University of Wollongong
Research Online

Department of Computing Science Working Paper Series

Faculty of Engineering and Information Sciences

1983

CSP at WSS

C. A. R. Hoare
University of Wollongong

Recommended Citation
Hoare, C. A. R., CSP at WSS, Department of Computing Science, University of Wollongong, Working Paper 83-3, 1983, 25p.
http://ro.uow.edu.au/compsciwp/71

Research Online is the open access institutional repository for the University of Wollongong. For further information contact the UOW Library: research-pubs@uow.edu.au

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

0)

0

COMMUNICATING SEQUENTIAL PROCESSES

C.A.R. HOARE

SPONAR'S CHALET

JAN - FEB 1983

1

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)

2

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

3

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.

1)

$CLOCK = (tick \rightarrow CLOCK)$

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

$= \ldots$

A simple vending machine

$VMS = coin \rightarrow choc \rightarrow VMS$

A change giving machine

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

$CHS = in5p \rightarrow out2p \rightarrow out1p \rightarrow out2p \rightarrow CHS$

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

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

A more complicated vending machine

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

$VMC = (in2p \rightarrow (large \rightarrow VMC$
$\quad\quad\quad\quad \mid small \rightarrow out1p \rightarrow VMC)$
$\quad\quad \mid in1p \rightarrow (small \rightarrow VMC$
$\quad\quad\quad\quad \mid in1p \rightarrow (large \rightarrow VMC$
$\quad\quad\quad\quad\quad\quad \mid in1p \rightarrow STOP)$
$\quad\quad\quad\quad)$
$\quad\quad)$

WARNING: DO NOT INSERT THREE COINS.

GENERAL CHOICE

Let A be a set of events

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

If $A \subseteq B$

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

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

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

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

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

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

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

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

CT_n is the behaviour at height n.

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

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

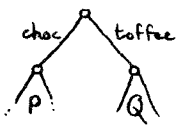
$$ROCKET = CT_0$$

PICTURES

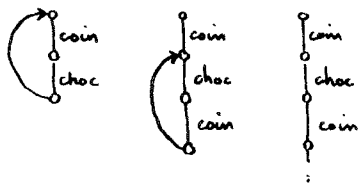
STOP

coin → STOP

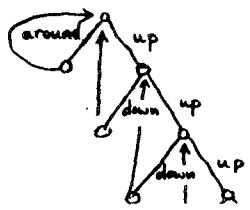
(choc → P | toffee → Q)



VMS



CT_0



3)

LAW

12

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

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

Corollaries

$\text{STOP} \neq (d \to P)$

$(c \to P) \neq (d \to Q) \qquad \text{if } c \neq d$

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

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

LAW

13

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')

14

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

$\lambda x.$ <u>if</u> ismember (x, A) <u>then</u> $P(x)$ <u>else</u> "BLEEP

$\text{STOP} \approx \lambda x$ "BLEEP

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

prefix $(c, P) = \lambda x.$ <u>if</u> $x = c$ <u>then</u> P <u>else</u> "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

<u>if</u> $n = 0$ <u>then</u> choice2 ("around, CT(0),
"up , CT(1))

<u>else</u> choice2 ("down , CT(n-1),
"up , CT(n+1)) (0)

TRACES

2.1

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

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

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

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

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

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

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

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

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

$\langle\rangle\restriction A = \langle\rangle$ strict distributive

$(s^\frown t)\restriction A = (s\restriction A)^\frown(t\restriction A)$

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

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

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

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

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 $As :: (s\restriction A)\restriction B = s\restriction(A\cap B)$

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

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

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

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

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

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

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

(3) $((s^\frown t)\restriction A)\restriction B = ((s\restriction A)^\frown(t\restriction A))\restriction B$

$= ((s\restriction A)\restriction B)^\frown((t\restriction A)\restriction B)$

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

$= (s^\frown t)\restriction(A\cap B)$

CHANGE OF SYMBOL 2.5

Let f be a function from symbols to symbols. – <u>total</u>

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

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

LAWS

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

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

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

$^*f^*(s\restriction A) = f^*(s)\restriction f(A)$

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

2.6

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

2.7

$\bar{s}$ 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

2.8

$\langle\rangle$	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> $f(\langle s_0\rangle) \frown f(s')$

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

EXAMPLE

2.9

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

$\langle s_0\rangle \upharpoonright A$ = <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

2.10

$s \le t$ 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., $\overset{n}{\le} ; \overset{m}{\le} \;=\; \overset{n+m}{\le}$

2.11

$s[0] = s_0$ 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

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

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

$\underline{sub}$ is a partial order

2.12

Let f be SM, with

$\overline{range(f)}$ infinite

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

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

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

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

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

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

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

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

4 A drinks dispenser DD

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

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 choccs

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

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

$VMI \triangleq \mu X.\; choc \to coin \to X$

Prove that

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

7) EXERCISES

1. $\alpha CTR = \{up, right\}$

CTRA

CTRB

CTRC

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

3. $\alpha CTRD = \{up, down, left, right\}$
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

A.4.

