# Introducing a Calculus of Effects and Handlers for Natural Language Semantics

Jirka Maršík    Maxime Amblard

August 20, 2016

LORIA, UMR 7503, Université de Lorraine, CNRS, Inria, Campus Scientifique,
F-54506 Vandœuvre-lès-Nancy, France
**jiri.marsik89@gmail.com**

# Introduction

# Fighting with Compositionality

**Setting**

Formal Semantics

# Fighting with Compositionality

**Setting**

Formal Semantics

**Compositionality**

The meaning of a complex expression is a function of its structure and the meanings of its constituents.

# Fighting with Compositionality

**Setting**

Formal Semantics

**Compositionality**

The meaning of a complex expression is a function of its structure and the meanings of its constituents.

**Case of quantification**

- use $\lambda\mu$ [de Groote, 2001] or `shift`/`reset` [Shan, 2005] functions

## Fighting with Compositionality

### Setting

Formal Semantics

### Compositionality

The meaning of a complex expression is a function of its structure and the meanings of its constituents.

### Case of quantification

- use $\lambda\mu$ [de Groote, 2001] or `shift/reset` [Shan, 2005] functions

# Fighting with Compositionality

**Setting**

Formal Semantics

**Compositionality**

The meaning of a complex expression is a function of its structure and the meanings of its constituents.

**Case of quantification**

- use $\lambda\mu$ [de Groote, 2001] or `shift`/`reset` [Shan, 2005] functions
- use continuized meanings [Barker, 2002], i.e. generalized quantifiers

# Fighting with Compositionality

**Setting**

Formal Semantics

**Compositionality**

The meaning of a complex expression is a function of its structure and the meanings of its constituents.

**Case of quantification**

- use $\lambda\mu$ [de Groote, 2001] or `shift`/`reset` [Shan, 2005] functions
- use continuized meanings [Barker, 2002], i.e. generalized quantifiers

# Fighting with Compositionality

**Setting**

Formal Semantics

**Compositionality**

The meaning of a complex expression is a function of its structure and the meanings of its constituents.

**Case of quantification**

- use $\lambda\mu$ [de Groote, 2001] or `shift`/`reset` [Shan, 2005] functions

  calculi with effects

- use continuized meanings [Barker, 2002], i.e. generalized quantifiers

  term encodings of effects

# Connections to Existing Work

**Pioneers**

- [Shan, 2002, Hobbs and Rosenschein, 1977]

**Parallel developments**

- [Charlow, 2014, Kiselyov, 2015]

**ESSLLI courses**

- [Barker and Bumford, 2015, Giorgolo and Asudeh, 2015]

**Effects and Handlers**

- [Cartwright and Felleisen, 1994, Plotkin and Pretnar, 2013, Bauer and Pretnar, 2015, Kammar et al., 2013, Kiselyov et al., 2013]

# Definition of the $\llparenthesis \lambda \rrparenthesis$ Calculus

# STLC with Computations

Simply typed lambda calculus (STLC) with computation types $\mathcal{F}_E(\gamma)$

**Constructors for $\mathcal{F}_E(\gamma)$**

$$\frac{\Gamma \vdash M : \gamma}{\Gamma \vdash \eta\, M : \mathcal{F}_E(\gamma)}\,[\eta]$$

$$\frac{\Gamma \vdash M_{\mathrm{p}} : \alpha \qquad \Gamma, x : \beta \vdash M_{\mathrm{c}} : \mathcal{F}_E(\gamma) \qquad \mathsf{op} : \alpha \rightarrowtail \beta \in E}{\Gamma \vdash \mathsf{op}\, M_{\mathrm{p}}\, (\lambda x.\, M_{\mathrm{c}}) : \mathcal{F}_E(\gamma)}\,[\mathsf{op}]$$

## Example Computation

$$\Gamma \vdash \texttt{speaker} \star (\lambda s.$$
$$\quad \texttt{implicate}\,(\mathbf{m} = \textbf{best-friend}(s))\,(\lambda\_.$$
$$\quad \eta\,(\textbf{love}(\mathbf{m}, s)))) : \mathcal{F}_E(o)$$

$$E = \{\, \texttt{speaker} : 1 \rightarrowtail \iota,$$
$$\qquad \texttt{implicate} : o \rightarrowtail 1 \,\}$$

# Example Computation

$$\Gamma \vdash \texttt{speaker} \star (\lambda s.\ \texttt{implicate}\,(\mathbf{m} = \textbf{best-friend}(s))\,(\lambda\_.\ \eta\,(\textbf{love}(\mathbf{m}, s)))) : \mathcal{F}_E(o)$$

$$E = \{\ \texttt{speaker} : 1 \rightarrowtail \iota,\ \texttt{implicate} : o \rightarrowtail 1\ \}$$

$[\![$Mary, my best friend, loves me.$]\!]$

# Computations as Programs

$$\Gamma \vdash \texttt{speaker} \star (\lambda s.$$
$$\texttt{implicate} (\mathbf{m} = \textbf{best-friend}(s)) (\lambda\_.$$
$$\eta (\textbf{love}(\mathbf{m}, s)))) : \mathcal{F}_E(o)$$

