

Extended Lambek Calculus

Yu Jiangsheng

Institute of Computational Linguistics
Peking University 100871

Abstract: *P.Aczel and R.Lunnon(1991) presented Simultaneous Abstraction Calculus to overcome the strict ordering requirements of standard λ -calculus, which suits some natural languages without strict word-ordering like Chinese. Then P.Ruhrberg(1995) simplified Aczel-Lunnon system. In this article, I propose Simple Typed Simultaneous Abstraction and Extended Lambek Calculus to be the fundamentals of Categorical Grammar. More details could be found in [Yu 1999].*

Keywords: type, simultaneous abstraction, category, Lambek calculus

扩展 Lambek 演算

于江生

北京大学计算语言学研究所 100871

摘要: 本文提出简单类型化的同时抽象和扩展的 Lambek 演算作为范畴语法的基本工具, 推广了 Aczel, Lunnon 和 Ruhrberg 等人的工作. 简单类型化的同时抽象克服了传统 λ -演算依赖于抽象次序的缺点, 有利于汉语等一类词序不很严格的自然语言的句法-语义分析. 文章最后给出了几个典型的例子.

关键词: 类型, 同时抽象, 范畴, Lambek 演算

1 Simple Typed Simultaneous Abstraction

Definition *belief lattice* BL is an ordered set $(BL; \preceq)$ satisfying:

1. $\top, t(\text{true}), f(\text{false}), \perp \in BL$
2. $\forall b \in BL. \perp \preceq b \preceq \top$
3. $L.u.b\{t, f\} = \top$

Definition \mathbf{Typ} is the least set of *types* defined recursively by:

1. the set of *basic types* (\mathbf{BasTyp}) includes \mathbf{InTyp} and \mathbf{BeTyp}
2. $\mathbf{BasTyp} \subseteq \mathbf{Typ}$
3. $\mathbf{Typ} = \mathbf{BasTyp} \cup \mathbf{Typ} \times \mathbf{Typ}$

[Note] Every type in $\mathbf{Typ} \times \mathbf{Typ}$ is in form $\sigma \rightarrow \tau$ or $\langle \sigma, \tau \rangle$.

\mathbf{Con}_σ and \mathbf{Var}_σ refer to the set of constants and the set of variables of type σ respectively.

$$\mathbf{Con} = \bigcup_{\sigma \in \mathbf{Typ}} \mathbf{Con}_\sigma, \quad \mathbf{Var} = \bigcup_{\sigma \in \mathbf{Typ}} \mathbf{Var}_\sigma, \quad \mathbf{Dom} = \bigcup_{\sigma \in \mathbf{Typ}} \mathbf{Dom}_\sigma$$

where \mathbf{Dom}_σ denotes the domain of type σ and $\mathbf{Dom}_{\sigma \rightarrow \tau} = \mathbf{Dom}_\tau^{\mathbf{Dom}_\sigma}$.

Definition \mathbf{Term}_σ is the least set of λ -terms of type σ defined by:

1. $\mathbf{Var}_\sigma \cup \mathbf{Con}_\sigma \subseteq \mathbf{Term}_\sigma$
2. *abstraction*: $\lambda\{x_i\}_{i \in I}. t \in \mathbf{Term}_\sigma$
if $\sigma = \tau^{|I|} \rightarrow \rho, \{x_i\}_{i \in I} \subseteq \mathbf{Var}_\tau$, and $t \in \mathbf{Term}_\rho$ ¹

$$\tau^{|I|} \rightarrow \rho \stackrel{\text{def}}{=} \begin{cases} \tau \rightarrow \rho & \text{if } |I| = 1 \\ \tau \rightarrow (\tau^{|I|-1} \rightarrow \rho) & \text{if } |I| > 1 \end{cases}$$

3. *total application*: $t(x_i.t_i)_{i \in I} \in \mathbf{Term}_\sigma$
if $t \in \mathbf{Term}_{\tau|I \rightarrow \sigma}$ and $t_i \in \mathbf{Term}_\tau$, where $r = (x_i.t_i)_{i \in I}$ is called a record
4. *partial application*: $t[x_i.t_i]_{i \in I} \in \mathbf{Term}_\sigma$
if $t \in \mathbf{Term}_{\tau|J \rightarrow \sigma}$ and $t_i \in \mathbf{Term}_\tau$, where $J \subseteq I$

