Church-Rosser in Morphogrammatics

Lambda calculus application of kenomic and morphic abstractions

Rudolf Kaehr Dr.phil
 ISSN 2041-4358

Abstract

Morphogrammatics is not presuming a multitude of contextures but is creating a polyverse of contextures in each situation of a realization of an operation. This continues studies proposed with “Notes on semi-Thue Systems in a Context of Morphogrammatics".

1. Further mimicry: Reduction theorems

1.1. Iteration vs. accretion

1.1.1. Motivation

"If term X can be reduced to both Y and Z, then there must be a further term V (possibly equal to either Y or Z) to which both Y and Z can be reduced."

Kenomic inversion

"If there are two morphogrammatically equivalent but semiotically different terms t_1 and t_2, $t_1 =_{MG} t_2$, and $t_1 \neq_{sem} t_2$, then two separated terms t_1 and t_2 can be produced from both t_1 and t_2, which then produce a prior term t_0 which can be produced from both t_1 and t_2."

Two representions, one solution

λ X, Y, Z:
CONS(((XY),Z):

CONS(XY), Z) ((ab), a).

$$\text{cons}\left(\left(\text{ab}\right), a\right) = \left(\text{aba}\right)\Big|\left(\text{abb}\right)\Big|\left(\text{abc}\right).$$

$$\left(\text{aba}\right) \neq_{MG} \left(\text{abb}\right) \neq_{MG} \left(\text{abc}\right).$$

Many representations, one solution.

$$\begin{pmatrix} (aba) \\ \text{II} \\ (abb) \\ \text{II} \\ (abc) \end{pmatrix} \Rightarrow \text{cons}\left((ab),\ a\right)$$

Deconstruction, step one
Church-Rosser: two representation, one solution.
Morphogrammatics: two solutions, one representation.

Example – I
Church –Rosser : Two representation, t_1, t_2,
of a formula t_0 with one solution $t_3 = t_1 = t_2$.

$t_0 \longrightarrow t_1$ and $t_0 \longrightarrow t_2$ imply $t_1 =_{ID} t_2$.

$$(br): \quad (\lambda v\, t)s \Rightarrow t\left[v/s\right]$$

$$\begin{array}{ll} (\lambda w.ww)((\lambda v.vv)u) & : t_0 \\ \swarrow \quad \searrow & \\ (\lambda v.vv)u((\lambda v.vv)u) \quad (\lambda w.ww)(uu) & : t_1, t_2\,(br) \\ \searrow \quad \swarrow & \\ uu(uu) =_{LC} uu(uu) & : t_3, t_1 = t_2. \end{array}$$

Example – II
Morphogrammatics : Two different solutions, t_1 ,
t_2 $t_1 \neq_{SEM} t_2$, of a formula t_0 with one representation t_0.
$t_0 \longrightarrow \ldots \longrightarrow t_1$ and $t_0 \longrightarrow \ldots \longrightarrow t_2$ imply $\left(t_1 \neq_{ID} t_2 \;\middle|\; t_1 =_{KENO} t_2\right)$.

$$(br_{keno}): \quad (\lambda v\, t)s \Rightarrow t\left[v/s_1\right] \;\middle|\; \ldots \;\middle|\; t\left[v/s_n\right]$$

$$\begin{array}{ll} (\lambda w.ww)((\lambda v.vv)u) & : t_0 \\ \swarrow \quad \searrow & \\ (\lambda v.vv)x((\lambda v.vv)x) \quad (\lambda w.ww)(uu) & : t_1, t_2,\ u =_{MG} x \\ \searrow \quad \swarrow & \\ xx(xx) =_{MG} uu(uu) & : t_1 \neq_{SEM} t_2,\ t_1 =_{MG} t_2. \end{array}$$

Example –I II
$\left(t_1 \neq_{ID} t_2 \;\middle|\; t_1 =_{KENO} t_2\right)$ imply $t_1 \longrightarrow \ldots \longrightarrow t_{01}$, $t_2 \longrightarrow \ldots \longrightarrow t_{02.}$, with $t_{01} =_{SEM} t_{02}$.

$$\left(br_{rev}\right):\ t\left[v/s_1\right] \mid \ldots \mid t\left[v/s_n\right] \ \Rightarrow\ \left(\lambda v\, t\right)s$$

$$\begin{array}{ll} xx\left(xx\right) =_{MG} uu\left(uu\right) & : t_1 \neq_{SEM} t_2,\ t_1 =_{MG} t_2. \\ \swarrow \qquad \searrow & \\ \left(\lambda v.vv\right)x\left(\left(\lambda v.vv\right)x\right)\left(\lambda w.ww\right)\left(uu\right) & : t_1, t_2\ ;\ u =_{MG} x \\ \searrow \qquad \swarrow & \\ \left(\lambda w.ww\right)\left(\left(\lambda v.vv\right)u\right) & : \text{syntactically}: t_{01} =_{SYNT} t_{02}\ . \end{array}$$

β-equivalence

> "Finally, β-equivalence is obtained by allowing reduction steps as well as *inverse* reduction steps, i.e., by making $\longrightarrow_\beta$ symmetric:
>
> *Definition*. We write M $=_\beta$ M' if M can be transformed into M' by zero or more reduction steps and/or *inverse* reduction steps.
> Formally, $=_\beta$ is defined to be the reflexive symmetric transitive closure of $\longrightarrow_\beta$, i.e., the smallest *equivalence* relation containing $\longrightarrow_\beta$." (Selinger, p.17)

Nonetheless, the β-equivalence is based on the principle *"One representation, one solution" a*nd is therefore not to be confused with the given examples of the concept *"Many solutions, one representation*".

1.1.2. Polycontexturality and morphogrammatics

What the difference to the polycontextural approach as it was presented under the motto *"Lambda calculi in polycontextural situations*"? As the title suggests polycontexturality of situations is presupposed to place, i.e. distribute and mediate, lambda calculi. The study of such distributed lambda calculi was partly elaborated and has delivered some interesting results. Nevertheless, polycontexturality wasn't generated by such dissemination of lambda calculi but presupposed.
Polycontexturality has its own calculus of generating distributed contextures in a polyverse.

On the other hand, a justification for the choice of polycontexturality isn't necessary. The reason is simple, there is no generally acceptable reason to opt for the classical monocontextural approach. The fact that monocontexturality is world-wide accepted and technically in many respects highly successful doesn't mean that this monocontextural approach has found a a secure foundation. Its legitimacy is pragmatically, its limits more and more obvious.
Therefore, an option for polycontexturality isn't less arbitrary than an option for the established monocontexturality.

In contrast to the contextural decision, keno- and morphogrammatic constructions as sketched with the paper *"Notes on semi-Thue Systems in a Context of Morphogrammatics"* are not presupposing but *generating* their polycontexturality on the base of their own operations. Each repetition and iteration has *per se* its retrograde recursive double role as iteration and as accretion.

Morphogrammatics is not presuming a multitude of contextures but is creating a polyverse of

contextures in each situation of a realization of an operation.