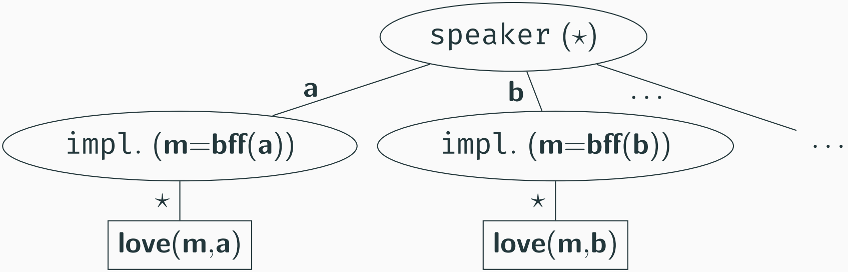
$$\Downarrow$$

$$\mathrm{do}\ s \leftarrow \texttt{speaker} \star$$
$$\texttt{implicate} (\mathbf{m} = \textbf{best-friend}(s))$$
$$\mathrm{return}\ (\textbf{love}(\mathbf{m}, s))$$

# Computations as Algebraic Expressions

$$\begin{aligned}\Gamma \vdash \ & \mathtt{speaker} \star (\lambda s. \\ & \mathtt{implicate}\,(\mathbf{m} = \mathbf{best\text{-}friend}(s))\,(\lambda\_. \\ & \eta\,(\mathbf{love}(\mathbf{m}, s)))) : \mathcal{F}_E(o)\end{aligned}$$

$$\Downarrow$$



# Closed Handlers

$$\frac{\begin{array}{c} E = \{\mathsf{op}_i : \alpha_i \rightarrowtail \beta_i\}_{i \in I} \\ [\Gamma \vdash M_i : \alpha_i \to (\beta_i \to \delta) \to \delta]_{i \in I} \\ \Gamma \vdash M_\eta : \gamma \to \delta \\ \Gamma \vdash N : \mathcal{F}_E(\gamma) \end{array}}{\Gamma \vdash \llparenthesis (\mathsf{op}_i \colon M_i)_{i \in I},\ \eta \colon M_\eta \rrparenthesis\, N : \delta} \, [\llparenthesis\rrparenthesis]$$

# Closed Handlers

$$\frac{\begin{array}{c} E = \{\mathsf{op}_i : \alpha_i \rightarrowtail \beta_i\}_{i \in I} \\ [\Gamma \vdash M_i : \alpha_i \to (\beta_i \to \delta) \to \delta]_{i \in I} \\ \Gamma \vdash M_\eta : \gamma \to \delta \\ \Gamma \vdash N : \mathcal{F}_E(\gamma) \end{array}}{\Gamma \vdash (\!|(\mathsf{op}_i\colon M_i)_{i \in I},\ \eta\colon M_\eta|\!)\, N : \delta}\ [(\!|\,|\!)]$$

## Types of constructors

- $\mathsf{op}_i : \alpha_i \to (\beta_i \to \mathcal{F}_E(\gamma)) \to \mathcal{F}_E(\gamma)$
- $\eta : \gamma \to \mathcal{F}_E(\gamma)$

# Closed Handlers

$$\frac{\begin{array}{c} E = \{\mathtt{op}_i : \alpha_i \rightarrowtail \beta_i\}_{i \in I} \\ [\Gamma \vdash M_i : \alpha_i \to (\beta_i \to \delta) \to \delta]_{i \in I} \\ \Gamma \vdash M_\eta : \gamma \to \delta \\ \Gamma \vdash N : \mathcal{F}_E(\gamma) \end{array}}{\Gamma \vdash (\!|(\mathtt{op}_i\!: M_i)_{i \in I},\ \eta\!: M_\eta|\!)\, N : \delta} \ [(\!|\,|\!)]$$

**Types of constructors**

- $\mathtt{op}_i : \alpha_i \to (\beta_i \to \mathcal{F}_E(\gamma)) \to \mathcal{F}_E(\gamma)$
- $\eta : \gamma \to \mathcal{F}_E(\gamma)$

# Closed Handlers

$$\frac{\begin{array}{c} E = \{\mathsf{op}_i : \alpha_i \rightarrowtail \beta_i\}_{i \in I} \\ [\Gamma \vdash M_i : \alpha_i \to (\beta_i \to \delta) \to \delta]_{i \in I} \\ \Gamma \vdash M_\eta : \gamma \to \delta \\ \Gamma \vdash N : \mathcal{F}_E(\gamma) \end{array}}{\Gamma \vdash \llparenthesis (\mathsf{op}_i \colon M_i)_{i \in I},\ \eta \colon M_\eta \rrparenthesis\, N : \delta} [\llparenthesis\rrparenthesis]$$

## Handlers are algebras

- $(\delta, M_i)$ — an algebra
  - $\delta$ — carrier
  - $M_i$ — operations
- $M_\eta$ — constants

## Types of constructors

- $\mathsf{op}_i : \alpha_i \to (\beta_i \to \mathcal{F}_E(\gamma)) \to \mathcal{F}_E(\gamma)$
- $\eta : \gamma \to \mathcal{F}_E(\gamma)$