Definition Given a model $\mathfrak{M} = \langle \mathbf{Dom}, [\cdot]_{\mathfrak{M}} \rangle^2$ and assignment function $\theta: \mathbf{Var} \rightarrow \mathbf{Dom}$:

1. $\Psi(d) : \mathbf{Dom}^{\mathbf{Var}} \rightarrow \mathbf{Dom}$ is the *applicative action* of $d \in \mathbf{Dom}$
where $\Psi : \mathbf{Dom} \rightarrow \mathbf{Dom}^{(\mathbf{Dom}^{\mathbf{Var}})}$
2. $\Phi : \mathbf{Dom}^{(\mathbf{Dom}^{\mathbf{Var}})} \rightarrow \mathbf{Dom}$ satisfies:
 $\Psi(\Phi(\varrho)) = \varrho$, where ϱ is an applicative action of some d
3. $\forall V \subseteq \mathbf{Var}$, given $\varepsilon : V \rightarrow \mathbf{Dom}$, define $\theta_V^\varepsilon : \mathbf{Var} \rightarrow \mathbf{Dom}$ satisfying:
$$\theta_V^\varepsilon(x) = \begin{cases} \varepsilon(x) & \text{if } x \in V \\ \theta(x) & \text{if } x \in \mathbf{Var} - V \end{cases}$$

Definition Denotation of a term with respect to \mathfrak{M} and θ :

1. $\forall c \in \mathbf{Con}, [c]_{\mathfrak{M}}^\theta = [c]_{\mathfrak{M}}$
2. $\forall x \in \mathbf{Var}, [x]_{\mathfrak{M}}^\theta = \theta(x)$
3. $[[t(x_i.t_i)_{i \in I}]_{\mathfrak{M}}^\theta = \Psi([t]_{\mathfrak{M}}^\theta)(\langle x_i \mapsto [t_i]_{\mathfrak{M}}^\theta \rangle_{i \in I})$
4. $[[\lambda V.t]_{\mathfrak{M}}^\theta = \Phi(\lambda \varepsilon \mathbf{Var} \rightarrow \mathbf{Dom}. [t]_{\mathfrak{M}}^{\theta_V^\varepsilon})$

Definition The set of free variables of λ -term t ($\mathbf{Free}(t)$) is defined by:

1. $\mathbf{Free}(c) = \emptyset$, if $c \in \mathbf{Con}$
2. $\mathbf{Free}(x) = \{x\}$, if $x \in \mathbf{Var}$
3. $\mathbf{Free}(t(x_i.t_i)_{i \in I}) = \mathbf{Free}(t) \cup (\bigcup_{i \in I} \mathbf{Free}(t_i))$
4. $\mathbf{Free}(\lambda V.t) = \mathbf{Free}(t) - V$

Definition Simultaneous substitution:

1. $(c)[x_i \mapsto t_i]_{i \in I} = c$
2. $(x)[x_i \mapsto t_i]_{i \in I} = \begin{cases} t_i & \text{if } \exists i \in I. x = x_i \\ x & \text{otherwise} \end{cases}$
3. $(t(x_j.t_j)_{j \in J})[x'_i \mapsto t'_i]_{i \in I} = t[x'_i \mapsto t'_i]_{i \in I}(x_j.t_j[x'_i \mapsto t'_i]_{i \in I})_{j \in J}$
4. $(\lambda V.t)[x_i \mapsto t_i]_{i \in I} = \begin{cases} \lambda V.(t[x_j \mapsto t_j]_{j \in J}) & \text{if } V \cap (\bigcup_{j \in J} \mathbf{Free}(t_j)) = \emptyset \\ \perp & \text{where } J = I - \{i | x_i \in V\} \\ & \text{otherwise} \end{cases}$

Definition $\models t = t'$ iff $\forall \mathfrak{M}, \forall \theta, [[t]_{\mathfrak{M}}^\theta = [t']_{\mathfrak{M}}^\theta$ holds.

