Natural language as a programming language

Michael Moortgat

Bolzano, November 2009

1. Outline

I compare two foundational principles of computational semantics:

- Montague: compositionality as a homomorphism syntax $\rightsquigarrow$ semantics
- Curry: formulas-as-types, proofs-as-programs

I discuss the role of these principles within two paradigms:

- Abstract Categorial Grammar
  - abstract language: tectogrammatical structure
  - object language: surface realization; semantic interpretation
  - encoding rewriting grammar formalisms in λ calculus
- Typelogical Grammar
  - directional type logics for syntax
  - symmetry: expressions as values or contexts for evaluation (continuations)
  - syntax $\rightsquigarrow$ semantics: continuation-passing-style translation

2. Compositionality

- central design principle of computational semantics: Frege's principle

  'the meaning of an expression is a function of the meaning of its parts and of the way they are syntactically combined' (Partee)

- Montague's Universal Grammar program: compositionality as a homomorphism

$$\langle (A_s)_{s \in S}, F \rangle \xrightarrow{\quad h \quad} \langle (B_t)_{t \in T}, G \rangle$$

  - syntactic algebra A with sorts (categories) S, operations F
  - semantic algebra B with sorts (types) T, operations G
  - homomorphism h: a mapping that respects (i) sorts and (ii) operations:

$$(i) \quad h[A_s] \subseteq B_{\sigma(s)}$$

$$(ii) \quad h(f(a_1, \ldots, a_n)) \quad = \quad g(h(a_1), \ldots, h(a_n)))$$

- the principle does not put interesting restrictions on the syntax/semantics itself

3. Simply typed lambda calculus

Simple types given a finite set of atomic types $\mathcal{A}$,

$$\mathcal{T}_{\mathcal{A}} \quad ::= \quad \mathcal{A} \quad | \quad \mathcal{T}_{\mathcal{A}} \rightarrow \mathcal{T}_{\mathcal{A}}$$

Signature type assignment to constants: $\Sigma = \langle \mathcal{A}, C, \tau \rangle$

- $\mathcal{A}$: finite set of atomic types
- C : finite set of constants
- $\tau : C \rightarrow \mathcal{T}_{\mathcal{A}}$ type assignment function

Terms Given set of variables $\mathcal{X}$ and signature $\Sigma = \langle \mathcal{A}, C, \tau \rangle$, the set of lambda terms built upon Σ is inductively defined as ($x \in \mathcal{X}$, $c \in C$)

$$T \quad ::= \quad x \quad | \quad c \quad | \quad \lambda x.T \quad | \quad (T\ T)$$

4. Curry-Howard Correspondence

Logic and computation deep connection between logical derivations and programs

INTUITIONISTIC LOGIC	LAMBDA CALCULUS
formulas	types
connectives	type constructors
proofs	terms
normalization	reduction
...	...

Typing rules

$$\Gamma, x : \alpha \vdash x : \alpha \quad \textit{(var)} \qquad \vdash c : \tau(c) \quad \textit{(cons)}$$

$$\frac{\Gamma, x : \alpha \vdash t : \beta}{\Gamma \vdash \lambda x.t : (\alpha \rightarrow \beta)} \ \textit{(abs)}$$

$$\frac{\Gamma \vdash t : (\alpha \rightarrow \beta) \quad \Gamma \vdash u : \alpha}{\Gamma \vdash (t\ u) : \beta} \ \textit{(app)}$$

5. Linear lambda calculus

Linear logic A suitable subsystem for natural language analysis:

- Intuitionistic logic: copying (Contraction), deletion (Weakening) of assumptions
- Linear Logic: assumptions as resources; every assumption is used exactly once

Linear typing rules

$$x : \alpha \vdash x : \alpha \quad \textit{(var)} \qquad \vdash c : \tau(c) \quad \textit{(cons)}$$

$$\frac{\Gamma, x : \alpha \vdash t : \beta}{\Gamma \vdash \lambda x.t : (\alpha \to \beta)} \quad x \notin \mathrm{dom}(\Gamma) \qquad \textit{(abs)}$$

$$\frac{\Gamma \vdash t : (\alpha \to \beta) \quad \Delta \vdash u : \alpha}{\Gamma, \Delta \vdash (t\ u) : \beta} \quad \mathrm{dom}(\Gamma) \cap \mathrm{dom}(\Delta) = \emptyset \qquad \textit{(app)}$$

Division of labour derivational semantics: LL; lexical semantics: IL.

