Lezione 25 MSC
Espressività: Operatori aggiuntivi (1/2)

Roberto Gorrieri

Espressiveness of CCS

- CCS is Turing-complete: it can compute all the Turing-computable functions. However, as we will see, Turing completeness in concurrency is not everything!
- Can CCS model additional operators of other languages? In many cases, it can!
  - CSP (Hoare): internal choice, hiding, multi-way synchronization
  - CCS (original version): relabeling
  - ACP (Bergstra&Klop): sequential composition, iteration (Kleene star)
  - *(Pi-calcolo (Milner&Parrow&Walker): replication)*

Internal choice

The binary, infix operator $\_ \oplus \_$ of internal choice, originally introduced in TCSP [Hoa85], can be added to the syntax of CCS, to get CCS$^{\oplus}$, as follows:[1]

$$p ::= \mathbf{0} \mid p+p \mid p|p \mid p \oplus p \mid (\nu a)p \mid C$$

with the intuition that $p \oplus q$ can perform the actions prescribed by the following SOS operational rules:

$$(\text{IntChoice}_1)\ \frac{}{p \oplus q \xrightarrow{\tau} p} \qquad (\text{IntChoice}_2)\ \frac{}{p \oplus q \xrightarrow{\tau} q}$$

where the choice between p and q is taken internally, without any intervention of the environment. $\mathscr{P}_{CCS^{\oplus}}$ denotes the set of CCS$^{\oplus}$ processes.

Properties

- Internal choice enjoys a few algebraic properties:

- $p \oplus q \sim q \oplus p$, i.e., commutativity holds for strong bisimilarity;
- $p \oplus (q \oplus r) \approxeq (p \oplus q) \oplus r$, i.e., associativity holds for weak similarity;
- $p \oplus p \approx p$, i.e., idempotence holds for weak bisimilarity;
- $p \oplus \mathbf{0} \approxeq p$, i.e., the law for $\mathbf{0}$ absorption holds for weak similarity.

- But some of these laws (notably, associativity and identity) do not hold for weak bisimilarity!

- Strong and weak bisimilarities are congruences for internal choice. For instance: if p ~ q then p $\oplus$ r ~ q $\oplus$ r

Encoding CCS$^{\oplus}$ into CCS

Internal choice is a derived operator in CCS: given a CCS$^{\oplus}$ process p, we can find a CCS process q such that $p \sim q$. This result is obtained by an encoding $[\![\_]\!]$ from CCS$^{\oplus}$ to CCS, defined as follows:

$$
\begin{aligned}
[\![\mathbf{0}]\!] &= \mathbf{0} \\
[\![\mu.p]\!] &= \mu.[\![p]\!] \\
[\![p_1 + p_2]\!] &= [\![p_1]\!] + [\![p_2]\!] \\
[\![p_1 \mid p_2]\!] &= [\![p_1]\!] \mid [\![p_2]\!] \\
[\![(\nu a)p]\!] &= (\nu a)[\![p]\!] \\
[\![A]\!] &= A' \qquad \text{where } A' \stackrel{def}{=} [\![q]\!] \text{ if } A \stackrel{def}{=} q
\end{aligned}
\qquad\qquad
[\![p_1 \oplus p_2]\!] = \tau.[\![p_1]\!] + \tau.[\![p_2]\!]
$$

where we assume that for each constant A used in CCS$^{\oplus}$ there exists a new constant A' for CCS. Observe that $[\![\_]\!]$ is *homomorphic* w.r.t. all the CCS operators (left column), and it is not only for internal choice (right column).

Example

Example 5.2. Consider process constant $B \stackrel{def}{=} a.A \oplus b.B$, with $A \stackrel{def}{=} c.A \oplus d.B$. The encoding of B, $[\![B]\!]$, gives rise to a new constant $B' \stackrel{def}{=} [\![a.A \oplus b.B]\!] = \tau.a.A' + \tau.b.B'$, with $A' \stackrel{def}{=} \tau.c.A' + \tau.d.B'$. □

Correctness of the encoding

It is an easy task to prove that $R = \{(p, [\![p]\!]) \mid p \text{ is a CCS}^{\oplus} \text{ process}\}$ is a strong bisimulation. One has to prove that $p \xrightarrow{\mu} p'$ implies $[\![p]\!] \xrightarrow{\mu} [\![p']\!]$ and, conversely, that $[\![p]\!] \xrightarrow{\mu} q$ implies there exists p' such that $q = [\![p']\!]$ and $p \xrightarrow{\mu} p'$. This can be proved by induction on the proof of the transitions – starting from the SOS axioms (Pref), (IntChoice_1) and (IntChoice_2) – and is left as an exercise for the reader.

