1983

CSP at WSS

C. A. R. Hoare
University of Wollongong

THE UNIVERSITY OF WOLLONGONG

DEPARTMENT OF COMPUTING SCIENCE

CSP at WSS

C.A.R. Hoare

Programming Research Group
Oxford University
45 Banbury Road
Oxford OX2 6PE

Abstract

This preprint contains a copy of the visuals on Communicating Sequential Processes which were presented at the First Wollongong Summer School on the Science of Programming at Sponar's Chalet, January 31 – February 9, 1983.

Preprint No 83-3 **March 25, 1983**

P.O. Box 1144, WOLLONGONG, N.S.W. AUSTRALIA
telephone (042)-282-981
telex AA29022

COMMUNICATING SEQUENTIAL PROCESSES

C.A.R. HOARE

SPONAR'S CHALET
JAN - FEB 1983

A symbol denotes a class of event in which a process may engage

coin — insertion of coin in vending machine VMS

choc — extraction of choc from VMS

The <u>alphabet</u> of a process is the set of events in which we are interested, e.g.,

$\alpha VMS = \{coin, choc\}$

(but not empty coin box)

1. $STOP_A$ never does anything.

$\alpha STOP_A = A$

2. If $x \in \alpha P$

$(x \rightarrow P)$ first does x and then behaves like P

$\alpha(x \rightarrow P) = \alpha P$

$(coin \rightarrow STOP_{\alpha VMS})$ accepts a coin before stopping

$(coin \rightarrow choc \rightarrow coin \rightarrow choc \rightarrow STOP)$ serves two customers

RECURSION

A clock does nothing but tick forever

$\alpha CLOCK = \{tick\}$

Let X be the "unknown" behaviour

$\therefore \quad X = (tick \rightarrow X)$

CLOCK is defined as the only solution

$CLOCK = \mu(X : \alpha X = \{tick\} : tick \rightarrow X)$

compare $ROOT2 = \mu(X : X \in \mathbb{R}^+ : X^2 + X - 2)$

WARNING. RHS of equation must be <u>guarded</u>: i.e., begin with $x \rightarrow ..$

$X = X$ has many solutions

$\therefore \mu X : X$ is NOT ALLOWED.

4

$CLOCK = (tick \rightarrow CLOCK)$

$= (tick \rightarrow (tick \rightarrow CLOCK))$

$= \ldots$

A simple vending machine

$VMS = coin \rightarrow choc \rightarrow VMS$

A change giving machine

$\alpha CH5 = \{in5p,\ out2p,\ out1p\}$

$CH5 = in5p \rightarrow out2p \rightarrow out1p \rightarrow out2p \rightarrow CH5$

5

CHOICE

Let $\alpha P = \alpha Q$

$\{x, y\} \subseteq \alpha P \ \wedge \ x \neq y$

Then $(x \rightarrow P \mid y \rightarrow Q)$

<u>either</u> does x and then behaves as P

<u>or</u> does y and then behaves as Q

The choice is made by some other process

A machine that offers a choice of goods

$VMCT = \mu X.\ coin \rightarrow (choc \rightarrow X \mid toffee \rightarrow X)$

6

A more complicated vending machine

$\alpha VMC = \{in1p,\ in2p,\ large,\ small,\ out1p\}$

$$
\begin{aligned}
VMC = (&in2p \rightarrow (large \rightarrow VMC \\
&\qquad\qquad \mid small \rightarrow out1p \rightarrow VMC) \\
&\mid in1p \rightarrow (small \rightarrow VMC \\
&\qquad\qquad \mid in1p \rightarrow (large \rightarrow VMC \\
&\qquad\qquad\qquad \mid in1p \rightarrow STOP) \\
&\qquad\qquad) \\
&)
\end{aligned}
$$

WARNING: DO NOT INSERT THREE COINS.

7.

GENERAL CHOICE

Let A be a set of events

Let Px be a process with alphabet B (for each x in A)

If $A \subseteq B$

$(x : A \rightarrow Px)$ First does some x in A then behaves as Px

A process which always does anything desired

$\alpha RUN_A = A$

$RUN_A = (x : A \rightarrow RUN_A)$

2)

All processes (so far) can be expressed by general choice

$STOP = (x:\{\} \to Px) = (y:\{\} \to Qy)$

$(b \to P) = (x:\{b\} \to P)$

$(b \to P \mid c \to Q) = (x:\{b,c\} \to$
$\underline{if}\ x = b\ \underline{then}\ P\ \underline{else}\ Q)$

$\mu X.(x:A \to P(x,X)) =$
$(x:A \to P(x, \mu X.(x:A \to P(x,X))))$

MUTUAL RECURSION

P and Q may be defined as the unique processes satisfying the simultaneous equations:

$P = coin \to Q$

$Q = choc \to P$

Theorem. $P = VMS$

Proof $P = coin \to (choc \to P)$ subst

VMS is the only solution of this equation

AN INFINITE SET OF EQUATIONS

$\alpha CT_n = \{up, down, around\}$ for $n \geq 0$

CT_n is the behaviour at height n.

$CT_0 = (around \to CT_0 \mid up \to CT_1)$

$CT_{n+1} = (down \to CT_n \mid up \to CT_{n+2})$

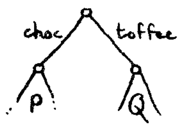
$ROCKET = CT_0$

PICTURES

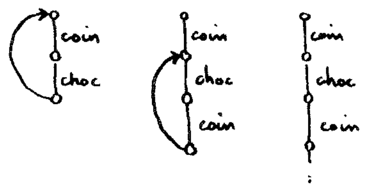
STOP

$coin \to STOP$

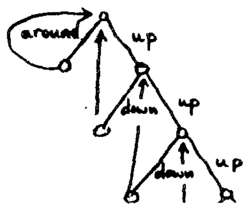
$(choc \to P \mid toffee \to Q)$



VMS



CT_0



3)

LAW

$$(x : A \to P\,x) = (y : B \to Q\,y)$$
$$\equiv \; A = B \wedge \underline{A}\,(x : x \in A : P\,x = Q\,x)$$

Corollaries

$STOP \neq (d \to P)$

$(c \to P) \neq (d \to Q)$ if $c \neq d$

$(c \to P \mid d \to Q) = (d \to Q \mid c \to P)$

$(c \to P) = (c \to Q) \;\equiv\; P = Q$

LAW

Let $F(Y)$ be guarded

$$(Y = F(Y)) \equiv (Y = \mu X.\, F(X))$$

Corollary

$$F(\mu X.\, F(X)) = \mu X.\, F(X)$$

recall $VMS = \mu X.\ coin \to choc \to X$

let $VMT = \mu X.\ choc \to coin \to X$

Theorem.

$$VMS = coin \to VMT$$

Proof

IMPLEMENTATION
(in LISP')

$(x : A \to P\,x)$ is implemented by

$\lambda x.$ $\underline{\text{if}}$ ismember (x, A) $\underline{\text{then}}$ $P(x)$ $\underline{\text{else}}$ "BLEEP

STOP $\approx$ λx "BLEEP

$(b \to P)$ $\approx$ prefix ("b, P)

prefix (c, P) = $\lambda x.$ $\underline{\text{if}}$ $x = c$ $\underline{\text{then}}$ P $\underline{\text{else}}$ "BLEEP

VMS $\approx$ LABEL X, prefix ("coin, prefix ("choc, X))

$(b \to P \mid c \to Q)$ $\approx$ choice2 ("b, P, "c, Q)

ROCKET $\approx$ LABEL CT. λn

$\underline{\text{if}}$ $n = 0$ $\underline{\text{then}}$ choice2 ("around, CT(0),
"up , CT(1))
$\underline{\text{else}}$ choice2 ("down , CT(n-1),
"up , CT(n+1)) (0)

TRACES

A trace of a process is a sequence of events recording its behaviour up to some moment.

