Functional modularity in the lambda calculus

Chung-chieh Shan
Cornell University

28 December 2011

A.I. Memo 357

May 1976

FROM UNDERSTANDING COMPUTATION TO UNDERSTANDING NEURAL CIRCUITRY

by

D. Marr and T. Poggio*

Abstract: The CNS needs to be understood at four nearly independent levels of description: (1) that at which the nature of a computation is expressed; (2) that at which the algorithms that implement a computation are characterized; (3) that at which an algorithm is committed to particular mechanisms; and (4) that at which the mechanisms are realized in hardware. In general, the nature of a computation is determined by the problem to be solved, the mechanisms that are used depend upon the available hardware, and the particular algorithms chosen depend on the problem and on the available mechanisms. Examples are given of theories at each level.

On the Design and Development of Program Families

DAVID L. PARNAS

One may consider a program development to be good, if the early decisions exclude only uninteresting, undesired, or unnecessary programs. The decisions which remove desired programs would be either postponed until a later stage or confined to a well delimited subset of the code. Objective criticism of a program's structure would be based upon the fact that a decision or assumption which was likely to change has influenced too much of the code either because it was made too early in the development or because it was not confined to an information hiding module.

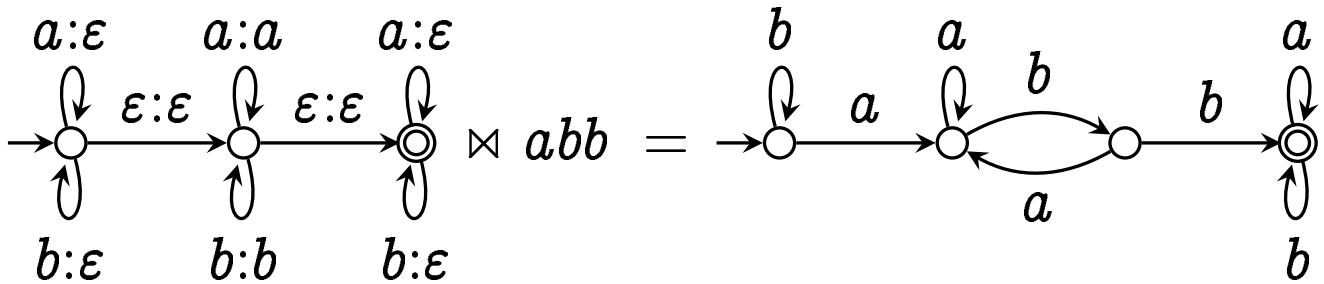
Clearly this is not the only criterion which one may use in evaluating program structures. Clarity (e.g., ease of understanding, ease of verification) is another quite relevant consideration. Although there is some reason to suspect that the two measures are not completely unrelated, there are no rea-

Functional modularity

A module is a part of a description of a system

- Modularity should be invariant under physically entangled emulation with dye pack
- Modularity makes a theory more concise, comprehensible
- ‘Functional structure’ (Gallistel)/
  ‘Wirkungsgefüge’ (behavioral physiology)/source code

general description $\xrightarrow{\text{specialize}}$ specific machinery



Functional modularity

A module is a part of a description of a system

- Modularity should be invariant under physically entangled emulation with dye pack
- Modularity makes a theory more concise, comprehensible
- 'Functional structure' (Gallistel)/
  'Wirkungsgefüge' (behavioral physiology)/source code

Good decomposition helps reuse when environment changes

- Utterances need not re-conventionalize
- Organisms need not re-learn
- Species need not re-evolve
- Researchers need not re-discover

Lambda the ultimate

The essence of reuse:
a module is a sub-expression.
Binding. Higher-order abstractions.

Types classify terms. Polymorphism circumscribes information flow.

1. Expressions and interpretations in Abstract Categorial Grammar
2. Layers of monads for quantification and state

Lambda the ultimate

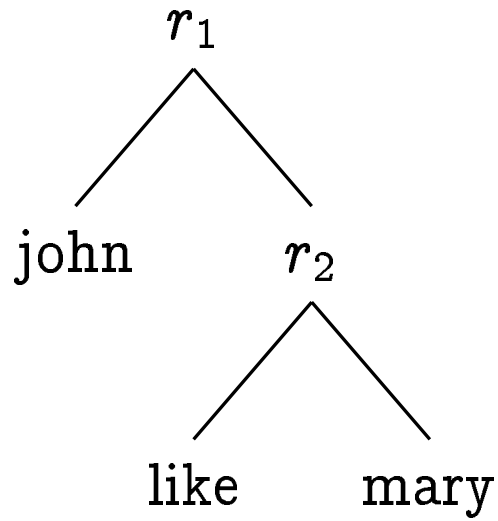
The essence of reuse:
a module is a sub-expression.
Binding. Higher-order abstractions.

Types classify terms. Polymorphism
circumscribes information flow.

