IDEALIZED CSP:
Procedures + Processes

Stephen Brookes
Department of Computer Science
Carnegie Mellon University

July 1997

THE ESSENCE OF CSP

- **Idealized CSP =**

  communicating processes

  + call-by-name λ-calculus

- **simply typed**

$$\theta ::= \mathbf{exp}[\tau] \;|\; \mathbf{var}[\tau] \;|\; \mathbf{chan}[\tau] \;|\; \mathbf{comm} \;|\; (\theta \to \theta') \;|\; \theta \times \theta'$$

phrase types

$$\tau ::= \mathbf{int} \;|\; \mathbf{bool}$$

data types

cf. Reynolds: Idealized Algol

cf. Brookes: Parallel Algol

CONNECTIONS

- **generalizes Hoare's CSP**
  - fairness
  - nested parallel
  - dynamic process creation
  - channel-based communication
  - asynchronous output, synchronous input
- **generalizes Idealized Algol**
  - typed channels
  - communicating processes
  - local channel declarations
- **generalizes Kahn networks**

RATIONALE

- **Programs can cooperate**
  - message-passing
  - access to shared memory
- **Procedures can encapsulate parallel idioms**
  - bounded buffer
  - communication protocols
- **Local variables and channels can limit scope of interference**

INTUITION

Procedures and parallelism should be orthogonal

BUFFERS

procedure *buff1*(*in*, *out*) =
  new[τ] x **in**
    while true do $(in?x; out!x)$

procedure *buff*(*in*, *out*) =
  newchan[τ] h **in**
    buff1(*in*, *h*) $\|$ *buff1*(*h*, *out*)

- Encapsulates common way to build buffers
- Relies on *locality* of x and h

SIEVE

procedure *filter*$(p,\ in,\ out) =$
  new[**int**] x **in**
    while true do
      $(in?x;$ **if** x **mod** $p \neq 0$ **then** $out!x);$

procedure *sift*$(in,\ out) =$
  newchan[**int**] h **in**
  new[**int**] p **in**
    begin
      $in?p;\ out!p;$
      filter$(p,\ in,\ h)$ $\|$ *sift*$(h,\ out)$
    end

SYNTAX

- **Input and Output**

$$\frac{\pi \vdash h : \mathbf{chan}[\tau] \quad \pi \vdash E : \mathbf{exp}[\tau]}{\pi \vdash h!E : \mathbf{comm}}$$

$$\frac{\pi \vdash h : \mathbf{chan}[\tau] \quad \pi \vdash X : \mathbf{var}[\tau]}{\pi \vdash h?X : \mathbf{comm}}$$

- **Parallel composition**

$$\frac{\pi \vdash P_1 : \mathbf{comm} \quad \pi \vdash P_2 : \mathbf{comm}}{\pi \vdash P_1 \| P_2 : \mathbf{comm}}$$

- **Local channel declaration**

$$\frac{\pi, \iota : \mathbf{chan}[\tau] \vdash P : \mathbf{comm}}{\pi \vdash \mathbf{newchan}[\tau]\ \iota\ \mathbf{in}\ P : \mathbf{comm}}$$

CATEGORY of WORLDS

- Objects are countable posets of "allowed states"

  $$V_1 \times \cdots \times V_k \times H_1^* \times \cdots H_n^*$$

  ordered by prefix, componentwise
- Morphisms $(f, Q) : W \to X$
  - function f from X to W
  - equivalence relation Q on X
  - f puts each Q-class into order-isomorphism with W

Generalizes Oles's category:

- channels as components of state
- morphisms respect queue structure

EXPANSIONS

- For each pair of objects W and V there is an expansion

$$- \times V : W \to W \times V$$

given by

$$- \times V = (\mathrm{fst} : W \times V \to W,\ Q)$$
$$(w_0, v_0) Q (w_1, v_1) \iff v_0 = v_1$$