Hiding

- The hiding operator (i a)P is a scoping operator, turning all the actions a and ‘a of P to tau.
- CCS^{hide} is the language:

$$p ::= \mathbf{0} \mid \mu.p \mid p+p \mid p|p \mid (\imath a)p \mid (\nu a)p \mid C$$

$$(\mathrm{H}_1)\ \frac{p \xrightarrow{\mu} p'}{(\imath a)p \xrightarrow{\mu} (\imath a)p'}\ \mu, \bar{\mu} \neq a \qquad (\mathrm{H}_2)\ \frac{p \xrightarrow{\mu} p'}{(\imath a)p \xrightarrow{\tau} (\imath a)p'}\ \mu = a \text{ or } \bar{\mu} = a$$

Algebraic properties for ~

$$
\begin{array}{lll}
(i) & (\iota a)p \sim p & \text{if } a \notin fn(p) \\
(ii) & (\iota a)((\iota b)p) \sim (\iota b)((\iota a)p) & \\
(iii) & (\iota a)(\mu.p) \sim \begin{cases} \tau.(\iota a)p \text{ if } \mu = a \text{ or } \mu = \bar{a} \\ \mu.(\iota a)p \text{ otherwise} \end{cases} & \\
(iv) & (\iota a)(p+q) \sim (\iota a)p + (\iota a)q & \\
(v) & (\iota a)p|q \sim (\iota a)(p|q) & \text{if } a \notin fn(q) \\
(vi) & p|(\iota a)q \sim (\iota a)(p|q) & \text{if } a \notin fn(p) \\
(vii) & (\iota a)p \sim (\iota b)(p\{b/a\}) & \text{if } b \notin fn(p) \cup bn(p) \\
(viii) & (\iota a)p + q \sim (\iota a)(p+q) & \text{if } a \notin fn(q) \\
(ix) & p + (\iota a)q \sim (\iota a)(p+q) & \text{if } a \notin fn(p) \\
(x) & (\iota a)((\nu a)p) \sim (\nu a)p & \\
(xi) & (\nu a)((\iota a)p) \sim (\iota a)p & \\
(xii) & (\iota a)((\nu b)p) \sim (\nu b)((\iota a)p) & \text{if } a \neq b \\
(xiii) & (\iota a)A \sim \begin{cases} A & \text{if } a \notin fn(A) \\ A^a & \text{otherwise, where } A^a \stackrel{def}{=} (\iota a)q \text{ if } A \stackrel{def}{=} q \end{cases} &
\end{array}
$$

Other properties

Exercise 5.5. (Algebraic property for $\approx^c$) Prove that the following algebraic property of the hiding operator holds for (rooted) weak bisimilarity: for all $p, q \in \mathscr{P}_{CCS^{hide}}$

$$(\iota a)(p|q) \approx^c (\iota a)p|(\iota a)q$$ □

Exercise 5.6. (Congruence) Prove that, for all $p, q \in \mathscr{P}_{CCS^{hide}}$, if $p \sim q$, then $(\iota a)p \sim (\iota a)q$ for all $a \in \mathscr{L}$, i.e., strong bisimilarity is a congruence for the hiding operator.

