Deconstructing CCS and CSP

Fairness, Asynchrony, and Full Abstraction

Stephen Brookes

MFPS 16

Special Session in honor of Robin Milner

MANIFESTO

CCS and CSP assumed

- *asynchronous processes* running at indeterminate rate
- *handshake communication*

Could equally well have chosen

asynchronous communication

as primitive. . .

- simpler semantics: *traces suffice*
- fairness and full abstraction
- unification of paradigms

Milner's CCS

A Calculus of Communicating Systems '80

- processes

$$
\begin{array}{lll}
P ::= & nil \mid & \textit{inaction} \\
& \lambda.P \mid & \textit{prefix} \\
& P_1 + P_2 \mid & \textit{sum} \\
& (P_1 | P_2) \mid & \textit{parallel} \\
& P \backslash a & \textit{restriction}
\end{array}
$$

- labels

$$\lambda ::= a?v \mid a!v \mid \tau$$

- transitions

$$P \xrightarrow{\lambda} Q$$

asynchronous parallel processes
with handshake communication

TRANSITION RULES

$$\frac{}{\lambda.P \xrightarrow{\lambda} P}$$

$$\frac{P \xrightarrow{\lambda} P'}{P+Q \xrightarrow{\lambda} P'} \qquad \frac{Q \xrightarrow{\lambda} Q'}{P+Q \xrightarrow{\lambda} Q'}$$

$$\frac{P \xrightarrow{\lambda} P'}{P|Q \xrightarrow{\lambda} P'|Q} \qquad \frac{Q \xrightarrow{\lambda} Q'}{P|Q \xrightarrow{\lambda} P|Q'}$$

$$\frac{P \xrightarrow{\lambda} P' \quad Q \xrightarrow{\bar{\lambda}} Q'}{P|Q \xrightarrow{\tau} P'|Q'}$$

$$\frac{P \xrightarrow{\lambda} P' \quad ch(\lambda) \neq a}{P \backslash a \xrightarrow{\lambda} P' \backslash a}$$

$$\overline{a?v} = a!v \qquad \overline{a!v} = a?v$$

WEAK BISIMILARITY

Park '80

$P \approx Q$ if and only if

- $\forall \lambda, P'.\ P \stackrel{\lambda}{\Longrightarrow} P'$ *implies*
  $\exists Q'.\ Q \stackrel{\lambda}{\Longrightarrow} Q'\ \&\ P' \approx Q'$
- $\forall \lambda, Q'.\ Q \stackrel{\lambda}{\Longrightarrow} Q'$ *implies*
  $\exists P'.\ P \stackrel{\lambda}{\Longrightarrow} P'\ \&\ P' \approx Q'$

where

$$\stackrel{\lambda}{\Longrightarrow}\ =\ (\stackrel{\tau}{\rightarrow})^{*} \circ \stackrel{\lambda}{\longrightarrow} \circ (\stackrel{\tau}{\rightarrow})^{*}$$

CONGRUENCE

$$P \approx^{c} Q \iff \forall R.\ P + R \approx Q + R$$

the largest CCS congruence contained in $\approx$

CALCULUS

- static laws

$$P|Q = Q|P$$

- dynamic laws

$$P + \tau.P = \tau.P$$

- expansion laws

$$(\lambda.P)|(\mu.Q) = \lambda.(P|(\mu.Q)) + \mu.((\lambda.P)|Q)$$

$$\textit{if } \bar{\lambda} \neq \mu$$

- unique fixed point

$$\frac{Q = P[Q/p]}{Q = \mathbf{rec}\ p.P}$$

$$\textit{if } p \textit{ is guarded in } P$$

Hoare's CSP

A Theory of Communicating Sequential Processes, HBR '81

- processes

$$
\begin{array}{lll}
P ::= & nil \mid & \textit{inaction} \\
 & \lambda.P \mid & \textit{prefix} \\
 & P_1 \Box P_2 \mid & \textit{external choice} \\
 & P_1 \sqcap P_2 \mid & \textit{internal choice} \\
 & (P_1 | P_2) \mid & \textit{parallel} \\
 & P/a & \textit{hiding}
\end{array}
$$