Theorem

1. $\models \lambda \{x_1, \dots, x_n\}.t = \lambda \{y_1, \dots, y_n\}.(t[x_i \mapsto y_i]_{i \in \{1, \dots, n\}})$

² $[\cdot]_{\mathfrak{M}} : \mathbf{Con}_\sigma \rightarrow \mathbf{Dom}_\sigma$

³ $\theta : \mathbf{Var} \rightarrow \mathbf{Dom}$ s.t. $\forall x \in \mathbf{Var}_\sigma. \theta(x) \in \mathbf{Dom}_\sigma$

$$2. \models \lambda\{x_1, \dots, x_n\}.(\lambda\{x_{n+1}, \dots, x_m\}.t) = \lambda\{x_1, \dots, x_m\}.t$$

Lemma Let $V = \{x_i\}_{i \in I}, \varepsilon = \langle x_i \mapsto \llbracket t_i \rrbracket_{\mathfrak{M}}^{\theta} \rangle_{i \in I}$:

If $t[x_i \mapsto t_i]_{i \in I}$ is defined, then $\llbracket t \rrbracket_{\mathfrak{M}}^{\theta \varepsilon} = \llbracket t[x_i \mapsto t_i]_{i \in I} \rrbracket_{\mathfrak{M}}^{\theta}$

Theorem

1. If $V' = V \cap \mathbf{Free}(t)$, then $\models \lambda V.t = \lambda V'.t$
2. $\models \lambda\{x_i\}_{i \in I}.t(x_j, t_j)_{j \in J} = t[x_j \mapsto t_j, x_i \mapsto \perp]_{j \in J', i \in I'}$
where $J' = J \cap I, I' = I - J$
3. $\models (\lambda\{x_i\}_{i \in I}.t)(x_i, t_i)_{i \in I} = \lambda\{y_i\}_{i \in I}.((t[x_i \mapsto y_i]_{i \in I})(y_i, t_i)_{i \in I})$
where $\forall i \in I, y_i$ is a fresh variable

2 Extended Lambek Calculus

2.1 Category and Lexicon

Definition \mathbf{Cat} is the least set of syntactic category defined recursively by:

1. *basic category*: a finite preordered set⁴
2. $\mathbf{BasCat} \subseteq \mathbf{Cat}$
3. *functor category*: $A/B, A \setminus B \in \mathbf{Cat}$ if $A, B \in \mathbf{Cat}$

Definition *type map* $\mathcal{T} : \mathbf{BasCat} \rightarrow \mathbf{Typ}$ is defined by

$$\begin{cases} \mathcal{T}(\mathbf{NP}) & = \mathbf{IndTyp} \\ \mathcal{T}(\mathbf{S}) & = \mathbf{BelTyp} \\ \mathcal{T}(B/A) & = \mathcal{T}(A) \rightarrow \mathcal{T}(B) \\ \mathcal{T}(A \setminus B) & = \mathcal{T}(A) \rightarrow \mathcal{T}(B) \end{cases}$$

categories	types	descriptions
S	BelTyp	sentence
NP	IndTyp(or e)	noun phase
$\mathbf{N} = \mathbf{NP} \setminus \mathbf{S}$	$\mathbf{IndTyp} \rightarrow \mathbf{BelTyp}$	noun
$\mathbf{Vi} = \mathbf{NP} \setminus \mathbf{S}$	$\mathbf{IndTyp} \rightarrow \mathbf{BelTyp}$	intransitive verb
$\mathbf{Vt} = \mathbf{NP} \setminus \mathbf{S} / \mathbf{NP}$	$\mathbf{IndTyp}^2 \rightarrow \mathbf{BelTyp}$	transitive verb
$\mathbf{Vt}' = \mathbf{NP} \setminus \mathbf{S} / \mathbf{NP} / \mathbf{NP}$	$\mathbf{IndTyp}^3 \rightarrow \mathbf{BelTyp}$	transitive verb
$\mathbf{Vt}'' = \mathbf{NP} \setminus \mathbf{S} / \mathbf{NP} / \mathbf{S}$	$\mathbf{BelTyp} \rightarrow (\mathbf{IndTyp}^2 \rightarrow \mathbf{BelTyp})$	transitive verb
$\mathbf{Vt}''' = \mathbf{NP} \setminus \mathbf{S} / \mathbf{S}$	$\mathbf{BelTyp} \rightarrow (\mathbf{IndTyp} \rightarrow \mathbf{BelTyp})$	transitive verb
$[\iota] = \mathbf{NP} \setminus \mathbf{N}$	$\mathbf{IndTyp}^2 \rightarrow \mathbf{BelTyp}$	article
$[\neg] = \mathbf{S} \setminus \mathbf{S}$	\mathbf{BelTyp}^2	negation
$[\vee] = \mathbf{S} \setminus \mathbf{S} / \mathbf{S}$	\mathbf{BelTyp}^3	disjunction
$[\wedge] = \mathbf{S} \setminus \mathbf{S} / \mathbf{S}$	\mathbf{BelTyp}^3	conjunction
$[\rightarrow] = \mathbf{S} \setminus \mathbf{S} / \mathbf{S}$	\mathbf{BelTyp}^3	implication

