General area

Applied mathematics: the formulation and study of mathematical structures to model interesting or puzzling phenomena in a useful and/or enlightening way. Need not be 'grand'.

My particular eccentricity: *constructive* mathematics, that is implementable **in principle**.

Phenomena:	master/slave	layering,
	call/return	levels of abstraction,
	instruction/results,	
	stimulus/response,	...
	angels/demons	

Aspects related to well–foundedness. The limitation imposed on us by working in a given formal (mechanisable) system.

Trinity of coalgebras

dynamical systems A set and a function on it: $(\exists A : \mathrm{Set}) A \to A$.

transition systems A set and a function from it to (indexed) families of its elements: $(\exists A : \mathrm{Set}) A \to \underbrace{(\exists B : \mathrm{Set}) B \to A}_{\mathit{Fam}(B)}$.

Typical element, $\phi = (A, \lambda a.\, (B(a), \lambda b.\, c(a,b)))$.

interactive systems A set and a function from it to families of families of its elements:

$(\exists A : \mathrm{Set}) A \to \underbrace{(\exists B : \mathrm{Set}) B \to \underbrace{(\exists C : \mathrm{Set}) C \to A}_{\mathit{Fam}(A)}}_{\mathit{Fam}(\mathit{Fam}(A))}$.

Typical element, $\Phi = (A, \lambda a.\, (B(a), \lambda b.\, (C(a,b), \lambda c.\, d(a,b,c))))$.

Main structures

Duples ('Ab's)	$A : \mathrm{Set}$,
	$b : (a : A) \to A$
Triples ('ABc's)	$A : \mathrm{Set}$,
	$B : (\forall a : A)\, \mathrm{Set}$,
	$c : (\forall a : A, b : B(a))\, A$
Quadruples ('ABCd's)	$A : \mathrm{Set}$,
	$B : (\forall a : A)\, \mathrm{Set}$,
	$C : (\forall a : A, b : B(a))\, \mathrm{Set}$,
	$d : (\forall a : A, b : B(a), c : C(a,b))\, A$

Notation:

$$a' \triangleq b(a)\ ,$$
$$a[b] \triangleq c(a,b)\ ,$$
$$a[b/c] \triangleq d(a,b,c)$$

Interactive system interpretation

$a : A$	a state
$b : B(a)$	an input legal (for user) in state a
$c : C(a,b)$	an output legal (for machine) in response to b
$a[b/c] : A$	the next state after interaction b, c.

$$\xrightarrow{\;b \;:\; B(a)\;} \boxed{a \mapsto d(a,b,c)} \xrightarrow{\;c \;:\; C(a,b)\;}$$

Rule set interpretation

$a : A$	a formula
$b : B(a)$	an inference (step) yielding a
$c : C(a,b)$	the name of a sub–proof (premise) required by b
$a[b/c] : A$	the formula to be proved by sub–proof c

$$\frac{\cdots \quad \overset{c \quad \cdots}{d(a,b,c)}}{a}\ b$$

Topological interpretation

$a : A$	a basic open neighbourhood
$b : B(a)$	index of a (basic) covering of neighbourhood a
$c : C(a,b)$	index of an element of covering b
$a[b/c] : A$	the neighbourhood indexed by c

$$\mathit{Cov}(a) \;=\; \{\ \{\ a[b/c] \mid c : C(a,b)\ \} \mid b : B(a)\ \}$$

With some other structure ('stability', *etc*), a presentation of a *locale*.

Inductive constructions

- $W : (\forall A : \mathrm{Set}, B : (\forall a : A)\, \mathrm{Set})\, \mathrm{Set}$

$$W \;=\; \Sigma_{a:A} W^{B(a)}$$

Per Martin–Löf. "Wellorderings".

- $T : (\forall A : \mathrm{Set},$
  $B : (\forall a : A)\, \mathrm{Set},$
  $C : (\forall a : A, b : B(a))\, \mathrm{Set},$
  $d : (\forall a : A, b : B(a), c : C(a,b))\, A,$
  $a : A$
  $)\, \mathrm{Set}$

$$T(a) \;=\; (\exists b : B(a))(\forall c : C(a,b))\, T(a[b/c])$$

Dan Synek and Kent Petersson. "General trees".