1. Expressions and interpretations in Abstract Categorial Grammar
2. Layers of monads for quantification and state

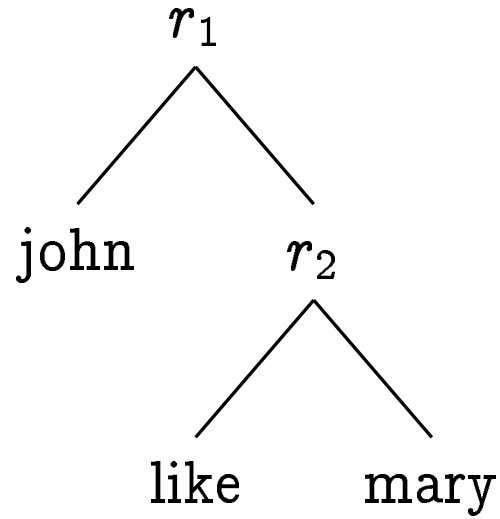
Abstract Categorial Grammar

$$e = \lambda\langle \text{john}, \text{mary}, \text{like}, r_1, r_2 \rangle .\\ r_1\ \text{john}\ (r_2\ \text{like}\ \text{mary})$$



Abstract Categorial Grammar

$$e = \lambda\langle \mathrm{john}, \mathrm{mary}, \mathrm{like}, r_1, r_2\rangle.\ r_1\ \mathrm{john}\ (r_2\ \mathrm{like}\ \mathrm{mary})$$



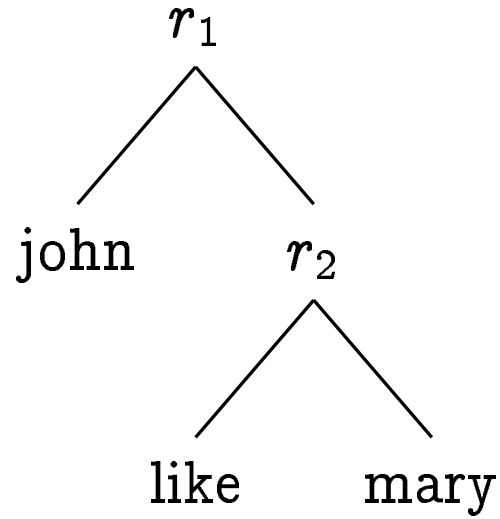
$$\mathrm{EN} = \langle \text{`John'}, \text{`Mary'}, \text{`likes'}, \lambda s.\,\lambda v.\, s \frown \text{`\ '} \frown v, \lambda v.\,\lambda o.\, v \frown \text{`\ '} \frown o\rangle$$

$$\mathrm{Sem} = \langle j', m', l', \lambda s.\,\lambda v.\, vs, \lambda v.\,\lambda o.\, vo\rangle$$

$$e(\mathrm{EN}) = \text{`John likes Mary'} \qquad e(\mathrm{Sem}) = lmj$$

Abstract Categorial Grammar

$$e = \lambda\langle \mathrm{john}, \mathrm{mary}, \mathrm{like}, r_1, r_2\rangle.\ r_1\ \mathrm{john}\ (r_2\ \mathrm{like}\ \mathrm{mary})$$



$$\mathtt{JA} = \langle \ldots \rangle$$

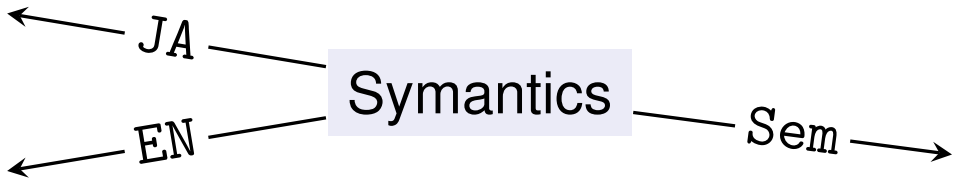
$$\mathtt{EN} = \langle \text{`John'}, \text{`Mary'}, \text{`likes'}, \lambda s.\lambda v.\, s^\frown\text{` '}^\frown v, \lambda v.\lambda o.\, v^\frown\text{` '}^\frown o\rangle$$

$$\mathtt{Sem} = \langle j', m', l', \lambda s.\lambda v.\, vs, \lambda v.\lambda o.\, vo\rangle$$

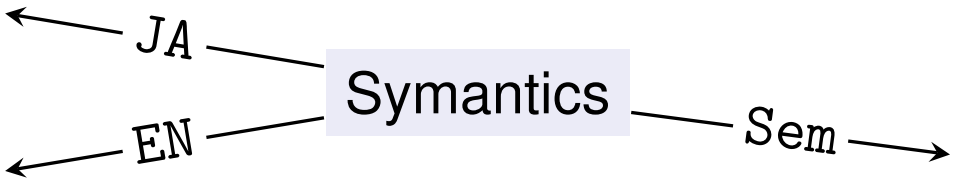
$e(\mathtt{JA}) =$ 'ジョンさんはメリさんのことが好きだ'

$e(\mathtt{EN}) =$ 'John likes Mary' $\qquad e(\mathtt{Sem}) = lmj$

Interpretations

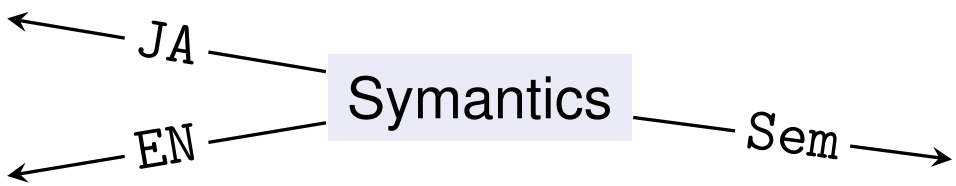


Interpretation transformers



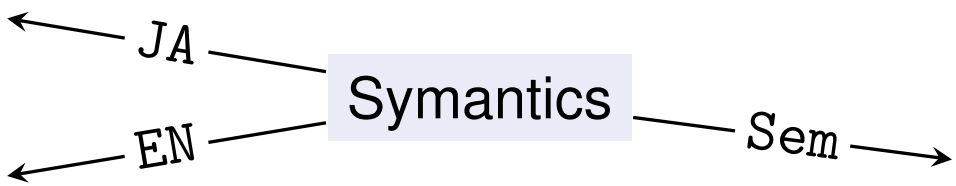
$$\begin{aligned}\texttt{Sem} = \langle & j', \\ & m', \\ & l', \\ & \lambda s.\, \lambda v.\, v s, \\ & \lambda v.\, \lambda o.\, v o \rangle\end{aligned}$$