Prove also that if $p \approx q$, then $(\iota a)p \approx (\iota a)q$ for all $a \in \mathscr{L}$, i.e., also weak bisimilarity is a congruence for the hiding operator. Finally, prove that rooted weak bisimilarity is a congruence for the hiding operator. □

Derived operator up to ~

The hiding operator is a *derived operator* in CCS. This means that we can encode any process p in CCS^{hide} into a CCS process q (with no occurrence of the hiding operator) such that $p \sim q$, hence showing that the extension is inessential as it adds no expressive power to CCS. The formal encoding $[\![-]\!]_1$ is defined homomorphically for all the CCS operators (as done in the previous section) and for the hiding operator as follows:

$$[\![(\iota a)p]\!]_1 = (\nu a)([\![p]\!]_1 \,|\, A_a) \qquad \text{where } A_a \stackrel{def}{=} a.A_a + \bar{a}.A_a$$

Example 5.3. For instance, if $B \stackrel{def}{=} a.(\iota a)(b.a.\mathbf{0} \,|\, \bar{a}.B)$, then the corresponding CCS process is $[\![B]\!]_1 = B'$ where $B' \stackrel{def}{=} [\![a.(\iota a)(b.a.\mathbf{0} \,|\, \bar{a}.B)]\!]_1 = a.[\![(\iota a)(b.a.\mathbf{0} \,|\, \bar{a}.B)]\!]_1 = a.(\nu a)([\![b.a.\mathbf{0} \,|\, \bar{a}.B]\!]_1 \,|\, A_a) = a.(\nu a)(b.a.\mathbf{0} \,|\, \bar{a}.B' \,|\, A_a)$. □

Correctness

It is not difficult to prove that relation

$$R_1 = \{(p, [\![p]\!]_1) \mid p \text{ is a CCS}^{hide} \text{ process}\}$$

is a strong bisimulation by induction on the proof of transitions from p and $[\![p]\!]_1$. In particular, take the pair $((\iota a)p, [\![(\iota a)p]\!]_1) \in R_1$. If $(\iota a)p \xrightarrow{\mu} q$ with $\mu \neq a, \bar{a}$, then this can be due only to SOS rule (H_1), with $q = (\iota a)p'$ and $p \xrightarrow{\mu} p'$. By induction, we have that $[\![p]\!]_1 \xrightarrow{\mu} [\![p']\!]_1$ and so $[\![(\iota a)p]\!]_1 = (\nu a)([\![p]\!]_1 | A_a) \xrightarrow{\mu} (\nu a)([\![p']\!]_1 | A_a) = [\![(\iota a)p']\!]_1$, by SOS rules ($Par_1$) and (Res), and the reached states form a pair in R_1. If $(\iota a)p \xrightarrow{\tau} q$, then this is due either to SOS rule (H_1), and this case is as above, or to SOS rule (H_2), with $q = (\iota a)p'$ and $p \xrightarrow{\alpha} p'$, $\alpha = a$ or $\alpha = \bar{a}$. By induction, $[\![p]\!]_1 \xrightarrow{\alpha} [\![p']\!]_1$ and so $[\![(\iota a)p]\!]_1 = (\nu a)([\![p]\!]_1 | A_a) \xrightarrow{\tau} (\nu a)([\![p']\!]_1 | A_a) = [\![(\iota a)p']\!]_1$, by SOS rules (Com) and (Res), and the reached states form a pair in R_1. Symmetrically, if $[\![(\iota a)p]\!]_1$ moves first.

Alternative econding: Derived operator up to $\approx^c$

An alternative way of encoding CCS^{hide} into CCS, up to rooted weak bisimilarity $\approx^c$, is to exploit syntactic substitution (see Definition 4.4 on page 169). The formal encoding $[\![-]\!]_2$ is defined homomorphically for all CCS operators, with the new rule for hiding defined as follows:

$$[\![(\iota a)p]\!]_2 = [\![p]\!]_2\{\tau/a\}$$

- It is not difficult to see that

$$R_2 = \{(p, [\![p]\!]_2) \mid p \text{ is a } \mathrm{CCS}^{hide} \text{ process}\}$$

is a weak bisimulation satisfying the rootedness conditions.