- labels

$$\lambda ::= a?v \mid a!v \mid \tau$$

- transitions

$$P \xrightarrow{\lambda} Q$$

asynchronous parallel processes
with handshake communication

TRANSITION RULES

$$\frac{}{P \sqcap Q \xrightarrow{\tau} P} \qquad \frac{}{P \sqcap Q \xrightarrow{\tau} Q}$$

$$\frac{P \xrightarrow{\lambda} P' \quad \lambda \neq \tau}{P \Box Q \xrightarrow{\lambda} P'} \qquad \frac{Q \xrightarrow{\lambda} Q' \quad \lambda \neq \tau}{P \Box Q \xrightarrow{\lambda} Q'}$$

$$\frac{P \xrightarrow{\tau} P'}{P \Box Q \xrightarrow{\tau} P' \Box Q} \qquad \frac{Q \xrightarrow{\tau} Q'}{P \Box Q \xrightarrow{\tau} P \Box Q'}$$

$$\frac{P \xrightarrow{\lambda} P' \quad ch(\lambda) \neq a}{P/a \xrightarrow{\lambda} P'/a} \qquad \frac{P \xrightarrow{\lambda} P' \quad ch(\lambda) = a}{P/a \xrightarrow{\tau} P'/a}$$

$$\frac{P \xrightarrow{\lambda} P'}{P|Q \xrightarrow{\lambda} P'|Q} \qquad \frac{Q \xrightarrow{\lambda} Q'}{P|Q \xrightarrow{\lambda} P|Q'}$$

$$\frac{P \xrightarrow{\lambda} P' \quad Q \xrightarrow{\bar{\lambda}} Q'}{P|Q \xrightarrow{\tau} P'|Q'}$$

FAILURES

- traces

$$t(P) = \{\alpha \in \Lambda^* \mid P \stackrel{\alpha}{\Longrightarrow} P'\}$$

- failures

$$f(P) = \{(\alpha, X) \mid P \stackrel{\alpha}{\Longrightarrow} P' \;\&\; P' \,\mathbf{ref}\, X\}$$

- refusals

$$P \,\mathbf{ref}\, X \iff \forall \lambda \in X \cup \{\tau\}.\, \neg(P \stackrel{\lambda}{\longrightarrow})$$

- failure equivalence

$$P \equiv_f Q \iff f(P) = f(Q)$$

PROPERTIES

- full abstraction for deadlock behavior

$\equiv_f$ is the coarsest reasonable congruence for CSP

- $\approx$ is also a CSP congruence

LIMITATIONS

- Infinite computations
  - no distinction between *fair* and *unfair*

$$\lambda | \mu^\omega \neq \mu^* \lambda \mu^\omega$$

  - divergence ignored by $\approx$ and $\equiv_f$
- Handshake communication
  - hard to model fairness
  - asynchronous communication seems more primitive

ASYNCHRONY

asynchronous output,
busy-waiting input

- channel = queue
- actions depend on *state*
  - $a!v$ is always enabled
  - $a?v$ is enabled if a is non-empty and v is the first item
  - $a?v$ must *wait* if a is empty

TRANSITIONS

$$\langle P, s \rangle \xrightarrow{\lambda} \langle P', s' \rangle$$

- $\lambda ::= a?v \mid a!v \mid \delta_X$
- δ_X = wait on channels in X
- state maps channels to contents

TRANSITION RULES

asynchronous parallel processes,
asynchronous communication

$$\frac{deq(a)(s) = (v, s')}{\langle a?v.P, s\rangle \xrightarrow{a?v} \langle P, s'\rangle} \qquad \frac{null(a)(s) \quad a \in X}{\langle a?v.P, s\rangle \xrightarrow{\delta_X} \langle a?v.P, s\rangle}$$

$$\frac{enq(a)(s) = s'}{\langle a!v.P, s\rangle \xrightarrow{a!v} \langle P, s'\rangle} \qquad \frac{}{\langle nil, s\rangle \xrightarrow{\delta_X} \langle nil, s\rangle}$$

$$\frac{\langle P, s\rangle \xrightarrow{\lambda} \langle P', s'\rangle \quad \lambda \neq \delta_X}{\langle P \Box Q, s\rangle \xrightarrow{\lambda} \langle P', s\rangle} \qquad \frac{\langle Q, s\rangle \xrightarrow{\lambda} \langle Q', s'\rangle \quad \lambda \neq \delta_X}{\langle P \Box Q, s\rangle \xrightarrow{\lambda} \langle Q', s'\rangle}$$

$$\frac{\langle P, s\rangle \xrightarrow{\delta_X} \langle P', s\rangle \quad \langle Q, s\rangle \xrightarrow{\delta_X} \langle Q', s\rangle}{\langle P \Box Q, s\rangle \xrightarrow{\delta_X} \langle P' \Box Q', s\rangle}$$

$$\frac{\langle P, s\rangle \xrightarrow{\lambda} \langle P', s'\rangle}{\langle P|Q, s\rangle \xrightarrow{\lambda} \langle P'|Q, s'\rangle} \qquad \frac{\langle Q, s\rangle \xrightarrow{\lambda} \langle Q', s'\rangle}{\langle P|Q, s\rangle \xrightarrow{\lambda} \langle P|Q', s'\rangle}$$