# Open Handlers

$$\frac{\begin{array}{c} E = \{\mathsf{op}_i : \alpha_i \rightarrowtail \beta_i\}_{i\in I} \uplus E_\mathrm{f} \\ E' = E'' \uplus E_\mathrm{f} \\ [\Gamma \vdash M_i : \alpha_i \to (\beta_i \to \mathcal{F}_{E'}(\delta)) \to \mathcal{F}_{E'}(\delta)]_{i\in I} \\ \Gamma \vdash M_\eta : \gamma \to \mathcal{F}_{E'}(\delta) \\ \Gamma \vdash N : \mathcal{F}_E(\gamma) \end{array}}{\Gamma \vdash (\!|\, (\mathsf{op}_i \!: M_i)_{i\in I},\ \eta \!: M_\eta \,|\!)\, N : \mathcal{F}_{E'}(\delta)}\ [(\!|\ |\!)]$$

# Open Handlers

$$\frac{\begin{array}{c} E = \{\mathsf{op}_i : \alpha_i \rightarrowtail \beta_i\}_{i \in I} \uplus E_{\mathrm{f}} \\ E' = E'' \uplus E_{\mathrm{f}} \\ [\Gamma \vdash M_i : \alpha_i \to (\beta_i \to \mathcal{F}_{E'}(\delta)) \to \mathcal{F}_{E'}(\delta)]_{i \in I} \\ \Gamma \vdash M_\eta : \gamma \to \mathcal{F}_{E'}(\delta) \\ \Gamma \vdash N : \mathcal{F}_E(\gamma) \end{array}}{\Gamma \vdash (\!|\, (\mathsf{op}_i \colon M_i)_{i \in I},\ \eta \colon M_\eta \,|\!)\, N : \mathcal{F}_{E'}(\delta)} \; [(\!|\ |\!)]$$

- $E$ — input effects
- $E_{\mathrm{f}}$ — forwarded effects
- $E'$ — output effects
- $E''$ — new effects

## Reduction Rules for Handlers

$$\llparenthesis\, (\mathsf{op}_i \colon M_i)_{i \in I},\ \eta \colon M_\eta \,\rrparenthesis\, (\eta\, N) \to$$
$$M_\eta\, N$$

$$\llparenthesis\, (\mathsf{op}_i \colon M_i)_{i \in I},\ \eta \colon M_\eta \,\rrparenthesis\, (\mathsf{op}_j\, N_{\mathrm{p}}\, (\lambda x.\, N_{\mathrm{c}})) \to$$
$$M_j\, N_{\mathrm{p}}\, (\lambda x.\, \llparenthesis\, (\mathsf{op}_i \colon M_i)_{i \in I},\ \eta \colon M_\eta \,\rrparenthesis\, N_{\mathrm{c}}) \qquad \text{where } j \in I$$

$$\llparenthesis\, (\mathsf{op}_i \colon M_i)_{i \in I},\ \eta \colon M_\eta \,\rrparenthesis\, (\mathsf{op}_j\, N_{\mathrm{p}}\, (\lambda x.\, N_{\mathrm{c}})) \to$$
$$\mathsf{op}_j\, N_{\mathrm{p}}\, (\lambda x.\, \llparenthesis\, (\mathsf{op}_i \colon M_i)_{i \in I},\ \eta \colon M_\eta \,\rrparenthesis\, N_{\mathrm{c}}) \qquad \text{where } j \notin I$$

# Reduction Rules for Handlers

$$\llparenthesis\, (\mathsf{op}_i \colon M_i)_{i \in I},\ \eta \colon M_\eta \,\rrparenthesis\, (\eta\, N) \to$$
$$M_\eta\, N$$

$$\llparenthesis\, (\mathsf{op}_i \colon M_i)_{i \in I},\ \eta \colon M_\eta \,\rrparenthesis\, (\mathsf{op}_j\, N_\mathrm{p}\, (\lambda x.\, N_\mathrm{c})) \to$$
$$M_j\, N_\mathrm{p}\, (\lambda x.\, \llparenthesis\, (\mathsf{op}_i \colon M_i)_{i \in I},\ \eta \colon M_\eta \,\rrparenthesis\, N_\mathrm{c}) \qquad \text{where } j \in I$$

$$\llparenthesis\, (\mathsf{op}_i \colon M_i)_{i \in I},\ \eta \colon M_\eta \,\rrparenthesis\, (\mathsf{op}_j\, N_\mathrm{p}\, (\lambda x.\, N_\mathrm{c})) \to$$
$$\mathsf{op}_j\, N_\mathrm{p}\, (\lambda x.\, \llparenthesis\, (\mathsf{op}_i \colon M_i)_{i \in I},\ \eta \colon M_\eta \,\rrparenthesis\, N_\mathrm{c}) \qquad \text{where } j \notin I$$