Definition *categorial lexicon*(\mathbf{Lex}) is a relation $\mathbf{Lex} \subseteq \mathbf{BasExp} \times (\mathbf{Cat} \times (\mathbf{Term}))$ such that $\forall \langle e, \langle C, t \rangle \rangle \in \mathbf{Lex}, t \in \mathbf{Term}_{\mathcal{T}(C)}$, where \mathbf{BasExp} is the set of basic expressions of \mathcal{L} .⁵

Definition Phrase structure applicative rules:

Forward application $t : A/B^n, \{t_i : B \mid i \in I\} \Rightarrow t[x_i, t_i]_{i \in I} : A$ if $n \leq |I|$

⁴In this article, we define $\mathbf{BasCat} = \{\mathbf{NP}, \mathbf{S}\}$

⁵Equivalent relation $\langle \cdot, \langle C, t \rangle \rangle$ classifies $\mathbf{Lex}, \forall \bar{e} \in \mathbf{Lex} / \langle \cdot, \langle C, t \rangle \rangle, \bar{e} \stackrel{\text{def}}{=} t : C$

Backward application $\{t_i : B \mid i \in I\}, t : B^n \setminus A \Rightarrow t[x_i.t_i]_{i \in I} : A$ if $n \leq |I|$

Definition *Denotation*($[\cdot]_{\text{Lex}}$) of phrase structure:

1. $t : A \in [\epsilon]_{\text{Lex}}$
if $\langle e, \langle A, t \rangle \rangle \in \text{Lex}$
2. $t[x_i.t_i]_{i \in I} : A \in [\epsilon \cdot \{e_j \mid j \in J\}]_{\text{Lex}}$
if $t : A/B^n \in [\epsilon]_{\text{Lex}}, |J| = n (J \subseteq I)$ and $\forall i \in I, t_i : B \in [\epsilon_i]_{\text{Lex}}$
3. $t[x_i.t_i]_{i \in I} : A \in [\epsilon \cdot \{e_j \mid j \in J\}]_{\text{Lex}}$
if $t : B^n \setminus A \in [\epsilon]_{\text{Lex}}, |J| = n (J \subseteq I)$ and $\forall i \in I, t_i : B \in [\epsilon_i]_{\text{Lex}}$

Theorem (Type soundness): $\forall t : C \in [\epsilon]_{\text{Lex}}, t \in \text{Term}_{\tau(C)}$

2.2 Extended Lambek Calculus

Definition *sequent rules*:

1. identity

$$\frac{}{t : A \Rightarrow t : A}^I$$

2. cutting

$$\frac{\Delta \Rightarrow t' : B \quad \Gamma, t' : B, \Gamma' \Rightarrow t : A}{\Gamma, \Delta, \Gamma' \Rightarrow t : A}^C$$

3. elimination

$$\frac{\{\Delta_i \Rightarrow t_i : B \mid i \in I\} \quad \Gamma, t[x_i.t_i]_{i \in I} : A, \Gamma' \Rightarrow t'' : C}{\Gamma, \Delta, t : B^l \setminus A / B^m, \Delta', \Gamma' \Rightarrow t'' : C}^E$$

where $\Delta \cup \Delta' = \{\Delta_i \mid i \in I\}$ and $l = |\Delta|, m = |\Delta'|$

Definition *Extended Lambek Calculus*:

$$\frac{\{x_i : B \mid i \in I\}, \Gamma, \{x_j : C \mid j \in J\} \Rightarrow t : A}{\Gamma \Rightarrow \lambda V.t : B^l \setminus A / C^m}^L$$

where $V = \{x_i \mid i \in I\} \cup \{x_j \mid j \in J\}$ and $l = |I|, m = |J|$