LOCAL CHANNELS

$$\textbf{local } a = \rho \textbf{ in } P \qquad (\rho \in V^*)$$

- generalizes hiding and restriction
  - local actions are invisible
  - only local output is "uncontrollable"

$$\frac{\langle P, (s, a:\rho)\rangle \xrightarrow{\lambda} \langle P', (s', a:\rho')\rangle \quad \mu \in \lambda/a}{\langle \textbf{local } a = \rho \textbf{ in } P, s\rangle \xrightarrow{\mu} \langle \textbf{local } a = \rho' \textbf{ in } P', s'\rangle}$$

where

$$\begin{array}{rcll} \delta_X/a & = & \{\delta_Y \mid X - \{a\} \subseteq Y\} & \\ a?v/a = a!v/a & = & \{\delta_X \mid X \subseteq Ch\} & \\ \lambda/a & = & \{\lambda\} & \textit{otherwise} \end{array}$$

TRACES

- communication traces

$$ct(P) = \{\alpha \in \Lambda^\omega \mid P \stackrel{\alpha}{\Longrightarrow} \mathit{fair}\}$$

$$P \stackrel{\lambda}{\Longrightarrow} P' \ \mathit{iff}\ \exists s, s'.\ \langle P, s\rangle \stackrel{\lambda}{\Longrightarrow} \langle P', s'\rangle$$

$$\Lambda = \{a?v, a!v\} \cup \{\delta_X\}$$

- transition traces

$$tt(P) = \{\beta \in (S \times S)^\omega \mid P \stackrel{\beta}{\Longrightarrow} \mathit{fair}\}$$

$$P \xRightarrow{(s,s')} P' \ \mathit{iff}\ \exists\lambda.\langle P, s\rangle \stackrel{\lambda}{\Longrightarrow} \langle P', s'\rangle$$

FAILURES

- asynchronous failures

$$f(P) = \{(\alpha, X) \mid P \stackrel{\alpha}{\Longrightarrow} P' \ \&\ P' \,\mathbf{ref}\, X\}$$

$$P \,\mathbf{ref}\, X \ \mathit{iff}\ P \ \mathit{stable}\ \&\ \forall a \in X. \neg(P \xrightarrow{a?})$$

CLOSURE

$ct(P)$ is closed under:

- stuttering

$$\alpha\beta \mapsto \alpha\delta_X\beta$$

- muttering

$$\alpha\delta_\phi\beta \mapsto \alpha\beta$$
$$\alpha\delta_X\delta_Y\beta \mapsto \alpha\delta_{X\cup Y}\beta$$

$tt(P)$ is closed under:

- stuttering

$$\alpha\beta \mapsto \alpha(s,s)\beta$$

- muttering

$$\alpha(s,s)(s,s')\beta \mapsto \alpha(s,s')\beta$$
$$\alpha(s,s')(s',s')\beta \mapsto \alpha(s,s')\beta$$

PROPERTIES

- ct and tt are *compositional*
- All CSP constructs are *monotone*

$$ct(P) \subseteq ct(Q) \Rightarrow ct(C[P]) \subseteq ct(C[Q])$$
$$tt(P) \subseteq tt(Q) \Rightarrow tt(C[P]) \subseteq tt(C[Q])$$