A <u>chain</u> of predicates satisfies

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

A predicate transformer F is <u>continuous</u> if

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

for <u>all</u> chains

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

Lemma. If F is continuous, $F^n(true)$ is a chain.

A.5

If F is continuous

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

Then $F(X) = X$

Proof. LHS =

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

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

$=$ RHS.

SPECIFICATIONS.

A.6.

A process is <u>specified</u> by giving all desired properties of its traces.

"It mustn't accept more than 70 coins"

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

"It mustn't accept three consecutive pennies"

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

"An improved VMC."

VMCOK = VMC

$\wedge$ VMCBAR3 $\wedge$ VMC70

ALWAYS.

A.7

Let P be a predicate intended to be true of <u>all</u> 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. A.8

"It mustn't lose money"

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

"It mustn't cheat the client"

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

"It must please owner and client"

NOLOSS $\wedge$ FAIR

This specifies VMS

A specification of CT_n:

$\Box$ (tr.down $\le$ tr.up + n

$\wedge$ last(tr) = around $\Rightarrow$ tr.down = tr.up + n)

LAWS A.9

$\Box P \Rightarrow P$

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

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

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

tr = <> v

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

if f is <u>healthy</u>

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

A.10

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

Proof. coin $\rightarrow$ choc $\rightarrow$ VMS

= (tr $\le$ <coin>

v(<coin, choc> $\le$ tr

$\wedge \Box\ 0 \le tr''.coin - tr''.choc < 2$))

= VMS

AFTER A.11

If s is a trace of P

P/s describes the behaviour of

P after it has engaged in s

VMC/<in2p, small> = out1p $\rightarrow$ VMC

VMC/<in1ps^3 = STOP

LAWS

P/<> = P

P/(s^t) = (P/s)/t

$(x : A \rightarrow P_x)/\langle y \rangle = P_y$ if $y \in A$

ASSERTION

Provided P[s/tr]

P/s = P[s^tr/tr]

$(\Box P)/s \overset{?}{=} \Box(P/s)$

10) EXAMPLE: BACKTRACKING

A.12

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 $\alpha P \cup \{backspace\}$

which specifies a mechanism which behaves as P except that backspace cancels the effect of the most recent uncancelled symbol in αP

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

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

$$clean(s^\frown\langle x\rangle) = (clean\ s)^\frown\langle x\rangle \quad x \in \alpha P$$

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

EXERCISES

Specify the following predicate transformers:

(1) $restart \in \alpha P$

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

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

<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>

IMPLEMENTATION

A.13

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))

INTERSECTION

3.1

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)

11)

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

α NOISYVM = {coin, clink, choc, toffee}

NOISYVM = (coin → clink → choc → NOISYVM)

α CUST = {coin, startcurse, endcurse, choc, toffee}

CUST = coin → (toffee → CUST
| startcurse → endcurse → choc → CUST)

3.3.

NOISYVM || CUST =

$\mu X.$ coin → (clink → startcurse → endcurse → choc → X
| startcurse → clink → endcurse → choc → X
| startcurse → endcurse → clink → choc → X)

α CTR = {up, down, left, right}

Let αP = {up, down}

P = (up → down → P)

αQ = {left, right}

Q = (right → left → Q
| left → right → 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 = $x{:}A \to Px$

Q = $y{:}B \to Qy$

Then P || Q =

$z{:}(A \ulcorner \alpha P \cap \alpha Q \urcorner B) \to$

(<u>if</u> $z \in A$ <u>then</u> Pz <u>else</u> P

|| <u>if</u> $z \in B$ <u>then</u> Qz <u>else</u> Q)