- Use $- \times V_\tau^*$ to model local channel declaration

- Each morphism is an expansion, up to order-isomorphism

- Some order-isomorphisms:

$$\mathit{swap} : W \times V \to V \times W$$
$$\mathit{assoc} : W \times (V \times U) \to (W \times V) \times U$$

but not $\mathit{rev} : V_\tau^* \to V_\tau^*$

SEMANTICS

- Types denote functors from worlds to domains, $[\![\theta]\!] : \mathbf{W} \to \mathbf{D}$

- Judgements $\pi \vdash P : \theta$ denote natural transformations

  $$[\![P]\!] : [\![\pi]\!] \stackrel{\cdot}{\to} [\![\theta]\!]$$

  i.e. when $h : W \to X$,

  $$\begin{array}{ccc} [\![\pi]\!]W & \xrightarrow{[\![P]\!]W} & [\![\theta]\!]W \\ {\scriptstyle [\![\pi]\!]h}\downarrow & & \downarrow{\scriptstyle [\![\theta]\!]h} \\ [\![\pi]\!]X & \xrightarrow[{[\![P]\!]X}]{} & [\![\theta]\!]X \end{array}$$

  commutes.

  Naturality enforces locality

CARTESIAN CLOSURE

- The functor category $\mathbf{D}^{\mathbf{W}}$ is cartesian closed
- Use ccc structure to interpret arrow and product types

$$[\![\theta \times \theta']\!] = [\![\theta]\!] \times [\![\theta']\!]$$
$$[\![\theta \to \theta']\!] = [\![\theta]\!] \Rightarrow [\![\theta']\!]$$

- Procedures are natural and therefore respect locality

PROCEDURES

A procedure of type $\theta \rightarrow \theta'$ at world W denotes a natural family of functions $p(-)$:
if $h : W \rightarrow X$ and $h' : X \rightarrow Y$,

$$\begin{array}{ccc} [\![\theta]\!]X & \xrightarrow{p(h)} & [\![\theta']\!]X \\ {\scriptstyle [\![\theta]\!]h'} \downarrow & & \downarrow {\scriptstyle [\![\theta']\!]h'} \\ [\![\theta]\!]Y & \xrightarrow[p(h;h')]{} & [\![\theta']\!]Y \end{array}$$

commutes.

Procedures can be called at expanded worlds, and naturality enforces locality constraints.

COMMANDS

$$[\![\mathbf{comm}]\!]W = \wp^{\dagger}((W \times W)^{\infty})$$

... as for shared-variable programs

- commands denote trace sets
- closed under stuttering and mumbling

$$\alpha\beta \in t \;\&\; w \in W \;\Rightarrow \alpha\langle w, w\rangle\beta \in t$$
$$\alpha\langle w, w'\rangle\langle w', w''\rangle\beta \in t \;\Rightarrow\; \alpha\langle w, w''\rangle\beta \in t$$

... with certain modifications

- message-passing as state change
- interference thus models communication by environment
- unrequited input = busy wait

INTUITION

- A trace

$$\langle w_0, w_0' \rangle \langle w_1, w_1' \rangle \ldots \langle w_n, w_n' \rangle \ldots$$

  models a fair interaction

- Each step $\langle w_i, w_i' \rangle$ represents a finite sequence of atomic actions

- If $(f, Q) : W \to X$ and $c \in [\![\mathbf{comm}]\!]W$

$$[\![\mathbf{comm}]\!](f, Q)c$$

  behaves like c on the W-component of state, has no effect elsewhere.

CHANNELS

An "object-oriented" semantics:

- output = acceptor
- input = expression with side-effect

$$[\![\mathbf{chan}[\tau]]\!]W = \\ (V_\tau \to [\![\mathbf{comm}]\!]W) \times [\![\mathbf{exp}[\tau]]\!]W$$

$$[\![\mathbf{exp}[\tau]]\!]W = \\ \wp((W \times W)^+ \times V_\tau \;\cup\; (W \times W)^\omega)$$

cf. Reynolds, Oles

PARALLEL COMPOSITION

Fair merge of traces

$$\llbracket P_1 \| P_2 \rrbracket W u = \\ \{\alpha \mid \exists \alpha_1 \in \llbracket P_1 \rrbracket W u,\ \alpha_2 \in \llbracket P_2 \rrbracket W u. \\ (\alpha_1, \alpha_2, \alpha) \in \mathit{fairmerge}_{W \times W}\}^{\dagger}$$