<coin, choc, coin> is a trace of VMS

<coin> is also a trace of VMS

<> is a trace of every process

<coin, choc>, <coin, toffee>
are traces of VMCT

<in1, in1, in1> is a trace of VMC

There is no x such that
<in1, in1, in1, x> is a trace of VMC

4)

CATENATION 2.2

$s \frown t$ is the result of writing t after s

$\langle coin, choc \rangle \frown \langle coin, toffee \rangle = \langle coin, choc, coin, toffee \rangle$

$s \frown \langle\rangle = \langle\rangle \frown s = s$

$s \frown (t \frown u) = (s \frown t) \frown u$

$s \frown t = s \frown u \Rightarrow t = u \quad , \quad t \frown s = u \frown s \Rightarrow t = u$

$s \frown t = \langle\rangle \Rightarrow s = t = \langle\rangle$

POWER

$t^0 = \langle\rangle$

$t^{n+1} = t \frown t^n = t^n \frown t$

$(s \frown t)^{n+1} = s \frown (t \frown s)^n \frown t$

Exercise: prove this last law from the preceding ones.

RESTRICTION 2.3

$t \upharpoonright A$ is formed from t by omitting symbols outside A

$\langle around, up, down, around \rangle \upharpoonright \{up, down\}$
$= \langle up, down \rangle$

$\langle\rangle \upharpoonright A = \langle\rangle$ strict

$(s \frown t) \upharpoonright A = (s \upharpoonright A) \frown (t \upharpoonright A)$ distributive

$\langle x \rangle \upharpoonright A = \langle x \rangle$ if $x \in A$

$= \langle\rangle$ if $x \notin A$

$s \upharpoonright \{\} = \langle\rangle$

$(s \upharpoonright A) \upharpoonright B = s \upharpoonright (A \cap B) = (s \upharpoonright B) \upharpoonright A$

NOTE: a strict distributive function is defined by its effect on singleton sequences.

Proofs. 2.4

To prove $\underline{A}\, s :: P(s)$

(1) prove $P(\langle\rangle)$

(2) prove $\underline{A}(x :: P(\langle x \rangle))$

(3) prove $\underline{A}(s,t :: Ps \wedge Pt \Rightarrow P(s \frown t))$

Example

To prove $\underline{A}\, s :: (s \upharpoonright A) \upharpoonright B = s \upharpoonright (A \cap B)$

(1) $(\langle\rangle \upharpoonright A) \upharpoonright B = \langle\rangle \upharpoonright B = \langle\rangle = \langle\rangle \upharpoonright (A \cap B)$

(2) $(\langle x \rangle \upharpoonright A) \upharpoonright B = \{\text{case 1: } x \in A \cap B\}$

$= (\langle x \rangle) \upharpoonright B = \langle x \rangle = \langle x \rangle \upharpoonright (A \cap B)$

" $= \{\text{case 2: } x \in A - B\}$

$= \langle x \rangle \upharpoonright B = \langle\rangle = \langle x \rangle \upharpoonright (A \cap B)$

" $= \{\text{case 3: } x \notin A\}$

$= \langle\rangle \upharpoonright B = \langle\rangle = \langle x \rangle \upharpoonright (A \cap B)$

(3) $((s \frown t) \upharpoonright A) \upharpoonright B = ((s \upharpoonright A) \frown (t \upharpoonright A)) \upharpoonright B$

$= ((s \upharpoonright A) \upharpoonright B) \frown ((t \upharpoonright A) \upharpoonright B)$

{ind. hyp} $= s \upharpoonright (A \cap B) \frown t \upharpoonright (A \cap B)$

$= (s \frown t) \upharpoonright (A \cap B)$

CHANGE OF SYMBOL 2.5

Let f be a function from symbols to symbols. – total

$f^*(s)$ is result of applying f to each symbol of s

$double^*(\langle 1, 5, 3, 1 \rangle) = \langle 2, 10, 6, 2 \rangle$

LAWS

$f^*(\langle\rangle) = \langle\rangle$

$f^*(\langle x \rangle) = \langle f(x) \rangle$

$f^*(s \frown t) = f^*(s) \frown f^*(t)$

$* \; f^*(s \upharpoonright A) = f^*(s) \upharpoonright f(A)$

Exercise: (1) give a counter-example.

(2) state a property of f which makes the law valid

(3) prove it! from the previous laws.

5)

MISCELLANEOUS

A^* is the set of all finite traces with no symbols outside A

$$A^* = \{s \mid s \upharpoonright A = s\}$$

If $s \neq \langle\rangle$ then

s_0 is its first element

s' is result of removing first element

$(\langle x\rangle^\frown s)_0 = x$, $(\langle x\rangle^\frown s)' = s$

$s \neq \langle\rangle \Rightarrow s = \langle s_0\rangle^\frown s'$

$\# s$ is the length of s

$s.b$ is the number of occurrences of b in s

$$s.b = \#(s \upharpoonright \{b\})$$

REVERSAL

$\bar{s}$ is s written backwards.

$$\overline{\langle a,b,c\rangle} = \langle c,b,a\rangle$$

$$last(s) = \bar{s}_0$$

$$trunc(s) = \overline{\bar{s}'}$$

Exercise.

Give 10 interesting laws governing reversal.

IMPLEMENTATION

$\langle\rangle$	NIL
$\langle x\rangle$	cons (x, NIL)
$\langle x\rangle^\frown s$	cons (x, s)
$s^\frown t$	append (s, t)
s_0	car (s)
s'	cdr (s)

THM. $\langle\rangle^\frown t = t \;\wedge\; (\langle x\rangle^\frown s)^\frown t = \langle x\rangle^\frown(s^\frown t)$

$\therefore$ append (s, t) = <u>if</u> $s = \langle\rangle$ <u>then</u> t
<u>else</u> cons (car(s), append (cdr(s), t))

Let f be a distributive function.

$f(s)$ = <u>if</u> $s = \langle\rangle$ <u>then</u> $f(\langle\rangle)$
<u>else</u> $f(\langle s_0\rangle)$ ^ $f(s')$

$f(s)$ = <u>if</u> s = NIL <u>then</u> $f(\langle\rangle)$
<u>else</u> append ($f(\langle s_0\rangle)$, f(cdr(s)))

EXAMPLE

$\langle\rangle \upharpoonright A = \langle\rangle$

$\langle s_0\rangle \upharpoonright A$ = <u>if</u> $s_0 \in A$ <u>then</u> $\langle s_0\rangle$ <u>else</u> $\langle\rangle$

$\therefore (\langle s_0\rangle \upharpoonright A)^\frown(s' \upharpoonright A) = s \upharpoonright A$
= <u>if</u> $s_0 \in A$ <u>then</u> $\langle s_0\rangle^\frown(s' \upharpoonright A)$ <u>else</u> $\langle\rangle^\frown(s' \upharpoonright A)$

$\therefore$ restrict (s, A) = <u>if</u> s = NIL <u>then</u> NIL
<u>else</u> <u>if</u> ismember (car(s), A)
<u>then</u> cons (car(s), restrict(cdr(s), A))
<u>else</u> restrict(cdr(s), A)

EXERCISE

(1) Implement $f^*(s)$ as the LISP function star(f, s); state explicitly all theorems used.

(2) Implement $\bar{s}$ as reverse(s), (together with the relevant theorems)

6}

ORDERING

$s \le t$ means t begins with a copy of s

$s \le t \;=\; \exists\,(u :: s^\frown u = t)$

$\le$ is a partial order, i.e:

$\langle\rangle \le s$

$s \le s$ reflexive

$s \le t \wedge t \le s \Rightarrow s = t$ antisymm

$s \le t \wedge t \le u \Rightarrow s \le u$ transitive

$s \ \underline{in}\ t \;=\; \underline{E}\,(u, v :: u^\frown s^\frown v = t)$

$\underline{in}$ is a partial order.

