Lezione 15 MSC
Proprietà algebriche

Roberto Gorrieri

Proprietà algebriche delle equivalenze comportamentali

- Valgono alcune leggi interessanti (ovvero proprietà algebriche) che sono utili in vari contesti, ad esempio quando si vuole costruire una bisimulazione up to ~/≈.
- Tipiche leggi sono quelle monoidali (associatività, elemento neutro), più commutatività, ma anche altre interessanti.

Proprietà di ~ (1)

- L'operatore + forma un monoide commutativo con identità **0**, ed è idempotente:

$$
\begin{array}{rcll}
p + (q + r) & \sim & (p + q) + r & \textit{(associativity)} \\
p + q & \sim & q + p & \textit{(commutativity)} \\
p + \mathbf{0} & \sim & p & \textit{(neutral element)} \\
p + p & \sim & p & \textit{(idempotency)}
\end{array}
$$

- Dimostrazione facile. Ad esempio una bisimulazione che giustifica la correttezza dell'ultima legge è:

$$R = \{(p + p, p) \mid p \in \mathscr{P}\} \cup \{(q, q) \mid q \in \mathscr{P}\}$$

Esercizio

- Obbligatorio: Trova una bisimulazione che giustifica l'associatività del +. Ripeti l'esercizio per la commutatività e il **0**-assorbimento.
- Osservazione: queste proprietà giustificano la scrittura $\sum_{1\leq i\leq n} p_i$ che a volte usiamo.
- Osservazione 2: queste leggi valgono ovviamente anche per tutte le equivalenze più deboli di bisimulazione (e.g., trace equivalence).
- Esercizio: queste leggi valgono per l'isomorfismo?

Distributività del prefisso rispetto alla somma

- Questa legge non vale per bisimulazione forte

$$\mu.(p+q) \not\sim \mu.p+\mu.q$$

e neanche per simulation equivalence, ma vale per trace equivalence!

Exercise 4.3. Check if $\mu.(p+q) \sim \mu.(p+q)+\mu.q$. Verify that this law holds for simulation equivalence $\simeq$. $\square$

Proprietà di ~ (2)

- L'operatore | forma un monoide commutativo con **0** come elemento neutro:

$$p \mid (q \mid r) \sim (p \mid q) \mid r$$
$$p \mid q \sim q \mid p$$
$$p \mid \mathbf{0} \sim p$$

- Dimostrazione facile. Ad esempio

$$R = \{(p \mid \mathbf{0}, p) \;\big|\; p \in \mathscr{P}\}$$

è una bisimulazione che dimostra la legge del nil.

- Esercizio obbligatorio: trova una bisimulazione che dimostra associatività (commutatività).

Osservazioni

- L'idempotenza $p|p \sim p$ non vale (ad es. $a.0|a.0$ non è bisimile ad $a.0$). Talvolta vale, ad es. se per p prendiamo $A = a.A$: $A|A \sim A$
- Queste proprietà del parallelo giustificano la scrittura $\prod_{1 \leq i \leq n} p_i$ che a volte usiamo.
- Queste leggi valgono ovviamente anche per tutte le equivalenze più deboli di bisimulazione (e.g., trace equiv.), ma anche per l'isomorfismo!
- Legge per la costante: $C \sim p$ se $C = p$. Questa legge non vale per l'isomorfismo! Prendiamo $A = a.A$: osserva che l'lts per A e per $a.A$ non sono isomorfi.