6. Abstract categorial grammar

Key idea Derive both surface forms and semantic interpretation from a more abstract source: Curry's tectogrammatical structure.

Interpretations Given source $\Sigma_1 = \langle \mathcal{A}_1, C_1, \tau_1 \rangle$, target $\Sigma_2 = \langle \mathcal{A}_2, C_2, \tau_2 \rangle$, a compositional interpretation $\mathcal{L}$ is a pair of functions $\langle \eta, \theta \rangle$ such that

- ▷ $\eta : \mathcal{A}_1 \longrightarrow \mathcal{T}_{\mathcal{A}_2}$
- ▷ $\theta : C_1 \longrightarrow \Lambda_{\Sigma_2}$
- ▷ $\vdash \theta(c) : \widehat{\eta}(\tau_1(c))$

(with $\widehat{\eta}$, $\widehat{\theta}$ the homomorphic extensions of η, θ)

Abstract categorial grammar $\mathcal{G} = \langle \Sigma_1, \Sigma_2, \mathcal{L}, s \rangle$ (start symbol s)

- abstract language: $\textsc{source}(\mathcal{G}) = \{t \in \Lambda_{\Sigma_1} \mid \ \vdash t : s \quad \text{is derivable}\}$
- object language: $\textsc{target}(\mathcal{G}) = \{t \in \Lambda_{\Sigma_1} \mid \ \exists u \in \textsc{source}(\mathcal{G}). t = \mathcal{L}(u)\}$

7. Example: 'John seeks a unicorn'

Source signature Σ_0 $(\{n, np, s\}, \{J, S, A, U\},$
$\{J : np, U : n, A : n \rightarrow ((np \rightarrow s) \rightarrow s), S : ((np \rightarrow s) \rightarrow s) \rightarrow (np \rightarrow s)\})$

Target signature Σ_1: form $(\{$*string*$\}, \{$john, seeks, a, unicorn$\}, \{$john, seeks, a, unicorn : *string*$\})$

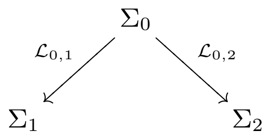
Interpretation: tecto $\rightsquigarrow$ form types: $\eta(n) = \eta(np) = \eta(s) =$ *string*; constants:

$$\theta : \begin{array}{lcl} J & : & \text{john} \\ S & : & \lambda p \lambda x.p(\lambda y.x \cdot \text{seeks} \cdot y) \\ A & : & \lambda x \lambda p.p(\text{a} \cdot x) \\ U & : & \text{unicorn} \end{array}$$

Target Σ_2: meaning $(\{e, t\}, \{\text{J}, \text{SEEK}, \text{UNICORN}, \wedge, \exists\}, \{\text{J} : e, \text{UNICORN} : e \rightarrow t, \wedge : t \rightarrow t \rightarrow t, \exists : (e \rightarrow t) \rightarrow t, \text{SEEKS} : ((e \rightarrow t) \rightarrow t) \rightarrow (e \rightarrow t)\})$

Interpretation: tecto $\rightsquigarrow$ meaning types: $\eta(np) = e, \eta(n) = e \rightarrow t, \eta(s) = t$; constants $J : \text{J}, U : \text{UNICORN}, S : \text{SEEK}, A : \lambda p \lambda q.(\exists\ \lambda x.(p\ x) \wedge (q\ x))$

8. ‘John seeks a unicorn’: derivations



Abstract terms $t_1 : (S\ (A\ U)\ J)$; $\quad t_2 : (A\ U\ \lambda x.(S\ \lambda k.(k\ x))\ J))$

Interpretation: form $\mathcal{L}_{0,1}(t_1) = \mathcal{L}_{0,1}(t_2) = \mathsf{john} \cdot \mathsf{seeks} \cdot \mathsf{a} \cdot \mathsf{unicorn}$

Interpretation: meaning

$$\mathcal{L}_{0,2}(t_1) = (\textsc{seek}\ \lambda q.(\exists\ \lambda x.(\textsc{unicorn}\ x) \wedge (q\ x))\ \textsc{j})$$

$$\mathcal{L}_{0,2}(t_2) = (\exists\ \lambda x.(\textsc{unicorn}\ x) \wedge (\textsc{seek}\ \lambda p.(p\ x)\ \textsc{j}))$$

- each of the interpretations (form, meaning) are compositional homomorphisms
- the relation form $\rightsquigarrow$ meaning is not: one surface form, two meanings

9. Example: context-free grammars

Source non-terminal symbols $\leadsto$ types; rules $\leadsto$ abstract constants.

$$
\begin{array}{ll}
S \longrightarrow \mathbf{(}\ S\ \mathbf{)} & R_1 : S \rightarrow S \\
S \longrightarrow S\ S & R_2 : S \rightarrow S \rightarrow S \\
S \longrightarrow \epsilon & R_3 : S
\end{array}
$$

Target type: *string*; constants: terminal symbols.

