



CCS: Operational Semantics And Process Algebra

Reading: Peled 8.3, 8.4, 8.6 – rest of ch. 8

Mads Dam

Value-passing CCS Combinators

Value-passing combinators and definitions as *abbreviations* using labels of the form $a(v)$ (receiving v) or $\bar{b}(v)$ (sending v)

Prefix	$a(x), \bar{b}(v)$	$a(x).P(x) = a(v_n).P(v_n) + \dots + a(v_1).P(v_1)$ $\bar{b}(v).P(v) = \bar{b}(v).P(v)$
Definition	$A(x) == P(x)$	$\text{Buf} == \text{in}(x).\text{Buf}_1(x)$ $\text{Buf}_1(x) == \text{out}(x).\text{Buf}(x)$
Conditional	if C then P	$\text{Teller}(x) == \text{Deposit}(x) + \text{Withdrawal}(x)$
Summation	$\Sigma y: P(y)$	$\text{Deposit}(x) == \text{deposit}(x).\text{Teller}(x + y)$ $\text{Withdrawal}(x) ==$ $\Sigma y: \text{if } y \leq x$ $\text{then withdraw}(y).\text{Teller}(x - y)$

Transition Semantics

To apply observational equivalence need a formalised semantics

Each CCS expression \rightarrow state in LTS derived from that expression

Compositionality: Construction of LTS follows expression syntax

Inference rules:

$$\frac{P_1 \rightarrow^\alpha P_2}{P_1 | Q \rightarrow^\alpha P_2 | Q}$$

Meaning: For all P_1, P_2, Q, α , if there is an α transition from P_1 to P_2 then there is an α transition from $P_1 | Q$ to $P_2 | Q$

CCS Transition Rules

(no rule for 0!) **Prefix** $\frac{}{\alpha.P \rightarrow^\alpha P}$ **Def** $\frac{P \rightarrow^\alpha Q}{A \rightarrow^\alpha Q} (A == P)$

Choice_L $\frac{P \rightarrow^\alpha P'}{P+Q \rightarrow^\alpha P'}$ **Choice_R** $\frac{Q \rightarrow^\alpha Q'}{P+Q \rightarrow^\alpha Q'}$

Com_L $\frac{P \rightarrow^\alpha P'}{P|Q \rightarrow^\alpha P'|Q}$ **Com_R** $\frac{Q \rightarrow^\alpha Q'}{P|Q \rightarrow^\alpha P|Q'}$ **Com** $\frac{P \rightarrow^\alpha P' \quad Q \rightarrow^\beta Q'}{P|Q \rightarrow^\alpha P'|Q'}$

Restr $\frac{P \rightarrow^\alpha P'}{P \setminus L \rightarrow^\alpha P' \setminus L} (\alpha, \bar{\alpha} \notin L)$ **Rel** $\frac{P \rightarrow^\alpha P'}{P[f] \rightarrow^\alpha P'[f]}$

CCS Transition Rules, II

Closure assumption: \rightarrow^α is least relation closed under the set of rules

Example derivation:

$$\begin{aligned} \text{Buf}_1 &== \text{in}.\overline{\text{comm}}.\text{Buf}_1 \\ \text{Buf}_2 &== \text{comm}.\overline{\text{out}}.\text{Buf}_2 \\ (\text{Buf}_1 | \text{Buf}_2) \setminus \{\text{comm}\} \\ &\rightarrow^{\text{in}} \overline{\text{comm}}.\text{Buf}_1 | \text{Buf}_2 \\ &\rightarrow^{\tau} \text{Buf}_1 | \overline{\text{out}}.\text{Buf}_2 \\ &\rightarrow^{\text{out}} \text{Buf}_1 | \text{Buf}_2 \end{aligned}$$

Extending the Language

Two ways of adding new operators:

- By equation
Example: Buffer composition

$$P \wedge Q = (P[\text{comm}/\text{out}][Q[\text{comm}/\text{in}]] \setminus \{\text{comm}\})$$


- By extending the transition semantics
Example: Sequential composition of processes
Assume special (non-label) action \checkmark for termination
Let $\checkmark = \checkmark$ and $f(\checkmark) = \checkmark$ for relabelling functions f

$$\text{Seq}_1 \frac{P \rightarrow^\alpha P'}{P;Q \rightarrow^\alpha P';Q} (\alpha \neq \checkmark) \quad \text{Seq}_2 \frac{P \rightarrow^{\checkmark} P' \quad Q \rightarrow^\alpha Q'}{P|Q \rightarrow^\alpha Q'}$$