$s \overset{n}{\le} t \;=\; s \le t \ \&\ \#t \le \#s + n$

$s \overset{0}{\le} t \;\equiv\; s = t$

$s \overset{n+m}{\le} t \;\equiv\; \exists u\ \ s \overset{n}{\le} u \wedge u \overset{m}{\le} t$

i.e., $\overset{n}{\le} ; \overset{m}{\le} \;=\; \overset{n+m}{\le}$

$s[0] = s_0$ if $0 < \#s$

$s[i+1] = s'[i]$

Let f, g be a total functions from $\mathbb{N}$ to $\mathbb{N}$

$(s \circ f)[i] = s[f(i)]$

$(s \circ f) \circ g = s \circ (f \circ g)$

f is strictly monotonic (SM) if

$\underline{A}\,(i, j : 0 \le i < j : f(i) < f(j))$

$s \ \underline{\mathbf{sub}}\ t \;=\; \underline{E}\,(f : f \text{ is SM} : s = t \circ f)$

$\underline{\mathbf{sub}}$ is a partial order

Let f be SM, with

$\overline{\text{range}(f)}$ infinite

$\bar{f}$ is the SM function s.t

$\text{range}(\bar{f}) = \overline{\text{range}(f)}$

- $\bar{\bar{f}} = f$

s interleaves $(t, u) \;=\; s \sim (t, u)$

$\quad \text{E}f : f \text{ is SM} : t = s \circ f \wedge u = s \circ \bar{f}$

$s \sim (t, u) \equiv s \sim (u, t)$

$s \sim (t, u) \wedge t \sim (v, w)$

$\quad \Rightarrow \exists x\ \ x \sim (u, w) \wedge s \sim (v, x)$

4 A drinks dispenser DD

$\alpha DD = \{\text{setgin}, \text{setwhisky}, \text{gin}, \text{whisky}, \text{coin}\}$

DD sells gin if the most recent setting was setgin, or whisky if setwhisky

5. VMS2 behaves like VMS, except that it accepts up to two coins before dispensing up to two chocs

6. Draw pictures of your answers to 3, 4 and 5

7. $VMS \triangleq \mu X.\ coin \to choc \to X$

$VMT \triangleq \mu X.\ choc \to coin \to X$

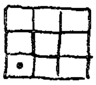
Prove that

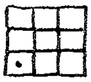
$VMS = \overset{coin}{\cancel{choc}} \to VMT$

7) EXERCISES

1. αCTR $= \{up, right\}$

CTRA 

CTRB 

CTRC 

2. αVMD $= \{in1p, in2p, choc\}$
choc costs 3p. Any combination and order of coins must be accepted.

3. αCTRD $= \{up, down, left, right\}$
moves on same board as CTRA

RECURSION AGAIN. A.2

To find the solution of the equation

$$P = F(P)$$

where F is guarded, and uses only approved* notations.

Define $F^0(P) = P$

$$F^{n+1}(P) = F(F^n(P))$$
$$= F(F(\ldots\ F(P)\ldots))$$

The required answer is

$$\underline{A}(n : n \geq 0 : F^n(true))$$

* i.e., continuous

ASSERTIONS A.1

Let's identify a process with a predicate $P(tr)$ which describes all its possible traces tr

$STOP = tr = \langle\rangle$

$(coin \to STOP) = tr = \langle\rangle \vee tr = \langle coin\rangle$

$CLOCK = tr \upharpoonright \{tick\} = tr$

$VMS = \underline{E}n : n \geq 0 : tr \leq \langle coin, choc\rangle^n$

In general

$$(x : A \to P_x) = tr = \langle\rangle$$
$$\vee (tr_0 \in A \wedge P(tr_0)[tr'/tr])$$

$\mu X.\ F(X) =$ the unique predicate P s.t
$$P = F(P)$$

$STOP = tr = \langle\rangle$

$(c \to P) = tr = \langle\rangle \vee (tr_0 = c \wedge P[tr'/tr])$

A.3

$$X \equiv coin \to choc \to X$$

$X \equiv tr = \langle\rangle \vee (tr_0 = coin \wedge (choc \to X)[tr'/tr])$

$\equiv tr = \langle\rangle \vee$
$(tr_0 = coin \wedge (tr' = \langle\rangle$
$\vee (tr'_0 = choc \wedge X[tr''/tr])))$

$\equiv tr = \langle\rangle \vee tr = \langle coin\rangle$
$\vee \langle coin, choc\rangle \leq tr \wedge X[tr''/tr]$

$P_0 \equiv true$

$P_1 \equiv tr \leq \langle coin, choc\rangle \vee \langle coin, choc\rangle \leq tr$

$P_2 \equiv tr \leq \langle coin\rangle \vee (\langle coin, choc\rangle \leq tr$
$\wedge (tr'' \leq \langle coin, choc\rangle \vee \langle coin, choc\rangle \leq tr''))$
$\equiv tr \leq \langle coin, choc\rangle^2 \vee \langle coin, choc\rangle^2 \leq tr$
$\ldots$
$P_n \equiv tr \leq \langle coin, choc\rangle^n \vee \langle coin, choc\rangle^n \leq tr$

$\therefore \mu X.\ coin \to choc \to X$

$$= \underline{A}(n : n \geq 0 : tr \leq \langle coin, choc\rangle^n \vee \langle coin, choc\rangle^n \leq tr)$$

$=$

8) CONTINUITY

A4.

A *chain* of predicates satisfies

$$[\underline{A}\, n : n \geq 0 : P_{n+1} \Rightarrow P_n]$$

A predicate transformer F is *continuous* if

$$F(\underline{A}(n : n \geq 0 : P_n)) = \underline{A}(n : n \geq 0 : F(P_n))$$

for *all* chains

Lemma. If F and G are continuous, so is $F \circ G$

Lemma. If F is continuous, $F^n(\mathrm{true})$ is a chain.

A5

If F is continuous

Let $X = \underline{A}(n : n \geq 0 : F^n(\mathrm{true}))$

Then $F(X) = X$

Proof. LHS =

$$\underline{A}(n : n \geq 0 : F(F^n(\mathrm{true})))$$

$$= \mathrm{true} \wedge \underline{A}(n : n \geq 1 : F^n(\mathrm{true}))$$

$$= \mathrm{RHS}.$$

SPECIFICATIONS.

A.6.

A process is *specified* by giving all desired properties of its traces.

"It mustn't accept more than 70 coins"

$VMC70 = \# tr \upharpoonright \{in1p, in2p\} \leq 70$

"It mustn't accept three consecutive pennies"

$VMCBAR3 = \neg \langle in1p \rangle^3 \; \underline{in} \; tr$

"An improved VMC."

$VMCOK = VMC$

$\wedge\, VMCBAR3 \wedge VMC70$

ALWAYS.

A.7

Let P be a predicate intended to be true of *all* traces tr of a process. ∴ it must be satisfied by all prefixes of tr too. So we define:

$$\Box P = \underline{A}(s : s \leq tr : P[s/tr])$$

Provided that $P[\langle\rangle/tr]$

$\Box P$ uniquely defines a process, with exactly those traces which satisfy it.

9) Examples.

"It mustn't lose money"

$NOLOSS = \Box(tr.choc \leq tr.coin)$

"It mustn't cheat the client"

$FAIR = \Box(tr.coin \leq tr.choc + 1)$

"It must please owner and client"

$NOLOSS \wedge FAIR$

This specifies VMS

A specification of CT_n:

$$\Box\,(tr.down \leq tr.up + n \;\wedge\; last(tr) = around \Rightarrow tr.down = tr.up + n)$$

LAWS

$$\Box P \Rightarrow P$$

$$\Box\Box P \equiv \Box P$$

$$\Box(P \wedge Q) \equiv (\Box P) \wedge (\Box Q)$$

$$\Box\,\underline{A}(n :: P n) \equiv \underline{A}(n :: \Box P n)$$

$tr = \langle\rangle \vee$

$$(\Box P)[tr'/tr] \equiv \Box(P[tr'/tr])$$

if f is <u>healthy</u>

$$(\Box P)[f(tr)/tr] = \Box(P[f(tr)/tr])$$

Prove $VMS = \Box\, 0 \leq tr.coin - tr.choc < 2$

Proof. $coin \rightarrow choc \rightarrow VMS$

$$= (tr \leq \langle coin\rangle$$
$$\vee (\langle coin, choc\rangle \leq tr$$
$$\wedge \Box\, 0 \leq tr''.coin - tr''.choc < 2))$$
$$= VMS$$

AFTER

If s is a trace of P

P/s describes the behaviour of P after it has engaged in s

$VMC/\langle in2p, small\rangle = out1p \rightarrow VMC$

$VMC/\langle in1p\rangle^3 = STOP$

LAWS

$$P/\langle\rangle = P$$

$$P/(s^\frown t) = (P/s)/t$$

$$(x : A \rightarrow Px)/\langle y\rangle = Py \quad \text{if } y \in A$$

ASSERTION

Provided $P[s/tr]$

$$P/s = P[s^\frown tr/tr]$$

$$(\Box P)/s \doteq \Box(P/s)$$

10) EXAMPLE: BACKTRACKING

Let P be specification of a mechanism driven by "menu-select" & keyboard.