- ct and tt are *equivalent*

$$ct(P) \subseteq ct(Q) \text{ iff } tt(P) \subseteq tt(Q)$$

- Failures can be recovered

$$f(P) = \{(\alpha, X) \mid \alpha\delta_X^\omega \in ct(P)\}$$

COMPOSITIONALITY

$$ct(nil) \;=\; \Delta^\omega$$

$$ct(a!v.P) \;=\; \{a!v\,\alpha \mid \alpha \in ct(P)\}^\dagger$$

$$ct(a?v.P) \;=\; \Delta_a^\omega \;\cup\; \{a?v\,\alpha \mid \alpha \in ct(P)\}^\dagger$$

$$ct(P \sqcap Q) \;=\; ct(P) \cup ct(Q)$$

$$\begin{aligned} ct(P \Box Q) \;=\; & ((ct(P) \cap ct(Q)) \cap \Delta^\omega) \cup \\ & ((ct(P) \cup ct(Q)) - \Delta^\omega) \end{aligned}$$

$$\begin{aligned} ct(P|Q) \;=\; & \{\gamma \mid \exists \alpha \in ct(P), \beta \in ct(Q). \\ & \quad (\alpha, \beta, \gamma) \in \mathit{fairmerge}\} \end{aligned}$$

$$\begin{aligned} & ct(\mathbf{local}\ a = \rho\ \mathbf{in}\ P) \;= \\ & \quad \{\gamma \mid \exists \alpha \in ct(P).\ \gamma \in \alpha/a\ \& \\ & \quad\quad \alpha\ \textit{int-free for a from}\ \rho\} \end{aligned}$$

CONNECTIONS

Let

$$[\![-]\!] : \Lambda \longrightarrow \mathcal{P}(S \times S)$$

be given by:

$$\begin{array}{rcl} [\![a?v]\!] & = & \{(s, s') \mid (v, s') = deq(a)(s)\} \\ [\![a!v]\!] & = & \{(s, s') \mid s' = enq(a, v)(s)\} \\ [\![\delta_X]\!] & = & \{(s, s) \mid \forall a \in X.\ null(a)(s)\} \end{array}$$

Extend to $[\![-]\!] : \Lambda^\omega \longrightarrow \mathcal{P}((S \times S)^\omega)$.

- $tt(P) \ = \ \bigcup\{[\![\alpha]\!] \mid \alpha \in ct(P)\}$
- $ct(P) \ = \ \{\alpha \mid [\![\alpha]\!] \subseteq tt(P)\}$

CALCULUS

$$P|Q = Q|P$$

$$P \sqcap Q \supseteq P \,\square\, Q$$

$$P \,\square\, nil = P$$

$$nil \sqcap (a?x \,\square\, b?x) = nil \sqcap a?x \sqcap b?x$$

$$(a?x.P) \,\square\, (a?x.Q) = a?x.(P \sqcap Q)$$

$$\begin{aligned}(a?x.P)|(b?y.Q) \supseteq\ & a?x.(P|(b?y.Q)) \,\square\, \\ & b?y.((a?x.P)|Q)\end{aligned}$$

$a?x|b?y$ has $(\delta_a \delta_b)^\omega$

$a?xb?y \,\square\, b?ya?x$ doesn't

OBSERVATIONS

- Start with channels empty
- Run fairly, without interference

- Observe *communication labels*

$$c(P) = \{\alpha \in \Lambda^\omega \mid P \overset{\alpha}{\Longrightarrow} \mathit{fair}, \mathit{int\text{-}free}\}$$

- Observe *state change*

$$t(P) = \{\beta \in (S \times S)^\omega \mid P \overset{\beta}{\Longrightarrow} \mathit{fair}, \mathit{int\text{-}free}\}$$

safety, liveness, deadlock
linear-time temporal logic

FULL ABSTRACTION

- ct is fully abstract for c

  $ct(P) \subseteq ct(Q)$ iff $\forall C.\, c(C[P]) \subseteq c(C[Q])$

- tt is fully abstract for t

  $tt(P) \subseteq tt(Q)$ iff $\forall C.\, t(C[P]) \subseteq t(C[Q])$

- c and t are *equivalent* as notions of observable

- $\{ct, tt\}$ fully abstract for $\{c, t\}$

trace equivalence is the coarsest reasonable congruence for fair asynchronous CSP

FULL ABSTRACTION

Proof Sketch

Let A be a finite set of channels.
Choose $z \notin A$.

Define a *testing process*:

$$RUN_A \;=\; \Box_{a \in A, v \in V}(a?v.z!0.RUN_A) \sqcap \Box_{a \in A, v \in V}(a!v.z!1.RUN_A) \;\sqcap\; nil$$

Let

$$\widehat{a!v} = a!v\, a?v\, z!0$$
$$\widehat{a?v} = a!v\, a?v\, z!1$$
$$\widehat{\delta_X} = \delta_X$$

Extend to traces in obvious way.