Interpretation $S \mapsto$ *string*,

$$
\begin{array}{lcl}
R_1 & : & \lambda x.\mathbf{(} \cdot x \cdot \mathbf{)} \\
R_2 & : & \lambda x \lambda y.x \cdot y \\
R_3 & : & \lambda x.x
\end{array}
$$

Etcetera λ calculus encoding of well-known grammar formalisms as ACG's, with interesting expressivity/complexity results (de Groote et al)

10. Categorial grammar: the Lambek calculi

Grammaticality judgements sequents $\Gamma \vdash t : B$, where

- Γ is a structure built from $x_1 : A_1, \ldots, x_n : A_n$
- t a term of type B built from the x_i of type A_i

Type logics Different kinds of resource management for Γ:

LOGIC	Γ	ASSOCIATIVE	COMMUTATIVE
LP (=LL)	multiset	✓	✓
L	string	✓	
NL	tree		

11. Directional type logics

Directional types given a finite set of atomic types $\mathcal{A}$,

$$A, B \quad ::= \quad \mathcal{A} \mid A \backslash B \mid B / A$$

Directional terms given a set of variables $\mathcal{X}$,

$$M, N \quad ::= \quad x \mid \lambda^r x.M \mid \lambda^l x.M \mid (M \,\text{`}\, N) \mid (N \,\text{'}\, M)$$

Directional typing rules

$$\frac{\Gamma, x : A \vdash M : B}{\Gamma \vdash \lambda^r x.M : B/A}\ I/ \qquad \frac{x : A, \Gamma \vdash M : B}{\Gamma \vdash \lambda^l x.M : A \backslash B}\ I\backslash$$

$$\frac{\Gamma \vdash M : B/A \quad \Delta \vdash N : A}{\Gamma, \Delta \vdash (M \,\text{`}\, N) : B}\ E/ \qquad \frac{\Gamma \vdash N : A \quad \Delta \vdash M : A \backslash B}{\Gamma, \Delta \vdash (N \,\text{'}\, M) : B}\ E\backslash$$

12. Compositional interpretation

$$\textbf{(N)L}^{\{n,np,s\}}_{/,\backslash} \quad \xrightarrow[\text{homomorphism}]{(\cdot)'} \quad \textbf{LP}^{\{e,t\}}_{\rightarrow}$$

SYNTACTIC CALCULUS — homomorphism — SEMANTIC CALCULUS

Types

$$\begin{array}{lcl} np' & = & e \\ s' & = & t \\ n' & = & e \rightarrow t \end{array}$$

Terms

$$\begin{array}{rcl} x' & = & \widetilde{x} \\ (\lambda^l x.M)' & = & \lambda\widetilde{x}.M' \\ (\lambda^r x.M)' & = & \lambda\widetilde{x}.M' \\ (N \text{ ' } M)' & = & (M' \; N') \\ (M \text{ ` } N)' & = & (M' \; N') \end{array}$$

13. Lost in translation

Desirable semantic terms are often unobtainable as image of **(N)L** proofs:

$$(\Lambda_{\mathbf{NL}})' \subset (\Lambda_{\mathbf{L}})' \subset \Lambda_{\mathbf{LP}}$$

Argument lowering valid in **NL**, hence also in **L**, **LP**

$$\begin{aligned} \mathbf{NL}: &\quad z : (B/(A\backslash B))\backslash C \vdash \lambda^l x.((\lambda^r y.(x \text{ ' } y)) \text{ ' } z) : A\backslash C \\ (\cdot)': &\quad \widetilde{z} \vdash \lambda\widetilde{x}.(\widetilde{z}\ \lambda\widetilde{y}.(\widetilde{y}\ \widetilde{x})) \end{aligned}$$