1.1.3. Modi of substitution

Definition. The (capture-avoiding) *substitution* of N for free occurrences of x in M, in symbols M[N/x], is defined as follows:

$$
\begin{aligned}
&x\left[N/x\right] \equiv N,\\
&y\left[N/x\right] \equiv y, \qquad \text{if } x \neq y,\\
&\left(MP\right)\left[N/x\right] \equiv \left(M\left[N/x\right]\right)\left(P\left[N/x\right]\right),\\
&\left(\lambda x.M\right)\left[N/x\right] \equiv \lambda x.M,\\
&\left(\lambda y.M\right)\left[N/x\right] \equiv \lambda y.\left(M\left[N/x\right]\right),\\
&\quad \text{if } x \neq y \text{ and } y \notin FV\left(N\right),\ FV:\ \text{free variable}\\
&\left(\lambda y.M\right)\left[N/x\right] \equiv \lambda y'.\left(M\left\{y/y'\right\}\left[N/x\right]\right),\ \text{if } x \neq y,\ y \in FV\left(N\right), \text{and } y\text{' fresh.}
\end{aligned}
$$

This opens up the possibility to introduce different kinds of abstractions involved in the process of substitution. The general table of different kinds of substitutions as they are introduced for morphogrammatic semi-Thue systems shall be applied.
http://memristors.memristics.com/semi-Thue/Notes%20on%20semi-Thue%20systems.pdf

Example

$$
\begin{aligned}
&\left(MP\right)\left[N/x_1\right] \equiv \left(M\left[N/x_2\right]\right)\left(P\left[N/x_3\right]\right) \text{ or}\\
&\left(MP\right)\left[x_1 := N\right] \equiv \left(M\left[x_2 := N\right]\right)\left(P\left[x_3 := N\right]\right):
\end{aligned}
$$

Identity (equality)
ID = $(x_1 \equiv x_2 \equiv x_3)$

Equality
$w \in C_{ID}$

$$\frac{u \underset{ID}{\Rightarrow} v}{w_1 u \underset{ID}{\Rightarrow} w_2 v,\ uw_3 \underset{ID}{\Rightarrow} vw_4}$$

Table $u \underset{id}{\Rightarrow} v$

Id	wx	xw	w_{12}	w_{34}
MG	+	+	+	+
SEM	+	+	+	+

Equivalence

$(x_1 =_{keno} x_2 =_{keno} x_3)$

Table $u \underset{keno}{\Rightarrow} v$

Equ	wx	xw	w_{12}	w_{34}	w_{13}	w_{24}
MG	+	+	+	+	+	+
SEM	−	−	−	−	+	+

Similarity

$(x_1 =_{sim} x_2 =_{sim} x_3)$

Table $u \underset{SIM}{\Rightarrow} v$

$u \underset{SIM}{\Rightarrow} v$	wx	xw	w_{12}	w_{34}	w_{13}	w_{24}
MG	+	+	+	+	+	+
SEM	−	−	−	−	−	−
$\frown$	+	+	.	.	.	.

Bisimilarity

$(x_1 =_{bis} x_2 =_{bis} x_3)$

Table: $u \underset{BIS}{\Longrightarrow} v$

$u \underset{BIS}{\Longrightarrow} v$	wx	xw	w_{12}	w_{34}	w_{13}	w_{24}
MG	−	−	−	−	+	+
SEM	−	−	−	−	−	−
$\frown$	+	+	.	.	.	.
length	−	−	−	−	+	+

Table of the modi of substitution for different types of lambda calculi LC.

MG	N	x	N	x	N	x	N	x	N	x
$=_{MG}$	+	+	+	+	+	+	−	−	+−	+−
$=_{sem}$	+	+	+	−	−	−	−	−	−	−
$\frown$	+	+	+	+	+	+	−	−	□	□
type	id	□	eq	□	sim	□	bisim	□	metamorph	□
LC	CLC	□	kenoLC	□	morphLC	□	bisLC	□	metamLC	□

Syntax

$x \in V \Rightarrow x \in \Lambda$

$M, N \in \Lambda \Rightarrow (MN) \in \Lambda$

$M \in \Lambda,\ x \in V \Rightarrow (\lambda x M) \in \Lambda$

$V - \{v, v', v'', \ldots\}$

Set ot free variables of M : $FV(M)$

$FV(x) = \{x\}$

$FV(M, N) = FV(M) \cup FV(N)$

$FV(\lambda x M) = FV(M) - \{x\}$

M *is a closed* $\lambda -$ *term*

if $FV(M) = \varnothing$

Substitution,

N *for free occurence of* x *in* M, $M[x := N]$

$x[x := N] \equiv N;$

$y[x := N] \equiv y,$ *if* $x \neq y;$

$(M_1 M_2)[x := N] \equiv (M_1[x := N])(M_2[x := N]);$

$(\lambda x.M_1)\lambda x[x := N] \equiv \lambda y.(M_1[x := N])$

Different version

"Reversing beta-reduction produces beta-abstraction rule."

Kenneth Slonneger, Formal syntax and semantics for programming languages, §5, p.149, 1995

Syntax of Lambda Expressions

1. t = x, x $\in$ Var

2 . t = λ t x M x and M are expressions

3. t = (MN), M, N expressions

β **- Reduction Rules**

$(br)\ (\lambda\, v\, t)\, s \Rightarrow t\ [v/s]$

b. $< s,\ t >_0 \Rightarrow s$

c. $< s,\ t >_1 \Rightarrow t.$

$(po1)\ t_1 \longrightarrow t_1$

$(po2)\ t_1 \longrightarrow t_2$ and $t_2 \longrightarrow t_3$ then $t_1 \longrightarrow t_3$

$(a1)$ if $t_1 \longrightarrow t_2$ then $(s\, t_1) \longrightarrow (s\, t_2)$

$(a2)$ if $t_1 \longrightarrow t_2$ then $(t_1\, s) \longrightarrow (t_2\, s)$

$(a3)$ if $t_1 \longrightarrow t_2$ then $(\lambda v\, t_1) \longrightarrow (\lambda v\, t_2)$

(rps) if $t_1 \longrightarrow t_2$ then if $t_1 = t_2$

2. Church-Rosser reductions for morphogrammatics

2.1. Lambda Calculus modus=ID

"In fact, when a form contains more than one lambda that can be reduced, it does not matter which one is reduced first, the result will be the same. This is known as the Church-Rosser property, or, informally, as the diamond property." (Barker)
$t_0 \longrightarrow \ldots \longrightarrow t_1$ and $t_0 \longrightarrow \ldots \longrightarrow t_2$ imply $t_1 =_{ID} t_2$.
Terms $= \{u, v, w\}$
$(br) : (\lambda v\, t)\, s \rightarrow t\left[v/s\right]_{ID}$

Identity , Equality
$u \Rightarrow_{id} v$:
$u \Rightarrow_{MG} v$, $w_1\, u \Rightarrow_{SEM} w_2\, v$ and $uw_3 \Rightarrow_{SEM} vw_4$ and
$w_1 =_{sem} w_2 =_{sem} w_3 =_{sem} w_4$. $: [++++]$

Example – I

$$(\lambda x.x + x)(\lambda y.y + y)\,1) \qquad : t_0$$

$$\swarrow \quad \searrow$$

$$(\lambda x.x + x)(1 + 1),\ ((\lambda y.y + y), 1) + ((\lambda y.y + y), 1) \ : (br),\ (a1);\ (br) : t_1, t_2$$

$$| \qquad |$$

$$|\quad (1 + 1) + ((\lambda y.y + y), 1) \qquad : (br),\ (a2) \qquad : t_2$$

$$\searrow \quad \swarrow$$

$$(1 + 1) + (1 + 1) \qquad : (br);\ (a1), (br) : t_1 = t_2.$$

General remarks to the modi of abstractions

Obviously this says that both branches are terminal and that both terminal results of the branches are semantically equal, therefore the branches are meeting in a common end. This corresponds to the graphematical equality or identity case.
Thus, we have :

$$\begin{array}{cc} \text{branch}_1 & \text{branch}_2 \\ | & | \\ t_{3.1} = (1 + 1) + (1 + 1) & (1 + 1) + (1 + 1) = t_{3.2} \end{array}$$

and $t_{3.1} =_{SEM} t_{3.2}$, hence the final result is $t_3 = (1 + 1) + (1 + 1)$.

Hence, for *equality* of terms we get: $t_{3.1} =_{SYNT} t_{3.2}$.

Again, this result is supposing a common arithmetic of the example, given by the term t_0. But what happens if we allow a game where arithmetics might be colored, and thus the results might still be syntactically equivalent but semantically different, because, say, blue is not green. Or if the number systems are running differently. This situation could result into a *kenogrammatical* calculus if we accept that the difference is not just in color but in the syntactical terms with results $t_{3.1}$ = (1 + 1) + (1 + 1) and $t_{3.2}$ = (2 + 2) +(2 + 2).

Hence, for *equivalence* of terms we get: $t_{3.1} \neq_{SEM} t_{3.2}$ but $t_{3.1} =_{KENO} t_{3.2}$.