Interpretation transformers



$$
\begin{aligned}
\texttt{Sem} = \lambda\langle j, m, l, @, \neg, \wedge, \ldots\rangle.\\
&\langle j,\\
&\ m,\\
&\ l,\\
&\ \lambda s.\, \lambda v.\, @vs,\\
&\ \lambda v.\, \lambda o.\, @vo\rangle
\end{aligned}
$$

Interpretation transformers



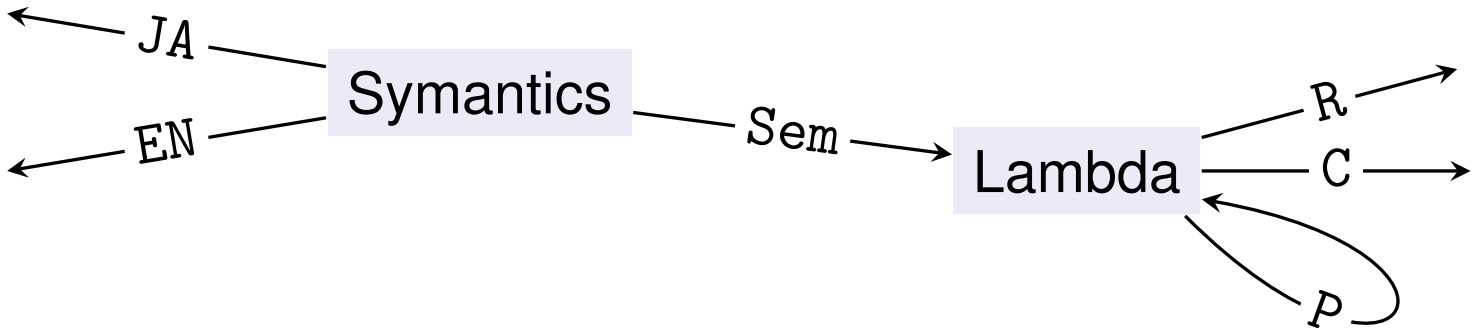
$$\begin{aligned}\texttt{Sem} = \lambda\langle j, m, l, @, \neg, \wedge, \ldots\rangle.\\ \langle j,\\ m,\\ l,\\ \lambda s.\,\lambda v.\,@vs,\\ \lambda v.\,\lambda o.\,@vo\rangle\end{aligned}$$

$$\texttt{R} = \langle j', m', l', \lambda f.\,\lambda x.\,fx, \ldots\rangle$$

$$\texttt{C} = \langle \text{`j'}, \text{`m'}, \text{`l'}, \lambda f.\,\lambda x.\,f \frown \text{`('} \frown x \frown \text{`)'}, \ldots\rangle$$

$$\texttt{P} = \lambda\langle j, m, l, @, \neg, \wedge, \ldots\rangle.\,\langle \ldots\rangle$$

Interpretation transformers



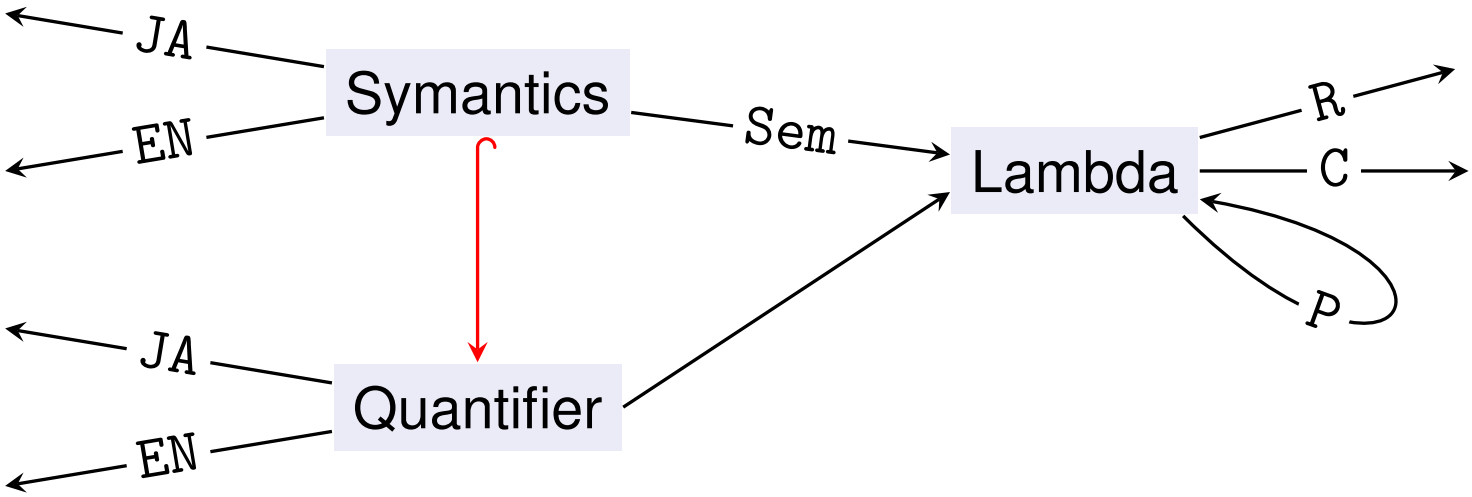
$$\texttt{Sem} = \lambda\langle j, m, l, @, \neg, \wedge, \ldots\rangle.\ \langle j,\ m,\ l,\ \lambda s.\,\lambda v.\,@vs,\ \lambda v.\,\lambda o.\,@vo\rangle$$