Example: Semaphores

Semaphore:

Result: $S^1 \mid S^1 \sim S^2$

Unary semaphore: 
 $S^1 == p.S^1_1$
 $S^1_1 == v.S^1$

Proof: Show that
 $((S^1 \mid S^1, S^2),$
 $(S^1_1 \mid S^1, S^2_1),$
 $(S^1 \mid S^1_1, S^2_1),$
 $(S^1_1 \mid S^1_1, S^2_2))$
 is a strong bisimulation relation

Binary semaphore:
 $S^2 == p.S^2_1$
 $S^2_1 == p.S^2_2 + v.S^2$
 $S^2_2 == v.S^2_1$

Example: Simple Protocol

Spec == in.out.Spec

Sender == in.Transmit
 Transmit == $\overline{\text{transmit}}$.WaitAck
 WaitAck == ack_v .Sender + ack_v .Transmit

Receiver == $\overline{\text{transmit}}$.Analyze
 Analyze == τ .out.ack $_v$.Receiver + τ .ack $_v$.Receiver

Protocol == (Sender | Receiver) \ {transmit,ack $_v$,ack}

Exercise: Prove Spec \approx Protocol

Example: Jobshop

i_E : input of easy job
 i_N : input of neutral job
 i_D : input of difficult job
 O: output of finished product

$A == i_E.A' + i_N.A' + i_D.A'$
 $A' == \overline{o}.A$

Spec = A | A

Hammer: H == gh.ph.H
 Mallet: M == gm.pm.M
 Jobber:
 $J == \sum_{x \in \{E, N, D\}} i_x.J_x$
 $J_E == o.J$
 $J_N == \overline{gh}.\overline{ph}.J_E + \overline{gm}.\overline{pm}.J_E$
 $J_D == \overline{gh}.\overline{ph}.J_E$
 Jobshop ==
 $(J \mid J \mid H \mid M) \setminus \{gh, ph, gm, pm\}$

Theorem:
 Spec = Jobshop

Exercise: Prove this.

Proving Equivalences

Two main methods for establishing an equivalence $P \approx Q$:

1. Establish a weak bisimulation relation S s.t. $P S Q$ (this is the canonical method)
2. Use equational reasoning

But: What about substitutivity – replacing equals for equals?

\approx Is Not a Congruence

Congruence: Equivalence preserved under substitution
 But: $P = P'$ does not imply $P + Q = P' + Q$
 Example: $a.0 = \tau.a.0$ but $a.0 + b.0 \neq \tau.a.0 + b.0$ does not hold

Exercise: Show that \approx is preserved by prefixing, parallel, restriction, and relabelling

Observational Congruence:
 Let $S \subseteq Q \times Q$. The relation S is an *observational congruence relation* if whenever $q_1 S q_2$ then:
 - $q_1 \xrightarrow{\alpha} q_1'$ implies $q_2 \Rightarrow \circ \xrightarrow{\alpha} \circ \Rightarrow q_2'$ for some q_2' such that $q_1' \approx q_2'$
 - $q_2 \xrightarrow{\alpha} q_2'$ implies $q_1 \Rightarrow \circ \xrightarrow{\alpha} \circ \Rightarrow q_1'$ for some q_1' such that $q_1' \approx q_2'$

Write $P = Q$ if o.c.r. S exists such that $P S Q$

Observational Congruence

Problem is initial τ 's in sums:

Proposition: $P = P'$ iff for all Q , $P + Q \approx P' + Q$

Exercise: Prove this.

Theorem: Observational congruence = is the largest congruence contained in \approx

Exercise: Prove this too (follow hints in class).

Let $C[_]$ be any CCS expression with a "hole" in it
Corollary: $P = Q$ iff for all $C[_]$, $C[P] \approx C[Q]$

Stable agents: P is stable if no Q exists such that $P \rightarrow^{\tau} Q$

Corollary: \approx and \approx coincides on stable processes

Laws for Observational Congruence

Too many... :-)

Summation:

$$\begin{aligned} P + Q &= Q + P \\ P + (Q + R) &= (P + Q) + R \\ P + P &= P \\ P + 0 &= P \end{aligned}$$

Prefixing:

$$\begin{aligned} l.\tau.P &= l.P \\ P + \tau.P &= \tau.P \\ l.(P + \tau.Q) + l.Q &= l.(P + \tau.Q) \\ P + \tau.(P + Q) &= \tau.(P + Q) \end{aligned}$$