αP contains a symbol for each key except backspace

We wish to transform P to a spec backable (P)

with alphabet $\alpha P \cup \{\text{backspace}\}$ which specifies a mechanism which behaves as P except that backspace cancels the effect of the most recent uncancelled symbol in αP

$$\text{backable}(P) = \Box(P[\text{clean}(tr)/tr])$$

$$\text{clean}(s) = s \quad \text{if } s \in \{\text{backspace}\}^*$$

$$\text{clean}(s\frown\langle x\rangle) = (\text{clean } s)\frown\langle x\rangle \quad x \in \alpha P$$

$$\text{clean}(s\frown\langle x\rangle\frown\langle\text{backspace}\rangle\frown t)$$
$$= \text{clean}(s\frown t) \qquad \text{for } x \in \alpha P$$

IMPLEMENTATION

back (Q, P) behaves like backable (P) except that an extra backspace will cause it to behave like Q instead.

back (Q, P) = λx.
  <u>if</u> x = backspace <u>then</u> Q
<u>else</u> <u>if</u> P(x) = "BLEEP <u>then</u> "BLEEP
  <u>else</u> back (back (Q, P), P(x))

'ackable (P) = λx.
  <u>if</u> P(x) = "BLEEP <u>then</u> "BLEEP
  <u>else</u> back (backable (P), P(x))

EXERCISES

Specify the following predicate transformers:

(1) restart $\tilde{\in}$ αP

<u>restartable (P)</u> behaves like P; but when <u>restart</u> is pressed, it starts from the beginning again.

(2) $\alpha P \cap \{\text{save}, \text{restore}\} = \{\}$

<u>saveable (P)</u> behaves Like P; but when <u>restore</u> is pressed it returns to the state just before the most recent <u>save</u>

INTERSECTION

Let $\alpha P = \alpha Q = \alpha(P \wedge Q)$

$P \wedge Q$ behaves like both P and Q

Each event that occurs requires simultaneous participation of both.

GRCUST = (choc → GRCUST
  | coin → choc → GRCUST)

GRCUST ∧ VMS = μX. coin → choc → X

FOOLC = (in2p → large → FOOLC
  | in1p → large → FOOLC)

FOOLC ∧ VMC = μX.(in2p → large → X
  | in1p → STOP)

CONCURRENCY

3.2

$$P \,||\, Q$$

$\alpha(P||Q) = \alpha P \cup \alpha Q$

If $x \in \alpha P - \alpha Q$, $(P||Q)$ does x whenever P does

If $y \in \alpha Q - \alpha P$, $(P||Q)$ does y whenever Q does

If $z \in \alpha P \cap \alpha Q$, $(P||Q)$ does z whenever both P and Q do z

$\alpha NOISYVM = \{coin, clink, choc, toffee\}$

$NOISYVM = (coin \to clink \to choc \to NOISYVM)$

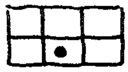
$\alpha CUST = \{coin, startcurse, endcurse, choc, toffee\}$

$$CUST = coin \to (toffee \to CUST \mid startcurse \to endcurse \to choc \to CUST)$$

3.3.

$NOISYVM \,||\, CUST =$

$$\mu X.\ coin \to (clink \to startcurse \to endcurse \to choc \to X$$
$$\mid startcurse \to clink \to endcurse \to choc \to X$$
$$\mid startcurse \to endcurse \to clink \to choc \to X)$$



$\alpha CTR = \{up, down, left, right\}$

Let $\alpha P = \{up, down\}$

$P = (up \to down \to P)$

$\alpha Q = \{left, right\}$

$Q = (right \to left \to Q \mid left \to right \to Q)$

$CTR = P||Q$

LAWS.

3.4.

$(P||Q)/s = (P/s{\restriction}\alpha P) \,||\, (Q/s{\restriction}\alpha Q)$

$||$ is symmetric and associative

Let $P = x : A \to P\,x$

$Q = y : B \to Q\,y$

Then $P||Q =$

$$z : (A \ulcorner \alpha P \cap \alpha Q \urcorner B) \to (\underline{\text{if}}\ z \in A\ \underline{\text{then}}\ P\,z\ \underline{\text{else}}\ P \;||\; \underline{\text{if}}\ z \in B\ \underline{\text{then}}\ Q\,z\ \underline{\text{else}}\ Q)$$

where $A \ulcorner X \urcorner B = (X \cap A \cap B) \cup (\overline{X} \cap A) \cup (\overline{X} \cap B)$

ASSERTION

$$P||Q \equiv tr \in (\alpha P \cup \alpha Q)^* \wedge P[(tr{\restriction}\alpha P)/tr] \wedge Q[(tr{\restriction}\alpha Q)/tr]$$

EXAMPLE

3.5

$\alpha P = \{a,c\}$ $\quad P = a \to c \to P$

$\alpha Q = \{b,c\}$ $\quad Q = c \to b \to Q$

1. $P||Q = a \to (c \to P \,||\, c \to b \to Q)$
$= a \to c \to (P \,||\, b \to Q)$

2. $P || b \to Q = (a \to (c \to P \,||\, b \to Q) \mid b \to (P||Q))$
$= (a \to b \to c \to (P \,||\, b \to Q) \mid b \to a \to c \to (P \,||\, b \to Q))$
$= \mu X.(a \to b \to c \to X \mid b \to a \to c \to X)$

3. $= (P||Q)/\langle a,c \rangle$

12)

3.6

Let $n \overset{k}{\leq} m \;\&\; n \leq m \wedge m \leq n+k$

$P = \Box\, tr.c \overset{1}{\leq} tr.a \;\; \Box\, 0 \leq tr.a - tr.c < 2$

$Q = \Box\, tr.b \overset{1}{\leq} tr.c$

$P \| Q = tr \in \{a,b,c\}^*$

$\wedge \Box\, tr \restriction \{a,c\}.c \overset{1}{\leq} tr \restriction \{a,c\}.a$

$\wedge \Box\, tr \restriction \{b,c\}.b \overset{1}{\leq} tr \restriction \{b,c\}.c$

$= \Box\, tr.b \overset{1}{\leq} tr.c \wedge tr.c \overset{1}{\leq} tr.a$

$\therefore (P\|Q)/\langle a,c\rangle =$

$= \Box\, (\langle a,c\rangle \frown tr).b \overset{1}{\leq} (\langle a,c\rangle \frown tr).c \overset{1}{\leq} (\langle a,c\rangle \frown tr).a$

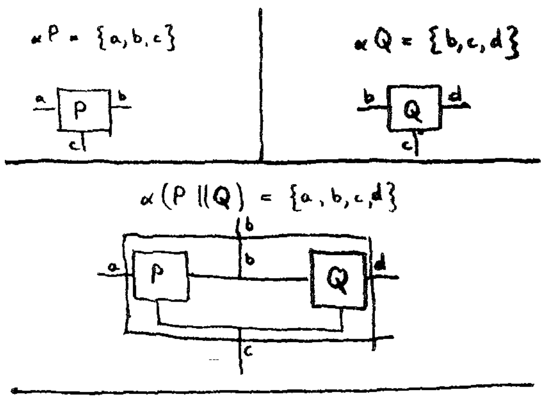
$= \Box\, tr.b \overset{1}{\leq} tr.c + 1 \overset{1}{\leq} tr.a + 1$

$\alpha(P \sim Q) = \alpha P \cup \alpha Q \cup \{switch\}$

$(P \sim Q)$ first behaves like P.
After switch it behaves like Q.
On each subsequent switch it suspends execution of the current one of $\{P, Q\}$, and resumes execution of the other at the point it was most recently suspended.