$$\texttt{R} = \langle j', m', l', \lambda f.\,\lambda x.\,fx, \ldots\rangle$$

$$\texttt{C} = \langle \text{`j'}, \text{`m'}, \text{`l'}, \lambda f.\,\lambda x.\,f \frown \text{`('} \frown x \frown \text{`)'}, \ldots\rangle$$

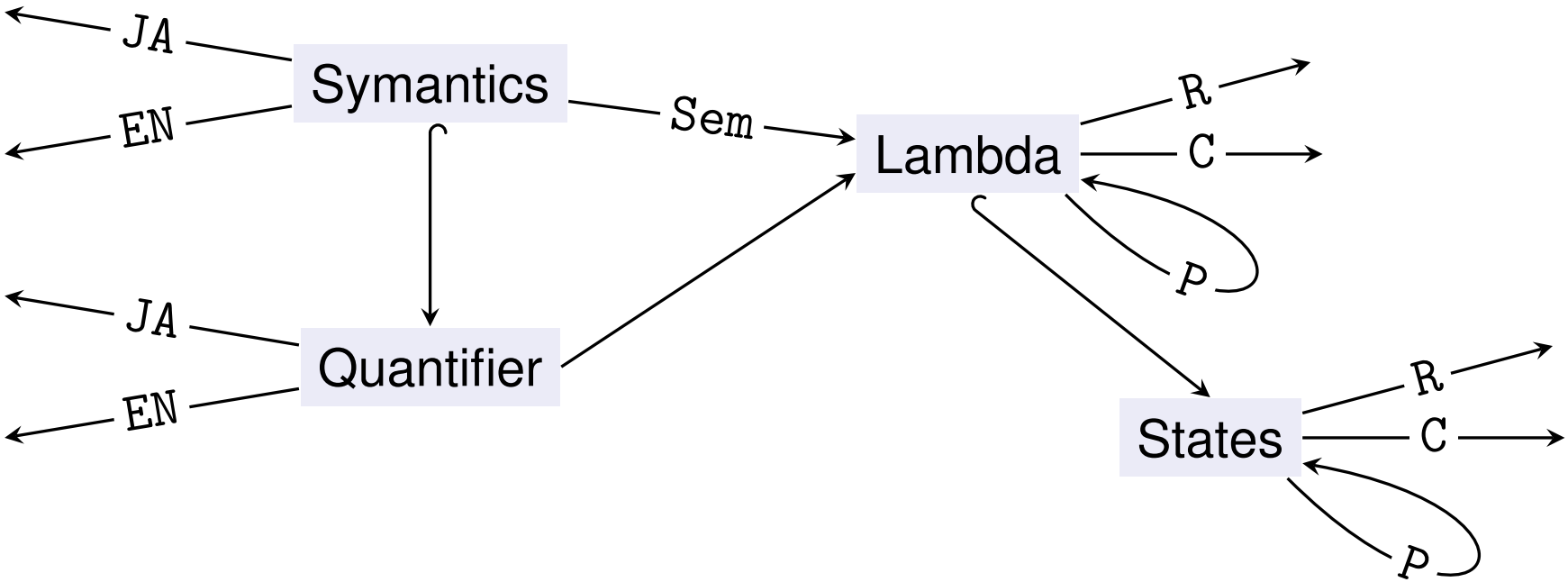
$$\texttt{P} = \lambda\langle j, m, l, @, \neg, \wedge, \ldots\rangle.\,\langle \ldots\rangle$$