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

ASSERTION

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

EXAMPLE

3.5

$\alpha P = \{a,c\}$ P = a → c → P

$\alpha Q = \{b,c\}$ Q = c → b → Q

1. P || Q = a → (c → P || c → b → Q)
   = a → c → (P || b → Q)

2. P || b → Q = (a → (c → P || b → Q)
   | b → (P || Q))
   = (a → b → c → (P || b → Q)
   | b → a → c → (P || b → Q))
   = $\mu X.$(a → b → c → X
   | b → a → c → X)

3. = (P || Q) / <a,c>

12)

3.6

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

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

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

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

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

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

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

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

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

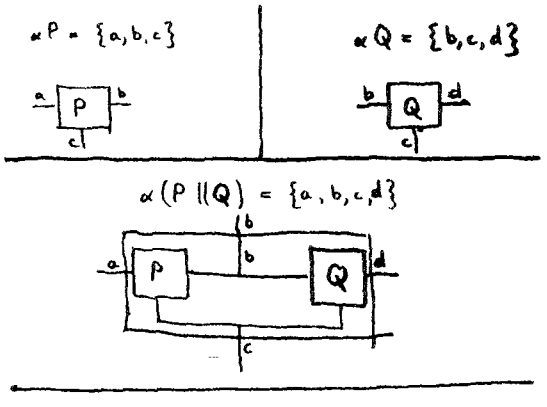
$= \Box\; tr.b \overset{1}{\leq} tr.c + 1 \overset{1}{\leq} tr.a + 1$

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

$(P \sim Q)$ 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.

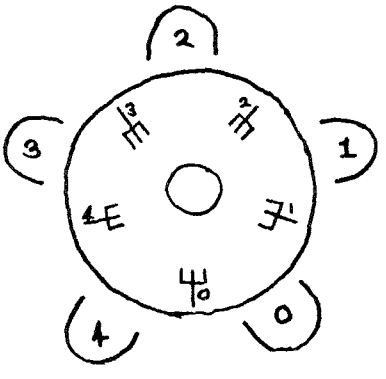
PICTURES.

3.7



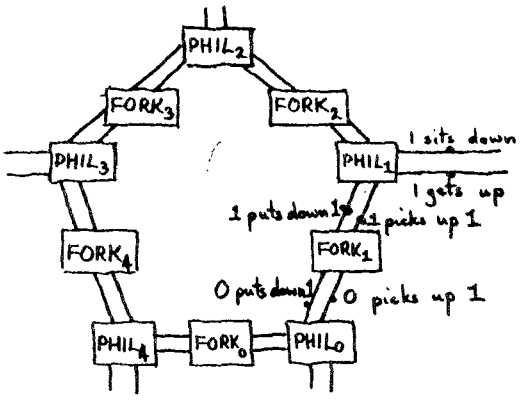
THE DINING PHILOSOPHERS.

3.8



13) ALPHABETS 3.9

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



$\alpha PHIL_i = \{ i \text{ sits down}, i \text{ gets up},$
$i \text{ picks up } i, i \text{ picks up } i\oplus 1,$
$i \text{ puts down } i, i \text{ puts down } i\oplus 1 \}$

$\alpha FORK_i = \{ i \text{ picks up } i, \; i\ominus 1 \text{ picks up } i,$
$i \text{ puts down } i, \; i\ominus 1 \text{ puts down } i \}$

BEHAVIOUR 3.10

$PHIL_i = \mu X.\ i \text{ sits down} \rightarrow i \text{ picks up } i \rightarrow$
$\rightarrow i \text{ picks up } i\oplus 1 \rightarrow i \text{ puts down } i$
$\rightarrow i \text{ puts down } i\oplus 1 \rightarrow i \text{ gets up}$
$\rightarrow X.$

$FORK_i = \mu X.(i \text{ picks up } i \rightarrow i \text{ puts down } i \rightarrow X$
$| i\ominus 1 \text{ picks up } i \rightarrow i\ominus 1 \text{ puts down } i \rightarrow X)$

$PHILOS = PHIL_0 \| PHIL_1 \| \ldots \| PHIL_4$

$FORKS = FORK_0 \| FORK_1 \| \ldots \| FORK_4$

$COLLEGE = PHILOS \| FORKS$

DEADLOCK 3.11

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

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

$FOOTMAN = \Box \; ups \leq downs$

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

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

$NEW\ COLLEGE = COLLEGE \| FOOTMAN$

CHANGE OF SYMBOL 3.12

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(in10p) = in2p f(small) = large
f(in5p) = in1p f(very small) = small
f(out5p) = out1p

$NEWVMC = f^{-1}(VMC)$

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

$LKY = g^{-1}(\mu X.\ \text{sits down} \rightarrow \text{gets up} \rightarrow X)$

IV) LAWS 3.13

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

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

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

$\times\ F^{-1}(P/s) = F^{-1}(P)/F^{*}(s)$

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

$F^{-1}(P \| Q) = F^{-1}(P) \| F^{-1}(Q)$

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

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

NONDETERMINISM 4.1

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.

