Formal Techniques for Software Engineering: Process Calculi

Rocco De Nicola

IMT Institute for Advanced Studies, Lucca
rocco.denicola@imtlucca.it

June 2013

Lesson 9

Process Algebras

What is a process algebra

- A set of terms
- An Operational Semantics associating LTs's to terms
- An Equivalence relations equating terms exhibiting "similar" behavior

Set of Operators

- Basic Processes
- Sequentialization, Choice
- Parallel Composition, Abstraction
- Recursion

Equivalences

- Trace, Testing, Bisimulation Equivalences
- ... many others ...
- Variants taking into account that some actions are unobservable

CCS: Calculus of Communicating Processes

Milner - 1980

The set of actions Act_τ consists of a set of labels Λ, of the set $\overline{\Lambda}$ of complementary labels and of the distinct action τ, the syntax is

$$E \ ::= \ nil \ \big| \ X \ \big| \ \mu.E \ \big| \ E\backslash L \ \big| \ E[f] \ \big| \ E_1 + E_2 \ \big| \ E_1|E_2 \ \big| \ recX.E$$

Moreover we have:

- $\mu \in Act_\tau$;
- $L \subseteq \Lambda$;
- $f : Act_\tau \rightarrow Act_\tau$;
- $f(\bar{\alpha}) = \overline{f(\alpha)}$ and $f(\tau) = \tau$.

CCS has been studied with Bisimulation and Testing Semantics

SCCS: Synchronous Calculus of Communicating Processes

Milner - 1983

The set of actions *Act* is an Abelian group containing a set of labels Λ, and of complementary actions $\overline{\Lambda}$ with over-dashed actions, the neutral element is 1, the syntax is

$$E \ ::= \ nil \ \Big| \ X \ \Big| \ \mu : E \ \Big| \ E \upharpoonright L \ \Big| \ E_1 + E_2 \ \Big| \ E_1 \times E_2 \ \Big| \ recX.E$$

where

- $\mu \in Act \cup \{1\}$,
- $L \subseteq \Lambda$,
- : denotes action prefixing

There is no relabelling operator, it is expressible via the other operators.

SCCS has been studied with Bisimulation Semantics

LOTOS: Language of Temporal Order Specification

Standard ISO - 1988

The set of actions Λ_i contains a set of labels Λ and the distinct label i, the syntax is

$$\begin{array}{rcl} E & ::= & \mathit{stop} \mid \mathit{exit} \mid \mu; E \mid E/L \mid E[f] \mid E_1 \gg E_2 \mid E_1 \, [> E_2 \\ & & \mid E_1 + E_2 \mid E_1 \parallel E_2 \mid E_1 \interleave E_2 \mid E_1 \, |[L]| \, E_2 \mid A \end{array}$$

- $\mu \in \Lambda_i$, $L \subseteq \Lambda$, $f : \Lambda \to \Lambda$;
- the operator ; denotes action prefixing;
- the operator $\gg$ denotes sequential composition;
- A is a process constant.

LOTOS has been studied with Bisimulation and Testing Semantics

ACP: Algebra of Communicating Processes

Bergstra-Klop - 1984

The set of actions Λ_τ consists of a finite set of labels Λ and of special action τ, the syntax is

$$\begin{array}{rcl} E & ::= & \surd \mid a \mid E\backslash L \mid E/L \mid E[f] \mid E_1 \cdot E_2 \mid E_1 + E_2 \\ & & \mid E_1 \parallel E_2 \mid E_1 \lfloor\!\lfloor E_2 \mid E_1 |_c E_2 \mid \partial_H(p) \mid \delta \mid A \end{array}$$

- $a \in \Lambda_\tau$, $L \subseteq \Lambda$, $f : \Lambda \to \Lambda$;
- the operator $\cdot$ denotes sequential composition;
- $\partial_H(p)$ is the hiding operator;
- δ is the deadlocked process;
- A is a process constant.

