3. Algebraic Properties of Bisimilarity

Lecture on Models of Concurrent Systems

(Summer 2022)

Stephan Mennicke

Apr 20-27, 2022

Calculi of Communicating Systems

Let $\mathcal{N}$ be a set names with $\tau \notin \mathcal{N}$. Define $\overline{\mathcal{N}} = \{\overline{a} \mid a \in \mathcal{N}\}$.

Definition 3.1: CCS is the process language defined over the set of actions $Act = \mathcal{N} \cup \overline{\mathcal{N}} \cup \{\tau\}$ and Pr defined by the following grammar:

$$P \quad ::= \quad \mathbf{0} \;\Big|\; \alpha.P \;\Big|\; P + P \;\Big|\; P \parallel P \;\Big|\; (\nu a)\,(P) \;\Big|\; K$$

where $\alpha \in Act$, $a \in \mathcal{N}$, and K is a constant from the set of constants $\mathbb{C}$.

Constants ($\mathbb{C}$) are accompanied with a **constant transition relation** $\mathcal{T}_{\mathbb{C}} \subseteq \mathbb{C} \times Act \times Pr$. Denote by $\mathbf{CCS}(Act, \mathbb{C}, \mathcal{T}_{\mathbb{C}})$ the CCS over set of actions Act and constants $\mathbb{C}$ with constant transition relation $\mathcal{T}_{\mathbb{C}}$.

Definition 3.2: The semantics of $\mathbf{CCS}(Act, \mathbb{C}, \mathcal{T}_{\mathbb{C}})$ is the labeled transition system $(Pr, Act, \rightarrow)$ where $\rightarrow$ is the smallest transition relation satisfying (1) $\mathcal{T}_{\mathbb{C}} \subseteq \rightarrow$ and (2) the rules ($\star$).

CCS Transition Rules ($\star$)

$$P \quad ::= \quad \mathbf{0} \;\Big|\; \alpha.P \;\Big|\; P + P \;\Big|\; P \parallel P \;\Big|\; (\nu a)\,(P) \;\Big|\; K$$

$$\text{(Pref)}\ \frac{}{\alpha.P \xrightarrow{\alpha} P} \qquad \text{(SumL)}\ \frac{P \xrightarrow{\alpha} P'}{P + Q \xrightarrow{\alpha} P'} \qquad \text{(SumR)}\ \frac{Q \xrightarrow{\alpha} Q'}{P + Q \xrightarrow{\alpha} Q'}$$

$$\text{(ParL)}\ \frac{P \xrightarrow{\alpha} P'}{P \parallel Q \xrightarrow{\alpha} P' \parallel Q} \qquad \text{(ParR)}\ \frac{Q \xrightarrow{\alpha} Q'}{P \parallel Q \xrightarrow{\alpha} P \parallel Q'} \qquad \text{(Com)}\ \frac{P \xrightarrow{\alpha} P' \quad Q \xrightarrow{\overline{\alpha}} Q'}{P \parallel Q \xrightarrow{\tau} P' \parallel Q'}$$

$$\text{(Res)}\ \frac{P \xrightarrow{\alpha} P'}{\nu a.P \xrightarrow{\alpha} \nu a.P'}\ \text{if } a \notin \{\alpha, \overline{\alpha}\}$$

In what follows, we always assume **equality of processes** up to bisimilarity ($\leftrightarroweq$) if not stated otherwise.

Some (Algebraic) Properties

Theorem 3.3: For each finite process $P \in \mathbf{CCS}(Act, \mathbb{C}, \mathcal{T}_{\mathbb{C}})$, there is a process $P' \in \mathbf{CCS}(Act, \emptyset, \emptyset)$ with $P \cong P'$. We call $\mathbf{CCS}(Act, \emptyset, \emptyset)$ **finCCS**.

Theorem 3.4: Parallel composition and choice are commutative and associative and have $\mathbf{0}$ as neutral element. Furthermore, choice is idempotent.