Expansion/Interleaving Law

- Legge che spiega come trasformare il parallelo di due processi in una somma. Ad es. a.0|b.0 risulta equivalente a a.b.0+b.a.0

Proposition 4.3. (Expansion Law) *Let* $p = \Sigma_{i=1}^{n} \mu_i.p_i$ *and* $q = \Sigma_{j=1}^{m} \mu'_j.q_j$. *Then*

$$p|q \sim \Sigma_{i=1}^{n} \mu_i.(p_i|q) + \Sigma_{j=1}^{m} \mu'_j.(p|q_j) + \Sigma_{\overline{\mu_i}=\mu'_j} \tau.(p_i|q_j)$$

Proof. *It follows immediately from the following observation: let* T_r *be the (finite) set of all transitions outgoing from* r, *i.e.* $\{(r, \mu_k, r_k) \in \rightarrow \ | \ \mu_k \in Act, r_k \in \mathscr{P}\}$; *then* $r \sim \Sigma_{(r,\mu_k,r_k) \in T_r} \mu_k.r_k$. □

- Osserva che anche le sincronizzazioni sono "interleavings"

Distributività del parallelo rispetto alla somma

Esercizio obbligatorio: dimostra che la distributività del parallelo rispetto alla somma non vale per bisimulation equivalence:

$(p+q) | r$ non è bisimile a $p|r + q|r$

Ma tale legge vale per trace equivalence!

Proprietà di ~ (3)

- L'operatore di restrizione possiede alcune proprietà:

$$
\begin{array}{ll}
(i) & (\nu a)\mathbf{0} \sim \mathbf{0} \\
(ii) & (\nu a)((\nu b)p) \sim (\nu b)((\nu a)p) \;\; \textit{if } a \neq b \\
(iii) & (\nu a)((\nu a)p) \sim (\nu a)p \\
(iv) & (\nu a)(\mu.p) \sim \begin{cases} \mathbf{0} & \textit{if } \mu = a \textit{ or } \mu = \bar{a} \\ \mu.(\nu a)p & \textit{otherwise} \end{cases}
\end{array}
$$

- La seconda e terza legge giustificano la notazione
  (v L)p dove L è un insieme di azioni
- Altra legge (su un'estensione sintattica di CCS):

$$(\nu a)(p+q) \sim (\nu a)p + (\nu a)q$$

Proprietà di ~ (4)

- Altre leggi interessanti per restrizione:

(i) $(\nu a)p \sim p$	*if* $a \notin fn(p)$		
(ii) $(\nu a)p\,|\,q \sim (\nu a)(p\,|\,q)$	*if* $a \notin fn(q)$	*(scope-enlargement$_1$)*	
(iii) $p\,|\,(\nu a)q \sim (\nu a)(p\,|\,q)$	*if* $a \notin fn(p)$	*(scope-enlargement$_2$)*	
(iv) $(\nu a)p \sim (\nu b)(p\{b/a\})$	*if* $b \notin fn(p) \cup bn(p)$	*(alpha-conversion)*	

- Ma dobbiamo definire cosa sono i free names fn(p), i bound names bn(p), e la sostituzione sintattica {b/a} tra nomi di azioni.

Free names fn(p)

Definition 4.1. (Free names) The *free names* of a process p, denoted $fn(p)$, are defined as the set $F(p,\emptyset)$, where $F(p,I)$, with I a set of process constants, is defined as follows:

$$
\begin{aligned}
F(\mathbf{0},I) &= \emptyset \\
F(a.p,I) = F(\bar{a}.p,I) &= F(p,I) \cup \{a\} \\
F(\tau.p,I) &= F(p,I) \\
F(p+q,I) = F(p|q,I) &= F(p,I) \cup F(q,I) \\
F((\nu a)p,I) &= F(p,I) \setminus \{a\} \\
F(C,I) &= \begin{cases} F(q, I \cup \{C\}) & \text{if } C \stackrel{def}{=} q \text{ and } C \notin I \\ \emptyset & \text{if } C \in I \end{cases}
\end{aligned}
$$

□

According to this definition, $fn(p)$ is effectively computable only for those processes p such that the set *Const(p)* of the constants used in p is finite, i.e., for processes in finitary CCS.

Free names (2)

- Proposizione 1:

per ogni p in Finitary CCS, l'insieme fn(p) è finito.