## Reduction Rules for Handlers

$$\llparenthesis\, (\mathsf{op}_i\colon M_i)_{i\in I},\ \eta\colon M_\eta \,\rrparenthesis\, (\eta\, N) \to$$
$$M_\eta\, N$$

$$\llparenthesis\, (\mathsf{op}_i\colon M_i)_{i\in I},\ \eta\colon M_\eta \,\rrparenthesis\, (\mathsf{op}_j\, N_{\mathrm{p}}\, (\lambda x.\, N_{\mathrm{c}})) \to$$
$$M_j\, N_{\mathrm{p}}\, (\lambda x.\, \llparenthesis\, (\mathsf{op}_i\colon M_i)_{i\in I},\ \eta\colon M_\eta \,\rrparenthesis\, N_{\mathrm{c}}) \qquad \text{where } j \in I$$

$$\llparenthesis\, (\mathsf{op}_i\colon M_i)_{i\in I},\ \eta\colon M_\eta \,\rrparenthesis\, (\mathsf{op}_j\, N_{\mathrm{p}}\, (\lambda x.\, N_{\mathrm{c}})) \to$$
$$\mathsf{op}_j\, N_{\mathrm{p}}\, (\lambda x.\, \llparenthesis\, (\mathsf{op}_i\colon M_i)_{i\in I},\ \eta\colon M_\eta \,\rrparenthesis\, N_{\mathrm{c}}) \qquad \text{where } j \notin I$$

# Chaining Computations

$$\_ \gg= \_ : \mathcal{F}_E(\alpha) \to (\alpha \to \mathcal{F}_E(\beta)) \to \mathcal{F}_E(\beta)$$

$$M \gg= N = (\!|\, \eta \colon N \,|\!)\, M$$

# Chaining Computations

$$\_ \gg\!= \_ : \mathcal{F}_E(\alpha) \to (\alpha \to \mathcal{F}_E(\beta)) \to \mathcal{F}_E(\beta)$$

$$M \gg\!= N = (\!|\, \eta \colon N \,|\!)\, M$$

$$\boxed{A}$$

$\texttt{speaker} \star (\lambda s.$

$\texttt{implicate}\,(\mathbf{m} = \mathbf{bff}(s))\,(\lambda\_.$

$\eta\,\mathbf{m}))$

# Chaining Computations

$$\_ \gg\!= \_ : \mathcal{F}_E(\alpha) \to (\alpha \to \mathcal{F}_E(\beta)) \to \mathcal{F}_E(\beta)$$

$$M \gg\!= N = (\!| \eta : N |\!)\, M$$

$$\boxed{A}$$

$\texttt{speaker} \star (\lambda s.$
$\texttt{implicate}\,(\mathbf{m} = \mathbf{bff}(s))\,(\lambda \_.$
$\eta\, \mathbf{m}))$

$$\boxed{B}$$

$\lambda x.\, \texttt{speaker} \star (\lambda s.$
$\eta\, (\mathbf{love}(x, s)))$

# Chaining Computations

$$\_ \gg\!= \_ : \mathcal{F}_E(\alpha) \to (\alpha \to \mathcal{F}_E(\beta)) \to \mathcal{F}_E(\beta)$$

$$M \gg\!= N = (\!|\, \eta \colon N \,|\!)\, M$$

$\boxed{A}$

`speaker` $\star\ (\lambda s.$

`implicate` $(\mathbf{m} = \mathbf{bff}(s))\ (\lambda \_.$

$\eta\, \mathbf{m}))$

$\boxed{B}$

$\lambda x.$ `speaker` $\star\ (\lambda s.$

$\eta\, (\mathbf{love}(x, s)))$

$\boxed{A \gg\!= B}$

`speaker` $\star\ (\lambda s.$

`implicate` $(\mathbf{m} = \mathbf{bff}(s))\ (\lambda \_.$

`speaker` $\star\ (\lambda s.$

$\eta\, (\mathbf{love}(\mathbf{m}, s)))))$

# Notation: Applying Operations to Computations

$$\_ \circ \_ : \alpha \to \beta \to \gamma$$

$$\_ \ll\!\circ \_ : \mathcal{F}_E(\alpha) \to \beta \to \mathcal{F}_E(\gamma)$$
$$X \ll\!\circ\, y = X \gg\!= (\lambda x.\, \eta\, (x \circ y))$$

$$\_ \circ\!\gg \_ : \alpha \to \mathcal{F}_E(\beta) \to \mathcal{F}_E(\gamma)$$
$$x \circ\!\gg Y = Y \gg\!= (\lambda y.\, \eta\, (x \circ y))$$

$$\_ \ll\!\circ\!\gg \_ : \mathcal{F}_E(\alpha) \to \mathcal{F}_E(\beta) \to \mathcal{F}_E(\gamma)$$
$$X \ll\!\circ\!\gg Y = X \gg\!= (\lambda x.\, Y \gg\!= (\lambda y.\, \eta\, (x \circ y)))$$

# Properties of the $(\!|\,\lambda\,|\!)$ Calculus

# $(\!|\lambda|\!)$ is Strongly Normalizing

**Confluence**

- Combinatory Reduction Systems [Klop et al., 1993]