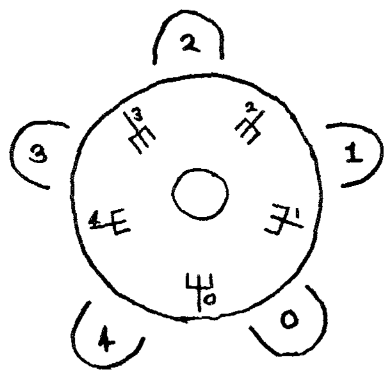
PICTURES.

3.7



THE DINING PHILOSOPHERS.

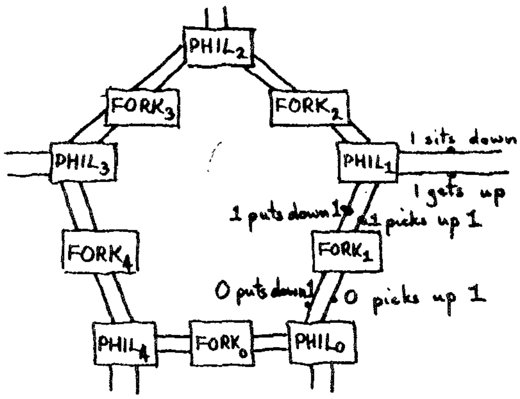
3.8



13)

ALPHABETS

$\oplus$ $\ominus$ mean arithmetic modulo 5.



αPHIL$_i$ = { i sits down, i gets up,
i picks up i , i picks up i$\oplus$1,
i puts down i , i puts down i$\oplus$1}

αFORK$_i$ = { i picks up i , i$\ominus$1 picks up i,
i puts down i , i$\ominus$1 puts down i }

BEHAVIOUR

PHIL$_i$ = μX. i sits down $\to$ i picks up i $\to$
$\to$ i picks up i$\oplus$1 $\to$ i puts down i
$\to$ i puts down i$\oplus$1 $\to$ i gets up
$\to$ X.

FORK$_i$ = μX.(i picks up i $\to$ i puts down i $\to$ X
| i$\ominus$1 picks up i $\to$ i$\ominus$1 puts down i $\to$ X)

PHILOS = PHIL$_0$ || PHIL$_1$ || ... || PHIL$_4$

FORKS = FORK$_0$ || FORK$_1$ || ... || FORK$_4$

COLLEGE = PHILOS || FORKS

DEADLOCK

Let sad = < 0 sits down, 1 sits down, ..., 4 sits down
0 picks up 0, 1 picks up 1, ... 4 picks up 4 >

COLLEGE / sad = STOP.

SOLUTION

introduce a FOOTMAN

α FOOTMAN = $\bigcup_{i=0}^{4}$ {i sits down, i gets up}

FOOTMAN = $\square$ ups $\overset{4}{\leq}$ downs

where ups = $\sum_{i=0}^{4}$ tr. i gets up

downs = $\sum_{i=0}^{4}$ tr. i sits down

NEW COLLEGE = COLLEGE || FOOTMAN

CHANGE OF SYMBOL

Let f map αP onto αQ

$P = f^{-1}(Q)$

means P performs x exactly on those occasions that Q performs f(x).

$\alpha f^{-1}(Q) = f^{-1}(\alpha Q)$

where $f^{-1}(B) = \{x \mid f(x) \in B\}$

To deal with inflation

f(in 10p) = in 2p   f(small) = large
f(in 5p) = in 1p   f(very small) = small
f(out 5p) = out 1p

NEWVMC = f^{-1}(VMC)

g(i sitsdown) = sitsdown   for $0 \leq i < 5$
g(i gets up) = gets up   "

LKY = g^{-1}(μX. sits down $\to$ gets up $\to$ X)

IV)

LAWS

$f^{-1}(STOP_A) = STOP_{f^{-1}(A)}$

$f^{-1}(x \to P) = (y : f^{-1}(\{x\}) \to f^{-1}(P))$

$f^{-1}(x : A \to P\,x) = (x : f^{-1}(A) \to (f^{-1} \circ P \circ f)\,x)$

$f^{-1}(P/s) = f^{-1}(P)/f^{*}(s)$

$f^{-1}(P/f^{*}(s)) = f^{-1}(P)/s$

$f^{-1}(P \parallel Q) = f^{-1}(P) \parallel f^{-1}(Q)$

$f^{-1}(\mu X.\, F) = \mu X.\, f^{-1}(F)$

$g^{-1}(f^{-1}(P)) = (f \circ g)^{-1}(P)$

NONDETERMINISM

Let $\alpha P = \alpha Q$

$P \sqcap Q$ behaves either like P or like Q
the choice is <u>not</u> made by another process.

$VMX = coin \to (choc \to VMX$
$\sqcap\ toffee \to VMX)$

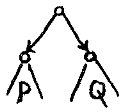
each transaction may be different

$VMY = (\mu X.\ coin \to choc \to X)$
$\sqcap\ (\mu X.\ coin \to toffee \to X)$

it's always the same.

Picture

$P \sqcap Q$



LAWS

$P \sqcap P = P$

$P \sqcap Q = Q \sqcap P$

$P \sqcap (Q \sqcap R) = (P \sqcap Q) \sqcap R$

An operator F is <u>distributive</u> if it distributes through $\sqcap$. i.e.,

$F(P \sqcap Q) = F(P) \sqcap F(Q)$

$G(P, Q \sqcap R) = G(P,Q) \sqcap G(P,R)$

$G(P \sqcap Q, R) = G(P,R) \sqcap G(Q,R)$

All operators so far are distributive, e.g.

$x \to (P \sqcap Q) = (x \to P) \sqcap (x \to Q)$

$(x : A \to (P\,x \sqcap Q\,x)) =$
$(x : A \to P\,x) \sqcap (x : A \to Q\,x)$

WARNING. $\mu X.$ is <u>not</u> distributive.
compare VMX with VMY.

IMPLEMENTATIONS

A nondeterministic process has more than one implementation, e.g.

$or1\,(P,Q) = P$

$or2\,(P,Q) = Q$

Or even allow environment to choose:

$or3\,(P,Q) = \lambda x.$ <u>if</u> $P(x) =$ "BLEEP <u>then</u> $Q(x)$
<u>else</u> <u>if</u> $Q(x) =$ "BLEEP <u>then</u> $P(x)$
<u>else</u> $or3(P(x), Q(x))$

or1 and or2 are efficient but asymmetric.
The <u>set</u> of implementations is symmetric

$\{or1\,(P,Q),\ or2\,(P,Q),\ or3\,(P,Q)\}$
$= \{or1\,(Q,P),\ or2\,(Q,P),\ or3\,(Q,P)\}$

15)

GENERAL CHOICE. 4.4

$P \,\square\, Q$ behaves like P or like Q.
The choice is made by the environment on the first step only.

$$(c \to P) \,\square\, (d \to Q) = (c \to P \mid d \to Q) \quad \text{if } c \neq d$$

$$(c \to P) \,\square\, (c \to Q) = c \to (P \sqcap Q)$$

LAWS

$\square$ is idempotent, symmetric, associative and distributive, and has unit STOP i.e.,

$P \,\square\, STOP = P$

$$(x : A \to P\,x) \,\square\, (y : B \to Q\,y) =$$
$$z : A \cup B \to \underline{\text{if}}\ z \in A - B\ \underline{\text{then}}\ P\,z$$
$$\underline{\text{else if}}\ z \in B - A\ \underline{\text{then}}\ Q\,z$$
$$\underline{\text{else}}\ (P\,z) \sqcap (Q\,z)$$

INTERLEAVING 4.5

$(P \,|||\, Q)$ does x whenever either P or Q does x. If both can do x, the choice is nondeterminate.

$$\alpha(P \,|||\, Q) = \alpha P = \alpha Q.$$

A Footman is constructed from four lackeys:

$$FOOTMAN = LKY \,|||\, LKY \,|||\, LKY \,|||\, LKY$$

LAWS

$|||$ is associative, commutative, distributive, with unit STOP and zero RUN i.e.,

$P \,|||\, RUN = RUN$.