- Proposizione 2:

se p-μ->p', allora fn(p') $\subseteq$ fn(p) e se $\mu \in$ {a,'a} allora a $\in$ fn(p)

Conseguenza: sort(p) $\subseteq$ fn(p) U 'fn(p) U {tau}, quindi tale insieme è un superset decidibile di sort(p) per tutti i p in Finitary CCS.

Bound names bn(p)

Definition 4.2. (Bound names) The *bound names* of a process p, denoted $bn(p)$, are defined as the set $B(p,\emptyset)$, where $B(p,I)$, with I a set of process constants, is defined as follows:

$$
\begin{aligned}
B(\mathbf{0},I) &= \emptyset \\
B(\mu.p,I) &= B(p,I) \\
B(p+q,I) = B(p\,|\,q,I) &= B(p,I) \cup B(q,I) \\
B((\nu a)p,I) &= B(p,I) \cup \{a\} \\
B(C,I) &= \begin{cases} B(q, I \cup \{C\}) & \text{if } C \stackrel{def}{=} q \text{ and } C \notin I \\ \emptyset & \text{if } C \in I \end{cases}
\end{aligned}
$$

Exercise 4.11. Compute the sets of bound and free names of the following CCS processes:

$$(\nu a)(a.b\,|\,\overline{a})\,|\,a \qquad b.d\,|\,(\nu a)(a.c) \qquad (\nu a)(b\,|\,(\nu a)(b.\overline{a}))$$

Substitution

Definition 4.3. (Substitutions) A substitution is a set $\{b_i/a_i\}_{i\in I}$ of associations of the form b_i/a_i for $i \in I$, meaning that action $a_i \in \mathscr{L}$ is to be replaced by action $b_i \in \mathscr{L}$, when applied to some term. A substitution $\{b_i/a_i\}_{i\in I}$ is *admissible* when $a_i \neq a_j$ for all $i \neq j$ and $b_i \neq a_j$ for all i, j. Hence, for instance, the following are not admissible: $\{a/a\}$, $\{b/a, c/a\}$ and $\{b/a, a/b\}$. On the contrary, $\{b/a\}$, $\{b/a, b/c\}$ and $\{b/a, d/c\}$ are admissible. We use θ to range over the set of admissible substitutions.

A substitution $\{b_i/a_i\}_{i\in I}$ is *empty*, denoted by ε, if $|I| = 0$, i.e., there is no association in the set. Of course, the empty substitution ε is admissible. A substitution $\{b_i/a_i\}_{i\in I}$ is *unary* if $|I| = 1$, i.e., it is of the form $\{b/a\}$. A unary substitution $\{b/a\}$ is admissible provided that $a \neq b$.

The composition of an admissible substitution $\theta = \{b_1/a_1, \ldots, b_n/a_n\}$ with a unary admissible substitution $\{b/a\}$, denoted by $\theta \circ \{b/a\}$, is defined — provided that $b \neq a_j$ for all $j = 1, \ldots, n$ — as follows:

- $\{b_1/a_1, \ldots, b_n/a_n\}$ if there exists an index j such that $a = a_j$;
- $\{b'_1/a_1, \ldots, b'_n/a_n, b/a\}$ if $a \neq a_j$ for all $j = 1, \ldots, n$, where $b'_i = b_i$ if $a \neq b_i$, otherwise $b'_i = b$, for $i = 1, \ldots, n$.

Substitution (2)

Note that $\theta \circ \{b/a\}$, if defined, is an admissible substitution. Examples of substitution composition are the following: $\varepsilon \circ \{b/a\} = \{b/a\}$, $\{b/a\} \circ \{b/a\} = \{b/a\}$ and $\{b/a, d/c\} \circ \{d/b\} = \{d/a, d/c, d/b\}$.