More interesting deviations happens with the case when the terms differ in the number of sub-terms too.
Say, if we get a result with different operators "+" and "x".
$t_{3.1}$ = (1 + 1) + (1 + 1) and $t_{3.2}$ = (2 x 2) +(2 x 2).
In this case, there is no equality (identity) or kenogrammatical equivalence but *similarity*. Both terms are similar in their abstract structure which conserves the complexity of the terms, i.e. the length of the formula.

Hence, *similarity* of terms we get: $t_{3.1} \neq_{SEM} t_{3.2}$, $t_{3.1} \neq_{KENO} t_{3.2}$, but $t_{3.1} =_{SIM} t_{3.2}$.

Also similar terms are still of the same length, the number of their elements might differ. Only the *metamorphic* abstraction over terms is introducing a *bisimilarity* between terms of different complexity and therefore different length of the involved terms.
Bisimilarity happens if we get structurally different answers from the 'same' constellation t_0.
Say, t_0= (λx.x + x)(λy.y +y)1) delivers the answers $t_{3.1}$ = (1 + 1) + (1 + 1) and $t_{3.2}$ = (2 x 2) + (2 x 2) x (1 +1).

Hence, for *bisimilarity* of terms we get: $t_{3.1} \neq_{SEM} t_{3.2}$, $t_{3.1} \neq_{KENO} t_{3.2}$, $t_{3.1} \neq_{SIM} t_{3.2}$ but $t_{3.1} =_{BIS} t_{3.2}$.

All cases, *equality*, *equivalence*, *similarity* and *bisimilarity* are still accepting the main scheme of term the development, here the branching of t_0 into t_1 and $t_2$terminating in t_3. Therefore, there is not yet any *retrograde* redefinition of the scheme involved during the term development.
This retrograde redefinition is attempted with interactional and interventional *metamorphosis* of general polycontextural formal systems. •

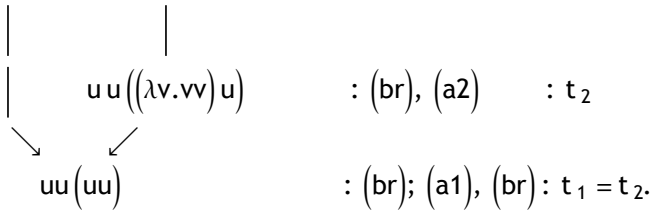
Example – II

$$(\mathrm{br}):\ \mathrm{subst}:(\lambda\, v\, t)\, s \to t\left[v/s\right]_{ID}$$

$$(\lambda w.ww)((\lambda v.vv)u) \qquad : t_0$$

$$\swarrow \qquad \searrow$$

$$(\lambda w.ww)(uu) \quad (\lambda v.vv)u((\lambda v.vv)u) \quad : (\mathrm{br}),\ (a1);\ (\mathrm{br}):\ t_1, t_2$$



"In these computations, the final values are syntactically equal, so we know that they are semantically equal. This is interesting in light of the fact that in the first computation the top-level function was applied to the value of its arguments, while the second computation, it was applied to the syntactic expressions defining its arguments!"
(Starck, p. 203)

2.2. Lambda Calculus modus=KENO

Equivalence

u ⇒ eq v :

$w_1 u \Rightarrow_{MG} w_2 v,\quad w_3 u \Rightarrow_{SEM} w_4 v,$

$u w_1 \Rightarrow_{MG} v w_2,\quad u w_3 \Rightarrow_{SEM} v w_4,$

$w_i \in C_{EQ},\ i = 1, \ldots, 4 :$

$w_1 \neq_{sem} w_2,\ w_1 =_{MG} w_2,$

$w_3 \neq_{sem} w_4,\ w_3 =_{MG} w_4,$

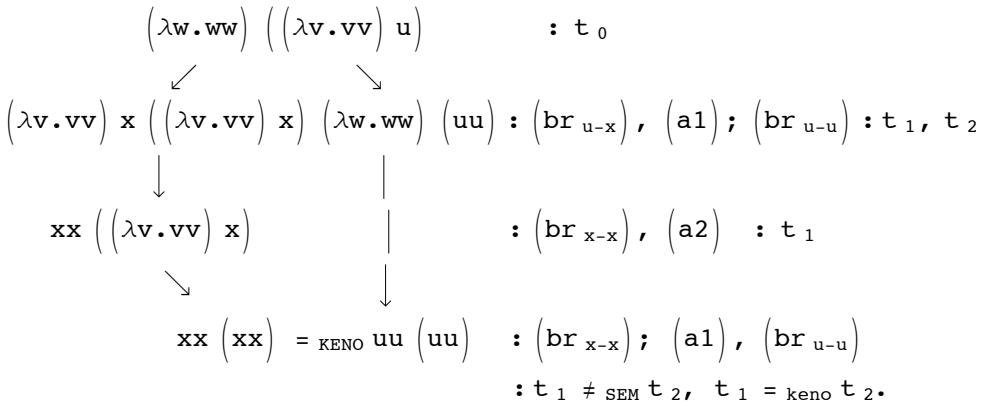
$w_1 =_{sem} w_3,\ w_1 =_{MG} w_3,$

$w_2 =_{sem} w_4,\ w_2 =_{MG} w_4.$

Example - II

$$\text{Terms} = \{u,\ v,\ w,\ x,\ y,\ z\}$$

$$(br_{keno}) :\ (\lambda\, v\, t)\ s \to t\ [v / s]_{keno}$$



Null

Example - III

$$\text{Terms} = \{u,\ v,\ w,\ x,\ y,\ z\}$$

$$\text{subst} : (\lambda\, v\, t)\ s \to t\left[v \,/\, s\right]_{\text{keno}}$$

$$(\lambda w.ww)\ ((\lambda v.vv)\ u) \qquad : t_0$$

$$\swarrow \qquad\qquad \searrow$$

$$\begin{pmatrix} (\lambda v.vv)\ x\ ((\lambda v.vv)\ x) \\ (\lambda v.vv)\ z\ ((\lambda v.vv)\ z) \end{pmatrix}, \begin{pmatrix} (\lambda w.ww)\ (uu) \\ (\lambda w.ww)\ (yy) \end{pmatrix}$$

$$\downarrow \qquad\qquad | \ :$$

$$\begin{pmatrix} (br_{u\text{-}x}),\ (a1);\ (br_{u\text{-}u}) : t_1,\ t_2 \\ (br_{u\text{-}z}),\ (a1);\ (br_{u\text{-}y}) : t'_1,\ t'_2 \end{pmatrix}$$

$$\begin{pmatrix} xx\ ((\lambda v.vv)\ x) \\ zz\ ((\lambda v.vv)\ z) \end{pmatrix} \qquad |$$

$$\searrow \qquad\qquad \swarrow$$

$$\begin{pmatrix} xx\ (xx) \\ zz\ (zz) \end{pmatrix}, \begin{pmatrix} uu\ (uu) \\ yy\ (yy) \end{pmatrix} \quad : t_1 \neq_{ID} t_2,\ t_1 =_{\text{keno}} t_2$$

