\delta)(c)(w) = \mu(\delta)(c)(w')$, 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 $\left\{ \begin{array}{c} \text{reference} \\ \text{utterance} \end{array} \right\}$.

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

4 Global Perspective

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

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

(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:

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

Definition

(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 $\left\{\begin{array}{c} deictic \\ a\ hyponym\ of\ \delta' \\ an\ a\ priori\ truth \end{array}\right\}$ in a global language iff δ is $\left\{\begin{array}{c} deictic \\ a\ hyponym\ of\ \delta' \\ an\ a\ priori\ truth \end{array}\right\}$ 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 $\{\mu_L(\varphi)(c)(w), \mu_L(\psi)(c)(w) \mid (w, c)$ is a point of reference of $L\}$ has 4 members.

Part III
Indirect Interpretation

5 Translation

(5.1)

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

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

(5.3) A *syntactic polynomial* (over a given syntax) is a term of the form $\boldsymbol{F}(\boldsymbol{X_1}, ..., \boldsymbol{X_n})$, 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 $g(S) = S'$;
- $t : \bigcup_{k\in K} L_k \to \Sigma'$ 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 $(C_i, k_1, ..., k_n, k^*) \in R$ and $\Delta_1, ..., \Delta_n$ are structures of the respective categories $k_1, ..., k_n$, then $T_i(\Delta_1, ..., \Delta_n)$ is of category $g(k^*)$. Given any $\Delta \in \Sigma$, then either $\Delta \in L_{k^*}$ and its *translation* $|\Delta|$ (according to $(g, t, (T_i)_{i\in I})$ is $t(\Delta)$; or else $\Delta = C_i(\Delta_1, ..., \Delta_n)$ and its translationtranslation [...] is $|C_i(\Delta_1, ..., \Delta_n))| = T_i(|\Delta_1|, ..., |\Delta_n|)$, where $|\Delta_1|, ..., |\Delta_n|$ are the respective translations [...] of $\Delta_1, ..., \Delta_n$.

6 Intensional Type Logic (ITL)

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

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

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

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 $h : \bigcup\limits_{a\in\mathbf{IT}} Var_a \to \bigcup\limits_{a\in\mathbf{IT}} E_a$ such that $h(\boldsymbol{x}) \in E_a$ whenever $\boldsymbol{x} \in Var_a$.

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

- Ξ is the syntax of ITL;
- $f(a) = f((\mathrm{VAR}, a)) = a$, for any $a \in \mathbf{IT}$;
- for any $b, b' \in \bigcup\limits_{a\in\mathbf{IT}} E_a, w, w' \in W$, and $h \in C$ the following hold:

  $M_{app}(b, b')(h)(w) = b(h)(w)(b'(h)(w))$ whenever $b \in M_{a,b}$ and $b' \in M_a$;
  $M_{abs}(\mu_0(\boldsymbol{x}), b)(h)(w)(u) = b(h[\boldsymbol{x}/u])(w)$ if $\boldsymbol{x} \in Var_a, u_a$, and $b \in M_{(a,b)}$, where $h[\boldsymbol{x}/u] = (h\backslash\{(\boldsymbol{x}, h(\boldsymbol{x}))\}) \cup \{(\boldsymbol{x}, u)\}$;
  $M_{id}(b, b')(h)(w) = \{\emptyset | b(h)(w) = b'(h)(w)\}$ whenever $b, b' \in M_a$;
  $M_{cup}(b)(h)(w) = b(h)(w)(w)$ whenever $b \in M_{(\mathbf{s},a)}$;
  $M_{cap}(b)(h)(w)(w') = b(h)(w')$.
- $\mu_0(\boldsymbol{c})(h)(w) = \mu_0(\boldsymbol{c})(h')(w)$ whenever $w \in W, h, h' \in C$ and $\boldsymbol{c} \in \bigcup\limits_{a\in\mathbf{IT}} Con_a$;

  $\mu_0(\boldsymbol{x})(h)(w) = h(\boldsymbol{x})$ whenever $w \in W, h \in C$ and $\boldsymbol{x} \in \bigcup\limits_{a\in\mathbf{IT}} Var_a$;
- $\boldsymbol{\Delta} = W \times C$.