A nonemtpy admissible substitution $\theta = \{b_1/a_1, \ldots, b_n/a_n\}$ can be represented as the composition of the unary admissible substitution $\{b_1/a_1\}$ and of the admissible substitution $\theta' = \{b_2/a_2, \ldots, b_n/a_n\}$, i.e., $\theta = \{b_1/a_1\} \circ \theta'$. $\square$

Costanti parametrizzate da sostituzioni

- Per uniformità di definizione, assumiamo che le costanti siano indicizzate da una sostituzione ammissibile; ad esempio, A_θ
- A_ε sta per A: all'inizio la sostituzione vuota ε.
- L'applicazione $A_\theta\{b/a\}$ di una sostituzione unaria $\{b/a\}$ ad A_θ genera una nuova costante con indice $\theta \circ \{b/a\}$, cioè, $A_{\theta \circ \{b/a\}}$.

Definition 4.4. (Syntactic Substitution) The syntactic substitution $p\{b/a\}$ of action b for a different action a inside a CCS process p is defined as follows:

$$
\begin{aligned}
\mathbf{0}\{b/a\} &= \mathbf{0} \\
(a.p)\{b/a\} &= b.(p\{b/a\}) \\
(\bar{a}.p)\{b/a\} &= \bar{b}.(p\{b/a\}) \\
(\mu.p)\{b/a\} &= \mu.(p\{b/a\}) \qquad \text{if } \mu \neq a, \bar{a} \\
(p+q)\{b/a\} &= p\{b/a\} + q\{b/a\} \\
(p\,|\,q)\{b/a\} &= p\{b/a\}\,|\,q\{b/a\} \\
((\nu c)p)\{b/a\} &= (\nu c)(p\{b/a\}) \qquad \text{if } c \neq a, b \\
((\nu a)p)\{b/a\} &= (\nu a)p \\
((\nu b)p)\{b/a\} &= \begin{cases} (\nu b)p & \text{if } a \notin fn(p) \\ (\nu c)((p\{c/b\})\{b/a\}) & \text{otherwise, with } c \notin fn(p) \cup bn(p) \end{cases} \\
C_\theta\{b/a\} &= \begin{cases} C_\theta & \text{if } a \notin fn(C_\theta) \\ C_{\theta \circ \{b/a\}} & \text{otherwise, with } C_{\theta \circ \{b/a\}} \stackrel{def}{=} q\{b/a\} \text{ if } C_\theta \stackrel{def}{=} q \end{cases}
\end{aligned}
$$

The application of an admissible substitution $\theta = \{b/a\} \circ \theta'$ to a process p can be computed as follows: $p\theta = (p\{b/a\})\theta'$, with the proviso that $p\varepsilon = p$. □

Esempio/Esercizi

Example 4.2. Let us consider again Example 3.9, where we have defined a pipelined buffer $Buf \stackrel{def}{=} B \frown B$, where $B \stackrel{def}{=} in.\overline{out}.B$ is the one-position buffer of Exercise 3.35 and the linking connects the *out* port of the left buffer to the *in* port of the right buffer. As a matter of fact, the explicit definition of Buf is

$$Buf \stackrel{def}{=} (\nu d)(B\{d/out\} \mid B\{d/in\})$$

and the effect of applying the substitution to B is the definition of a new constant where the substitution is applied to its body: $B_{\{d/out\}} \stackrel{def}{=} in.\overline{d}.B_{\{d/out\}}$ and $B_{\{d/in\}} \stackrel{def}{=} d.\overline{out}.B_{\{d/in\}}$. □

Exercise 4.7. Let us consider constants $A \stackrel{def}{=} a.A + b.B$ and $B \stackrel{def}{=} c.d.A + a.B$. Suppose we want to replace action a with action c. Compute the result of $A\{c/a\}$. □

Exercise 4.8. Consider $A \stackrel{def}{=} (\nu a)(a.b.A \mid \overline{a}.c.A)$. Compute $A\{b/a\}$ and $A\{a/b\}$. □