Further examples

Example1

$$(\lambda x.x + x)(\lambda y.y + y)a \qquad : t_0$$

$$\swarrow \searrow$$

$$(\lambda x.x + x)(a + a), ((\lambda y.y + y)b) + ((\lambda y.y + y)b) : 'b'/t_2(br, a1); 'a'/t_1(br)$$

$$| \qquad |$$

$$| \qquad (b + b) + ((\lambda y.y + y)b) \qquad : 'b'/t_2(br,\ a2)$$

$$\searrow \qquad \swarrow$$

$$(a + a) + (a + a), (b + b) + (b + b) \qquad : 'a'/t_1(br); 'b'/t_2(a1,\ br)$$

Null

Example1 + accr + BIF

$$(\lambda x.x + x)(\lambda y.y + y)a \qquad : t_0$$

$$\swarrow \qquad \searrow$$

$$\begin{array}{c}
(\lambda x.x+x)(a+a),\ ((\lambda y.y+a)a)+((\lambda y.y+b)b) \qquad : t_1, t_2 \\
\downarrow \qquad\qquad\qquad\qquad \downarrow \\
\begin{pmatrix} (\lambda x.x+x)(a+a)_{1.1} \\ \amalg_{1.2} \\ (\lambda x.x+x)(b+b)_{1.2} \end{pmatrix}, \begin{pmatrix} (\lambda y.y+y)a)+((\lambda y.y+y)a)_{2.1} \\ \amalg_{1.2} \\ ((\lambda y.y+y)b)+((\lambda y.y+y)b)_{2.2} \end{pmatrix} \\
\\
\begin{pmatrix} (a+a)+((\lambda y.y+y)a)_{2.1} \\ \amalg_{1.2} \\ (b+b)+((\lambda y.y+y)b)_{2.2} \end{pmatrix} \\
\\
\searrow \qquad\qquad \swarrow \\
\begin{pmatrix} ((a+a)+(a+a))_{1.1} \\ \amalg_{1.2} \\ ((b+b)+(b+b))_{1.2} \end{pmatrix}, \begin{pmatrix} ((a+a)+(a+a))_{2.1} \\ \amalg_{1.2} \\ ((b+b)+(b+b))_{2.2} \end{pmatrix} \quad : t_1 =_{KENO} t_2
\end{array}$$

Counter – Example3 :

$$\begin{array}{c}
(\lambda x.x+x)(\lambda y.y+y)a) \qquad\qquad : t_0 \\
\swarrow \quad \searrow \\
(\lambda x.x+x)(a+a) \mid (\lambda x.x+x)(b+b),\ ((\lambda y.y+a)a)+((\lambda y.y+b)b) \qquad : t_1, t_2 \\
\searrow \quad \swarrow \\
((a+a)+(a+a) \mid (b+b)+(b+b)), ((a+a)+(b+b) \mid (a+b)+(b+c)) \ : \neq (a1)\, t_2 \\
\\
((a+a) \mid (b+b)), ((a+a) \mid (b+b), (b+b) \mid (b+c)) \quad : t_1 \neq_{KENO} t_2.
\end{array}$$

2.3. Lambda Calculus modus=similarity

Substitutions had been quite harmonious for the *identity* (equality) and the *equivalence* case.

Substitution in the mode of identity which is the case for classical lambda calculi is a prototype of a transformation or rewriting system whithout any deviation from identity.

Kenomic substitutions happens in the mode of *equivalence*, in contrast to the equality of identity, enabled by the kenomic abstraction or the Stirling Turn.

Nevertheless, both abstractions are *balanced* in respect of the number of occurrence of their terms.

This presumption is abandoned with the abstraction of *similarity*. Also similar terms are still of the same length, the number of their elements might differ.

Only the *metamorphic* abstraction over terms is introducing a *bisimilarity* between terms of different numbers of elements and different length of the involved terms.

Similarity

$$\frac{\left(u \Rightarrow_{SIM} v\right) \Rightarrow \begin{pmatrix} w_1 u \Rightarrow_{SIM} w_2 v \\ u w_3 \Rightarrow_{SIM} v w_4 \end{pmatrix}}{}$$

$wu \neg \Rightarrow_{SEM} wv,\ uw \neg \Rightarrow_{SEM} vw$

$w \in C_{SIM}$:

$w_1 \neq_{sem} w_2,\ w_1 =_{MG} w_2$

$w_3 \neq_{sem} w_4,\ w_3 =_{MG} w_4$

$w_1 \neq_{sem} w_3,\ w_1 =_{MG} w_3$

$w_2 \neq_{sem} w_4,\ w_2 =_{MG} w_4$

Null

Table $u \underset{SIM}{\Rightarrow} v$

$u \underset{SIM}{\Rightarrow} v$	wx	xw	w_{12}	w_{34}	w_{13}	w_{24}
MG	+	+	+	+	+	+
SEM	−	−	−	−	−	−
$\frown$	+	+	.	.	.	.

Example: $u \underset{SIM}{\Rightarrow} v$

u = [aab], v = [bba]

$u \Rightarrow_{MG} v$, $u \neg\Rightarrow_{SEM} v$

w_1 = [cc], w_2 = [dd] : $w_1 \neq_{sem} w_2$, $w_1 =_{MG} w_2$

w_3 = [ee], w_4 = [ff] : $w_3 \neq_{sem} w_4$, $w_3 =_{MG} w_4$

$w_i \in$SIM, i=1,2,3,4

length(w_1) = length(w_2)

$w_1 \neq_{sem} w_2$

sem(w_i) $\frown$ sem(u) = ø, i = 1,2

length(w_3) = length(w_4)

$w_3 \neq_{sem} w_4$

sem(w_i) $\frown$ sem(v) = ø, i = 3,4

length(w_1) = length(w_3)

$w_1 \neq_{sem} w_3$

sem(w_i) $\frown$ sem(v) = ø, i = 1, 3

length(w_2) = length(w_4)

$w_2 \neq_{sem} w_4$

$\mathrm{sem}(w_i) \frown \mathrm{sem}(v) = \emptyset,\ i = 2, 4$

$$\frac{\left[aab\right] \underset{SIM}{\Rightarrow} \left[bba\right]}{\left[aab\right]\left[cc\right] \underset{SIM}{\Rightarrow} \left[bba\right]\left[dd\right],\ \left[ee\right]\left[aab\right] \underset{SIM}{\Rightarrow} \left[ff\right]\left[bba\right]}$$

Example I – SIM

$$\mathrm{Terms} = \{u,\ v,\ w,\ x,\ y,\ z\}$$

$$\left(br_{SIM}\right):\ (\lambda\, v\, t)\, s \to t\left[v/s\right]_{SIM}$$

$$(\lambda w.ww)((\lambda v.vv)\, u) \qquad : t_0$$

$$\swarrow \qquad \searrow$$

$$(\lambda v.vv)\, x((\lambda v.vv)\, x) \quad (\lambda w.ww)(uu) : (br_{u-x}), (a1); (br_{u-u}) : t_1, t_2$$

$$\downarrow \qquad\qquad |$$

$$xx((\lambda v.vv)\, x) \qquad | \qquad : (br_{x-x}), (a2) \ : t_1$$

$$\searrow \qquad \downarrow$$

$$xx(yy) =_{SIM} uu(uu) \qquad : (br_{x-y});\ (a1), (br_{u-u})$$

$$: t_1 \neq_{SEM} t_2,\ t_1 \neq_{keno} t_2,$$

$$: t_1 =_{SIM} t_2,\ .$$

Example II – SIM

$$\mathrm{Terms} = \{u,\ v,\ w,\ x,\ y,\ z\}$$

$$\mathrm{subst} : (\lambda\, v\, t)\, s \to t\left[v/s\right]_{SIM}$$

$$(\lambda w.ww)((\lambda v.vv)\, u) \qquad : t_0$$

$$\swarrow \qquad \searrow$$

$$(\lambda v.vv)\, y((\lambda v.vv)\, x) \quad (\lambda w.ww)(uu) : (br_{u-y,\,x}), (a1); (br_{u-u}) : t_1, t_2$$

$$\downarrow \qquad\qquad |$$

$$yy((\lambda v.vv)\, x) \qquad | \qquad : (br_{y-y}), (a2) \ : t_1$$

$$\searrow \qquad \downarrow$$

$$yy(xx) =_{SIM} uu(uu) \qquad : (br_{x-y});\ (a1), (br_{u-u})$$

$$: t_1 \neq_{SEM} t_2,\ t_1 \neq_{keno} t_2,$$

$$: t_1 =_{SIM} t_2,\ .$$

Example III – SIM

$$\mathrm{Terms} = \{u,\ v,\ w,\ x,\ y,\ z\}$$

$$\text{subst} : (\lambda\, v\, t)\, s \to t\left[v / s\right]_{SIM}$$

$$(\lambda w.ww)((\lambda v.vv)\, u) \qquad : t_0$$

$$\swarrow \qquad \searrow$$

$$(\lambda v.vv)\, y((\lambda v.vv)\, x) \quad (\lambda w.ww)(uu) : (br_{u-y,u-x}), (a1); (br_{v-u}) : t_1, t_2$$

$$\downarrow \qquad\qquad |$$

$$yy((\lambda v.vv)\, x) \qquad\qquad | \qquad : (br_{v-y}), (a2) \;\; : t_1$$

$$\searrow \qquad\qquad \downarrow$$

$$yy(xx) =_{SIM} zz(uu) \qquad : (br_{v-x}); (a1), (br_{w-z})$$

$$: t_1 \neq_{SEM} t_2,\; t_1 \neq_{keno} t_2,$$

$$: t_1 =_{SIM} t_2,\; .$$

2.4. Lambda Calculus modus=bisimilarity

2.4.1. First steps to LC-bisimilarity

Table : $u \underset{BIS}{\Rightarrow} v$

$u \underset{BIS}{\Rightarrow} v$	wx	xw	w_{12}	w_{34}	w_{13}	w_{24}
MG	−	−	−	−	+	+
SEM	−	−	−	−	−	−
$\frown$	+	+	.	.	.	.
length	−	−	−	−	+	+