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

Given a local ITL-language M, the exists a function $F : \bigcup\limits_{a\in IT} Con_a \to \bigcup\limits_{a\in IT} I_a$ 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) $[\![\alpha]\!]^{M,g,w} = \{(u, [\![\alpha_1]\!]^{M,g[x/u],w})|u \in D_b\}$, if $\alpha = (\boldsymbol{\lambda x}\ \alpha_1)$ und $\boldsymbol{x} \in Var_b$;

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

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

(vii) $[\![\alpha]\!]^{M,g,w} = \{(w', [\![\alpha_1]\!]^{M,g,w'})|w' \in W\}$, 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) $\alpha \in \bigcup_{a \in IT} Var_a$; 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) $\alpha = (\lambda\alpha_1\alpha_2)$, or (ii-c) $\alpha = (\alpha_1 = \alpha_2)$.

Down-Up Cancellation [Gallin1975]

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

Up-Down Cancellation

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

(6.4) Abbreviations in ITL and Ty2:

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

(6.5) Special ITL-conventions:

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

(6.6) $(\boldsymbol{\lambda x}.\, \Box(\boldsymbol{x} = \boldsymbol{d}))(\boldsymbol{c}) \not\equiv \Box(\boldsymbol{c} = \boldsymbol{d})$

Restricted $\boldsymbol{\beta}$-conversion (ITL) [Gallin1975]

$((\boldsymbol{\lambda x}\alpha)(\beta)) \equiv \alpha[\boldsymbol{x}/\beta]$, 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 ${}^\wedge$, or (ii-b) β is modally closed.

Two-sorted Type Theory

$\mathbf{2T}$ 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}.

$\boldsymbol{\beta}$-conversion (Ty2)

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

$$(\boldsymbol{\lambda x}\mathbf{R}(\boldsymbol{x})(\boldsymbol{x})(\boldsymbol{x}))(\mathbf{f}(\mathbf{c})(\mathbf{c})(\mathbf{c})) >_\beta \mathbf{R}(\mathbf{f}(\mathbf{c})(\mathbf{c})(\mathbf{c}))(\mathbf{f}(\mathbf{c})(\mathbf{c})(\mathbf{c}))(\mathbf{f}(\mathbf{c})(\mathbf{c})(\mathbf{c}))$$

$\boldsymbol{\eta}$-conversion (Ty2 & ITL)

$(\boldsymbol{\lambda x}\ \beta(\boldsymbol{x})) \equiv \beta$; if $\boldsymbol{x} \notin Fr(\beta)$

$\boldsymbol{\alpha}$-conversion (Ty2 & ITL)

$(\boldsymbol{\lambda x}\ \alpha) \equiv (\boldsymbol{\lambda y}\ \alpha[\boldsymbol{x}/\boldsymbol{y}])$ iff
no occurrence of $\boldsymbol{x}$ in α lies within the scope of (some) $\boldsymbol{\lambda y}$ and $\boldsymbol{y} \notin Fr((\boldsymbol{\lambda x}\alpha))$.

Definition

(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 $\delta_1 >_\eta \delta_2]$
for some γ, δ_1, δ_2 and variable x (of the appropriate types)

[Notation: $\alpha > \beta$; transitive closure: $\alpha \rhd \beta$]

(b) α is *normal* iff $\alpha \rhd \beta$ implies $\alpha \rhd_{\boldsymbol{\alpha}} \beta$ (where $\rhd_{\boldsymbol{\alpha}}$ is the transitive closure of $>_{\boldsymbol{\alpha}}$)

(c) β is a *normal form of* α iff $\alpha \rhd \beta$ and β is normal.

Normal Form Theorem (Ty2 & IL)

Every Ty2-formula has a normal form.

Church-Rosser Theorem (Ty2)

If β and β' are normal forms of α, then $\beta \rhd_{\boldsymbol{\alpha}} \beta'$.