2.3 Examples — comparison of Lambek calculi

topic sentence: 《牛虻》梁山伯读过

		读过	: NP\S/NP	$x : \text{NP}$
	梁山伯	: NP	Has_read	: $e \rightarrow (e \rightarrow t)$
	Liang	: e	读过 x	: NP\S
《牛虻》	: NP	Has_read(x) : $e \rightarrow t$		
Gadfly	: e	梁山伯读过 x	: S	
		Has_read(x)(Liang)	: t	
			↓ abstraction	
		梁山伯读过 \emptyset	: S\NP	
		$\lambda x.(Has_read(x)(Liang))$: $e \rightarrow t$	
	《牛虻》 ₁ 梁山伯读过 \emptyset_1	: S		
	$\lambda x.(Has_read(x)(Liang))(Gadfly)$: t		
			↓ reduction	
	《牛虻》 ₁ 梁山伯读过 \emptyset_1	: S		
	Has_read(Gadfly)(Liang)	: t		

$\left\{ \begin{array}{l} \text{《牛虻》} : \text{NP} \\ \text{Gadfly} : e \\ \text{梁山伯} : \text{NP} \\ \text{Liang} : e \end{array} \right.$	$\left. \begin{array}{l} y : \text{NP} \text{ 读过} : \text{NP} \backslash \text{S} / \text{NP} \quad x : \text{NP} \\ y : e \quad \text{Has_read} : e \rightarrow (e \rightarrow t) \quad x : e \end{array} \right\}$
	$\left. \begin{array}{l} y \text{ 读过 } x : \text{S} \\ \text{Has_read}(x)(y) : t \end{array} \right\}$
	\Downarrow abstraction
	$\emptyset_1 \text{ 读过 } \emptyset_2 : \text{S} \text{NP} \text{NP}$
	$\lambda\{x, y\} . (\text{Has_read}(x)(y)) : e^2 \rightarrow t$
	$\text{《牛虻》}_2 \text{ 梁山伯}_1 \emptyset_1 \text{ 读过 } \emptyset_2 : \text{S}$
	$\lambda\{x, y\} . (\text{Has_read}(x)(y))(y . \text{Liang}, x . \text{Gadfly}) : t$
	\Downarrow total application
	$\text{《牛虻》}_2 \text{ 梁山伯}_1 \emptyset_1 \text{ 读过 } \emptyset_2 : \text{S}$
	$\text{Has_read}(\text{Gadfly})(\text{Liang}) : t$

focus sentence: 梁山伯《牛虻》读过（《红楼梦》没读过）

	$\text{读过} : \text{N} \backslash \text{S} / \text{N} \quad x : \text{N}$
$\left. \begin{array}{l} \text{《牛虻》} : \text{N} \\ \text{Gadfly} : e \end{array} \right.$	$\left. \begin{array}{l} \text{Has_read} : e \rightarrow (e \rightarrow t) \quad x : e \\ \text{读过 } x : \text{N} \backslash \text{S} \\ \text{Has_read}(x) : e \rightarrow t \end{array} \right\}$
	\Downarrow abstraction
	$\text{读过 } \emptyset : (\text{N} \backslash \text{S}) \text{N}$
$\left. \begin{array}{l} \text{梁山伯} : \text{N} \\ \text{Liang} : e \end{array} \right.$	$\left. \begin{array}{l} \lambda x . \text{Has_read}(x) : e \rightarrow (e \rightarrow t) \\ \text{《牛虻》}_1 \text{ 读过 } \emptyset_1 : \text{N} \backslash \text{S} \\ \lambda x . \text{Has_read}(x)(\text{Gadfly}) : e \rightarrow t \end{array} \right\}$
	\Downarrow reduction
	$\text{《牛虻》}_1 \text{ 读过 } \emptyset_1 : \text{N} \backslash \text{S}$
	$\text{Has_read}(\text{Gadfly}) : e \rightarrow t$
	$\text{梁山伯} \text{《牛虻》}_1 \text{ 读过 } \emptyset_1 : \text{S}$
	$\text{Has_read}(\text{Gadfly})(\text{Liang}) : t$

The analysis by Extended Lambek Calculus is the same with topic case.