Let $P = (x : A \to P\,x)$, $Q = (y : B \to Q\,y)$

Then
$$P \,|||\, Q = (x : A \to (P\,x \,|||\, Q))$$
$$\square\, (y : B \to (P \,|||\, Q\,y))$$

INTERLEAVING 4.5a

Let s be a sequence of sequences

$$\frown/s = s_0 \frown s_1 \frown s_2 \frown \ldots \frown s_{\#s-1}$$

zip(s,t) takes alternate elements from s & t:

$$zip(s,t) = \underline{\text{if}}\ s = \langle\rangle\ \underline{\text{then}}\ t$$
$$\underline{\text{else}}\ \langle s_0 \rangle \frown zip(t, s')$$

$$u \text{ interleaves } (v, w) \equiv$$
$$\exists\, s\, t.\ u = \frown/zip(s,t) \wedge v = \frown/s \wedge w = \frown/t$$

ASSERTION

$$P \,|||\, Q = \underline{E}\,(s, t : tr \text{ interleaves } (s,t)$$
$$\wedge\ P\,[s/tr]$$
$$\wedge\ Q\,[t/tr])$$

CONCEALMENT 4.6

$P \backslash C$ behaves like P except that events in set C occur autonomously and invisibly.

$$\alpha(P \backslash C) = \alpha P - C.$$

Soundproofing

$$VMS = NOISYVM \backslash \{clink, toffee\}$$

$$((\mu X.\, a \to c \to X) \,||\, (\mu X.\, c \to b \to X)) \backslash \{c\}$$

$$= (a \to c \to (\mu X.(a \to b \to c \to X$$
$$\mid b \to a \to c \to X))) \backslash \{c\}$$

$$= (a \to \mu X.(a \to b \to X$$
$$\mid b \to a \to X))$$

16)

LAWS 4.7

$P \backslash \{\} = P$

$(P \backslash B) \backslash C = P \backslash (B \cup C)$

$(P \sqcap Q) \backslash C = (P \backslash C) \sqcap (Q \backslash C)$

$STOP_A \backslash C = STOP_{A-C}$

$(x \to P) \backslash C = x \to (P \backslash C)$ if $x \notin C$

$= P \backslash C$ if $x \in C$

If $A \cap C = \{\}$

$(x : A \to P\,x) \backslash C = (x : A \to (P\,x \backslash C))$

If $A \neq \{\} \wedge A \subseteq C$ A finite.

$(x : A \to P\,x) \backslash C = \bigsqcap_{x \in A} P\,x \backslash C$

If $A \cap C \neq \{\}$ A finite.

$(x : A \to P\,x) \backslash C = Q \sqcap (Q \,\square\, x : A - C \to (P\,x \backslash C))$

where $Q = \bigsqcap_{x \in A \cap C} (P\,x \backslash C)$

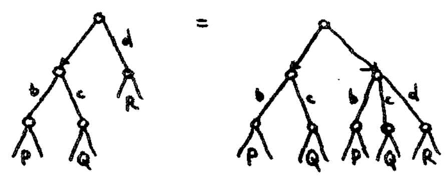
If $X = \{x, y, \ldots, z\}$

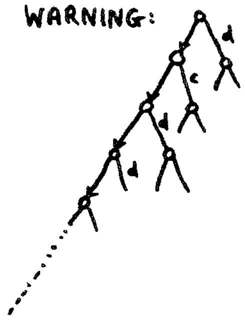
$\bigsqcap_{w \in X} P\,w = P\,x \sqcap P\,y \sqcap \ldots \sqcap P\,z$

PICTURES 4.8

Just remove hidden symbols from arcs.

Single unlabelled arcs may be suppressed:



WARNING: 

is **NOT ALLOWED**

$(\mu X.(c \to X \mid d \to STOP)) \backslash \{c\}$

IMPLEMENTATION 4.9

hide (P,"c) implements $P \backslash \{c\}$

hide(P, x) = <u>if</u> P(x) = "BLEEP <u>then</u>
  λy. <u>if</u> P(y) = "BLEEP <u>then</u> "BLEEP
    <u>else</u> hide (P(y), x)
  <u>else</u> *hide (P(x), x)

*NOTE danger of infinite recursion:

such hiding is not allowed.

ALPHABET EXTENSION 4.10

Let $\alpha P \cap B = \{\}$

$P \underline{\times} B$ behaves like P, except that

$\alpha(P \underline{\times} B) = \alpha P \cup B$

It never performs any action from B

$(P \underline{\times} B) \backslash B = P$

Alphabet extension usually implicit:

If $\alpha P \cap \alpha Q = \{\}$

$P \,|||\, Q = P \,||\, Q$

(means: $(P \underline{\times} \alpha Q) \,|||\, (Q \underline{\times} (\alpha P)) = P \,||\, Q$)

17)

PROCESS NAMING 4.11

A compound event

$m.e$

records performance of e by a process named m

$m:P$ does $m.x$ whenever P would do x.

$$m:P = strip_m^{-1}(P)$$

where $strip_m(m.x) = x$

A pair of vending machines.

$$left:VMS \parallel right:VMS$$

NOTE: $VMS \parallel VMS = VMS$

MASTER AND SLAVE 4.12

Let $\alpha(m:P) \subseteq \alpha U$

$[m:P \parallel U]$ is a process in which U controls $m:P$ by actions in $\alpha(m:P)$, and these are concealed from environment

$$[m:P \parallel U] = ((m:P) \parallel U) \setminus \alpha(m:P)$$

A PILOT flies a rocket named m

$$[m:ROCKET \parallel PILOT]$$

To land the rocket she does:

$$\mu X.(m.down \rightarrow X \\ \quad | m.around \rightarrow \ldots)$$

A pilot with two rockets:

$$[n:ROCKET \parallel [m:ROCKET \parallel PILOT]]$$

RECURSION 4.13

Another definition of ROCKET

$$Z = (around \rightarrow Z \\ \quad | up \rightarrow [m:Z \parallel LOOP])$$

$$LOOP = (up \rightarrow m.up \rightarrow LOOP \\ \quad | down \rightarrow (m.down \rightarrow LOOP \\ \qquad | m.around \rightarrow Z \ \underline{x}\ \alpha(m:Z)))$$

$$\alpha Z = \{up, down, around\}$$

$$\alpha LOOP = \alpha Z \cup \alpha(m:Z) \\ = \{up, down, around, \\ m.up, m.down, m.around\}$$

STACK 4.14

$$\alpha STACK = \{in1p, out1p, in2p, out2p, isempty\}$$

$$STACK = EMPTY$$

$$EMPTY = (isempty \rightarrow EMPTY \\ \quad | in1p \rightarrow [m:EMPTY \parallel LOOP1] \\ \quad | in2p \rightarrow [m:EMPTY \parallel LOOP2])$$

$$LOOP1m = (in1p \rightarrow m.in1p \rightarrow LOOP1 \\ \quad | in2p \rightarrow m.in1p \rightarrow LOOP2 \\ \quad | out1p \rightarrow CONTRACT)$$

$LOOP2(m) =$

$$CONTRACTm = (m.out1p \rightarrow LOOP1 \\ \quad | m.out2p \rightarrow LOOP2 \\ \quad | m.isempty \rightarrow EMPTY \\ \qquad \underline{x}\ \alpha(m:EMPTY))$$

$LOOP1n =$

$\vdots$

WARNING

RECURSION UNDER HIDING

is not guarded.