(6.7a) $(\boldsymbol{\lambda x}\mathbf{P}((\boldsymbol{\lambda y}\ ({}^\wedge\boldsymbol{y}))(\boldsymbol{x})))(\boldsymbol{c})$ (where $\boldsymbol{x} \in Var_{\boldsymbol{se}}, \boldsymbol{y} \in Var_{\boldsymbol{e}}, \boldsymbol{c} \in Con_{\boldsymbol{se}}, \mathbf{P} \in Con_{\boldsymbol{(se)t}}$)

(b) $\mathbf{P}((\boldsymbol{\lambda y}\ ({}^\wedge\boldsymbol{y}))(\boldsymbol{c}))$

(c) $(\boldsymbol{\lambda x}\ \mathbf{P}(({}^\wedge\boldsymbol{x})))(\boldsymbol{c})$ [Friedman and Warren1980]

Gallin's translation ($\boldsymbol{i}$ is a fixed variable in Var_s) [Gallin1975]

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

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

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

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

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

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

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

Four observations on *: [Gallin1975]; [Zimmermann1989]

- $({}^{\vee\wedge}\alpha)^* >_{\boldsymbol{\beta}} \alpha^*$.
- An ITL-formula α is modally closed iff $\boldsymbol{i} \notin Fr(\alpha^*)$.
- If α is modally closed, $({}^{\vee\wedge}\alpha)* >_{\boldsymbol{\eta}} \alpha^*$.
- 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.

(6.8a) $(\lambda\boldsymbol{i}(\lambda\boldsymbol{j}(\boldsymbol{i} = \boldsymbol{j})))$ (where $\boldsymbol{i}, \boldsymbol{j} \in Var_s$)

(b) $(\lambda\boldsymbol{i}(\lambda\boldsymbol{F}(\lambda\boldsymbol{i}(\boldsymbol{F} = (\lambda\boldsymbol{p}\ \boldsymbol{p}(\boldsymbol{i})))))((\lambda\boldsymbol{p}\ \boldsymbol{p}(\boldsymbol{i})))\ )$ (where $F \in Var_{(\boldsymbol{st})\boldsymbol{t}}, p \in Var_{st}$)

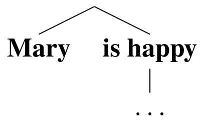
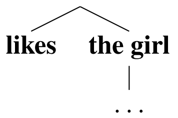
(c) $({}^{\wedge}(\lambda\boldsymbol{F}\ ({}^{\wedge}(\boldsymbol{F} = (\lambda\boldsymbol{p}\ ({}^{\vee}\boldsymbol{p})))))((\lambda\boldsymbol{p}\ ({}^{\vee}\boldsymbol{p}))))$

Part IV

Descriptive Montague Grammar

7 Extensional Constructions

(7.1) Simple constructions

Construction	Corresponding Rule(s)	Example
$F_{pred}(\Delta, \Delta') =$ $\Delta\Delta', pred$ 	(F_{pred},VP,NP,S)	F_{pred}(**is happy**, **Mary**) = **Mary is happy**, *pred* 
$F_{obj}(\Delta, \Delta') =$ $\Delta\Delta', obj$ 	(F_{obj},TV,NP,VP)	F_{obj}(**likes**, **the girl**) = **likes the girl**, *obj* 
$F_{cop}(\Delta) =$ **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**	$\boldsymbol{e(et)}$
Adj	**happy**	$\boldsymbol{et}$
N	**girl**; **boy**	$\boldsymbol{et}$

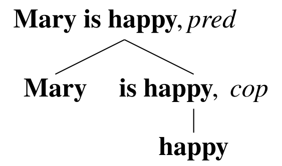
(7.3) Naive lexical translation

Item	Example	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** [$\in Con_{\boldsymbol{et}}$]	$\mathbf{H}_{\boldsymbol{i}}$

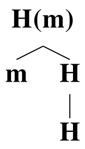
(7.4) Naive meaning combinations

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

(7.5a)



(a′)



$= |F_{pred}(\mathbf{Mary}, F_{cop}(\mathbf{happy}))|$

$= G_{pred}(|\mathbf{Mary}|, |F_{cop}(\mathbf{happy})|)$

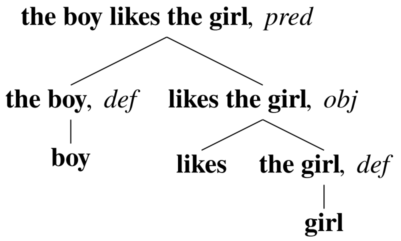
$= G_{pred}(|\mathbf{Mary}|, G_{cop}(|\mathbf{happy}|))$