Interpretation transformers

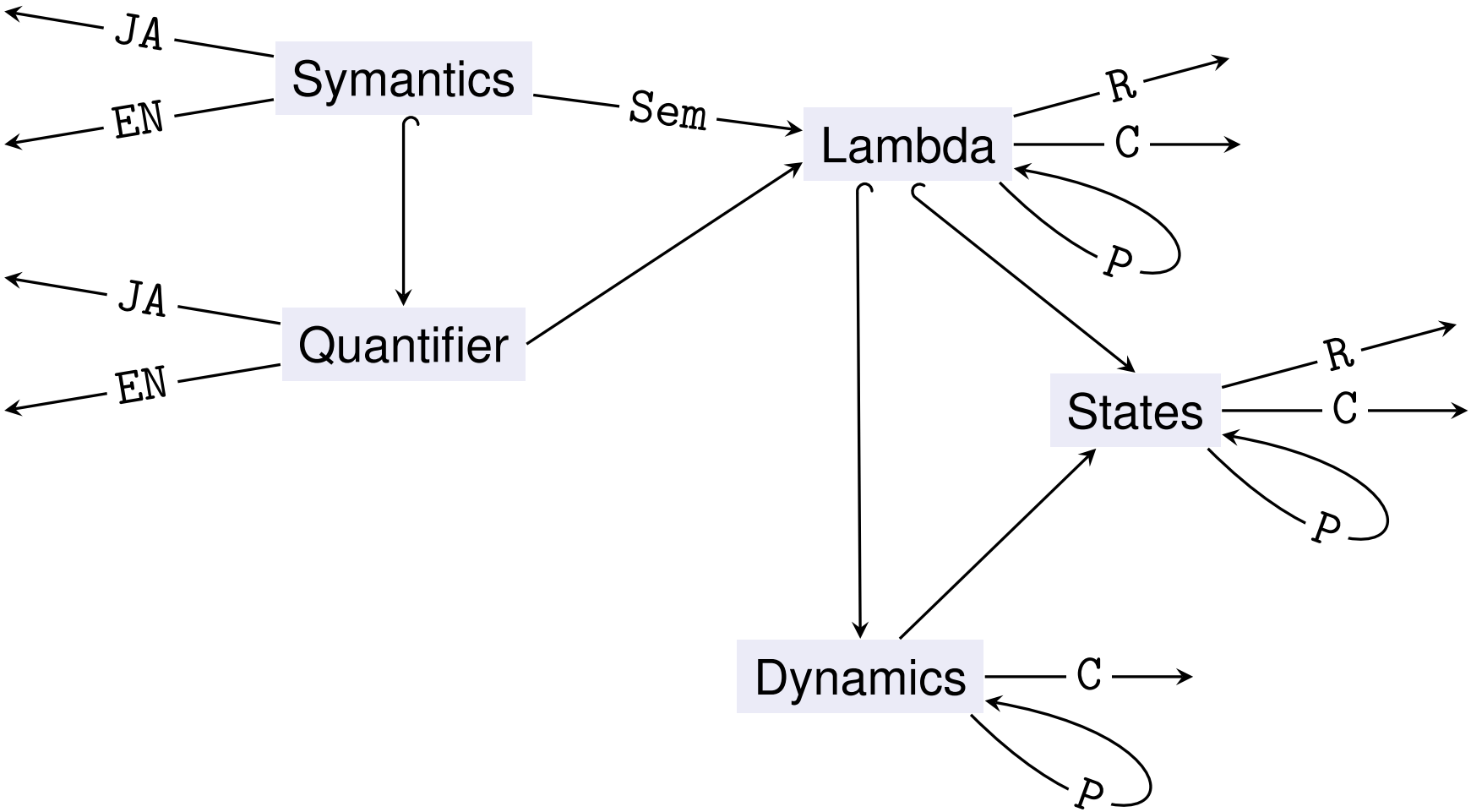


$$\lambda\langle \text{john}, \text{mary}, \text{like}, r_1, r_2, \text{every}, \text{some}, r_4, r_5\rangle.$$
$$\langle \text{john}, \text{mary}, \text{like}, r_1, r_2\rangle$$