Let $F(X) = a \rightarrow (X \setminus \{a\}) \underline{\times} \{a\}$
$\alpha X = \{a, b, c\}$

$F^2(X) = a \rightarrow ((a \rightarrow (X \setminus \{a\}) \underline{\times} \{a\}) \setminus \{a\}) \underline{\times} \{a\}$

$= a \rightarrow (X \setminus \{a\} \underbrace{\underline{\times} \{a\} \setminus \{a\}}_{} \underline{\times} \{a\})$

$= a \rightarrow X \setminus \{a\} \underline{\times} \{a\}$

$= F(X)$

$\therefore$ For <u>all</u> X, $F(X)$ is a solution of

$Y = F(Y)$

i.e., there are many solutions.

$F(\cancel{X}) / \langle a \rangle = X \setminus \{a\} \underline{\times} \{a\}$
$\cancel{X} = \cancel{\mu} X b \rightarrow X$
$\cancel{Y} = \mu X c \rightarrow X$

Let τ be a new event

$\mu X . F(X) = (\mu X . \tau \rightarrow F(X)) \setminus \{\tau\}$

The recursion fails just when the hiding fails.

REFUSALS.

Let $x \neq y$, $\alpha P = \alpha Q = \{x, y\}$,
$P = (x \rightarrow P)$, $Q = (y \rightarrow Q)$

$(P \,\square\, Q) \| P = (x \rightarrow P \mid y \rightarrow Q) \| (x \rightarrow P)$
$= (x \rightarrow (P \| P)) = P$

$(P \sqcap Q) \| P = (P \| P) \sqcap (Q \| P)$
$= P \sqcap (y \rightarrow Q \| x \rightarrow P)$
$= P \sqcap STOP$

$\therefore P \,\square\, Q \neq P \sqcap Q$

because $P \sqcap Q$ can <u>refuse</u> $\{x\}$ on its first step

$P \,\square\, Q$ can't.

$B \in$ refusals (P) means

$P \| (x : B \rightarrow Q x)$

can deadlock immediately.

refusals $(x : A \rightarrow P x) = \{ B \mid \overset{B \subseteq \alpha P}{A \cap B = \{\}} \}$

refusals $(P \sqcap Q)$ = refusals $(P) \,\underline{\cup}\,$ refusals (Q)

refusals $(P \,\square\, Q)$ = refusals $(P) \,\underline{\cap}\,$ refusals (Q)

refusals $(P \| Q) = \{ X \cup Y \mid X \in$ refusals (P)
$\wedge Y \in$ refusals $(Q) \}$

refusals $(P \,|||\, Q)$ = refusals $(P \,\square\, Q)$

refusals $(F^{-1}(P)) = \{ X \mid F(X) \in$ refusals $(P) \}$

refusals $(P \setminus C) = \bigcup_{s \in C^*} \{ X \mid X \in$ refusals (P/s)
$s \in$ traces$(P) \wedge X \cap C = \{\} \}$
provided this is finite union

19)

LAWS

.3

$X \in refusals(P) \Rightarrow X \subseteq \alpha P$

$\{\} \in refusals(P)$

$(X \cup Y) \in refusals(P) \Rightarrow X \in refusals(P$

$X \in refusals(P) \wedge x \in \alpha P \Rightarrow$

$\quad X \cup \{x\} \in refusals(P)$

$\quad \vee \langle x \rangle \in traces(P)$

OBSERVATIONS.

.4

(s, B) is an <u>observation</u> of P if

$\quad B \in refusals(P/s) \wedge s \in traces(P)$

$(s, B) \leq (t, C)$ means.

$\quad s = t \wedge B \subseteq C$

$\quad \vee s < t \wedge B = \{\}$

ASSERTIONS

.5

Let's identify a process with a predicate $P(tr, ref)$ which describes all its possible observations. (tr, ref)

$(x : A \to P_x) = (tr = \langle\rangle \wedge \overline{A} \subseteq ref \subseteq \alpha P)$

$\qquad \vee (tr_0 \in A \wedge P(tr_0)[tr'/tr])$

$STOP_A = tr = \langle\rangle \hat{\&} ref \subseteq A$

$(c \to P) = (tr = \langle\rangle \wedge c \tilde{\in} ref \subseteq \alpha P)$

$tr \neq \langle\rangle \wedge \quad \cancel{\vee}(tr_0 = c \wedge P[tr'/tr])$

$P/s = P[s \frown tr/tr]$

$(P \| Q) = \exists X, Y : ref = X \cup Y :$

$\qquad P[X/ref, tr \upharpoonright \alpha P/tr]$

$\qquad \wedge Q[Y/ref, tr \upharpoonright \alpha Q/tr]$

$f^{-1}(P) = P[f(ref)/ref, f^*(tr)/tr]$

.6

$P \sqcap Q = P \vee Q$

$P \,\square\, Q = \underline{if}\ tr = \langle\rangle\ \underline{then}\ (P \wedge Q)\ \underline{else}\ (P \vee Q)$

$P \,|||\, Q = \underline{E}\, s, t : tr \text{ interleaves } (s, t) :$

$\qquad P[s/tr] \wedge Q[t/tr]$

$P \backslash C = \underline{E}\, s : s \upharpoonright \overline{C} = tr :$

$\qquad P[s/tr, (ref \cup C)/ref]$

provided that

$\underline{A}\, t :: \{tr \mid tr \upharpoonright \overline{C} = t \wedge P[\{\}/ref]\}$

is FINITE

$P \ulcorner tr = \langle\rangle \urcorner Q$

Bill Roscoe
Steve Brookes.

20) **EXAMPLES** .7

VM with buffering of coins perhaps.

Let $BAL = tr.coin - tr.choc$

$$\{coin \not\subseteq ref$$
$$VM = \Box((BAL = 0 \land \{coin\} \subseteq \overline{ref}) \lor (BAL > 0 \land \{choc\} \subseteq \overline{ref}))$$

$$= VM_0 = coin \to VM_1$$

$$VM_{n+1} = choc \to VM_n$$
$$\sqcap (coin \to VM_{n+2} \ \Box \ choc \to VM_n)$$

$$VMS = VM \land \Box BAL \leq 1$$

$$VMS \Rightarrow VM$$

i.e VMS is a valid implementation of VM

8.

CD is a cash dispenser

$$\alpha CD = \{withdraw, deposit\}$$

$$BAL = tr.deposit - tr.withdraw$$

$$CD = \Box(deposit \in \overline{ref} \land (BAL \leq 0 \lor withdraw \in \overline{ref}))$$

EXERCISE

A. Write process CD

B. Write a process CDB such that

$$CDB \Rightarrow CD$$

COMMUNICATING PROCESSES. 5.1

A compound event

$c.v$

denotes communication of a message with value v on channel c

$$\alpha c(P) = \{v \mid c.v \in \alpha P\}$$

gives the set of messages which P can communicate on channel c

If E is an expression with value v

$$(c!E \to P) = (c.v \to P)$$

outputs v on channel c

$$(c?x \to P_x) = (y : strip_c^{-1}(\alpha c(P)) \to P(strip_c(y)))$$

inputs a message x on channel c

Loosely: $c \in \alpha P$ means $\alpha c(P) \neq \{\}$

EXAMPLES 5.2

$$COPY = \mu X.\ left?y \to right!y \to X$$

$$DOUBLE = \mu X.\ left?y \to right!(y+y) \to X$$

$UNPACK = P_{\langle\rangle}$ $\qquad \alpha left(P) = (\alpha right(P))^*$

$$P_{\langle\rangle} = left?s \to P_s$$
$$P_{\langle x\rangle^\frown s} = right!x \to P_s$$

$PACK = P_{\langle\rangle}$ $\qquad (\alpha left(P))^{125} = \alpha right(P)$

$P_l = left?x \to P_{l^\frown\langle x\rangle}$ $\qquad$ if $\#l < 125$

$P_l = right!l \to P_{\langle\rangle}$ $\qquad$ if $\#l = 125$

$SQUASH = \mu X.\ left?x \to$
  <u>if</u> $x = "*"$ <u>then</u> $(right!x \to X)$
  <u>else</u> $left?y \to$
    <u>if</u> $y = "*"$ <u>then</u> $(right!"\uparrow" \to X)$
    <u>else</u> $(right!"*" \to right!y \to X)$