“被” sentence: 梁山伯被《牛虻》迷住了

$\left. \begin{array}{l} \text{梁山伯} : \text{NP} \\ \text{Liang} : e \end{array} \right.$	$\left. \begin{array}{l} y : \text{NP} \text{ 迷住了} : \text{NP} \backslash \text{S} / \text{NP} \quad x : \text{NP} \\ y : e \quad \text{Abstracts} : e \rightarrow (e \rightarrow t) \quad x : e \end{array} \right\}$
$\left. \begin{array}{l} \text{被} \text{《牛虻》} : \text{NP} \\ \text{Gadfly} : e \end{array} \right.$	$\left. \begin{array}{l} y \text{ 迷住了 } x : \text{S} \\ \text{Abstracts}(x)(y) : t \end{array} \right\}$
	\Downarrow abstraction
	$\emptyset_1 \text{ 迷住了 } \emptyset_2 : \text{S} \text{NP} \text{NP}$
	$\lambda\{x, y\} . (\text{Abstracts}(x)(y)) : e^2 \rightarrow t$
	$\text{梁山伯}_2 \text{ [被] 《牛虻》}_1 \emptyset_1 \text{ 迷住了 } \emptyset_2 : \text{S}$
	$\lambda\{x, y\} . (\text{Abstracts}(x)(y))(y . \text{Gadfly}, x . \text{Liang}) : t$
	\Downarrow total application
	$\text{梁山伯}_2 \text{ [被] 《牛虻》}_1 \emptyset_1 \text{ 迷住了 } \emptyset_2 : \text{S}$
	$\text{Abstracts}(\text{Liang})(\text{Gadfly}) : t$

“把” sentence: 梁山伯把《牛虻》卖了⁶

梁山伯	:	NP	y	:	NP	卖了	:	NP\S/NP	x	:	NP	
<i>Liang</i>	:	e	y	:	e	<i>Sold</i>	:	$e \rightarrow (e \rightarrow t)$	x	:	e	
把 《牛虻》	:	NP	y	卖了	x		:	S				
<i>Gadfly</i>	:	e	$Sold(x)(y)$:	t				
								↓	abstraction			
								\emptyset_1	卖了	\emptyset_2	:	S NP NP
								$\lambda\{x, y\}.$	$(Sold(x)(y))$:	$e^2 \rightarrow t$
梁山伯 ₁ [把] 《牛虻》		\emptyset_2	\emptyset_1	卖了	\emptyset_2		:	S				
$\lambda\{x, y\}.$		$(Sold(x)(y))$	$(y.Liang, x.Gadfly)$:	t				
								↓	total application			
梁山伯 ₁ [把] 《牛虻》		\emptyset_2	\emptyset_1	卖了	\emptyset_2		:	S				
<i>Sold(Gadfly)(Liang)</i>							:	t				

Acknowledgement: With the encouragement of Prof. Yu Shiwen, I've been focusing on Computational Semantics and have learned much in those symposiums of Institute of Computational Linguistics. Another one I must mention is my wife who has still endured my being lazy to do housework.

References

- [Aczel and Lunnon 1991] P. Aczel and R. Lunnon. *Universes and Parameters*. in *Situation Theory and its applications*. Vol. 2, Chap. 1, pp. 3-24. J. Barwise et al (eds.). CSLI and Univ. of Chicago, Stanford.
- [Barwise et al 1981] J. Barwise et al (eds), *The Lambda Calculus-Its Syntax and Semantics*. in *Studies in Logic and the Foundations of Mathematics*, Vol. 103. Amsterdam: North-Holland Publishing Company.
- [Carpenter 1995] B. Carpenter, *Lectures on Type-Logical Semantics*. MIT Press.
- [Oehrle et al 1988] R. T. Oehrle, E. Bach and D. Wheeler (eds.) *Categorial Grammars and Natural Language Structures*. D. Reidel Publishing Company.
- [Partee et al 1990] B. H. Partee, A. ter Meulen, R. E. Wall *Mathematical Methods in Linguistics*. Kluwer Academic Publishers. R. Johnson, *Computational Linguistics and Formal Semantics*. Chicago Univ. Press.
- [Ruheberg 1995] P. Ruheberg. *Simultaneous Abstraction and Semantic Theories*. Ph.D. thesis. also in *Building the Framework* (FraCas-1996).
- [van Benthem 1991] J. van Benthem. *Language in Action-Categories, Lambdas and Dynamic Logic*.
- [Yu 1999] J. S. Yu. *Mathematical Foundations of Semantics*. Ph.D thesis of Peking University.
- [蒋严、潘海华 1998] 蒋严; 潘海华, 《形式语义学引论》. 中国社会科学出版社.

⁶ { 被 } points 《牛虻》 to be the logic { subject } of the main verb { 迷住了 } directly.
 { 把 }