Conditions : $u \underset{BIS}{\Rightarrow} v$

$wu\neg \Rightarrow_{SEM} wv,\;\; uw\neg \Rightarrow_{SEM} vw$

$w \in C_{BIS}$:

$w_1 \neq_{sem} w_2,\; w_1 \neq_{MG} w_2$

$w_3 \neq_{sem} w_4,\; w_3 \neq_{MG} w_4$

$w_1 \neq_{sem} w_3,\; w_1 =_{MG} w_3$

$w_2 \neq_{sem} w_4,\; w_2 =_{MG} w_4$

Bisimilarity

$w_1 \underset{BIS}{\Rightarrow} w_2$ iff

$\exists\, w_1, w_2 : (w_1\, w_2) \in (\text{Dec, Vk, Vs, EVk, EVs})$

$\text{length}(w_1) \neq \text{length}(w_2)$:

$\exists\, (x_1, x_2) :\; \text{EVk}([w_1]) = (x_1, x_2)$

$\exists\, (y_1, y_2) :\; \text{EVs}([w_2]) = (y_1, y_2),$

$$\text{IF}\left(\begin{matrix}\text{Vs}\left(x_1, x_2\right) = \left[w_2\right] \\ \text{Vk}\left(y_1, y_2\right) = \left[w_1\right]\end{matrix}\right)\text{THEN}\left(w_1 \underset{\text{BIS}}{\Rightarrow} w_2\right).$$

Bisimilarity

$$\text{Vs}\left(\text{EVk}\left(w_1\right)\right) =_{\text{BIS}} \text{Vk}\left(\text{EVs}\left(w_2\right)\right)$$

Example

$u \underset{\text{BIS}}{\Rightarrow} v,\ w_1,\ w_2$

For

$$\text{length}\left(w_1\right) \neq \text{length}\left(w_2\right)$$

and

$$\text{EVk}\left(\left[w_1\right]\right) = \left(\left[ab\right], \left[ab\right]\right)$$

$$\text{EVs}\left(\left[w_2\right]\right) = \left(\left[ab\right], \left[ab\right]\right)$$

and

$$\frac{\left(\begin{matrix}\text{Vs}\left(\left[ab\right], \left[ab\right]\right) = \left[w_2\right] \\ \text{Vk}\left(\left[ab\right], \left[ab\right]\right) = \left[w_1\right]\end{matrix}\right) \quad \textbf{then} \quad \text{Vs}\left(\text{EVk}\left(w_1\right)\right) = \text{Vk}\left(\text{EVs}\left(w_2\right)\right)}{\left(w_1\, u \underset{\text{BIS}}{\Rightarrow} w_2\, v\right)}$$

Example – BIS

$$\text{Terms} = \left\{u,\ v,\ w,\ x,\ y,\ z\right\}$$

$$\left(\text{br}_{\text{BIS}}\right) : \left(\lambda\, v\, t\right) s \to t\left[v/s\right]_{\text{BIS}}$$

$$\left(\lambda w.ww\right)\left(\left(\lambda v.vv\right)u\right) \qquad : t_0$$

$$\left(\lambda v.vvv\right) y\left(\left(\lambda v.vvv\right)x\right) \quad \left(\lambda w.ww\right)\left(uu\right) \quad : \left(\text{br}, xx =_{\text{BIS}} vvv\right), \left(a1\right); \left(\text{br}_{v-u}\right)$$

$$yyy\left(\left(\lambda v.vvv\right)x\right) \quad \Big| \quad : \left(br\right), \left(a2\right) \ : t_1$$

$$\searrow \qquad \downarrow$$

$$yyy\left(xxx\right) =_{BIS} uu\left(uu\right) \qquad : \left(br_{v-x}\right); \left(a1\right), \left(br_{w-u}\right)$$

$$: t_1 =_{BIS} t_2, \ .$$

http://memristors.memristics.com/semi-Thue/Notes%20on%20semi-Thue%20systems.pdf

2.4.2. Linguistic example for LC-bisimilarity

A simple application "Fruit flies like a banana."

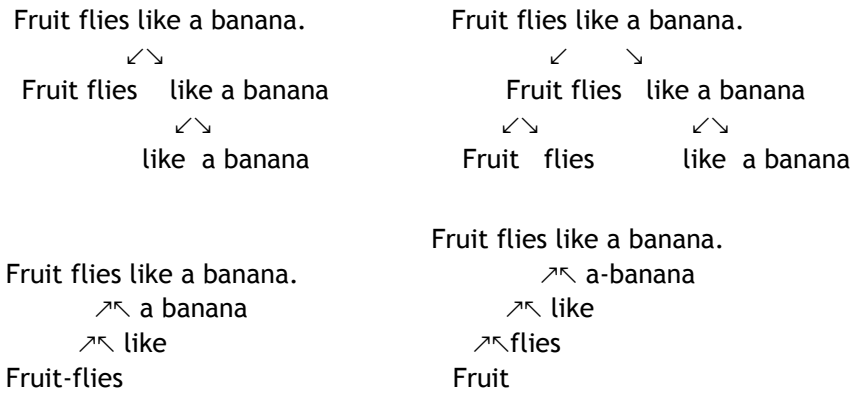
What's the meaning of an ambiguous sentence like *"Fruit flies like a banana."*?

As it is well known, the sentence has, at least, two meanings:

1. "The insects called fruit flies are positively disposed towards bananas."
2. "Something called fruit is capable of the same type of trajectory as a banana."

"*These two potential meanings are partly based on the (at least) two ways in which the phrase can be parsed.*" (Alan P. Parkes, Introduction to Languages, Machines and Logic, 2002, p. 42)

Ambiguity in languages is reflected in the existence of more than one *parse tree* for one sentence of that language.



Compositionality

The sentence X = "Fruit flies like a banana." is de/composable into two different term-trees.

X' = (x_1, x_2, x_3) and X" = (x_1, x_2, x_3, x_4) with

X' = like(banana, fruit-flies), length(X') = 3

X" = flies((like, banana), fruit), length(X") = 4.

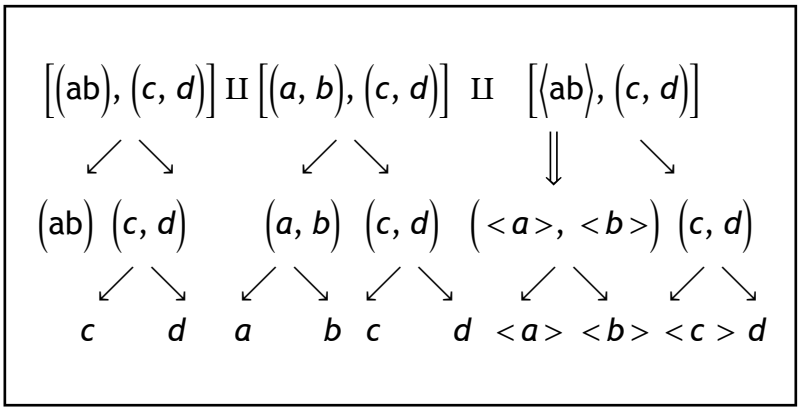
Because the meaning of the two sentences is not decomposable into the same amount of sub-terms, the relation between the two interpretations

of the full sentence is neither equal, equivalent or similar, therefore, the concept of polysemy is not properly applicable. The 2 interpretations are also not in an asymmetric meaning relation of one-many-mappings.