Function composition invalid in **NL**, valid in **L(P)**

$$\begin{aligned} \mathbf{L}: &\quad y : A\backslash B, z : B\backslash C \vdash \lambda^l x.((x \text{ ' } y) \text{ ' } z) : A\backslash C \\ (\cdot)': &\quad \widetilde{y}, \widetilde{z} \vdash \lambda\widetilde{x}.(\widetilde{z}\ (\widetilde{y}\ \widetilde{x})) \end{aligned}$$

Argument raising valid only in **LP**

$$\mathbf{LP}: \quad x : A \to (B \to C) \vdash \lambda w.\lambda z.(w\ \lambda y.((x\ y)\ z)) : ((A \to C) \to C) \to (B \to C)$$

14. Symmetric categorial grammar

Through the looking glass We introduce a dual perspective on linguistic resources:

value	context (*aka* continuation)
supply	demand
producer	consumer
credit	debet

$$\underbrace{x_1 : A_1, \quad \ldots \quad , x_n : A_n}_{\text{multiplicative conjunction}} \vdash \underbrace{\alpha_1 : B_1, \quad \ldots \quad , \alpha_m : B_m}_{\text{multiplicative disjunction}}$$

- sequents with multiple hypotheses and multiple conclusions
- equilibrium: balance between the supply and the demand side
- computation: surplus on the supply (focus on A_i) or demand side (focus on B_j)

15. Symmetry: types and terms

Types given a finite set of atomic types $\mathcal{A}$,

$$A, B \quad ::= \quad \mathcal{A} \mid A \backslash B \mid B / A \mid A \oslash B \mid B \obslash A$$

Two ways of putting together A value and B continuation:

- implications $A \backslash B$, B / A
- coimplications $A \oslash B$ ('A minus B'), $B \obslash A$ ('B from A')

Terms given a set of regular variables Var and a set of continuation variables $CoVar$,

$$\begin{array}{rrcl}
\text{(commands)} & c & ::= & \langle v \mid e \rangle \\
\text{(terms)} & v & ::= & x \mid \mu\alpha.c \mid v \oslash e \mid e \obslash v \mid \lambda(x,\beta).c \mid \lambda(\beta,x).c \\
\text{(contexts)} & e & ::= & \alpha \mid \widetilde{\mu}x.c \mid v \setminus e \mid e \,/\, v \mid \widetilde{\lambda}(x,\beta).c \mid \widetilde{\lambda}(\beta,x).c
\end{array}$$

16. LG: typing rules

Sequents Three types (cf commands, terms, contexts):

$$X \overset{c}{\vdash} Y \quad | \quad X \overset{v}{\vdash} A \quad | \quad A \overset{e}{\vdash} Y$$

Identity Axiom, co-axiom; cut.

$$x : A \overset{x}{\vdash} A \qquad \frac{X \overset{v}{\vdash} A \quad A \overset{e}{\vdash} Y}{X \overset{\langle v|e\rangle}{\vdash} Y}\ \textit{cut} \qquad A \overset{\alpha}{\vdash} \alpha : A$$

Focusing Activate a passive formula.

$$\frac{X \overset{c}{\vdash} \alpha : A}{X \overset{\mu\alpha.c}{\vdash} A}\ \mu \qquad \frac{x : A \overset{c}{\vdash} Y}{A \overset{\widetilde{\mu}x.c}{\vdash} Y}\ \widetilde{\mu}$$

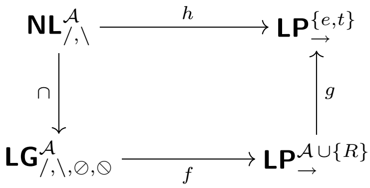
17. LG: logical rules

Left and right introduction rules for the connectives.

$$\frac{X \overset{v}{\vdash} A \quad B \overset{e}{\vdash} Y}{A\backslash B \overset{v \backslash e}{\vdash} X \cdot \backslash \cdot Y} \backslash L \qquad \frac{X \overset{v}{\vdash} A \quad B \overset{e}{\vdash} Y}{X \cdot \oslash \cdot Y \overset{v \oslash e}{\vdash} A \oslash B} \oslash R$$

$$\frac{X \overset{c}{\vdash} x : A \cdot \backslash \cdot \beta : B}{X \overset{\lambda(x,\beta).c}{\vdash} A\backslash B} \backslash R \qquad \frac{x : A \cdot \oslash \cdot \beta : B \overset{c}{\vdash} X}{A \oslash B \overset{\widetilde{\lambda}(x,\beta).c}{\vdash} X} \oslash L$$