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

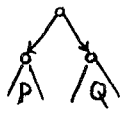
each transaction may be different

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

it's always the same.

Picture

$P \sqcap Q$



LAWS 4.2

$P \sqcap P = P$

$P \sqcap Q = Q \sqcap P$

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

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.

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

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

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

IMPLEMENTATIONS 4.3

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> $Q(x)$
$\qquad$ <u>else</u> <u>if</u> $Q(x) =$ "BLEEP <u>then</u> $P(x)$
$\qquad\qquad$ <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

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

$(c \to P) \,[\!]\, (d \to Q) = (c \to P \mid d \to Q)$ if $c \neq d$

$(c \to P) \,[\!]\, (c \to Q) = c \to (P \sqcap Q)$

LAWS

$[\!]$ is idempotent, symmetric, associative and distributive, and has unit STOP i.e.,
$P \,[\!]\, STOP = P$

$(x : A \to P\,x) \,[\!]\, (y : B \to Q\,y) =$
$z : A \cup B \to$ <u>if</u> $z \in A - B$ <u>then</u> $P\,z$
<u>else if</u> $z \in B - A$ <u>then</u> $Q\,z$
<u>else</u> $(P\,z) \sqcap (Q\,z)$

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 lackey

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 $P \,|||\, Q = (x : A \to (P\,x \,|||\, Q))$
$[\!]\, (y : B \to (P \,|||\, Q\,y))$

INTERLEAVING 4.~~6~~ 5

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) = <u>if</u> $s = \langle\rangle$ <u>then</u> t
<u>else</u> $\langle s_0 \rangle{}^\frown$ zip (t, s')

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

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

$(x{:}A \to Px)\backslash C = (x{:}A \to (Px\backslash C))$

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

$(x{:}A \to Px)\backslash C = \prod_{x \in A} Px\backslash C$

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

$(x{:}A \to Px)\backslash C = Q \sqcap (Q \,[\!]\, x{:}A-C \to (Px\backslash C))$

where $Q = \prod_{x \in A \cap C} (Px\backslash C))$

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

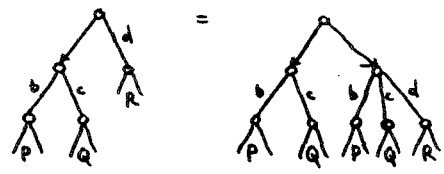
$$\prod_{w \in X} Pw = Px \sqcap Py \sqcap \ldots \sqcap Pz$$

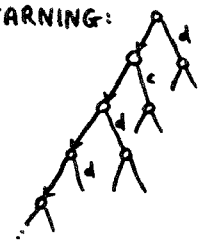
PICTURES

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

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

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

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

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

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

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

STACK

4.14

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

$STACK = EMPTY$

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

$LOOP1_m = (in1p \rightarrow m.in1p \rightarrow LOOP1$
$\quad | in2p \rightarrow m.in1p \rightarrow LOOP2$
$\quad | out1p \rightarrow CONTRACT)$

$LOOP2(m) =$

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

$LOOP1_n =$

8) WARNING

RECURSION UNDER HIDING is not guarded.

Let $F(X) = a \to (X \setminus \{a\}) \underline{\times} \{a\}$

$\alpha X = \{a, b, c\}$

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

$$= a \to (X \setminus \{a\} \underline{\times} \{a\} \setminus \{a\} \underline{\times} \{a\})$$

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

$$= F(X)$$

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

$$Y = F(Y)$$

i.e., there are many solutions.

$F(X)/\langle a \rangle = X \setminus \{a\} \underline{\times} \{a\}$

$\mathbf{X} = \mu X b \to X$

$\mathbf{Y} = \mu X c \to X$

REFUSALS.

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

$P = (x \to P)$, $Q = (y \to Q)$

$$(P \,[\!]\, Q) \| P = (x \to P \mid y \to Q) \| (x \to P)$$

$$= (x \to (P \| P)) = P$$

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

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

$$= P \sqcap STOP$$

$\therefore$ $P \,[\!]\, Q \neq P \sqcap Q$

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

$P \,[\!]\, Q$ can't.