Two predicate transformers

$$^{\circ}(P(a : A) : \mathrm{Set}) \;\triangleq\; \lambda a.\, (\exists b : B(a))(\forall c : C(a,b))\, P(a[b/c])$$
$$^{\bullet}(P(a : A) : \mathrm{Set}) \;\triangleq\; \lambda a.\, (\forall b : B(a))\, (\exists c : C(a,b)) P(a[b/c])$$

$$\mathit{Bar} \;\triangleq\; \mu P :\ {}^{\circ}P \subseteq P$$
$$\mathit{Pos} \;\triangleq\; \nu P :\ P \subseteq {}^{\bullet}P$$

Composition

Give ρ_1, ρ_2, define $\rho_1 ;_\circ \rho_2$ such that $(\rho_1 ;_\circ \rho_2)^\circ = \rho_1{}^\circ \circ \rho_2{}^\circ$

$$
;_\circ \left\{ \begin{array}{lcl}
B(a) & \triangleq & \underbrace{(\exists b : B_1(a))(\forall c : C_1(a,b))\, B_2(a[b/c]_1)}_{\rho_1{}^\circ(B_2,a)} \\
C(a,(b,f)) & \triangleq & (\exists c : C_1(a,b)) C_2(a[b/c]_1, f(c)) \\
a[(b,f)/(c_1,c_2)] & \triangleq & a[b/c_1]_1[f(c_1)/c_2]_2
\end{array} \right.
$$

Bar

Constructor: $\dfrac{x : {}^\circ Bar(a)}{\mathsf{bar}(x) : Bar(a)}$

Recursor:

$$
\begin{array}{l}
BarRec : (\forall\, G : (\forall a : A, x : Bar(a))\ \mathrm{Set} \\
\quad g : (\forall\, a : A, b : B(a), \\
\quad\quad f : (\forall c : C(a,b))\ Bar(a[b/c]), \\
\quad\quad h : (\forall c : C(a,b))\ G(a[b/c], f(c))) \\
\quad)\ G(a, \mathsf{bar}(b,f)) \\
\quad a : A, x : Bar(a) \\
)\ G(a,x)
\end{array}
$$

Equation: $(\!|G,g|\!) \triangleq BarRec(G,g)$

$$(\!|G,g|\!)(a, \mathsf{bar}(b,f)) = g(a,b,f,\lambda c.(\!|G,g|\!)(a,f(c)))$$

Bar’

Constructor: $\dfrac{x : {}^\circ Bar(a)}{\mathsf{bar}(x) : Bar(a)}$

Recursor:

$$
\begin{array}{l}
BarRec : (\forall\, G : (\forall a : A, x : Bar(a))\ \mathrm{Set} \\
\quad g : (\forall\, a : A, \\
\quad\quad (b, \lambda c.\,(f(c), h(c))) : \mathbf{let}\ \ G'(a) \ \triangleq\ (\exists t : Bar(a)) G(a,t) \\
\quad\quad\quad\quad\quad\quad\quad\quad \mathbf{in}\ \ {}^\circ G'(a) \\
\quad)\ G(a, \mathsf{bar}(b,f)) \\
\quad a : A, x : Bar(a) \\
)\ G(a,x)
\end{array}
$$

Equation: $(\!|G,g|\!) \triangleq BarRec(G,g)$

$$(\!|G,g|\!)(a, \mathsf{bar}(b,f)) = g(a,b,f,\lambda c.(\!|G,g|\!)(a,f(c)))$$

Pos

Coconstructor: $\dfrac{x : Pos(a)}{\mathsf{pos}(x) : {}^\bullet Pos(a)}$

Corecursor:

$$
\begin{array}{l}
MkPos : (\forall\, G : (\forall a : A)\ \mathrm{Set}, \\
\quad f : (\forall a : A, x : Pos(a), b : B(a))\ C(a,b), \\
\quad g : (\forall a : A, x : Pos(a), b : B(a))\ G(a[b/f(a,x,b)], \\
\quad a : A, x : G(a) \\
)\ Pos(a)
\end{array}
$$

Coequation: $[\!(G,f,g)\!] \triangleq MkPos(G,f,g)$

$$
\begin{array}{rl}
\mathsf{pos}([\!(G,f,g)\!](a,x)) = \lambda b.\,\mathbf{let} & c = f(a,x,b) : C(a,b), \\
 & x' = g(a,x,b) : G(a[b/c]) \\
\mathbf{in} & (c, [\!(G,f,g)\!](a[b/c], x'))
\end{array}
$$