**Termination**

- Inductive Data Type Systems [Blanqui, 2000]
- Higher-Order Semantic Labelling [Hamana, 2007]

# Linguistic Phenomena as Effects

## Deixis

# Deixis

$$\textsc{John}, \textsc{Mary}, \textsc{me} : \mathit{NP}$$

$$\textsc{loves} : \mathit{NP} \multimap \mathit{NP} \multimap S$$

$$\begin{aligned}
[\![\textsc{John}]\!] &= \eta\, \mathbf{j} \\
[\![\textsc{Mary}]\!] &= \eta\, \mathbf{m} \\
[\![\textsc{me}]\!] &= \mathtt{speaker} \star (\lambda x.\, \eta\, x) \\
[\![\textsc{loves}]\!] &= \lambda OS.\, \mathbf{love} \cdot\!\gg S \ll\!\cdot\!\gg O
\end{aligned}$$

# Deixis — Examples

$$
\begin{aligned}
[\![\textsc{John}]\!] &= \eta\,\mathbf{j} \\
[\![\textsc{Mary}]\!] &= \eta\,\mathbf{m} \\
[\![\textsc{me}]\!] &= \texttt{speaker} \star (\lambda x.\, \eta\, x) \\
[\![\textsc{loves}]\!] &= \lambda OS.\, \mathbf{love} \cdot\!\gg S \ll\!\cdot\!\gg O
\end{aligned}
$$

John loves Mary.

$$[\![\textsc{loves}\,\textsc{Mary}\,\textsc{John}]\!] \twoheadrightarrow \eta\,(\mathbf{love}\,\mathbf{j}\,\mathbf{m})$$

Mary loves me.

$$[\![\textsc{loves}\,\textsc{me}\,\textsc{Mary}]\!] \twoheadrightarrow \texttt{speaker} \star (\lambda x.\, \eta\,(\mathbf{love}\,\mathbf{m}\,x))$$

# Deixis — Handler

$$\text{withSpeaker} : \iota \to \mathcal{F}_{\{\texttt{speaker}:1 \rightarrowtail \iota\} \uplus E}(\alpha) \to \mathcal{F}_E(\alpha)$$

$$\text{withSpeaker} = \lambda s M.\, (\!|\, \texttt{speaker}: (\lambda\_k.\, k\, s) \,|\!)\, M$$

# Deixis — Handler

$$\text{withSpeaker} : \iota \to \mathcal{F}_{\{\texttt{speaker}:1 \rightarrowtail \iota\} \uplus E}(\alpha) \to \mathcal{F}_E(\alpha)$$

$$\text{withSpeaker} = \lambda s M. \llparenthesis\, \texttt{speaker}: (\lambda\_k.\, k\, s) \,\rrparenthesis\, M$$

$$\begin{aligned} & \text{withSpeaker}\ s\ \ [\![\textsc{loves me Mary}]\!] \\ \twoheadrightarrow\ & \text{withSpeaker}\ s\ (\texttt{speaker} \star (\lambda x.\, \eta\, (\mathbf{love}\, \mathbf{m}\, x))) \\ \twoheadrightarrow\ & \eta\, (\mathbf{love}\, \mathbf{m}\, s) \end{aligned}$$

# Deixis — Shifting the Index

$$\textsc{said}_{\mathrm{IS}} : S \multimap NP \multimap S$$

$$\textsc{said}_{\mathrm{DS}} : S \multimap NP \multimap S$$

$$\begin{aligned}
[\![\textsc{said}_{\mathrm{IS}}]\!] &= \lambda CS.\, \mathbf{say} \cdot\!\gg S \ll\!\cdot\!\gg C \\
&= \lambda CS.\, S \ggg= (\lambda s.\, \mathbf{say}\, s \cdot\!\gg C) \\
[\![\textsc{said}_{\mathrm{DS}}]\!] &= \lambda CS.\, S \ggg= (\lambda s.\, \mathbf{say}\, s \cdot\!\gg (\mathsf{withSpeaker}\ s\, C))
\end{aligned}$$

# Deixis — Shifting the Index, Examples

$$
\begin{aligned}
[\![\textsc{said}_{\text{IS}}]\!] &= \lambda CS.\, \textbf{say} \cdot\!\gg S \ll\!\cdot\!\gg C \\
&= \lambda CS.\, S \ggg\!= (\lambda s.\, \textbf{say}\, s \cdot\!\gg C) \\
[\![\textsc{said}_{\text{DS}}]\!] &= \lambda CS.\, S \ggg\!= (\lambda s.\, \textbf{say}\, s \cdot\!\gg (\text{withSpeaker}\; s\, C))
\end{aligned}
$$

John said Mary loves me.

$$
\begin{aligned}
&[\![\textsc{said}_{\text{IS}}\,(\textsc{loves me Mary})\,\textsc{John}]\!] \\
&\twoheadrightarrow \texttt{speaker} \star (\lambda x.\, \eta\, (\textbf{say}\, \textbf{j}\, (\textbf{love}\, \textbf{m}\, x)))
\end{aligned}
$$

John said, "Mary loves me".