Correctness

- An alternative proof is by inspecting the definition of syntactic substitution and the algebraic properties of hiding. In fact, the definition of syntactic substitution parallels those ones, up to rooted weak bisimilarity:

$$
\begin{array}{rcl@{\qquad}rcl@{\qquad}l}
\mathbf{0}\{\tau/a\} & = & \mathbf{0} & (\imath a)\mathbf{0} & \sim & \mathbf{0} & \\
(a.p)\{\tau/a\} & = & \tau.(p\{\tau/a\}) & (\imath a)((a.p)) & \sim & \tau.((\imath a)p) & \\
(\bar{a}.p)\{\tau/a\} & = & \tau.(p\{\tau/a\}) & (\imath a)((\bar{a}.p)) & \sim & \tau.((\imath a)p) & \\
(\mu.p)\{\tau/a\} & = & \mu.(p\{\tau/a\}) & (\imath a)((\mu.p)) & \sim & \mu.((\imath a)p) & \text{if } \mu \neq a,\bar{a} \\
(p+p')\{\tau/a\} & = & p\{\tau/a\}+p'\{\tau/a\} & (\imath a)(p+p') & \sim & (\imath a)p+(\imath a)p' & \\
(p\,|\,p')\{\tau/a\} & = & p\{\tau/a\}\,|\,p'\{\tau/a\} & (\imath a)(p\,|\,p') & \approx^c & (\imath a)p\,|\,(\imath a)p' & \\
((\nu b)p)\{\tau/a\} & = & (\nu b)(p\{\tau/a\}) & (\imath a)((\nu b)p) & \sim & (\nu b)((\imath a)p) & \text{if } b \neq a \\
((\nu a)p)\{\tau/a\} & = & (\nu a)p & (\imath a)((\nu a)p) & \sim & (\nu a)p &
\end{array}
$$

and for constants,

$$
A\{\tau/a\} = \begin{cases} A & \text{if } a \notin fn(A) \\ A_{\{\tau/a\}} & \text{otherwise, where } A_{\{\tau/a\}} \stackrel{def}{=} q\{\tau/a\} \text{ if } A \stackrel{def}{=} q \end{cases}
$$

$$
(\imath a)A \sim \begin{cases} A & \text{if } a \notin fn(A) \\ A^a & \text{otherwise, where } A^a \stackrel{def}{=} (\imath a)q \text{ if } A \stackrel{def}{=} q \end{cases}
$$

Comparison of the two encodings

The two different encodings differ not only for the equivalence they respect – $\sim$ for $[\![\_]\!]_1$ and $\approx^c$ for $[\![\_]\!]_2$ – but may differ also for the size of generated CCS process, as the following exercise shows.

Exercise 5.7. Consider process constant $C \stackrel{def}{=} (\iota a)(a.C)$. (i) Show that C generates an lts with infinitely many states. (ii) Compute $[\![C]\!]_1$ and show that its lts is infinite as well. (iii) Compute $[\![C]\!]_2$ and show that its lts is finite-state. □

Relabeling

- The relabeling operator P[b/a] is a scoping operator, turning all the actions a of P into b (and ‘a into ‘b).
- CCS^{rel} is the language:

$$p ::= \mathbf{0} \mid \mu.p \mid p+p \mid p|p \mid p[b/a] \mid (\nu a)p \mid C$$

$$(\mathbf{Rel}_1)\ \frac{p \xrightarrow{\mu} p'}{p[b/a] \xrightarrow{\mu} p'[b/a]}\ \mu \neq a \wedge \mu \neq \bar{a}$$

$$(\mathbf{Rel}_2)\ \frac{p \xrightarrow{\alpha} p'}{p[b/a] \xrightarrow{\beta} p'[b/a]}\ (\alpha = a \wedge \beta = b) \vee (\alpha = \bar{a} \wedge \beta = \bar{b})$$