Pos’

Coconstructor: $\dfrac{x : Pos(a)}{\mathsf{pos}(x) : {}^\bullet Pos(a)}$

Corecursor:

$$
\begin{array}{l}
MkPos : (\forall\, G : (\forall a : A)\ \mathrm{Set}, \\
\quad \lambda a, x, b.\,(f(a,x,b), h(a,x,b)) : (\forall a : A, x : G(a))\ {}^\bullet G(a) \\
\quad a : A, x : G(a) \\
)\ Pos(a)
\end{array}
$$

Coequation: $[\!(G,f,g)\!] \triangleq MkPos(G,f,g)$

$$
\begin{array}{rl}
\mathsf{pos}([\!(G,f,g)\!](a,x)) = \lambda b.\,\mathbf{let} & c = f(a,x,b) : C(a,b), \\
 & x' = g(a,x,b) : G(a[b/c]) \\
\mathbf{in} & (c, [\!(G,f,g)\!](a[b/c], x'))
\end{array}
$$

Trees in summer

$$
\begin{array}{l}
B^*(a : A) : \mathrm{Set} \\
\quad B^*(a : A) \ = \ \{\ \mathsf{leaf}\ \} \\
\quad\quad\quad \cup \{\ \mathsf{fork}(b,f) \mid\ b : B(a), \\
\quad\quad\quad\quad\quad f : (\forall c : C(a,b))\ B^*(a[b/c])\ \} \\
C^*(a : A, bt : B^*(a)) : \mathrm{Set} \\
\quad C^*(a, \mathsf{leaf}) \ = \ \{\ \mathsf{nil}\ \} \\
\quad C^*(a, \mathsf{fork}(b,f)) \ = \ \{\ \mathsf{cons}(c, cs) \mid\ c : C(a : A, b : B(a)) \\
\quad\quad\quad\quad\quad cs : C^*(a[b/c], f(c))\ \} \\
d^*(a : A, bt : B^*(a), cs : C^*(a, bt)) : A \\
\quad d^*(a, \mathsf{leaf}, \mathsf{nil}) \ = \ a \\
\quad d^*(a, \mathsf{fork}(b,f), \mathsf{cons}(c, cs)) \ = \ d^*(a[b/c], f(c), cs)
\end{array}
$$

A strange kind of 'dual': $\Sigma\Pi \mapsto \Pi\Sigma$

$$
\begin{array}{l}
B_*(a : A) : \mathrm{Set} \\
\quad B_*(a : A) \;=\; \{\, \mathsf{fael} \,\} \\
\qquad\qquad \cup \{\, \mathsf{krof}(f, g) \mid f : (\forall b : B(a))\, C(a, b), \\
\qquad\qquad\qquad g : (\forall b : B(a))\, B_*(a[b/f(b)]) \,\} \\
C_*(a : A, ct : B_*(a)) : \mathrm{Set} \\
\quad C_*(a, \mathsf{fael}) \;=\; \{\, \mathsf{lin} \,\} \\
\quad C_*(a, \mathsf{krof}(f, g)) \;=\; \{\, \mathsf{snoc}(b, bs) \mid b : B(a) \\
\qquad\qquad\qquad bs : C_*(a[b/f(b)], g(b)) \,\} \\
d_*(a : A, ct : B_*(a), bs : C_*(a, ct)) : A \\
\quad d_*(a, \mathsf{fael}, \mathsf{lin}) \;=\; a \\
\quad d_*(a, \mathsf{krof}(f, g), \mathsf{snoc}(b, bs)) \;=\; d_*(a[b/f(b)], g(b), bs)
\end{array}
$$

What are these things?

- – $bt : B^*(a)$ is a prescription or program for the user to deal with (feed stimuli to) a machine. Eventually the user leafs, or drives machine into deadlock.
  – $cs : C^*(a, b)$ is a trace of machine responses that leads to a state in which the user leafs. (Machine escapes deadlock.)
- – $tb : B_*(a)$ is a prescription or program for the machine to deal with (respond to stimuli from) a user. Eventually the machine faels, or drives user into deadlock.
  – $sc : C_*(a, tb)$ is a trace of user commands that leads to a state where the machine faels. (User escapes deadlock.)