where

$$\mathit{fairmerge}_A = \mathit{both}_A^{*} \cdot \mathit{one}_A \cup \mathit{both}_A^{\omega}$$
$$\mathit{both}_A = \{(\alpha, \beta, \alpha\beta), (\alpha, \beta, \beta\alpha) \mid \alpha, \beta \in A^{+}\}$$
$$\mathit{one}_A = \{(\alpha, \epsilon, \alpha), (\epsilon, \alpha, \alpha) \mid \alpha \in A^{\infty}\}$$

fairmerge is natural

INPUT and OUTPUT

In world $V_{int} \times V_{int}^*$ and a suitable environment

- $h?x$ has traces

  $\langle(v, n\rho), (n, \rho)\rangle \qquad (v, n \in V_{int},\ \rho \in V_{int}^*)$

  and

  $\langle(v_0, \epsilon), (v_0, \epsilon)\rangle \ldots \langle(v_k, \epsilon), (v_k, \epsilon)\rangle \ldots$

- $h!(x + 1)$ has traces

  $\langle(m, \sigma), (m, \sigma(m{+}1))\rangle \quad (m \in V_{int},\ \sigma \in V_{int}^*)$

- $h?x \| h!(x + 1)$ has traces

  $\langle(v, n\rho), (n, \rho)\rangle\langle(m, \sigma), (m, \sigma(m + 1))\rangle$

  and

  $\langle(m, \sigma), (m, \sigma(m + 1))\rangle\langle(v, n\rho), (n, \rho)\rangle$

  and

  $\langle(m, \epsilon), (m + 1, \epsilon)\rangle$

CHOICE

An external choice

$$(a?x \rightarrow P_1) \Box (b?x \rightarrow P_2)$$

can

- input on a and behave like P_1
- input on b and behave like P_2
- busy-wait while a and b are *both* empty

However, an internal choice

$$(a?x \rightarrow P_1) \sqcap (b?x \rightarrow P_2)$$

can busy-wait if *either* a or b is empty

LOCAL CHANNELS

The traces of

$$\mathbf{newchan}[\tau]\ h\ \mathbf{in}\ P$$

at world W are projected from the traces of P in world $W \times V_\tau^*$ in which

- initially h is empty
- contents of h never change across step boundaries

EXAMPLES

- $\mathbf{newchan}[\tau]\ h\ \mathbf{in}\ (h!e \| h?x) \ = \ x{:=}e$
- $\mathbf{newchan}[\tau]\ h\ \mathbf{in}\ (h!0; P) \ = \ P$
  if h does not occur free in P
- $\mathbf{newchan}[\tau]\ h\ \mathbf{in}\ (h?x; P)$
  has only infinite stuttering traces, because of unrequited input

BUFFERS

In world $V_{int}^* \times V_{int}^*$ and a suitable environment

$$\mathit{buff1}(\mathit{left},\ \mathit{right})$$

has trace

$$\langle(0,\epsilon),(\epsilon,\epsilon)\rangle \quad \text{input 0}$$
$$\langle(1,\epsilon),(1,0)\rangle \quad \text{output 0}$$
$$\langle(1,0),(\epsilon,0)\rangle \quad \text{input 1}$$
$$\ldots$$

Similarly

$$\mathit{buff}(\mathit{left},\ \mathit{right})$$

has trace

$$\langle(0,\epsilon),(\epsilon,\epsilon)\rangle \quad \text{input 0}$$
$$\langle(1,\epsilon),(\epsilon,\epsilon)\rangle \quad \text{input 1}$$
$$\langle(2,\epsilon),(2,0)\rangle \quad \text{output 0}$$
$$\ldots$$