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

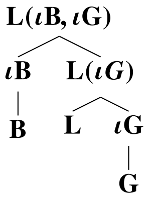
$= G_{pred}(\mathbf{m}, \mathbf{H})$

$= \mathbf{H(m)}$

(b)



(b′)



$= |F_{pred}(F_{def}(\mathbf{boy})), F_{obj}(\mathbf{likes}, F_{def}(\mathbf{girl}))|$

$= G_{pred}(|F_{def}(\mathbf{boy})|, |F_{obj}(\mathbf{likes}, F_{def}(\mathbf{girl}))|)$

$= G_{pred}(G_{def}(|\mathbf{boy}|), G_{obj}(|\mathbf{likes}|, G_{def}(|\mathbf{girl}|)))$

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

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

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

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

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

(7.6) **Every boy likes Mary**

Deriving functional extensions [Frege1891]

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

- $\rho_\alpha(\mu(\beta)(c)(w)) = \mu(F(\alpha, \beta))(c)(w)$ direct version
- $|\alpha|(|\beta|) \equiv |F(\alpha, \beta)|$ indirect version
- $|\alpha| = (\boldsymbol{\lambda x}\ |F(\alpha, \beta)|[^{|\beta|}/_{\boldsymbol{x}}])$ abstract version

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

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

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

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

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

$|\textbf{every}|(|\textbf{girl}|) \equiv (\boldsymbol{\lambda Q_{et}}(\forall \boldsymbol{x})[|\textbf{girl}|(\boldsymbol{x}) \rightarrow \boldsymbol{Q}(\boldsymbol{x})])$ ...

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

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

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

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

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

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

$|\textbf{every boy or every girl}|(|\beta|)$

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

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

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

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

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

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

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

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

$\equiv$ $(\boldsymbol{\lambda Q_{et}}[|\textbf{every boy}|(\boldsymbol{Q}) \vee |\textbf{some girl}|(\boldsymbol{Q})])$ ...

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

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

(7.11) |**Mary or every boy**|

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

$\not\equiv$ $(\lambda Q_{et}[|\textbf{Mary}|(Q)] \vee |\textbf{every girl}|(Q)])$ wrong type ($\boldsymbol{e}$ vs. $\boldsymbol{q}$)

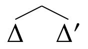
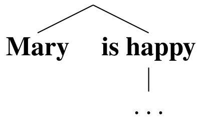
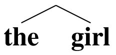
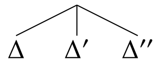
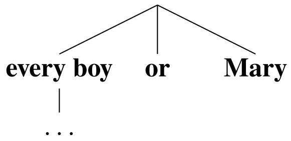
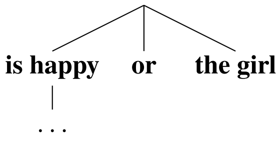
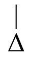
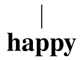
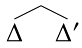
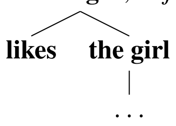
(7.11a) $|\textbf{Mary}|(|\textbf{likes Mary}|) \equiv |\textbf{likes Mary}|(\mathbf{m})$

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

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

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

(7.12) Revised rules and constructions

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

(7.13) Revised (and expanded) type assignment

Category	(Extension) Type
S	$\boldsymbol{t}$
NP	$\boldsymbol{q}$
VP	$\boldsymbol{et}$
TP	$\boldsymbol{e(et)}$
Adj	$\boldsymbol{et}$
Conj	$\boldsymbol{q(qq)}$

(7.14) Lexical translation: revisions and additions

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

(7.15) New meaning combinations

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

8 Intensional Constructions

Attitude verbs

(8.1a) **John thinks that Mary is happy.**

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

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

(8.1d) **John thinks that every boy likes Mary.**

Opaque verbs

(8.2a) **John is looking for a book on Clinton.**

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

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

(8.2d) **John is looking for a book by Clinton.**

Core-intensional verbs [Montague1973]