18. Two-step interpretation



- f : continuation-passing-style translation (CPS)
  - maps multiple conclusion source logic to intuitionistic linear logic
  - special type R: ‘response of computation’, logic: absurdum $\perp$
- g : the usual syntax-semantics homomorphism
  - $g[\mathcal{A}] = h[\mathcal{A}]$,
  - $g(R) = t$
- target interpretation: $g \circ f$

19. Continuation-passing-style translation

Types Values $\lceil A \rceil$, continuations $\lceil A \rceil \rightarrow R$, computations $(\lceil A \rceil \rightarrow R) \rightarrow R$

$$\begin{array}{rcccl} \lceil B/A \rceil = \lceil A \backslash B \rceil & = & \lceil B \rceil^{\perp} \rightarrow \lceil A \rceil^{\perp} & & \\ \lceil B \oslash A \rceil = \lceil A \oslash B \rceil & = & \lceil A \backslash B \rceil^{\perp} & = & (\lceil B \rceil^{\perp} \rightarrow \lceil A \rceil^{\perp})^{\perp} \end{array}$$

Invariants of the translation

source: $\mathbf{LG}^{\mathcal{A}}_{/,\backslash,\oslash,\oslash} \xrightarrow{\text{CPS}}$ target: $\mathbf{LP}^{\mathcal{A}\cup\{R\}}_{\rightarrow}$

$$\begin{array}{lrcl} X \overset{v}{\vdash} B & \lceil X^{\bullet} \rceil, \lceil X^{\circ} \rceil^{\perp} & \vdash & \lceil v \rceil : \lceil B \rceil^{\perp\perp} \\ A \overset{e}{\vdash} Y & \lceil Y^{\bullet} \rceil, \lceil Y^{\circ} \rceil^{\perp} & \vdash & \lceil e \rceil : \lceil A \rceil^{\perp} \\ X \overset{c}{\vdash} Y & \lceil X^{\bullet} \rceil, \lceil Y^{\bullet} \rceil, \lceil X^{\circ} \rceil^{\perp}, \lceil Y^{\circ} \rceil^{\perp} & \vdash & \lceil c \rceil : R \end{array}$$

($X^{\bullet}$ input/value parts of X; X° output/continuation parts of X)

20. CPS translation: terms

$$
\begin{array}{llcl}
\text{(terms)} & \lceil x \rceil & = & \lambda k.(k\ \widetilde{x}) \\
 & \lceil \lambda(x,\beta).c \rceil = \lceil \lambda(\beta,x).c \rceil & = & \lambda k.(k\ \lambda\widetilde{\beta}\lambda\widetilde{x}.\lceil c \rceil) \\
 & \lceil v \oslash e \rceil = \lceil e \mathbin{\text{⦸}} v \rceil & = & \lambda k.(k\ \lambda u.(\lceil v \rceil\ (u\ \lceil e \rceil))) \\
 & \lceil \mu\alpha.c \rceil & = & \lambda\widetilde{\alpha}.\lceil c \rceil \\
 & & & \\
\text{(contexts)} & \lceil \alpha \rceil & = & \widetilde{\alpha} \quad (= \lambda x.(\widetilde{\alpha}\ x)) \\
 & \lceil v \backslash e \rceil = \lceil e / v \rceil & = & \lambda u.(\lceil v \rceil\ (u\ \lceil e \rceil)) \\
 & \lceil \widetilde{\lambda}(x,\beta).c \rceil = \lceil \widetilde{\lambda}(\beta,x).c \rceil & = & \lambda u.(u\ \lambda\widetilde{\beta}\lambda\widetilde{x}.\lceil c \rceil) \\
 & \lceil \widetilde{\mu}x.c \rceil & = & \lambda\widetilde{x}.\lceil c \rceil \\
 & & & \\
\text{(commands)} & \lceil \langle v \mid e \rangle \rceil & = & (\lceil v \rceil\ \lceil e \rceil)
\end{array}
$$

21. Two step interpretation: illustration

Source

$$\begin{aligned}\Sigma_1 \quad = \quad & (\{np, s\}, \{\textit{mary}, \textit{left}, \textit{teases}, \textit{everybody}, \textit{someone}\}, \\ & \{\textit{mary} \mapsto np, \\ & \textit{left} \mapsto np \backslash s, \\ & \textit{teases} \mapsto (np \backslash s)/np, \\ & \textit{everybody} \mapsto s/(np \backslash s), \\ & \textit{someone} \mapsto (s \oslash s) \obslash np \quad \} \quad)\end{aligned}$$