Two Relation transformers

${}^{\bullet\circ}R(a_v, a) \;\triangleq$

$$(\forall b_v : B_v(a_v))\, \underbrace{(\exists bt : B^*(a_v))(\forall cs : C^*(a, bt))}_{\text{Callbacks}}\, (\exists c_v : C_v(a_v, b_v))\; R(a_v[b_v/c_v], a[bt/cs]^*)$$

Typical element: $\lambda b_v.\, (bt(b_v), \lambda cs.\, (c_v(b_v, cs), r(b_v, cs)))$

${}^{\circ\bullet}R(a_v, a) \;\triangleq$

$$(\exists b_v : B_v(a_v))\, \underbrace{(\forall bt : B^*(a))\, (\exists cs : C^*(a, bt))}_{\text{Callbacks}}\, (\forall c_v : C_v(a_v, b_v))\; R(a_v[b_v/c_v], a[bt/cs]^*)$$

Typical element: $(b_v, \lambda bt.\, (cs(bt), \lambda c_v.\, r(bt, c_v)))$

What follows is detritus

What's an triple?

A triple is a binary relation (directed graph) on A:

- $b : B(a)$ is a token of something's relationship with a.
- $c(a, b) : A$ is that thing $(a[b])$.

Reflexive and transitive closure (free category on a graph):

$$
\begin{array}{l}
B^*(a : A) : \mathrm{Set} \\
\quad B^*(a) \;=\; \{\, \mathsf{nil} \,\} \cup \{\, \mathsf{cons}(b, bt) \mid b : B(a), bt : B^*(c(a, b)) \,\} \\
c^*(a : A, b : B^*(a)) : A \\
\quad c^*(a, \mathsf{nil}) \;=\; a \\
\quad c^*(a, \mathsf{cons}(b, bt)) \;=\; c^*(c(a, b), bt)
\end{array}
$$

In what sense is * a *closure* operation?

Triples. Form reflexive and transitive closure using $\mathsf{nil} : B(a)$ and $\mathsf{cons}(b : B(a), bs : B^*(a[b])) : B^*(a)$. Then define *concatenation*

$$(bs : B^*(a)) \frown (bs' : B^*(a[bs]^*)) : B^*(a)$$

by induction on bs. This is (like) a *monoid* (multiplication).

Quadruples. Form ... closure using given $\mathsf{leaf} : B^*(a)$, and $\mathsf{fork}(b : B(a), f : (\forall c : C(a, b))\, B^*(a[b/c])) : B^*(a)$. Then define *tree–forest composition*

$$(bt : B^*(a)) \star (f : (\forall cs : C^*(a, bt))\, B^*(a[bt/cs]^*)) : B^*(a)$$

by induction on bt. This is (like) a *monad* (bind).

What's a morphism?

$$
\begin{array}{rcccl}
\alpha & : & A & \longrightarrow & A' \\
\beta(a : A) & : & B(a) & \longrightarrow & B'(\alpha(a)) \\
\gamma(a : A, b : B(a)) & : & C(a, b) & \longleftarrow & C'(\alpha(a), \beta(a, b))
\end{array}
$$

$$\alpha(a[b/\gamma(a, b, c')]) = \alpha(a)[\beta(a, b)/c']$$

Triples and quadruples are coalgebras

$$Fam(X : \mathsf{Type}) \;\triangleq\; (\exists I : \mathrm{Set}) I \to X : \mathsf{Type}$$

Typical element: $(I, \lambda i.\, x(i))$

$$
\begin{array}{rcl}
Tr & \triangleq & (\exists A : \mathrm{Set}) S \to \underbrace{(\exists B : \mathrm{Set}) B \to A}_{Fam(A)} \\
 & \triangleq & (\exists A : \mathrm{Set}) A \to Fam(A)
\end{array}
$$

Typical element: $(A, \lambda a.\, (B(a), \lambda b.\, c(a, b)))$

$$
\begin{array}{rcl}
Qu & \triangleq & (\exists S : \mathrm{Set}) S \to Fam(Fam(S)) \\
 & \triangleq & (\exists A : \mathrm{Set}) A \to (\exists B : \mathrm{Set}) B \to (\exists C : \mathrm{Set}) C \to A
\end{array}
$$

Typical element: $(A, \lambda a.\, (B(a), \lambda b.\, (C(a, b), \lambda c.\, a[b/c])))$