21)

5.3

$VAR = left?x \rightarrow VAR_x$

$VAR_x = (left?y \rightarrow VAR_y$
$\quad | right!x \rightarrow VAR_x)$

$MERGE =$
$(left1?x \rightarrow right!x \rightarrow MERGE$
$| left2?x \rightarrow right!x \rightarrow MERGE)$

$BUFFER = P_{\langle\rangle}$
$P_{\langle\rangle} = left?x \rightarrow P_{\langle x\rangle}$
$P_{\langle x\rangle s} = (left?y \rightarrow P_{\langle x\rangle s\langle y\rangle}$
$\quad | right!x \rightarrow P_s)$

$STACK = P_{\langle\rangle}$
$P_{\langle\rangle} = (isempty \rightarrow P_{\langle\rangle} | left?x \rightarrow P_x)$
$P_{\langle x\rangle s} =$

TRACES

5.4

$tr.c = strip_c^*(tr \upharpoonright \alpha c)$

IF $tr = \langle left.3, right.3, left.37\rangle$
then $tr.left = \langle 3, 37\rangle$
$\quad tr.right = \langle 3\rangle$

Define $t \geq^n s \;=\; s \leq t \;\&\; \#t \leq \#s + n$

$COPY = \square\, tr.left \geq^1 tr.right$

$DOUBLE = \square\, double^*(tr.left) \geq^1 tr.right.$

$UNPACK = \square\, {}^\frown/\, tr.left \geq tr.right$
$\quad \& \; {}^\frown/(trunc(tr.left)) \leq tr.right.$
where $trunc(s\langle x\rangle) = s.$

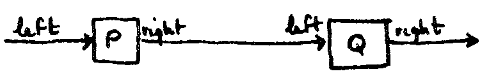
$PACK = \square\, tr.left \geq^{125} {}^\frown/tr.right.$
$\quad \& \; \forall s : s \in range\ tr.right : \#s = 125$

PIPES

5.5.

A pipe is a process with only two channels left and right

$(P \gg Q)$ is formed by joining right channel of P to left channel of Q



Communications on connecting channel hidden

Provided that $\alpha right(P) = \alpha left(Q)$

$(P \gg Q) = (P[mid/right] \,\|\, Q[mid/left]) \setminus \alpha mid$

where $P[d/c] = f^{-1}(P)$
$\quad f(d.v) = c.v$
$\quad f(b.v) = b.v \quad$ for $b \neq d$

$\alpha(P \gg Q) = \alpha left(P) \cup \alpha right(Q)$

$(P \gg Q) \gg R = P \gg (Q \gg R)$

EXAMPLES

5.6.

$QUADRUPLE = DOUBLE \gg DOUBLE$

$LISTING = UNPACK \gg PACK$

$CONWAY = UNPACK \gg SQUASH \gg PACK$

$BUFFERED = UNPACK \gg BUFFER \gg PACK$

$BUFFER3 = COPY \gg COPY \gg COPY$

ALGEBRA 5.7

$\gg$ is associative

$(right!v \rightarrow P) \gg (left?x \rightarrow Q\ y) = P \gg (Q\ v)$

$(right!v \rightarrow P) \gg (right!w \rightarrow Q)$

$= (right!w \rightarrow ((right!v \rightarrow P) \gg Q))$

$(left?x \rightarrow P\ x) \gg (left?y \rightarrow Q\ y)$

$= (left?x \rightarrow (P\ x \gg (left?y \rightarrow Q\ y)))$

$(left?x \rightarrow P\ x) \underline{\gg} (right!w \rightarrow Q)$

$= (left?x \rightarrow (P\ x \underline{\gg} (right!w \rightarrow Q))$

$\mid right!w \rightarrow ((left?x \rightarrow P\ x) \underline{\gg} Q))$

As above, with $\underline{\gg}$ replaced by $\underline{\gg R \gg}$

RELATIONS 5.8

$l(S;T)r = \exists u :: lSu \;\&\; uTr$

$;$ is associative, with unit $=$.

$(\geq);(\geq) = \geq$

$\geq^m ; \geq^n = \geq^{m+n}$

$f^*; \geq^m = \geq^m ; f^*$

where $l(f^*)r = r = f^*(l)$

$f^*; g^* = (g \circ f)^*$

S is $\underline{\text{grounded}}$ if for all l, r:

$lSr \Rightarrow l = r = \langle\rangle$

$\vee\ trunc(l)\, S\, r$

$\vee\ l\, S\, trunc(r)$

If S and T are grounded so is $S;T$

5.9

Let R and S be grounded:

If $P = \square\ tr.left\ R\ tr.right$

$Q = \square\ tr.left\ S\ tr.right$

then $P \gg Q = \square\ tr.left\ (R;S)\ tr.right$

provided that for all s, t

$\{u \mid s R u \;\&\; u S t\}$ is finite.

$BUFFER \gg BUFFER$

$= \square\ tr.left\ (\geq;\geq)\ tr.right$

$= BUFFER$

$(COPY \gg COPY) = \square\ tr.left\ \geq^2\ tr.right$

$(DOUBLE \gg DOUBLE) =$

$\square\ tr.left\ (double^*; \geq^1; double^*; \geq^1)\ tr.right$

$= \qquad (double^*; double^*; \geq^1; \geq^1)$

$= \qquad (quadruple^*; \geq^2)$

SEQUENTIAL PROCESSES. 6.1

$P;Q$ behaves as P until P terminates successfully; then behaves as Q

$SKIP_A$ does nothing but terminate successfully.

$\alpha SKIP_A = A$

$SKIP_A \neq STOP_A$

If P is guarded.

$*P = \mu X.\ P;X$

23)

EXAMPLES

$$VM1 = (coin \rightarrow choc \rightarrow SKIP)$$

$$VM3 = VM1 ; VM1 ; VM1$$

$$VMS = * VM1$$

$$AnBCn = \mu X.\,(b \rightarrow SKIP$$
$$\mid a \rightarrow (X ; (c \rightarrow SKIP)))$$

accepts n a's followed by a b

followed by n c's, for any n

$\exists n :: tr \leq \langle a \rangle^n \frown \langle b \rangle \frown \langle c \rangle^n \frown \checkmark$

where ✓ denotes successful termination

$$POS = (down \rightarrow SKIP \mid up \rightarrow (POS ; POS))$$

POS terminates when number of down's first exceeds the number of up's

$$C_0 = (around \rightarrow C_0 \mid up \rightarrow C_1)$$

$$C_{n+1} = POS ; C_n \qquad \text{for all } n \geq 0$$

$$C_0 = CT_0 \qquad !$$

LAWS

; is associative with unit SKIP
distributive

$$(x : A \rightarrow P_x) ; Q = (x : A \rightarrow (P_x ; Q))$$

Theorem $C_0 = CT_0$

Proof: C_0 satisfies equations of CT_0

Lemma (1) $C_0 = (around \rightarrow C_0 \mid up \rightarrow C_1)$ by def. C_0

Lemma (2) $C_{n+1} = (up \rightarrow C_{n+2} \mid down \rightarrow C_n)$

Proof. $LHS = POS ; C_n$ def C_{n+1}

$= (down \rightarrow SKIP \mid up \rightarrow POS ; POS) ; C_n$ def POS

$= (down \rightarrow SKIP ; C_n \mid up \rightarrow POS ; POS ; C_n)$ dist

$= (down \rightarrow C_n \mid up \rightarrow POS ; (POS ; C_n))$ assoc

$= (down \rightarrow C_n \mid up \rightarrow POS ; C_{n+1})$ def C_{n+1}

$= (down \rightarrow C_n \mid up \rightarrow C_{n+2})$ def C_{n+2}

$= RHS$