LAWS

- **Symmetry**

$$\begin{aligned}&\textbf{newchan}[\tau_1]\ h_1\ \textbf{in}\\&\quad\textbf{newchan}[\tau_2]\ h_2\ \textbf{in}\ P\\&\qquad=\ \textbf{newchan}[\tau_2]\ h_2\ \textbf{in}\\&\qquad\qquad\textbf{newchan}[\tau_1]\ h_1\ \textbf{in}\ P\end{aligned}$$

- **Frobenius**

$$\begin{aligned}&\textbf{newchan}[\tau]\ h\ \textbf{in}\ (P_1 \| P_2)\ =\\&\qquad(\textbf{newchan}[\tau]\ h\ \textbf{in}\ P_1) \| P_2\end{aligned}$$

provided h does not occur free in P_2

LAWS

- **Local variables**

$$\mathbf{newvar}[\tau]\ \iota\ \mathbf{in}\ P' = P'$$
$$\mathbf{newvar}[\tau]\ \iota\ \mathbf{in}\ (P \| P') = (\mathbf{newvar}[\tau]\ \iota\ \mathbf{in}\ P) \| P'$$

when ι does not occur free in P'

- **Functional laws**

$$(\lambda\iota : \theta.P)(Q) = P[Q/\iota]$$
$$\mathbf{rec}\ \iota\mathbf{.}P = P[\mathbf{rec}\ \iota\mathbf{.}P/\iota]$$

LOCAL LAWS

- **Local output**

  $$\textbf{newchan}[\tau]\ h = \rho\ \textbf{in}\ P_1 \| h!v; P_2 \\ \quad = \ \textbf{newchan}[\tau]\ h = \rho v\ \textbf{in}\ P_1 \| P_2$$

  if $h!$ not in P_1

- **Local input**

  $$\textbf{newchan}[\tau]\ h = v\rho\ \textbf{in}\ P_1 \| h?v; P_2 \\ \quad = \ \textbf{newchan}[\tau]\ h = \rho\ \textbf{in}\ P_1 \| P_2$$

  if $h?$ not in P_1

. . . help when channels are used in at most one direction by each process

FAIRNESS LAWS

- **Fair prefix**
  If h not free in P_1 and

  $$\textbf{newchan}[\tau]\ h = \rho\ \textbf{in}\ P$$

  diverges, then

  $$\begin{aligned}&\textbf{newchan}[\tau]\ h = \rho\ \textbf{in}\ P \| (P_1; P_2)\\ &\quad = P_1; \textbf{newchan}[\tau]\ h = \rho\ \textbf{in}\ P \| P_2\end{aligned}$$

- **Unrequited input**
  If h not free in P_1 then

  $$\begin{aligned}&\textbf{newchan}[\tau]\ h\ \textbf{in}\ (h?x; P) \| (P_1; P_2)\\ &\quad = P_1; \textbf{newchan}[\tau]\ h\ \textbf{in}\ (h?x; P) \| P_2\end{aligned}$$

CONCLUSIONS

- Transition traces are fundamental, general and unifying
  - shared-variable
  - communicating process
- Fairness incorporated smoothly
- Deadlock = busy-waiting
  - avoids need for failure sets
- Implicit treatment of channels
  - no channel names in traces
  - object-oriented model
  - channels kept separate

FURTHER WORK

- Relational parametricity
  - representation independence
  - concurrent objects
- Full abstraction at ground types
  - observing sequence of states
- Disjoint processes

  $$[\![P_1 \| P_2]\!](W_1 \times W_2, H) = [\![P_1]\!](W_1, H) \parallel [\![P_2]\!](W_2, H)$$
- Unreliable communication
  - lossy channels
  - bounded channels
- Synchronous communication

CONCURRENT OBJECTS

newchan$[\tau]$ h **in**
  procedure $put(y) = h!y;$
  procedure $get(z) =$
      new$[\tau]$ x **in** $(h?x;\ z{:=}x);$
  begin
    $P(put,\ get)$
  end

newchan$[\tau]$ h **in**
  procedure $put(y) = h!(-y);$
  procedure $get(z) =$
      new$[\tau]$ x **in** $(h?x;\ z{:=}(-x));$
  begin
    $P(put,\ get)$
  end