The 'ambiguous' meaning of the sentence X 'as such' is therefore understandable as a *bisimilar* interplay between its two different realizations X' and X". The morphogram of X, i.e. its deep-structural meaning prior to any phonological interpretation is thus the morphogram of the whole situation MG = (X, X', X").

It is a reasonable option to interpret this linguistic example which is an overlapping of two sentences of different structural 'length' with the techniques of *morphogrammatic bisimilarity*. With such an interpretation the former interpretation with the help of morphogrammatic bifunctoriality based on *fusion* gets a further and technically more unifying treatment.

Morphogrammatic interpretation based of "*fusion*" and concatenation.



http://memristors.memristics.com/Memristics%20LISPs/Memristics%20LISPs.html

Bisimilarity approximation model

$$\text{Terms} = \{u,\ v,\ w,\ x,\ y,\ z\}$$

$$\text{subst} : (\lambda\, v\, t)\, s \to t\left[v/s\right]_{BIS}$$

$$(\lambda w.ww)((\lambda v.vv)u) \quad : t_0 = \text{"}\textit{Fruit flies like a banana.}\text{"}$$

$$(\lambda v.vvv)\, y\,((\lambda v.vvv)\, x) \quad (\lambda w.ww)(uu) \quad : (br,\ xx =_{BIS} vvv), (a1); (br_{v-u})$$

$$t_1 = \text{"}\textit{Fruit flies like a banana.}\text{"},$$

$$t_2 = \text{"}\textit{Fruit} - \textit{flies like a banana.}\text{"}$$

$$yyy\left(\left(\lambda v.vvv\right)x\right) \qquad \Big| \qquad : \left(br\right), \left(a2\right) \ : t_1$$

$$\searrow \qquad \downarrow$$

$$yyy\left(xxx\right) =_{BIS} uu\left(uu\right) \qquad : \left(br_{v-x}\right); \left(a1\right), \left(br_{w-u}\right)$$

$$: t_1 =_{BIS} t_2.$$

" *Fruit flies like a banana.* " $=_{BIS}$ " *Fruit –flies like a banana.* "

X as t_0 is the *bisimilar interplay*
between t_1 and t_2 as X' and X'' as answers to t_0.

3. Types of iterability

3.1. Identity

> *"For instance, if f = λx.x is the identity function, then we have f(x) = x for any x.*
> *In particular, we can take x = f , and we get*
> *f(f) = (λx.x)(f) = f .*
> *Note that the equation f(f) = f never makes sense in ordinary mathematics, since it is not possible (for set-theoretic reasons) for a function to be included in its own domain." (Selinger, p. 7)*
> *http://www.mscs.dal.ca/~selinger/papers/lambdanotes.pdf*

The trick to produce circularity is simple, well accepted and has lost any strangeness:
"In *particular*, we can take x = f, and we get f(f) = (λx.x)(f) = f."

But it is guided by *logical* decisions and not by the means of the lambda calculus *per se*.
What holds in general, holds in particular too. But all that belongs to innocent hierarchical systems of total governance.

From a quadralectic point of view such naivety is not worth to be followed. Identity, especially in the disguise as beginnings, archai, are not unified by a simple umbrella of identity as it appears with the formula "f = λx.x" and its consequences.

3.2. Terminal terms

"For example, the lambda term (λx.y)((λz .z z)(λw.w)) can be reduced as follows.

Here, we underline each redex just before reducing it:

$$(\lambda x.y)((\lambda z\ .z\ z\ )(\lambda w.w)\ ) \rightarrow_\beta (\lambda x.y)((\lambda w.w)(\lambda w.w))$$
$$\rightarrow_\beta (\lambda x.y)(\lambda w.w)$$
$$\rightarrow_\beta y.$$

The last term, y, has no redexes and is thus in normal form. We could reduce the
same term differently, by choosing the redexes in a different order:

$$(\lambda x.y)((\lambda z\ .z\ z\ )(\lambda w.w)) \rightarrow_\beta y.$$

"This example also shows that the size of a lambda term need not decrease during reduction it can increase,or remain the same." (Selinger)

3.3. Non-terminal terms

Not every term evaluates to something; some terms can be reduced forever without reaching a normal form.The following is an example:

$(\lambda x.xx)(\lambda x.xx)$:

$$(\lambda x.xx)(\lambda x.xx) \rightarrow_\beta (\lambda x.xx)(\lambda x.xx)(\lambda x.xx)$$
$$\rightarrow_\beta (\lambda x.xx)(\lambda x.xx)(\lambda x.xx)(\lambda x.xx)(\lambda x.xx).$$
$$\rightarrow_\beta \ldots$$

In fact, most terms don't terminate.

3.4. Chiastic terms

Two infinite developments with a finite inter-relation between the 2 loops are called chiastic terms. Chiastic terms are a new kind of terminal,i.e.finite,interactions between non-terminal terms. Chiasms themselves might have non-terminal developments too.

Chiastic terms belong to a class of new term configurations and are motivated by polycontextural considerations.

$(\lambda x.xx)(\lambda x.xx) \rightarrow \beta_{\text{keno}} (\lambda x.xx)\left((\lambda x.xx)(\lambda x.xx)\,\middle|\,(\lambda y.yy)(\lambda y.yy)\right)$:

$$(\lambda x.xx)(\lambda x.xx) \rightarrow \beta_{keno} \begin{pmatrix} (\lambda x.xx)(\lambda x.xx)\ldots \\ \diamond \\ (\lambda y.yy)(\lambda y.yy)\ldots \\ \diamond \\ (\lambda z.zz)(\lambda z.zz)\ldots \\ \vdots \end{pmatrix}$$

4. From Y to WHY

4.1. Transitions to morphogrammatic fixed-points

Identity: $F =_{ID} F'$

$Y \equiv (\lambda f.\ (\lambda x.f(xx))\ (\lambda x.\ f(xx)))$
$YF \equiv (\lambda f.\ (\lambda x.f(xx))\ (\lambda x.\ f(xx)))F$
$\Rightarrow\ (\lambda x.\ F'(xx))\ (\lambda x.\ F(xx))$
$\Rightarrow\ F(\lambda x.\ F'(xx))\ (\lambda x.\ F(xx))$
$\Rightarrow\ F'(YF) = F(YF).$

Equivalence: $F \neq_{ID} F'$, $F =_{keno} F'$

$Y \equiv (\lambda f.\ (\lambda x.f(xx))\ (\lambda x.\ f(xx)))$
$YF \equiv (\lambda f.\ (\lambda x.f(xx))\ (\lambda x.\ f(xx)))F$
$\Rightarrow\ (\lambda x.\ F'(xx))\ (\lambda x.\ F(xx))$
$\Rightarrow\ F(\lambda x.\ F'(xx))\ (\lambda x.\ F(xx))$
$\Rightarrow\ F'(YF).$

Some motivation

If the meaning of a term is defined by its *use* (Wittgenstein)then we should analyze the use of all crucial terms in the construction of the FixedPoint theorem.

Because this theorem is of crucial importance for formal systems in general and for questions of computability in particular it shouldn't be exaggerated to test all the cases of the use of the main terms involved.

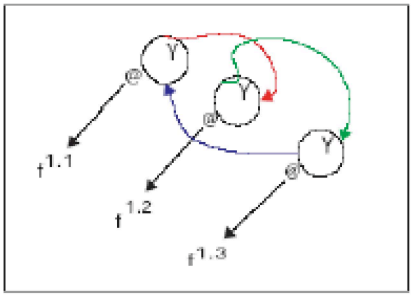
With Barendregt's formula for the construction of the Fixed Point Theorem, the term "W" is used 6 times and therefore we have to check all its possible ways of use. The comfortable excuse to use the distinction of a

syntactic and a semantic or of an object- and meta-language use of the terms to avoid further analysis isn't of leading importance in this case.