Let τ be a new event

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

The recursion fails just when the hiding fails.

.1

$B \in$ refusals (P) means

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

can deadlock immediately.

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

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

refusals $(P \,[\!]\, Q)$ = 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 $(P \,[\!]\, Q)$

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

refusals $(P \setminus C) = \bigcup_{\substack{s \in C^* \\ s \in traces(P) \wedge X \cap C = \{\}}} \{X \mid X \in$ refusals $(P/s)\}$

provided this is finite union

.2

19)

LAWS

.3

$X \in \text{refusals}(P) \Rightarrow X \subseteq \alpha P$

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

$(X \cup Y) \in \text{refusals}(P) \Rightarrow X \in \text{refusals}(P$

$X \in \text{refusals}(P) \wedge x \in \alpha P \Rightarrow$
$\quad X \cup \{x\} \in \text{refusals}(P)$
$\quad \vee \ \langle x \rangle \in \text{traces}(P)$

OBSERVATIONS.

.4

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

$B \in \text{refusals}(P/s) \wedge s \in \text{traces}(P)$

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

$s = t \wedge B \subseteq C$
$\vee \ s < t \wedge B = \{\}$

ASSERTIONS

.5

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

$(x : A \to Px) = (tr = \langle\rangle \wedge \bar{A} \subseteq ref \subseteq \alpha P)$
$\quad \vee (tr_0 \in A \wedge P(tr_0)[tr'/tr])$

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

$(c \to P) = (tr = \langle\rangle \wedge c \tilde{\in} ref \subseteq \alpha P)$
$\quad \vee (tr_0 = c \wedge P[tr'/tr])$

$P/s = P[s \hat{\ } tr/tr]$

$(P \| Q) = \exists X, Y : ref = X \cup Y :$
$\quad P[X/ref, tr \restriction \alpha P/tr]$
$\quad \wedge Q[Y/ref, tr \restriction \alpha Q/tr]$

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

.6

$P \sqcap Q = P \vee Q$

$P \,[\!]\, Q = \underline{\text{if}}\ tr = \langle\rangle\ \underline{\text{then}}\ (P \wedge Q)\ \underline{\text{else}}\ (P \vee Q)$

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

$P \backslash C = \underline{E}s : s \restriction \bar{C} = tr :$
$\quad P[s/tr, (ref \cup C)/ref]$

provided that

$\underline{A}t :: [tr \mid tr \restriction \bar{C} = t \wedge P[\{\}/ref]]$
is FINITE

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

Bill Roscoe
Steve Brookes.

20) EXAMPLES

VM with buffering of coins perhaps.

Let BAL = tr.coin − tr.choc

$$VM = \Box((BAL = 0 \wedge \{coin\} \le \overline{ref}) \vee (BAL > 0 \wedge \{choc\} \le \overline{ref}))$$

$$= VM_0 = coin \rightarrow VM_1$$

$$VM_{n+1} = choc \rightarrow VM_n \sqcap (coin \rightarrow VM_{n+2} \;[]\; choc \rightarrow VM_n)$$

$$VMS = VM \wedge \Box BAL \le 1$$

$$VMS \Rightarrow VM$$

i.e. VMS is a valid implementation of VM

CD is a cash dispenser

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

BAL = tr.deposit − tr.withdraw

$$CD = \Box(|deposit \in \overline{ref} \wedge (BAL \le 0 \vee withdraw \in \overline{ref})$$

EXERCISE

A. Write process CD

B. Write a process CDB such that

$$CDB \Rightarrow CD$$

COMMUNICATING PROCESSES.

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

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

outputs v on channel c

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

inputs a message x on channel c

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

EXAMPLES

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

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

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

$P_{<>} = left?s \rightarrow P_s$

$P_{<x>s} = right!x \rightarrow P_s$

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

$P_\ell = left?x \rightarrow P_{\ell<x>}$ $\qquad$ if $\#\ell < 125$

$P_\ell = right!\ell \rightarrow P_{<>}$ $\qquad$ if $\#\ell = 125$

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

21)