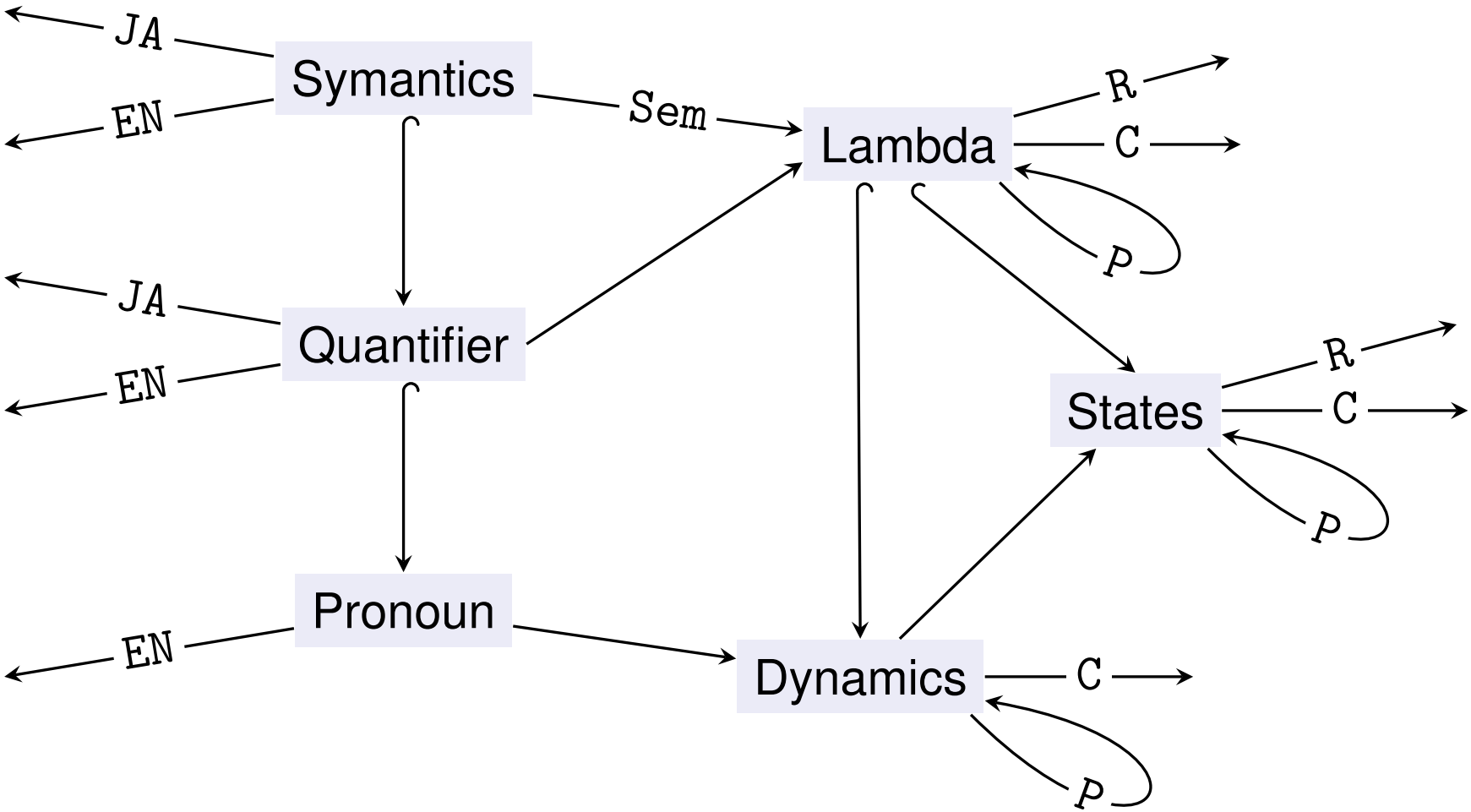
Dynamic logic



Dynamic logic



Dynamic logic



Expression transformers

Macros are maps from expressions to expressions.

$$\begin{aligned} \vee \quad = \quad & \lambda\langle j, m, l, @, \neg, \wedge, \ldots\rangle. \\ & \lambda e_1.\, \lambda e_2.\, \neg(\wedge(\neg e_1)(\neg e_2)) \end{aligned}$$

Also for analyzing unquotation.

Ralph warned that he has
'long suspected that [Ortcutt] is a spy'.

Ralph warned that he has
'long suspected that [Ortcutt's beach alias] is a spy'.

Lambda the ultimate

The essence of reuse:
a module is a sub-expression.
Binding. Higher-order abstractions.

Types classify terms. Polymorphism circumscribes information flow.

1. Expressions and interpretations in Abstract Categorial Grammar
2. Layers of monads for quantification and state

Generalization to the worst case

extensional

$$\begin{aligned}\mathrm{john} &= j\\ \mathrm{mary} &= m\\ r_1 &= \lambda s.\,\lambda v.\,vs\\ r_2 &= \lambda v.\,\lambda o.\,vo\end{aligned}$$

possible worlds

$$\begin{aligned}\mathrm{john} &= \lambda w.\,j\\ \mathrm{mary} &= \lambda w.\,m\\ r_1 &= \lambda s.\,\lambda v.\,\lambda w.\,vw(sw)\\ r_2 &= \lambda v.\,\lambda o.\,\lambda w.\,vw(ow)\end{aligned}$$

alternative sets

$$\begin{aligned}\mathrm{john} &= \{j\}\\ \mathrm{mary} &= \{m\}\\ r_1 &= \lambda s.\,\lambda v.\,\{\,fx \mid x \in s,\ f \in v\,\}\\ r_2 &= \lambda v.\,\lambda o.\,\{\,fx \mid f \in v,\ x \in o\,\}\end{aligned}$$

Generalization to the worst case

extensional

$$\begin{aligned}\text{john} &= j\\ \text{mary} &= m\\ r_1 &= \lambda s.\,\lambda v.\,vs\\ r_2 &= \lambda v.\,\lambda o.\,vo\end{aligned}$$

state

$$\begin{aligned}\text{john} &= \lambda i.\,\langle i, j\rangle\\ \text{mary} &= \lambda i.\,\langle i, m\rangle\\ r_1 &= \lambda s.\,\lambda v.\,\lambda i.\,\langle i'', fx\rangle \quad \text{where} \quad \langle i', x\rangle = si \quad \langle i'', f\rangle = vi'\\ r_2 &= \lambda v.\,\lambda o.\,\lambda i.\,\langle i'', fx\rangle \quad \text{where} \quad \langle i', f\rangle = vi \quad \langle i'', x\rangle = oi'\end{aligned}$$

continuations

$$\begin{aligned}\text{john} &= \lambda c.\,cj\\ \text{mary} &= \lambda c.\,cm\\ r_1 &= \lambda s.\,\lambda v.\,\lambda c.\,s\lambda x.\,v\lambda f.\,c(fx)\\ r_2 &= \lambda v.\,\lambda o.\,\lambda c.\,v\lambda f.\,o\lambda x.\,c(fx)\end{aligned}$$

Generalization of the worst case

Three components of a monad:

$$\mathbb{M}, \qquad \eta : \alpha \to \mathbb{M}\alpha, \qquad \star : \mathbb{M}\alpha \to (\alpha \to \mathbb{M}\beta) \to \mathbb{M}\beta$$

$$\begin{aligned} \mathrm{john} &= \eta(j) \\ \mathrm{mary} &= \eta(m) \\ r_1 &= \lambda s.\, \lambda v.\, s \star \lambda x.\, v \star \lambda f.\, \eta(f x) \\ r_2 &= \lambda v.\, \lambda o.\, v \star \lambda f.\, o \star \lambda x.\, \eta(f x) \end{aligned}$$