There are certainly better arguments to motivate the decision for a morphogrammatic turn towards a new understanding of the use of signs. For the purpose of a simplified introduction of the techniques of morphogrammatics it should suffice the desire for justification.

From Y functor to WHY interaction

A similar constellation appeared in *"Lambda Calculi in..."* with the modeling of a 3-contextural Y and WHY function.



Chiasm of operator and operand

"In YF, the term Y is an operator and F is an operand of the application YF. Because of the highly abstract definition of the Lambda Calculus it is possible to change the operand, step by step, to an operator.
Now, F is an *operator* to (YF) and also an *operand* to Y in (YF). This double-functionality of Y is saved in the mind of the reader, the difference is nullified by notional abstraction.
Polycontextural strategy tries, in contrast and additionally, to inscribe such a notational abstraction into a graphematic play. Because there is no trust in mental representations we have to write it down." (polyLC, p. 63)

4.2. Morphogrammatic Fixed Point scheme

4.2.1. Barendregt's Fixed Point

The ingenious construction of the FixedPoint Theorem is given by Barendregt's formula. It is a simplified definition of the Turing operator "by recognizing that we don't need to delay the evaluation of the first expression".
The following analysis of the Fixed Point Theorem is not considering the

computational aspects of “eager” and “lazy” programming.

Proof of the theorem by Barendregt:

Barendregt' proof

$\forall F \exists X FX = X.$
$Y \equiv (\lambda f.(\lambda x.f(xx))(\lambda x.f(xx)))$ such that
$\forall F F(YF) = YF.$

Define $W \equiv \lambda x.F(xx)$ and $X \equiv WW.$
Then
$X \equiv WW \equiv \lambda x.F(xx)W \equiv F(WW) \equiv FX.$

To analyze this “lazy” construction of the Y combinator we have to consider all occurrences of the main term “W”. Hence, we count 6 different uses of the term W. Therefore we offer each use of the term W a number: W^i, with $i \in \{1, ...,6\}$.

Thus, the full scheme of the Fixedpoint theorem is now marked by its numbers of occurrence of term “W” in the application.

Fixed – point scheme

Define $W^1 \equiv \lambda x.F(xx)$ and $X \equiv W^2W^3.$
Then
$X \equiv W^2W^3 \equiv \lambda x.F(xx)W^4 \equiv F(W^5W^6) \equiv FX.$

There is in fact also a difference of use in “*define* $X \equiv WW$”, W^2W^3, and the logical application “*Then* $X \equiv W$”, W^2W^3. This difference is not specially marked in this discussion.

Fixed Point scheme for W

$W^1 \equiv \lambda x.F(xx),\ X \equiv W^2W^3,\ \lambda x.F(xx)W^4,\ F(W^5W^6).$

Eager Fixed Point Combinator (Selinger)

"The lambda calculus contrasts with arithmetic in that every lambda term has a fixpoint. This is perhaps the first surprising fact about the

lambda calculus we learn in this course."

"Theorem 3.1. *In the untyped lambda calculus, every term F has a fixpoint.*

Proof. Let $A = \lambda xy.y(xxy)$, and define $\Theta = AA$. Now suppose F is any lambda term, and let $N = \Theta F$. We claim that N is a fixpoint of F. This is shown by the following calculation:

$$
\begin{aligned}
N &= \Theta F \\
&= AAF \\
&= (\lambda xy.y(xxy))AF \\
&\longrightarrow_\beta F(AAF) \qquad : (\beta)(\lambda x.M)N \longrightarrow_\beta M[N/x] \\
&= F(\Theta F) \\
&= FN.
\end{aligned}
$$

The term Θ used in the proof is called *Turing's fixpoint combinator*." (Selinger)

Analysis

$A = \lambda xy.y(xxy)$, $\Theta = AA$,
$N = \Theta F = AAF = (\lambda xy.y(xxy))AF \longrightarrow_\beta F(AAF) = F(\Theta F) = FN.$

$A^1 = \lambda xy.y(xxy)$, $\Theta = A^2A^3$,
$N = \Theta F = A^4A^5F = (\lambda xy.y(xxy))A^6F \longrightarrow_\beta F(A^7A^8F) = F(\Theta F) = FN.$

4.2.2. Distributed Fixed points

Eager Fixed Point combinator in Scheme

```
(define Y
  (lambda (f)
    ((lambda (x) (f (lambda (y) ((x x) y)))
     (lambda (x) (f (lambda (y) ((x x) y))))))).
```

Recursive FACT with Y:

```
(define fact
  (Y (lambda (f)
       (lambda (n)
         (if (zero? n) 1 (* n (f (- n 1))))))))).
```

$Y^{(3)}$-KENO

```
(define Y(3)
  (lambda (f)
    ((lambda (x) (f (lambda (y) ((x x) y)))
       (lambda (x) (f (lambda (y) ((x x) y))
         (lambda (x) (f (lambda (y) ((x x) y))))))).
```

Heterarchic distribution of $Y^{(3)}$

$$Y^{(1.2\quad .3)} \equiv \left[\lambda f^{(1.2\quad .3)}\right].\left(\begin{array}{c}\left(\left(\lambda x.f^{1}\left(xx\right)\right)\right)\\ \diamond \\ \left(\lambda x.f^{2}\left(xx\right)\right)\\ \diamond \\ \left(\lambda x.f^{3}\left(xx\right)\right)\end{array}\right)$$

Heterarchy of $Y^{(3)}$

$$Y^{(1.2\quad .3)} \equiv \lambda f^{(1.2\quad .3)}.\left(\begin{array}{c}\left(\lambda x.f^{1}\left(xx\right)\right)\left(\lambda x.f^{1}\left(xx\right)\right)\\ \text{II} \\ \left(\lambda x.f^{2}\left(xx\right)\right)\left(\lambda x.f^{2}\left(xx\right)\right)\\ \text{II} \\ \left(\lambda x.f^{3}\left(xx\right)\right)\left(\lambda x.f^{3}\left(xx\right)\right)\end{array}\right)$$

Null

Recursive 3 –*FACT with $Y^{(3)}$*

$$\lambda \text{fact}^{(1.2\quad .3)}.\left(\begin{array}{c}Y^{1}\ \left(\lambda\left(f^{1}\right)\lambda\left(n^{1}\right)\ \left(\text{if}\left(\text{zero?}\ n^{1}\right)1\ (*\ n^{1}\left(f^{1}\ \left(-\ n^{1}\ 1\right)\right)\right)\right)\\ \text{II} \\ Y^{2}\ \left(\lambda\left(f^{2}\right)\lambda\left(n^{2}\right)\ \left(\text{if}\ \left(\text{zero?}\ n^{2}\right)\ 1\ (*\ n^{2}\left(f^{2}\ \left(-\ n^{2}\ 1\right)\right)\right)\\ \text{II} \\ Y^{3}\ \left(\lambda\left(f^{3}\right)\lambda\left(n^{3}\right)\ \ \left(\text{if}\left(\text{zero?}\ n^{3}\right)1\ (*\ n^{3}\left(f^{3}\ \left(-\ n^{3}\ 1\right)\right)\right)\end{array}\right)$$

$$\lambda \mathrm{fact}^{(3)}.\ Y^{(3)}\left[\lambda\left(f^{(3)}\right)\lambda\left(n^{(3)}\right)\right]\begin{pmatrix}\left(\mathrm{if}\left(\mathrm{zero?}\ n^{1}\right)1\ (*\ \mathrm{n}^{1}\left(\mathrm{f}^{1}\left(-\ \mathrm{n}^{1}\ 1\right)\right)\right)\\ \amalg\\ \left(\mathrm{if}\left(\mathrm{zero?}\ \mathrm{n}^{2}\right)\ 1\ (*\ \mathrm{n}^{2}\left(\mathrm{f}^{2}\left(-\ \mathrm{n}^{2}\ 1\right)\right)\right)\\ \amalg\\ \left(\mathrm{if}\left(\mathrm{zero?}\ n^{3}\right)1\ (*\ \mathrm{n}^{3}\left(\mathrm{f}^{3}\left(-\ \mathrm{n}^{3}\ 1\right)\right)\right)\end{pmatrix}.$$

4.3. Morphogrammatics of Fixed Point Theorems

4.3.1. Semiotic equality

Identity : [1,1,1,1,1,1]

The most obvious use of "W" is the use in the mode of *identity*:
Thus, for all i, j W^i, W^j: $W^i \equiv W^j$, with $i,j \in \{1, \ldots,6\}$.