5.3

$VAR = left?x \rightarrow VAR_x$

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

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

$BUFFER = P_{<>}$

$P_{<>} = left?x \rightarrow P_{<x>}$

$P_{<x>s} = (left?y \rightarrow P_{<x>s<y>}$
$\quad | right!x \rightarrow P_s)$

$STACK = P_{<>}$

$P_{<>} = (isempty \rightarrow P_{<>} | left?x \rightarrow Px)$

$P_{<x>s} =$

TRACES

5.4

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

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

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

$tr.right = \langle 3 \rangle$

Define $t \overset{n}{\geq} s = s \leq t \;\&\; \#t \leq \#s + n$

$COPY = \Box\, tr.left \overset{1}{\geq} tr.right$

$DOUBLE = \Box\, double^*(tr.left) \overset{1}{\geq} tr.right.$

$UNPACK = \Box\, /tr.left \geq tr.right$
$\quad \& \; \hat{}/(trunc(tr.left)) \leq tr.right.$

where $trunc(s<x>) = s.$

$PACK = \Box\, tr.left \overset{125}{\geq} \hat{}/tr.right.$
$\quad \& \; \forall s : s \in range\ tr.right : \#s = 125$

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

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

where $P[d/c] = f^{-1}(P)$

$f(d.v) = c.v$

$f(b.v) = b.v \quad$ 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 >> DOUBLE

LISTING = UNPACK >> PACK

CONWAY = UNPACK >> SQUASH >> PACK

BUFFERED = UNPACK >> BUFFER >> PACK

BUFFER3 = COPY >> COPY >> COPY

22) ALGEBRA 5.7

$\gg$ is associative

$(\text{right}!v \to P) \gg (\text{left}?x \to Q\ y) = P \gg (Q\ v)$

$(\text{right}!v \to P) \gg (\text{right}!w \to Q)$
$= (\text{right}!w \to ((\text{right}!v \to P) \gg Q))$

$(\text{left}?x \to P\ x) \gg (\text{left}?y \to Q\ y)$
$= (\text{left}?x \to (P\ x \gg (\text{left}?y \to Q\ y)))$

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

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

RELATIONS 5.8

$\ell(S;T)r = \exists u :: \ell S u \ \&\ u T r$

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

$(\geq);(\geq) = \geq$
$\geq^m ; \geq^n = \geq^{m+n}$
$f^* ; \geq^m = \geq^m ; f^*$
where $\ell(f^*)r = r = f^*(\ell)$
$f^* ; g^* = (g \circ f)^*$

S is <u>grounded</u> if for all ℓ, r:

$\ell S r \Rightarrow \ell = r = \langle\rangle$
$\vee\ \text{trunc}(\ell)\ S\ r$
$\vee\ \ell\ S\ \text{trunc}(r)$

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

Let R and S be grounded: 5.9

If $P = \Box\ \text{tr.left}\ R\ \text{tr.right}$
$Q = \Box\ \text{tr.left}\ S\ \text{tr.right}$

then $P \gg Q = \Box\ \text{tr.left}\ (R;S)\ \text{tr.right}$
provided that for all s, t
$\{u \mid s R u\ \&\ u S t\}$ is finite.

BUFFER $\gg$ BUFFER
$= \Box\ \text{tr.left}\ (\geq;\geq)\ \text{tr.right}$
$=$ BUFFER

$(\text{COPY} \gg \text{COPY}) = \Box\ \text{tr.left}\ \geq^2\ \text{tr.right}$

$(\text{DOUBLE} \gg \text{DOUBLE}) =$
$\Box\ \text{tr.left}\ (\text{double}^*; \geq^1; \text{double}^*; \geq^1)\ \text{tr.right}$
$= (\text{double}^*; \text{double}^*; \geq^1; \geq^1)$
$= (\text{quadruple}^*; \geq^2)$

SEQUENTIAL PROCESSES. 6.1

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

SKIP_A does nothing but terminate successfully.
$\alpha\text{SKIP}_A = A$

$\text{SKIP}_A \neq \text{STOP}_A$

If P is guarded.

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

23) EXAMPLES

6.2

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

$VM3 = VM1; VM1; VM1$

$VMS = *VM1$

$AnBCn = \mu X. (b \rightarrow SKIP$
$\quad | a \rightarrow (X; (c \rightarrow SKIP)))$

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

6.3

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

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

$C_0 = (around \rightarrow C_0 \mid up \rightarrow C_1)$
$C_{n+1} = POS; C_n$ for all $n \geq 0$

$C_0 = CT_0$!

LAWS

6.4

; is associative with unit SKIP
distributive

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

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}
$= RHS$