Algebraic properties for ~

$$
\begin{array}{lll}
(i) & p[b/a] \sim p & \text{if } a \notin fn(p) \\
(ii) & (p[b/a])[c/a] \sim p[b/a] & \\
(iii) & (p[b/a])[d/c] \sim (p[d/c])[b/a] & \text{if } a \neq d,\ c \neq b \text{ and } a \neq c \\
(iv) & (\mu.p)[b/a] \sim \begin{cases} b.(p[b/a]) & \text{if } \mu = a \\ \bar{b}.(p[b/a]) & \text{if } \mu = \bar{a} \\ \mu.(p[b/a]) & \text{otherwise} \end{cases} & \\
(v) & (p+q)[b/a] \sim p[b/a] + q[b/a] & \\
(vi) & p \mid q[b/a] \sim (p \mid q)[b/a] & \text{if } a \notin fn(p) \\
(vii) & p[b/a] \mid q \sim (p \mid q)[b/a] & \text{if } a \notin fn(q) \\
(viii) & p[b/a] + q \sim (p+q)[b/a] & \text{if } a \notin fn(q) \\
(ix) & p + q[b/a] \sim (p+q)[b/a] & \text{if } a \notin fn(p) \\
(x) & ((\nu a)p)[b/a] \sim (\nu a)p & \\
(xi) & ((\nu c)p)[b/a] \sim (\nu c)(p[b/a]) & \text{if } c \neq a, b \\
(xii) & ((\nu b)p)[b/a] \sim \begin{cases} (\nu b)p & \text{if } a \notin fn(p) \\ (\nu c)((p\{c/b\})[b/a]) & \text{o.w., with } c \notin fn(p) \cup bn(p) \end{cases} &
\end{array}
$$

Other properties

$$(xiii) \qquad A[b/a] \sim \begin{cases} A & \text{if } a \notin fn(A) \\ A_{[b/a]} & \text{otherwise, where } A_{[b/a]} \stackrel{def}{=} q[b/a] \text{ if } A \stackrel{def}{=} q \end{cases}$$ □

Note that one important law is missing: $(p \,|\, q)[b/a] \sim p[b/a] \,|\, q[b/a]$. Unfortunately, this law is invalid in general. For instance, consider $p = a.\mathbf{0}$ and $q = \bar{b}.\mathbf{0}$. Of course, the set of completed traces of $(p \,|\, q)[b/a]$ is $\{b\bar{b}, \bar{b}b\}$, while the set of completed traces for $p[b/a] \,|\, q[b/a]$ includes additionally trace τ.

Exercise 5.10. Take inspiration from the example above to study sufficient conditions on the sorts of p and q so that $(p \,|\, q)[b/a] \sim p[b/a] \,|\, q[b/a]$ is valid. □

Exercise 5.8. (Congruence) Prove that, for all $p, q \in \mathscr{P}_{CCS^{rel}}$ and for all unary substitutions b/a, if $p \sim q$, then $p[b/a] \sim q[b/a]$, i.e., $\sim$ is a congruence for the relabeling operator. □

Is relabeling a derived operator?

As a consequence, we cannot use an encoding $[\![-]\!]_3$ based on the same idea of $[\![-]\!]_2$, i.e., homomorphically defined for all CCS operators, with the new rule for relabeling defined as follows:

$$[\![p[b/a]]\!]_3 = [\![p]\!]_3\{b/a\}$$

However, such an encoding is correct, up to $\sim$, for CCS subcalculi that are not using parallel composition, as the following example illustrates. And it may be even used for processes of CCS, including parallel composition, when the conditions to be studied in Exercise 5.10 are satisfied.

Full relabeling

- A relabeling is a f: L → L, where L is the set of visible actions (inputs), extended to Act as follows: f(‘a) = ‘f(a) and f(tau) = tau.

$$\text{(Rel)}\ \frac{p \xrightarrow{\mu} p'}{p[f] \xrightarrow{f(\mu)} p'[f]}$$

This new relabeling operator is more powerful than the basic version we have described above, as it may be used to generate a process that is not bisimulation equivalent to any finitary CCS process. Indeed, assume that $Act = \{a_i \mid i \in \mathbb{N}\}$ and consider f defined thus: $f(a_i) = a_{i+1}$ for all $i \in \mathbb{N}$. The constant $A \stackrel{def}{=} a_0.(A[f])$ gives rise to an lts with infinitely many states:

$$A \xrightarrow{a_0} A[f] \xrightarrow{a_1} (A[f])[f] \xrightarrow{a_2} ((A[f])[f])[f] \xrightarrow{a_3} \ldots.$$

Note that A is not equivalent to any finitary CCS process p, as $sort(A) = Act$, while $sort(p)$ is always a finite set (see Corollary 4.1 on page 167). Hence, finitary CCS with the (full) relabeling operator is more powerful than finitary CCS.