$$
\begin{aligned}
&[\![\textsc{said}_{\text{DS}}\,(\textsc{loves me Mary})\,\textsc{John}]\!] \\
&\twoheadrightarrow \eta\, (\textbf{say}\, \textbf{j}\, (\textbf{love}\, \textbf{m}\, \textbf{j}))
\end{aligned}
$$

# Linguistic Phenomena as Effects

## Conventional Implicature

# Conventional Implicature

$$\begin{aligned}
\textsc{appos} &: \mathit{NP} \multimap \mathit{NP} \multimap \mathit{NP} \\
\textsc{best-friend} &: \mathit{NP} \multimap \mathit{NP} \\
\textsc{not-the-case} &: S \multimap S
\end{aligned}$$

$$\begin{aligned}
[\![\textsc{appos}]\!] &= \lambda XY.\ X \gg\!= (\lambda x. \\
&\quad Y \gg\!= (\lambda y. \\
&\quad \texttt{implicate}\,(x = y)\,(\lambda_. \\
&\quad \eta\, x))) \\
[\![\textsc{best-friend}]\!] &= \lambda X.\, \textbf{best-friend} \cdot\!\gg X \\
[\![\textsc{not-the-case}]\!] &= \lambda S.\, \neg \cdot\!\gg S
\end{aligned}$$

# Conventional Implicature — Handler

$$\text{accommodate} : \mathcal{F}_{\{\mathtt{implicate}:o \rightarrowtail 1\} \uplus E}(o) \to \mathcal{F}_E(o)$$
$$\text{accommodate} = \lambda M.\, (\!|\, \mathtt{implicate}\colon (\lambda ik.\, i \mathbin{\wedge\!\!\gg} k\,\star) \,|\!)\; M$$

# Conventional Implicature — Handler

$$\text{accommodate} : \mathcal{F}_{\{\mathtt{implicate}:o \rightarrowtail 1\} \uplus E}(o) \to \mathcal{F}_E(o)$$
$$\text{accommodate} = \lambda M.\, (\!|\, \mathtt{implicate}\!: (\lambda ik.\, i \wedge\!\!\gg k \star) \,|\!)\, M$$

$$[\![\mathrm{SAID}_{\mathrm{DS}}]\!] := \lambda CS.\, S \ggg\!= (\lambda s.$$
$$\textbf{say}\, s \cdot\!\!\gg (\text{withSpeaker}\; s\, (\text{accommodate}\; C)))$$

# Conventional Implicature — Examples

It is not the case that John, Mary's best friend, loves Alice.

$[\![\textsc{not-the-case}\,(\textsc{loves}\,\textsc{Alice}\,(\textsc{appos}\,\textsc{John}\,(\textsc{best-friend}\,\textsc{Mary})))]\!]$

$\twoheadrightarrow \texttt{implicate}\,(\mathbf{j} = \mathbf{best\text{-}friend}\,\mathbf{m})\,(\lambda\_.\,\eta\,(\neg(\mathbf{love}\,\mathbf{j}\,\mathbf{a})))$

accommodate $[\![\ldots]\!]$

$\twoheadrightarrow \eta\,((\mathbf{j} = \mathbf{best\text{-}friend}\,\mathbf{m}) \wedge \neg(\mathbf{love}\,\mathbf{j}\,\mathbf{a}))$

John said, "I, Mary's best friend, love Mary".

$[\![\textsc{said}_{\textsc{ds}}\,(\textsc{loves}\,\textsc{Mary}\,(\textsc{appos}\,\textsc{me}\,(\textsc{best-friend}\,\textsc{Mary})))\,\textsc{John}]\!]$

$\twoheadrightarrow \eta\,(\mathbf{say}\,\mathbf{j}\,((\mathbf{j} = \mathbf{best\text{-}friend}\,\mathbf{m}) \wedge \mathbf{love}\,\mathbf{j}\,\mathbf{m}))$

# Linguistic Phenomena as Effects

## Quantification

# Quantification

$$\textsc{every}, \textsc{a} : N \multimap NP$$
$$\textsc{man}, \textsc{woman} : N$$

$$\begin{aligned}
[\![\textsc{every}]\!] &= \lambda N.\, \mathtt{scope}\, (\lambda c.\, \forall \cdot\!\gg (\mathcal{C}\, (\lambda x.\, (N \ll\!\cdot\, x) \ll\!\!\rightarrow\!\!\gg c\, x)))\, (\lambda x.\, \eta\, x) \\
[\![\textsc{a}]\!] &= \lambda N.\, \mathtt{scope}\, (\lambda c.\, \exists \cdot\!\gg (\mathcal{C}\, (\lambda x.\, (N \ll\!\cdot\, x) \ll\!\wedge\!\gg c\, x)))\, (\lambda x.\, \eta\, x) \\
[\![\textsc{man}]\!] &= \eta\, \mathbf{man} \\
[\![\textsc{woman}]\!] &= \eta\, \mathbf{woman}
\end{aligned}$$