Hence, we may write for indexed processes $P_1, P_2, \ldots, P_n$:

$$\sum_{i=1}^{n} P_i = P_1 + P_2 + \ldots + P_n$$

$$\prod_{i=1}^{n} P_i = P_1 \parallel P_2 \parallel \ldots \parallel P_n$$

Theorem 3.5: Let $\Omega_\alpha \in \mathbb{C}$ with $\Omega_\alpha \xrightarrow{\alpha} \Omega_\alpha$. For all $n \geq 1$, $\prod_{i=1}^{n} \Omega_\alpha \leftrightarrows \Omega_\alpha$.

Expressivity of CCS

Theorem 3.6: There are Act, $\mathbb{C}$, and $\mathcal{T}_{\mathbb{C}}$, so that **CCS**$(Act, \mathbb{C}, \mathcal{T}_{\mathbb{C}})$ is Turing-complete.

$\rightsquigarrow$ bisimilarity of CCS processes is undecidable.

Proof Plan:

1. Pick a Turing-complete model $\rightsquigarrow$ Minsky machines
2. Encode computations by means of **CCS** using only finitely many actions, constants, and a finite constant transition relation per Minsky machine

1. Minsky Machine (or Counter Machine)

Definition 3.7: A **Minsky machine** is a pair $\mathcal{M} = (R, P)$, where $R = \{c_1, c_2, \ldots, c_n\}$ is a finite set of counters (or registers) and $P = \{l_0, l_1, \ldots, l_m\}$ is a finite set of **instructions** l_i ($i = 0, 1, \ldots, m$) over $\mathcal{M}$, such that $l_i = \langle X_i, \texttt{inc}\ k : j \rangle$, $l_i = \langle X_i, \texttt{dec}\ k : j : j' \rangle$, and $l_m = \texttt{halt}$, where $i, j, j' \in \{0, 1, \ldots, m\}$ are line indizes and $k \in \{1, \ldots, n\}$ are counter indizes.

Definition 3.8: For Minsky machine $\mathcal{M} = (R, P)$ we call a pair $\langle i, \beta \rangle$ a **configuration of** $\mathcal{M}$ if $l_i \in P$ and $\beta : R \to \mathbb{N}$. A configuration $\langle 0, \beta \rangle$ is called an initial configuration. Define a step of $\mathcal{M}$ by $\langle i, \beta \rangle \triangleright \langle j, \beta' \rangle$ if, and only if, (1) $l_i = \langle X_i, \texttt{inc}\ k : j \rangle$ and $\beta' = \beta[c_k \mapsto \beta(c_k) + 1]$, (2) $l_i = \langle X_i, \texttt{dec}\ k : j : j' \rangle$, $\beta(c_k) > 0$ and $\beta' = \beta[c_k \mapsto \beta(c_k) - 1]$, and (3) $l_i = \langle X_j, \texttt{dec}\ k : j' : j \rangle$ and $\beta(c_k) = 0$.

1. Minsky and Turing

The **Halting Problem for Minsky Machines** is the language

$$\mathbf{L}_{\mathsf{HALT}} := \{\langle \mathcal{M}, \beta \rangle \mid \exists n \in \mathbb{N} : \langle 0, \beta \rangle \triangleright^* \langle n, \mathtt{halt} \rangle\}.$$

> $\mathbf{L}_{\mathsf{HALT}}$ is undecidable, even if only two counters are used.

Proof idea: Simulate a Turing machine (wlog, binary tape alphabet) as follows: Cut the tape at the TM head in two halfs, resulting in two tapes that are bounded at one of their sides. Read the binary string from left (or right) end of the tape and encode it as a natural number assigned to counters c_1 and c_2. Manipulation of the tape is deferred to manipulation of the numbers stored in the counters. Furthermore, left- or right-head movement is implemented by shifting bits between the registers.

> Minsky Machines are Turing-complete.

2. Implementing Minsky Machines in CCS

Construction: in two steps.