extensional

$\mathbb{M}\alpha = \alpha$

$\eta(a) = a$

$m \star q = q m$

possible worlds

$\mathbb{M}\alpha = s \to \alpha$

$\eta(a) = \lambda w.\, a$

$m \star q = \lambda w.\, q(m w) w$

state

$\mathbb{M}\alpha = i \to (i \times \alpha)$

$\eta(a) = \lambda i.\, \langle i, a \rangle$

$m \star q = \lambda i.\, q a i'$ where $\langle i', a \rangle = m i$

alternative sets

$\mathbb{M}\alpha = \alpha \to t$

$\eta(a) = \{a\}$

$m \star q = \bigcup_{a \in m} q a$

continuations

$\mathbb{M}\alpha = (\alpha \to r) \to r$

$\eta(a) = \lambda c.\, c a$

$m \star q = \lambda c.\, m \lambda a.\, q a c$

Generalization of the worst case

Three components of a monad:

$$\mathbb{M}, \qquad \eta : \alpha \to \mathbb{M}\alpha, \qquad \star : \mathbb{M}\alpha \to (\alpha \to \mathbb{M}\beta) \to \mathbb{M}\beta$$

$$\begin{aligned} \mathrm{john} &= \eta(j) \\ \mathrm{mary} &= \eta(m) \\ r_1 &= \lambda s.\, \lambda v.\, s \star \lambda x.\, v \star \lambda f.\, \eta(f x) \\ r_2 &= \lambda v.\, \lambda o.\, v \star \lambda f.\, o \star \lambda x.\, \eta(f x) \end{aligned}$$

extensional

$$\begin{aligned} \mathbb{M}\alpha &= \alpha \\ \eta(a) &= a \\ m \star q &= q m \end{aligned}$$

possible worlds

$$\begin{aligned} \mathbb{M}\alpha &= s \to \alpha \\ \eta(a) &= \lambda w.\, a \\ m \star q &= \lambda w.\, q(m w) w \end{aligned}$$

state

$$\begin{aligned} \mathbb{M}\alpha &= i \to (i \times \alpha) \\ \eta(a) &= \lambda i.\, \langle i, a \rangle \\ m \star q &= \lambda i.\, q a i' \text{ where } \langle i', a \rangle = m i \end{aligned}$$

alternative sets

$$\begin{aligned} \mathbb{M}\alpha &= \alpha \to t \\ \eta(a) &= \{a\} \\ m \star q &= \bigcup_{a \in m} q a \end{aligned}$$

continuations

$$\begin{aligned} \mathbb{M}\alpha &= (\alpha \to r) \to r \\ \eta(a) &= \lambda c.\, c a \\ m \star q &= \lambda c.\, m \lambda a.\, q a c \end{aligned}$$

Generalization of the worst case

Three components of a monad:

$$\mathbb{M}, \qquad \eta : \alpha \to \mathbb{M}\alpha, \qquad \star : \mathbb{M}\alpha \to (\alpha \to \mathbb{M}\beta) \to \mathbb{M}\beta$$

$$\begin{aligned} \mathrm{john} &= \eta(j) \\ \mathrm{mary} &= \eta(m) \\ r_1 &= \lambda s.\, \lambda v.\, s \star \lambda x.\, v \star \lambda f.\, \eta(f x) \\ r_2 &= \lambda v.\, \lambda o.\, v \star \lambda f.\, o \star \lambda x.\, \eta(f x) \\ \mathrm{her} &= \lambda i.\, \langle i, i(5) \rangle \end{aligned}$$

extensional

$$\begin{aligned} \mathbb{M}\alpha &= \alpha \\ \eta(a) &= a \\ m \star q &= q m \end{aligned}$$

possible worlds

$$\begin{aligned} \mathbb{M}\alpha &= s \to \alpha \\ \eta(a) &= \lambda w.\, a \\ m \star q &= \lambda w.\, q(m w) w \end{aligned}$$

state

$$\begin{aligned} \mathbb{M}\alpha &= i \to (i \times \alpha) \\ \eta(a) &= \lambda i.\, \langle i, a \rangle \\ m \star q &= \lambda i.\, q a i' \text{ where } \langle i', a \rangle = m i \end{aligned}$$

alternative sets

$$\begin{aligned} \mathbb{M}\alpha &= \alpha \to t \\ \eta(a) &= \{a\} \\ m \star q &= \bigcup_{a \in m} q a \end{aligned}$$

continuations

$$\begin{aligned} \mathbb{M}\alpha &= (\alpha \to r) \to r \\ \eta(a) &= \lambda c.\, c a \\ m \star q &= \lambda c.\, m \lambda a.\, q a c \end{aligned}$$

Generalization of the worst case

Three components of a monad:

$$\mathbb{M}, \qquad \eta : \alpha \to \mathbb{M}\alpha, \qquad \star : \mathbb{M}\alpha \to (\alpha \to \mathbb{M}\beta) \to \mathbb{M}\beta$$

extensional

$\mathbb{M}\alpha = \alpha$

$\eta(a) = a$

$m \star q = qm$

$$\begin{aligned} \text{john} &= \eta(j) \\ \text{mary} &= \eta(m) \\ r_1 &= \lambda s.\, \lambda v.\, s \star \lambda x.\, v \star \lambda f.\, \eta(fx) \\ r_2 &= \lambda v.\, \lambda o.\, v \star \lambda f.\, o \star \lambda x.\, \eta(fx) \\ \text{john and mary} &= \lambda c.\, cj \wedge cm \end{aligned}$$