# Quantification

$$\textsc{every}, \textsc{a} : N \multimap NP$$

$$\textsc{man}, \textsc{woman} : N$$

$$\llbracket \textsc{every} \rrbracket = \lambda N.\, \mathtt{scope}\,(\lambda c.\, \forall \cdot\!\gg (\mathcal{C}\,(\lambda x.\,(N \ll\!\cdot\; x) \ll\!\rightarrow\!\gg c\,x)))\,(\lambda x.\, \eta\, x)$$

$$\llbracket \textsc{a} \rrbracket = \lambda N.\, \mathtt{scope}\,(\lambda c.\, \exists \cdot\!\gg (\mathcal{C}\,(\lambda x.\,(N \ll\!\cdot\; x) \ll\!\wedge\!\gg c\,x)))\,(\lambda x.\, \eta\, x)$$

$$\llbracket \textsc{man} \rrbracket = \eta\, \mathbf{man}$$

$$\llbracket \textsc{woman} \rrbracket = \eta\, \mathbf{woman}$$

## Quantification

$$\textsc{every}, \textsc{a} : N \multimap NP$$

$$\textsc{man}, \textsc{woman} : N$$

$$\begin{aligned}
[\![\textsc{every}]\!] &= \lambda N.\, \texttt{scope}\, (\lambda c.\, \forall \cdot\!\gg (\mathcal{C}\, (\lambda x.\, (N \ll\!\cdot\, x) \ll\!\!\rightarrow\!\!\gg c\, x)))\, (\lambda x.\, \eta\, x) \\
[\![\textsc{a}]\!] &= \lambda N.\, \texttt{scope}\, (\lambda c.\, \exists \cdot\!\gg (\mathcal{C}\, (\lambda x.\, (N \ll\!\cdot\, x) \ll\!\wedge\!\gg c\, x)))\, (\lambda x.\, \eta\, x) \\
[\![\textsc{man}]\!] &= \eta\, \mathbf{man} \\
[\![\textsc{woman}]\!] &= \eta\, \mathbf{woman}
\end{aligned}$$

# Quantification — Handler

$$\mathsf{SI} = \lambda M.\, (\!|\, \mathtt{scope} \colon (\lambda c k.\, c\, k) \,|\!)\, M$$

# Quantification — Handler

$$\mathsf{SI} = \lambda M.\, (\!|\, \mathtt{scope}\colon (\lambda ck.\, c\, k)\, |\!)\, M$$

$$\begin{aligned}
[\![\textsc{loves}]\!] &:= \lambda OS.\ \mathsf{SI}\,([\![\textsc{loves}]\!]\ O\, S)\\
[\![\textsc{said}_{\textsc{is}}]\!] &:= \lambda CS.\ \mathsf{SI}\,([\![\textsc{said}_{\textsc{is}}]\!]\ C\, S)\\
[\![\textsc{said}_{\textsc{ds}}]\!] &:= \lambda CS.\ \mathsf{SI}\,([\![\textsc{said}_{\textsc{ds}}]\!]\ C\, S)\\
[\![\textsc{appos}]\!] &:= \lambda XY.\ X \gg\!= (\lambda x.\\
&\qquad \mathsf{SI}\,(\eta\, x \lll\!\Leftrightarrow\!\ggg Y) \gg\!= (\lambda i.\\
&\qquad \mathtt{implicate}\, i\, (\lambda _.\\
&\qquad \eta\, x)))
\end{aligned}$$

# Quantification — Examples

John, my best friend, loves every woman.

$$\text{w. S. } s\,(\text{acc. } [\![\textsc{loves}\,(\textsc{every woman})\,(\textsc{appos John}\,(\textsc{best-friend me}))]\!]$$

$$\twoheadrightarrow \eta\,((\mathbf{j} = \textbf{best-friend}\, s) \wedge (\forall x.\ \textbf{woman}\, x \to \textbf{love}\,\mathbf{j}\, x))$$

Mary, everyone's best friend, loves John.

$$\text{acc. } [\![\textsc{loves John}\,(\textsc{appos Mary}\,(\textsc{best-friend everyone}))]\!]$$

$$\twoheadrightarrow \eta\,((\forall x.\ \mathbf{m} = \textbf{best-friend}\, x) \wedge (\textbf{love}\,\mathbf{m}\,\mathbf{j}))$$

A man said, "My best friend, Mary, loves me".

$$[\![\textsc{said}_{\textsc{ds}}\,(\textsc{loves me}\,(\textsc{appos}\,(\textsc{best-friend me})\,\textsc{Mary}))\,(\textsc{a man})]\!]$$

$$\twoheadrightarrow \eta\,(\exists x.\ \textbf{man}\, x \wedge \textbf{say}\, x\,((\textbf{best-friend}\, x = \mathbf{m}) \wedge (\textbf{love}\,(\textbf{best-friend}\, x)\, x)))$$

# Linguistic Phenomena as Effects

## Summary

## Effects — Independence & Interactions

lexical entries (almost) independent

- no generalized quantifiers in $[\![\textsc{John}]\!]$, $[\![\textsc{me}]\!]$, $[\![\textsc{best-friend}]\!]$

changes needed only to account for interactions

- scope islands (SI) in tensed clauses and appositives
- blocking conversational implicature (accommodate) in direct quotation

old results preserved when extending fragment

- adding handlers for new effects preserves old meanings
- e.g. SI is a nop in sentences without quantification