1. Implementing unbounded counters using finitely many actions and constants;
2. Implementing the program instructions

We do the second step first. As an interface to the counters c_1 and c_2, we assume action names $\overline{u^1}, \overline{d^1}, z^1$ to control the first counter and $\overline{u^2}, \overline{d^1}, z^2$ for the second. For each $l_i \in P$, $X_i \in \mathbb{C}$, which we translate using the following theme (assuming $k \in \{1, 2\}$):

1. $\langle X_i, \texttt{inc}\ k\ :\ j \rangle \mapsto X_i$ with $X_i \xrightarrow{\overline{u^k}} X_j$;
2. $\langle X_i, \texttt{dec}\ k\ :\ j\ :\ j' \rangle \mapsto X_i$ with $X_i \xrightarrow{\overline{d^k}} X_j$ and $X_i \xrightarrow{z^k} X_{j'}$;
3. $\langle X_i, \texttt{halt} \rangle \mapsto X_i$ with $X_i \xrightarrow{h} \mathbf{0}$.

2.1 Implementing Counters

A single counter may be realized using constants $C, C_1, C_2 \in \mathbb{C}$ and actions $u, d, \overline{z} \in Act$.

1. Define $C \xrightarrow{\overline{z}} C$ and $C \xrightarrow{u} (\nu a)\,(C_1 \parallel a.C)$;
2. Define $C_1 \xrightarrow{d} \overline{a}.\mathbf{0}$ and $C_1 \xrightarrow{u} (\nu b)\,(C_2 \parallel b.C_1)$;
3. Define $C_2 \xrightarrow{d} \overline{b}.\mathbf{0}$ and $C_2 \xrightarrow{u} (\nu a)\,(C_1 \parallel a.C_2)$.

For any process P, reachable from C, define $val(P)$ inductively:

Base: $val(P) = 0$ if $P = C$.

Step: For process Q with $val(Q) = n$ $(n > 0)$, $val(Q') = n + 1$ if $Q \xrightarrow{u} Q'$ and $val(Q') = n - 1$ if $Q \xrightarrow{d} \cdot \xrightarrow{\tau} Q'$.

For two processes P and Q, reachable from C, we get $val(P) = val(Q)$ iff $P \leftrightarrows Q$.

Putting Everything Together

Let $\mathcal{M} = (R, P)$ be a Minsky machine with $R = \{c_1, c_2\}$ and $P = \{l_0, l_1, \ldots, l_n\}$.

Our construction uses $Act = \{u^1, d^1, z^1, u^2, d^2, z^2, \tau, \overline{u^1}, \overline{d^1}, \overline{z^1}, \overline{u^2}, \overline{d^2}, \overline{z^2}\}$ and $\mathbb{C} = \{C_1^1, C_2^1, C^1, C_1^2, C_2^2, C^2, X_0, X_1, \ldots, X_n\}$, where n is the maximal line index of P. $\mathcal{T}_{\mathbb{C}}$ defined as before.

For $\beta_0 = \{c_1 \mapsto 0, c_2 \mapsto 0\}$, $\langle 0, \beta_0 \rangle \triangleright^* \langle i, \beta \rangle$ with $\beta(c_1) = n_1$ and $\beta(c_2) = n_2$, we get $(\nu u^1, u^2, d^1, d^2, z^1, z^2)\,(X_0 \parallel C^1 \parallel C^2) \xrightarrow{\tau}{}^* (\nu u^1, u^2, d^1, d^2, z^1, z^2)\,(X_i \parallel \underline{C^1} \parallel \underline{C^2})$ such that $val(\underline{C_1}) = n_1$ and $val(\underline{C_2}) = n_2$.

$\rightsquigarrow$ halting problem for CCS is undecidable.

Bisimilarity is a Congruence for CCS

In a previous version I explained the whole subject of this slide using the notion of sub-processes. In retrospect, this is misleading overhead. So, I got rid of it.

A **CCS context** is
a **CCS** process C using single special process constant $\bullet \in \mathbb{C}$ (called a hole), also denoted as $C[\bullet]$.

Theorem 3.9: For all processes P, Q of **CCS** with $P \leftrightarrows Q$ and all CCS contexts $C[\bullet]$, $C[P] \leftrightarrows C[Q]$.

$\rightsquigarrow$ Bisimilarity is a congruence for **CCS**.

What happens if $\bullet$ occurs more than once? What happens if we use finitely many holes (i. e., $\bullet_1, \ldots, \bullet_n \in \mathbb{C}$)?