Proprietà di $\approx$

- Legge fondamentale di weak bisimilarity

  $$\tau.p \approx p$$

  che vale anche weak trace equivalence $=_{wtr}$

  (ma che non vale per rooted weak bis $\approx^c$)

- Osserva che $p \approx q$ sse $\mu.p \approx^c \mu.q$ per ogni μ

Proprietà di $\approx^c$

- Le tre leggi del tau

$$\mu.\tau.p \approx^c \mu.p$$

$$p+\tau.p \approx^c \tau.p$$

$$\mu.(p+\tau.q) \approx^c \mu.(p+\tau.q)+\mu.q$$

- La prima vale perché τ.p ≈ p e per l'osservazione del lucido precedente.
- La seconda e la terza dimostratele per esercizio.

Exercise 4.11. (*i*) Argue that $\mu.(p+\tau.q) \approx^c \mu.(p+q)+\mu.q$ is not valid. Find a suitable weak equivalence for which it holds. (*ii*) Argue that also $\mu.(p+q) \approx^c \mu.(\tau.p+\tau.q)$ is not valid as well. □

Exercise 4.12. (*i*) Prove that $p \,|\, \tau.q \approx p \,|\, q$. (*ii*) Show that $p \,|\, \tau.q \not\approx^c p \,|\, q$. (*iii*) Prove also that $p \,|\, \tau.q \approx^c \tau.(p \,|\, q)$. □

Lemma di Hennessy

Lemma 4.1. (Hennessy Lemma)*[Mil89]*
For any processes p and q, $p \approx q$ if and only if ($p \approx^c q$ or $p \approx^c \tau.q$ or $\tau.p \approx^c q$).

Proof. $\Leftarrow$) We have three cases: (i) If $p \approx^c q$, then $p \approx q$ by Exercise 2.76. (ii) If $p \approx^c \tau.q$, then $p \approx \tau.q \approx q$. (iii) Symmetric to the previous one.

$\Rightarrow$) We assume that $p \approx q$. We consider three cases: (i) If there exists p' such that $p \xrightarrow{\tau} p' \approx q$, then it is easy to observe that $p \approx^c \tau.q$. As a matter of fact, in one direction, if $\tau.q \xrightarrow{\tau} q$, then $p \xrightarrow{\tau} p'$ with $p' \approx q$, as required. In the other direction, if $p \xrightarrow{\tau} p'$ with $p' \approx q$, then $\tau.q \xrightarrow{\tau} q$, as required; if $p \xrightarrow{\tau} p''$ with $p'' \not\approx q$, then $\tau.q \xrightarrow{\tau} q \stackrel{\varepsilon}{\Longrightarrow} q''$ with $p'' \approx q''$, because $p \approx q$; finally, if $p \xrightarrow{\alpha} p''$, then $\tau.q \xrightarrow{\tau} q \stackrel{\alpha}{\Longrightarrow} q''$ with $p'' \approx q''$, because $p \approx q$; hence the rooted weak bisimulation condition is respected also in this direction. (ii) Symmetrically to the above case, if $q \xrightarrow{\tau} q' \approx p$, then $\tau.p \approx^c q$. (iii) If neither of the above two holds, then we can show that $p \approx^c q$ as follows. If $p \xrightarrow{\alpha} p'$, then $q \stackrel{\alpha}{\Longrightarrow} q'$, with $p' \approx q'$, and the definition of rooted weak bisimilarity $\approx^c$ is respected. If $p \xrightarrow{\tau} p'$, then $q \stackrel{\varepsilon}{\Longrightarrow} q'$, with $p' \approx q'$; note that q' cannot be q itself, otherwise we would be in case (i) above, This means that $q \stackrel{\tau}{\Longrightarrow} q'$ and the definition of rooted weak bisimilarity $\approx^c$ is respected. Symmetrically, if q moves first. $\square$