For all i, j W^i, W^j: $W^i \equiv W^j$, $i,j \in \{1, \ldots,6\}$
$\Rightarrow$
$X \equiv W^1W^1 \equiv \lambda x.\ F(xx)W^1 \equiv F(W^1W^1) \equiv FX.$

That is the classical case of identity:
$X \equiv WW \equiv \lambda x.\ F(xx)W \equiv F(WW) \equiv FX.$
$\Rightarrow X \equiv FX.$

Logical interpretation
$X \equiv WW \equiv \lambda x.\ non(xx)W \equiv non(WW) \equiv nonX.$
$\Rightarrow X \equiv nonX.$
$\{X , nonX\} \in$ Contradiction.

4.3.2. Kenomic equivalence

The kenomic equivalence takes the *Stirling turn*. What counts in this case is not anymore the identity of the signs in consideration but the structure of their distribution. Also the complexity of distribution is minimal in this application of the fixedpoint construction there are nevertheless enough distinctions available to enable to differentiate between an identity and a kenogrammatic use of signs.

Example1: [1,1,1; 4,4,4]

$W^1 \equiv W^2 \equiv W^3$, $W^4 \equiv W^5 \equiv W^6$

$W^1 \equiv \lambda x.\ F(xx)$, $X \equiv W^1W^1$, $\lambda x.\ F(xx)W^4$, $F(W^4W^4)$:

$W^1 \equiv \lambda x.\ F(xx)$
$X^1 \equiv W^1W^1 \cong_{keno} \lambda x.\ F(xx)W^4 \equiv F(W^4W^4) \cong_{keno} FX^4$

$\Rightarrow X^1 \cong_{keno} FX^4.$

Example 2: [1,1,1,4,1,1]
$W^1 \equiv W^2 \equiv W^3 \equiv W^5 \equiv W^6,\ W^4 \neq W^1$

$W^1 \equiv \lambda x.\ F(xx)$
$X^1 \equiv W^1W^1 \cong_{keno} \lambda x.\ F(xx)W^4 \cong_{keno} F(W^1W^1) \cong_{keno} FX^1$

$\Rightarrow X^1 \cong_{keno} FX^1.$

Example 2 – par : $\left[1,\ 1,\ 1,\ 1 \shortparallel 4,\ 1 \shortparallel 4,\ 1 \shortparallel 4\right]$
$W^1 \equiv W^2 \equiv W^3 \equiv W^5 \equiv W^6,\ W^4 \neq W^1$

$W^1 \equiv \lambda x.F\left(xx\right)$

$$X^1 \equiv W^1W^1 \cong_{keno} \begin{pmatrix} \lambda x.\ F\left(xx\right)W^1 \\ \lambda x.\ F\left(xx\right)W^4 \end{pmatrix} \cong_{keno} \begin{pmatrix} F\left(W^1W^1\right) \\ F\left(W^4W^4\right) \end{pmatrix} \cong_{keno} \begin{pmatrix} FX^1 \\ FX^4 \end{pmatrix}$$

$$\Rightarrow X^1 \cong_{keno} \begin{pmatrix} FX^1 \\ FX^4 \end{pmatrix}.$$

Example3: [1,1,1; 4; 5,5]

$W^1 \equiv W^2 \equiv W^3,\ W^4,\ W^5 \equiv W^6$

$W^1 \equiv \lambda x.\ F(xx)$
$X^1 \equiv W^1W^1 \cong_{keno} \lambda x.\ F(xx)W^4 \cong_{keno} F(W^5W^5) \cong_{keno} FX^5$

$\Rightarrow X^1 \cong_{keno} FX^5.$

Logical interpretation
a) $X^1 \equiv W^1W^1 \cong_{keno} \lambda x.\ non(xx)W^1 \cong_{keno} non(W^1W^1) \cong_{keno} nonX^1$

$\Rightarrow X^1 \cong_{keno} non^1X^1 \Rightarrow \{X^1, non^1X^1\} \in$ Noncontradiction.

b) $X^1 \equiv W^1W^1 \cong_{keno} \lambda x.\ non(xx)W^4 \cong_{keno} non(W^5W^5) \cong_{keno} nonX^5$

$\Rightarrow X^1 \cong_{keno} non^5X^5.$

$\Rightarrow \{X^1, non^5X^5\} \in$Incompatible.

4.3.3. Morphic similarity

While the kenogrammatic use in the case of equivalence is still accepting the symmetry of substitutions the case of morphic similarity is abandoning this restriction too. Hence, the necessity to replace the same term in a function like “f(xx)" is deliberated to the possibility to replace the repetition of “x” in “f(xx)" with different occurrences of the term “W”.

Example4: [1,1,1; 4; 5; 6]

$W^1 \equiv W^2 \equiv W^3, W^4, W^5, W^6$

$W^1 \equiv \lambda x.\ F(xx)$
$X^1 \equiv W^1W^1 \cong_{keno} \lambda x.\ F(xx)W^4 \cong_{sim} F(W^5W^6) \cong_{sim} FX^{5.6}$

$\Rightarrow X^1 \cong_{sim} FX^{5.6}.$

Example5: [1,2,3; 4; 2; 3]

$W^1, W^2, W^3, W^4, W^2, W^3$

$W^1 \equiv \lambda x.\ F(xx)$
$X^1 \equiv W^2W^3 \cong_{sim} \lambda x.\ F(xx)W^4 \cong_{sim} F(W^2W^3) \cong_{sim} FX^{2.3}.$
$\Rightarrow X^1 \cong_{sim} FX^{2.3}.$

Logical interpretation
$X^1 \equiv W^2W^3 \cong_{sim} \lambda x.\ non(xx)W^4 \cong_{sim} non(W^2W^3) \cong_{sim} nonX^{2.3}.$

$\Rightarrow \{X^1, non(X^2|X^3)\} \in$Incompatible.

4.3.4. Morphic bisimilarity

Morphic bisimilarity abandons the security of the process of substitution established by the presumption in charge for the identity, the equivalence and the similarity option, of the same "length" of the substituted terms. Hence, the situation "$W^1 \equiv \lambda x.\ F(xx)$" might be replaced by the *bisimilar* term "$\lambda x.\ F(xxx)$" delivering not WW but WWW with the abstraction $W^1W^1 \cong_{bis} W^5W^6W^7$.

Example6: [1,1,1; 4; 5; 6; 7]

$W^1 \equiv W^2 \equiv W^3, W^4, W^5, W^6, W^7$

$W^1 \equiv \lambda x.\ F(xx)$
$X^1 \equiv W^1W^1 \cong_{bis} \lambda x.\ F(xxx)W^4 \cong_{sim} F(W^5W^6W^7) \cong_{sim} FX^{5.6\quad .7}$

$\Rightarrow X^1 \cong_{bis} FX^{5.6\quad .7}$.

Example7: [1,2,3,3'; 4; 4; 5]

$W^1, W^2, W^3, W^3, W^4, W^2, W^3$

$W^1 \equiv \lambda x.\ F(xxx)$
$X^1 \equiv W^2W^3W^4 \cong_{bis} \lambda x.\ F(xx)W^5 \cong_{sim} F(W^6W^7) \cong_{sim} FX^{6.7}$.
$\Rightarrow X^1 \cong_{bis} FX^{6.7}$.