# Universal Semantic Glue

$$[\![\textsc{loves}]\!] = \lambda OS.\, \textbf{love} \cdot\!\gg S \ll\!\cdot\!\gg O$$

instead of

$$[\![\textsc{loves}]\!] = \lambda OSi.\, \textbf{love}\, (S\, i)\, (O\, i)$$

$$[\![\textsc{loves}]\!] = \lambda OS.\, \textbf{love} \cdot\!\gg S \ll\!\cdot\!\gg O$$

instead of

$$[\![\textsc{loves}]\!] = \lambda OS.\, \langle \textbf{love}\, (\pi_1\, S)\, (\pi_1\, O), \pi_2\, S \wedge \pi_2\, O \rangle$$

$$[\![\textsc{loves}]\!] = \lambda OS.\, \textbf{love} \cdot\!\gg S \ll\!\cdot\!\gg O$$

instead of

$$[\![\textsc{loves}]\!] = \lambda OSk.\, S\, (\lambda s.\, O\, (\lambda o.\, k\, (\textbf{love}\, s\, o)))$$

# Conclusion

## Conclusion

[Shan, 2002] : monads are prevalent in NL semantics

$(\!| \lambda |\!) =$ strongly normalizing $\lambda$-calculus with free monads $\mathcal{F}_E$

# Conclusion

[Shan, 2002] : monads are prevalent in NL semantics

$(\!|\,\lambda\,|\!)$ = strongly normalizing $\lambda$-calculus with free monads $\mathcal{F}_E$

$$\Downarrow$$

We can use $(\!|\,\lambda\,|\!)$ to

- write less semantic glue
- study more interactions of phenomena

# Future

add more effects

- anaphora and presupposition in upcoming dissertation
    - sketched in [Maršík and Amblard, 2014]

what phenomena can we treat?

- all that project?

use effects directly

- glue-free!
- have to define a uniform evaluation strategy

**Thank you!**

**Questions?**

# References i


- Barker, C. (2002).
  **Continuations and the nature of quantification.**
  *Natural language semantics*, 10(3):211–242.
- Barker, C. and Bumford, D. (2015).
  **Monads for natural language.**
- Bauer, A. and Pretnar, M. (2015).
  **Programming with algebraic effects and handlers.**
  *J. Log. Algebr. Meth. Program.*, 84(1):108–123.

# References ii


Blanqui, F. (2000).
**Termination and confluence of higher-order rewrite systems.**
In *Rewriting Techniques and Applications*, pages 47–61. Springer.

Cartwright, R. and Felleisen, M. (1994).
**Extensible denotational language specifications.**
In *Theoretical Aspects of Computer Software*, pages 244–272. Springer.

Charlow, S. (2014).
***On the semantics of exceptional scope*.**
PhD thesis, New York University.

# References iii


de Groote, P. (2001).
**Type raising, continuations, and classical logic.**
In *Proceedings of the thirteenth Amsterdam Colloquium*.

Giorgolo, G. and Asudeh, A. (2015).
**Natural language semantics with enriched meanings.**

Hamana, M. (2007).
**Higher-order semantic labelling for inductive datatype systems.**
In *Proceedings of the 9th ACM SIGPLAN international conference on Principles and practice of declarative programming*, pages 97–108. ACM.

# References iv


Hobbs, J. and Rosenschein, S. (1977).
**Making computational sense of montague's intensional logic.**
*Artificial Intelligence*, 9(3):287–306.

Kammar, O., Lindley, S., and Oury, N. (2013).
**Handlers in action.**
In *Proceedings of the 18th ACM SIGPLAN international conference on Functional programming*, pages 145–158. ACM.

Kiselyov, O. (2015).
**Applicative abstract categorial grammars in full swing.**
In *Proceedings of the Twelth Workshop on Logic and Engineering of Natural Language Semantics*.

# References v


Kiselyov, O., Sabry, A., and Swords, C. (2013).
**Extensible effects: an alternative to monad transformers.**
In *Proceedings of the 2013 ACM SIGPLAN symposium on Haskell*, pages 59–70. ACM.

Klop, J. W., Van Oostrom, V., and Van Raamsdonk, F. (1993).
**Combinatory reduction systems: introduction and survey.**

*Theoretical computer science*, 121(1):279–308.

Maršík, J. and Amblard, M. (2014).
**Algebraic effects and handlers in natural language interpretation.**
In *Natural Language and Computer Science*. Center for Informatics and Systems of the University of Coimbra.

Plotkin, G. D. and Pretnar, M. (2013).
**Handling algebraic effects.**
*arXiv preprint arXiv:1312.1399*.

Shan, C. (2002).
**Monads for natural language semantics.**
*arXiv preprint cs/0205026*.

Shan, C. (2005).
**Linguistic side effects.**
In *In Proceedings of the Eighteenth Annual IEEE Symposium on Logic and Computer Science (LICS 2003) Workshop on Logic and Computational*, pages 132–163. University Press.