Target: level 1 The CPS image of the source types and proofs.

$$\begin{aligned}\Sigma_2 \quad = \quad & (\{np, s, R\}, \{\text{mary}, \text{left}, \text{teases}, \text{everybody}, \text{someone}\}, \\ & \{\text{mary} \mapsto np, \\ & \text{left} \mapsto s^{\perp} \rightarrow np^{\perp}, \\ & \text{teases} \mapsto (s^{\perp} \rightarrow np^{\perp})^{\perp} \rightarrow np^{\perp}, \\ & \text{everybody} \mapsto s^{\perp} \rightarrow (s^{\perp} \rightarrow np^{\perp})^{\perp}, \\ & \text{someone} \mapsto ((s^{\perp} \rightarrow s^{\perp})^{\perp\perp} \rightarrow np^{\perp})^{\perp} \quad \} \quad)\end{aligned}$$

22. Step two

Target: level 2 Final target: regular Montagovian λ recipes

$$\begin{aligned}\Sigma_3 \quad = \quad & (\{e, t\}, \{\text{MARY}, \text{LEFT}, \text{TEASES}\},\\ & \{\text{MARY} \mapsto e,\\ & \text{LEFT} \mapsto e \to t,\\ & \text{TEASES} \mapsto e \to (e \to t) \quad \} \quad)\end{aligned}$$

Interpretation from level 1 $\rightsquigarrow$ 2 A mapping $[\![\cdot]\!]$ *lowering* the CPS terms.

$$\begin{aligned}\mathcal{L}_{2,3} \quad = \quad & (\{np \mapsto e, s \mapsto t, R \mapsto t\},\\ & \{\mathsf{mary} \mapsto \text{MARY},\\ & \mathsf{left} \mapsto \lambda c.\lambda x.(c\ (\text{LEFT}\ x)),\\ & \mathsf{teases} \mapsto \lambda v.\lambda y.(v\ \lambda c.\lambda x.(c\ ((\text{TEASES}\ y)\ x))),\\ & \mathsf{everybody} \mapsto \lambda c.\lambda v.(c\ (\forall\ \lambda x.((v\ \mathit{id})\ x))),\\ & \mathsf{someone} \mapsto \lambda h.(\exists\ \lambda x.((h\ \lambda u.(u\ \mathit{id}))\ x)) \quad \} \quad)\end{aligned}$$

23. Derivational ambiguity

In the **LG** source calculus, the following sentence has two proofs.

$$\underbrace{s/(np\backslash s)}_{\text{everybody}} \cdot \otimes \cdot \underbrace{((np\backslash s)/np}_{\text{teases}} \cdot \otimes \cdot \underbrace{(s \oslash s) \obslash np)}_{\text{someone}} \overset{v_1|v_2}{\vdash} s$$

Taking the composition of the CPS translation $\lceil\cdot\rceil$ and the lowering translation $\llbracket\cdot\rrbracket$ produces the desired readings.

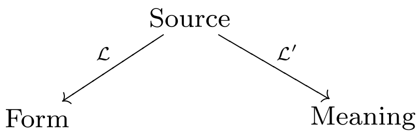
$$\textit{eval}(\llbracket(\lceil v_1 \rceil)\rrbracket) \quad = \quad (\forall\ \lambda x.(\exists\ \lambda y.((\text{TEASES}\ y)\ x)))$$

$$\textit{eval}(\llbracket(\lceil v_2 \rceil)\rrbracket) \quad = \quad (\exists\ \lambda y.(\forall\ \lambda x.((\text{TEASES}\ y)\ x)))$$

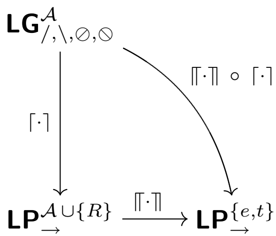
(*eval* provides the identity mapping for the final continuation)

24. Comparison: ACG, LG

Abstract Categorial Grammar abstract tectogrammatical level; diverging compositional mappings to surface forms, semantic readings



Symmetric categorial grammar composition of compositional mappings



25. More to explore

Abstract categorial grammar See the ACG homepage at

`http://www.loria.fr/equipes/calligramme/acg/`

Symmetric categorial grammar See the course wiki

`http://symcg.pbworks.com/`

and

Moortgat (2009) 'Symmetric categorial grammar'. JPL, 38 (6) 681-710.

□