(8.3a) **The temperature is rising.**

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

(8.3c) **Ninety is rising.**

(8.4) Attitude reports: type assignment

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

(8.5) Attitude reports:additional rules and constructions

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

(8.6)

Item	ITL	Ty2
thinks	$\mathbf{T}$ $[\in Con_{(\boldsymbol{st})(\boldsymbol{et})}]$	$\mathbf{T}_{\boldsymbol{i}}$
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}	$G_{hat}(\alpha) = C_{cap}(\alpha)$ $[=^{\wedge} \alpha]$	$G_{hat}(\alpha) = C_{abs}(\boldsymbol{i}, \alpha)$ $[= (\boldsymbol{\lambda i}\ \alpha)]$
F_{att}	$G_{att}(\alpha, \beta) = C_{app}(\alpha, \beta)$ $[= \alpha(\beta)]$	$G_{att}(\alpha, \beta) = C_{app}(\alpha, \beta)$

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

(8.9a) $|\textbf{seeks}|(|\textbf{a book}|)$

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

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

. . .

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

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

(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}	$G_{obj}(\alpha, \beta) = C_{app}(\alpha, C_{cap}(\beta))$ $[= \alpha(^\wedge\beta)]$	$G_{obj}(\alpha, \beta) = C_{app}(\alpha, C_{abs}(\boldsymbol{i}, \beta))$ $[= \alpha(\lambda \boldsymbol{i} \beta)]$

(8.12) Transitive verbs: revised lexical translation

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

(8.13a) **Mary is looking for a** [certain] **book on Clinton.**

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

(8.14) Scope construction

Construction	Rule(s)	Example
$F_{scope,\boldsymbol{x}}(\Delta, \Delta') = \Delta'[\boldsymbol{x}/\Delta], scope, \boldsymbol{x}$ $\Delta \quad \Delta'$ $(\boldsymbol{x} \in Var_{\boldsymbol{e}})$	$(F_{scope,\boldsymbol{x}}, NP, S, S)$	$F_{scope,\boldsymbol{x}}$(**a book**, **Mary seeks** $\boldsymbol{x}$) = **Mary seeks a book**, $scope, \boldsymbol{x}$ **a book** **Mary seeks** $\boldsymbol{x}$ | | . . . . . .

(8.15) Variables in the lexicon

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

(8.16) Scope: meaning combination

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

(8.17) $(\exists \boldsymbol{j})(\exists \boldsymbol{k})[\boldsymbol{j} < \boldsymbol{i} < \boldsymbol{k} \wedge (\forall \boldsymbol{h})[\boldsymbol{j} < \boldsymbol{h} < \boldsymbol{k} \rightarrow \mathfrak{A}_{\boldsymbol{j}}(\boldsymbol{\lambda x}.\ \mathfrak{A}_{\boldsymbol{h}}(\boldsymbol{\lambda y}.\ \boldsymbol{x} < \boldsymbol{y})) \wedge \mathfrak{A}_{\boldsymbol{h}}(\boldsymbol{\lambda x}.\ \mathfrak{A}_{\boldsymbol{k}}(\boldsymbol{\lambda y}.\ \boldsymbol{x} < \boldsymbol{y}))]$

(8.18) **The temperature is ninety and it is rising.**

References

[Frege1891] Frege, G. (1891). *Function und Begriff.* Pohle, Jena.

[Friedman and Warren1980] Friedman, J. and Warren, D. S. (1980). λ-Normal Forms in an Intensional Logic for English. *Studia Logica*, XXXIX(2-3):311–324.

[Gallin1975] Gallin, D. (1975). *Intensional and Higher-order Modal Logic*. North-Holland Pub. Company, Amsterdam.

[Montague1970] Montague, R. (1970). Universal Grammar. *Theoria*, 36(3):373–398.

[Montague1973] Montague, R. (1973). The Proper Treatment of Quantification in Ordinary English. In Hintikka, J., Moravcsik, J., and Suppes, P., editors, *Approaches to Natural Language*, pages 221–242. Reidel, Dordrecht.

[Zimmermann1989] Zimmermann, T. E. (1989). Intensional Logic and Two-sorted Type Theory. *Journal of Symbolic Logic*, 54(1):65–77.