possible worlds

$\mathbb{M}\alpha = s \to \alpha$

$\eta(a) = \lambda w.\, a$

$m \star q = \lambda w.\, q(mw)w$

state

$\mathbb{M}\alpha = i \to (i \times \alpha)$

$\eta(a) = \lambda i.\, \langle i, a \rangle$

$m \star q = \lambda i.\, qai'$ where $\langle i', a \rangle = mi$

alternative sets

$\mathbb{M}\alpha = \alpha \to t$

$\eta(a) = \{a\}$

$m \star q = \bigcup_{a \in m} qa$

continuations

$\mathbb{M}\alpha = (\alpha \to r) \to r$

$\eta(a) = \lambda c.\, ca$

$m \star q = \lambda c.\, m\lambda a.\, qac$

Semantic cartography

state monad	**continuation monad transformer**
$\mathbb{M}\alpha = i \to (i \times \alpha)$	$\mathbb{M}'\alpha = (\alpha \to \mathbb{M}r) \to \mathbb{M}r$
$\eta(a) = \cdots$	$\eta'(a) = \cdots$
$m \star q = \cdots$	$m \star' q = \cdots$
	$\ell(m) = \lambda c.\, m \star c$
$\mathrm{her} = \ell(\lambda i.\, \langle i, i(5)\rangle)$	$\mathrm{j\&m} = \lambda c.\, cj \star \lambda x.\, cm \star \lambda y.\, \eta(x \wedge y)$

Semantic cartography

state monad

$$\mathbb{M}\alpha = i \to (i \times \alpha)$$
$$\eta(a) = \cdots$$
$$m \star q = \cdots$$

$$\mathrm{her} = \ell(\lambda i.\, \langle i, i(5)\rangle)$$

continuation monad transformer

$$\mathbb{M}'\alpha = (\alpha \to \mathbb{M}r) \to \mathbb{M}r$$
$$\eta'(a) = \cdots$$
$$m \star' q = \cdots$$
$$\ell(m) = \lambda c.\, m \star c$$

$$\mathrm{j\&m} = \lambda c.\, cj \star \lambda x.\, cm \star \lambda y.\, \eta(x \wedge y)$$

Semantic cartography

state monad

$$\mathbb{M}\alpha = i \to (i \times \alpha)$$
$$\eta(a) = \cdots$$
$$m \star q = \cdots$$

$$\mathrm{her} = \ell(\lambda i.\, \langle i, i(5)\rangle)$$

continuation monad transformer

$$\mathbb{M}'\alpha = (\alpha \to \mathbb{M}r) \to \mathbb{M}r$$
$$\eta'(a) = \cdots$$
$$m \star' q = \cdots$$
$$\ell(m) = \lambda c.\, m \star c$$

$$\mathrm{j\&m} = \lambda c.\, cj \star \lambda x.\, cm \star \lambda y.\, \eta(x \wedge y)$$

Semantic cartography

state monad	**continuation monad transformer**
$\mathbb{M}\alpha = i \to (i \times \alpha)$	$\mathbb{M}'\alpha = (\alpha \to \mathbb{M}r) \to \mathbb{M}r$
$\eta(a) = \cdots$	$\eta'(a) = \cdots$
$m \star q = \cdots$	$m \star' q = \cdots$
	$\ell(m) = \lambda c.\, m \star c$
$\mathrm{her} = \ell(\lambda i.\, \langle i, i(5)\rangle)$	$\mathrm{j\&m} = \lambda c.\, cj \star \lambda x.\, cm \star \lambda y.\, \eta(x \wedge y)$

$$\mathbb{M}'\alpha = (\alpha \to i \to (i \times r)) \to i \to (i \times r)$$

$$\mathrm{her} = \lambda c.\, \lambda i.\, c(i(5))i$$

$$\mathrm{j\&m} = \lambda c.\, \lambda i.\, \langle i'',\, x \wedge y\rangle \quad \text{where} \quad \langle i', x\rangle = cji \quad \langle i'', y\rangle = cmi'$$

Semantic cartography

state monad	**continuation monad transformer**
$\mathbb{M}\alpha = i \to (i \times \alpha)$	$\mathbb{M}'\alpha = (\alpha \to \mathbb{M}r) \to \mathbb{M}r$
$\eta(a) = \cdots$	$\eta'(a) = \cdots$
$m \star q = \cdots$	$m \star' q = \cdots$
	$\ell(m) = \lambda c.\, m \star c$
$\mathrm{her} = \ell(\lambda i.\, \langle i, i(5)\rangle)$	$\mathrm{j\&m} = \lambda c.\, cj \star \lambda x.\, cm \star \lambda y.\, \eta(x \wedge y)$

$$\mathbb{M}'\alpha = (\alpha \to i \to (i \times r)) \to i \to (i \times r)$$

$$\begin{aligned}
\mathrm{her} &= \lambda c.\, \lambda i.\, c(i(5))i \\
&\neq \lambda c.\, \lambda i.\, \langle i,\ c(i(5))i \vee c(i(6))i\rangle \\
\mathrm{j\&m} &= \lambda c.\, \lambda i.\, \langle i'',\ x \wedge y\rangle \quad \text{where} \quad \langle i', x\rangle = cji \quad \langle i'', y\rangle = cmi' \\
&\neq \lambda c.\, \lambda i.\, \langle i',\ x \wedge y\rangle \quad \text{where} \quad \langle i', x\rangle = cji \quad \langle i'', y\rangle = cmi'
\end{aligned}$$

Summary

Functional modules are description parts that can be reused in the face of change

- Expressions
- Interpretations
- Side effects
- Lexical entries
- ...

Types enforce information hiding