Two **non**-laws:

$$\begin{aligned} P &= \tau.P \\ l.(P + Q) &= l.P + l.Q \end{aligned}$$

Definition:

$$\text{If } A == P \text{ then } A = P$$

Expansion law:

$$\begin{aligned} (P \mid Q) \setminus L &= \\ \Sigma\{l.(P' \mid Q) \setminus L \mid P \rightarrow^l P', l \notin L\} + \\ \Sigma\{l.(P \mid Q') \setminus L \mid Q \rightarrow^l Q', l \notin L\} + \\ \Sigma\{\tau.(P' \mid Q') \setminus L \mid \exists l.P \rightarrow^l P', Q \rightarrow^l Q'\} \end{aligned}$$

Example Derivation

$$\begin{aligned} P &== a.P + \tau.b.0 \\ Q &== a.Q + c.0 \\ R &== \bar{c}.b.0 \\ S &== (Q \mid R) \setminus c \\ \text{Task: Prove } P &= S \end{aligned}$$

$$\begin{aligned} S &= (Q \mid R) \setminus c \\ &= (a.Q + c.0 \mid \bar{c}.b.0) \setminus c \\ &= a.(Q \mid \bar{c}.b.0) \setminus c + \tau.(0 \mid b.0) \setminus c \\ &= a.(Q \mid R) \setminus c + \tau.(0 \mid b.0) \setminus c \\ &= a.S + \tau.b.(0 \mid 0) \setminus c \\ &= a.S + \tau.b.0 \end{aligned}$$

Observe:

$$P = a.P + \tau.b.0$$

and

$$S = a.S + \tau.b.0$$

Can we conclude $P = S$??

Unique Fixed Point Induction

Let $A == P$ be any definition

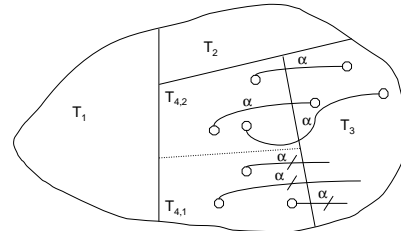
UFI principle: If

- A is guarded in P (every occurrence of A in P in scope of prefix l.-)
- A does not occur in scope of |, \ or [f] in P
- $Q = P[Q/A]$

Then $P = Q$

Partition Refinement

Algorithm for deciding strong bisimulation equivalence



Note: Doesn't work for on-the-fly state generation!

Partition Refinement, II

Q = set of states

create initial partition $\mathcal{P} = \{Q\}$;

change := true;

while change do

 change := false

 if exists partitions $T_1, T_2 \in \mathcal{P}$ and action α such that

$T_{1,1} = \{q \in T_1 \mid \exists q' \in T_2. q \rightarrow^\alpha q'\}$ and $T_{1,2} = T_1 \setminus T_{1,1}$ are both nonempty

 then

$\mathcal{P} := (\mathcal{P} \setminus \{T_1\}) \cup \{T_{1,1}, T_{1,2}\}$;

 change := true

 fi

od

Partition Refinement, Correctness

Let \mathcal{P}_i be partition at step i of algorithm, and $\text{lim}\mathcal{P}$ be the final partition

Define:

1. $p \equiv q$ iff $\exists T \in \text{lim}\mathcal{P}$ such that $p, q \in T$
2. $p \approx_i q$ iff $\exists T \in \mathcal{P}_i$ such that $p, q \in T$

Claim: \equiv is a strong bisimulation relation

Exercise: Check this

Claim: If R is a strong bisimulation relation then $R \subseteq \equiv_i$ for all i

Proof: Induction on i . The statement holds for $i=0$. Suppose that $p R q$ and $p, q \in T \in \mathcal{P}_i$. Either $T \in \mathcal{P}_{i+1}$ as well (and we're done) or $T = T_1 \cup T_2$, $T_1 \cap T_2 = \emptyset$, and $T_1, T_2 \in \mathcal{P}_{i+1}$. In the latter case, if $p \in T_1$ and $q \in T_2$, say, we find some $T' \in \mathcal{P}_i$ and $p' \in T'$ such that $p \rightarrow^\alpha p' \in T'$ and whenever $q \rightarrow^\alpha q'$ then $\text{not}(q' \in T')$. But then $\text{not}(p' R q')$ by the induction hypothesis, so R cannot be a strong bisimulation relation.