$P \,|\, RUN_A$ has int-free trace $\widehat{\alpha}$
iff P has trace α

FAIR BISIMILARITY

$P \approx_{fair} Q$ if and only if

- $ct(P) = ct(Q)$
- $\forall \lambda, P'.\ P \stackrel{\lambda}{\Longrightarrow} P'$ *implies*
  $\exists Q'.\ Q \stackrel{\lambda}{\Longrightarrow} Q'\ \&\ P' \approx_{fair} Q'$
- $\forall \lambda, Q'.\ Q \stackrel{\lambda}{\Longrightarrow} Q'$ *implies*
  $\exists P'.\ P \stackrel{\lambda}{\Longrightarrow} P'\ \&\ P' \approx_{fair} Q'$

Conjecture:

the finest reasonable congruence for fair asynchronous CSP

FAIR BISIMILARITY

$P \approx_{fair} Q$ if and only if

- $tt(P) = tt(Q)$
- $\forall s, s', P'.\ P \xrightarrow{(s,s')} P'$ *implies* $\exists Q'.\ Q \xrightarrow{(s,s')} Q'\ \&\ P' \approx_{fair} Q'$
- $\forall s, s', Q'.\ Q \xrightarrow{(s,s')} Q'$ *implies* $\exists P'.\ P \xrightarrow{(s,s')} P'\ \&\ P' \approx_{fair} Q'$

CALCULUS

- fairness laws

$$\textbf{local } a{=}\epsilon \textbf{ in } (a?x.P)|(Q_1;Q_2)$$
$$= \; Q_1; \textbf{local } a{=}\epsilon \textbf{ in } (a?x.P)|Q_2$$
$$\textit{if } a \notin ch(Q_1)$$

- recursion

$$\textbf{rec } p.P \;=\; P[\textbf{rec } p.P/p]$$

  – uniqueness fails

- fair parallel expansion laws MFPS'99

$$\frac{P = \sum_i A_i P_i \quad Q = \sum_j B_j Q_j}{P|Q = \sum_{i,j} (A_i \| B_j)(P_i|Q_j)}$$

What about +?

$$\frac{\langle P,s\rangle \xrightarrow{\lambda} \langle P',s'\rangle}{\langle P+Q,s\rangle \xrightarrow{\lambda} \langle P,s'\rangle} \qquad \frac{\langle Q,s\rangle \xrightarrow{\lambda} \langle Q',s'\rangle}{\langle P+Q,s\rangle \xrightarrow{\lambda} \langle Q',s'\rangle}$$

hence

$$ct(P+Q) = ct(P) \cup ct(Q)$$

PROBLEMS

- $P + Q = P \sqcap Q$
- $P + nil \neq P$
- $a?x + b?x$ should be $a?x \,\Box\, b?x$
- $a?x + b!0$ should be $a?x \,\Box\, b!0$

With asynchronous i/o,
$\Box$ plays role of +

RELATED WORK

handshake communication

- Milner '82
  - finite delay operator for synchronous CCS
- Costa, Stirling '84-'85
  - weak and strong fairness in CCS
- Parrow Ph.D. '85
  - fairness *via* infinitary restriction $P\langle\langle\phi\rangle\rangle$
  - infinitary charts
  - unfair notion of *weak* ω*-bisimulation*
- Hennessy TCS '87
  - acceptance trees with fair paths
  - divergence is catastrophic
  - stronger fairness notion: $\lambda^\omega | nil \neq \lambda^\omega$

RELATED WORK

asynchronous communication

- de Boer, Klop, Palamidessi LICS '92
  - intended *vs.* complete actions
  - "queue failure sets"
  - no fairness
- de Boer, Kok, Palamidessi, Rutten CONCUR '91
  - concurrent constraint programs
  - used $\mathcal{P}((S \times S)^*)$, similar closure properties
  - no fairness

full abstraction

- Milner TCS '77
  - full abstraction for typed λ-calculus

RELATED WORK

fairness

- Park '79-'82
  - shared-variable programs
  - *fairmerge* relation
  - no closure

- Brookes
  - full abstraction for shared-variable programs
  - Parallel Algol
  - Idealized CSP
  - Non-deterministic Kahn networks

CONCLUSIONS

If we assume

asynchronous communication

traces suffice:

- full abstraction
  - safety and liveness
- can unify paradigms
  - shared variable programs
  - Kahn networks
  - communicating processes
- can be generalized
  - procedures
  - objects