ACP has been studied with Bisimulation and Branching Bis. Semantics

Axiomatic Semantics

Groups in Abstract Algebra

A *group* is a set G of abstract objects and of an operator $\star : G \times G \to G$ such that the following axioms hold:

- $a \star (b \star c) = (a \star b) \star c)$,
- $\exists u \in G : \; u \star a = a = a \star u$,
- $\forall a \in G, \exists a^{-1} \in G : \; a^{-1} \star a = a \star a^{-1} = \mathrm{u}$.

A group is any model of the above equational theory. The notion of groups is used to abstract from details and work with symbols rather than numbers.

Within ACP a process algebra is any mathematical structure, consisting of a set of objects and set of operators, like, e.g., sequential, nondeterministic or parallel composition, that enjoy the a given number of properties as specified by given axioms.

ACP and Axiomatic Semantics

Atomic Actions

Λ is a finite set of atomic actions: a, b, ... denote specific actions, while v and w denote generic actions.

ACP Syntax

$$\begin{array}{lll}
\text{BPA} & p & ::= v \mid p_1 + p_2 \mid p_1 \cdot p_2 \\
\text{CPA} & p & ::= v \mid p_1 + p_2 \mid p_1 \cdot p_2 \mid p_1 \| p_2 \mid p_1 \mathbin{\|\!_} p_2 \mid p_1 |_c p_2 \\
\text{ACP} & p & ::= v \mid p_1 + p_2 \mid p_1 \cdot p_2 \mid p_1 \| p_2 \mid p_1 \mathbin{\|\!_} p_2 \mid p_1 |_c p_2 \mid \partial_H(p) \mid \delta
\end{array}$$

Communication Functions

$\gamma : \Lambda \times \Lambda \to \Lambda \cup \{\delta\}$ (δ not in Λ), yields the corresponding communication action $\gamma(a, b)$, if a e b are meant to communicate and yields δ otherwise. Function γ can be defined freely but it has to satisfy:

$$\gamma(a, b) = \gamma(b, a) \qquad \gamma(\gamma(a, b), c) = \gamma(a, \gamma(b, c))$$

Axioms for ACP

Axioms for BPA

(A1) $x + y = y + x$
(A2) $(x + y) + z = x + (y + z)$
(A3) $x + x = x$
(A4) $(x + y) \cdot z = x \cdot z + y \cdot z$
(A5) $(x \cdot y) \cdot z = x \cdot (y \cdot z)$

Axioms for CPA

New Axioms for CPA

(M1) $x \| y = x \lfloor\!\lfloor y + y \lfloor\!\lfloor x + x |_c y$

(LM2) $v \lfloor\!\lfloor y = v \cdot y$
(LM3) $(v \cdot x) \lfloor\!\lfloor y = v \cdot (x \| y)$
(LM4) $(x + y) \lfloor\!\lfloor z = x \lfloor\!\lfloor z + y \lfloor\!\lfloor z$

(CM5) $v |_c w = \gamma(v, w)$
(CM6) $v |_c (w \cdot y) = \gamma(v, w) \cdot y$
(CM7) $(v \cdot x) |_c w = \gamma(v, w) \cdot x$
(CM8) $(v \cdot x) |_c (w \cdot y) = \gamma(v, w) \cdot (x \| y)$
(CM9) $(x + y) |_c z = x |_c z + y |_c z$
(CM10) $x |_c (y + z) = x |_c y + x |_c z$

Axioms for ACP

New Axioms for ACP

(A6) $x + \delta = x$
(A7) $\delta \cdot x = \delta$

(LM11) $\delta \mathbin{\|\!\!\_} x = \delta$

(D2) $\partial_H(v) = \delta$ if $v \in H$
(D3) $\partial_H(\delta) = \delta$
(D4) $\partial_H(x + y) = \partial_H(x) + \partial_H(y)$
(D5) $\partial_H(x \cdot y) = \partial_H(x) \cdot \partial_H(y)$
(D1) $\partial_H(v) = v$ if $v \notin H$

(CM12) $\delta |_c x = \delta$
(CM13) $x |_c \delta = \delta$

Models for ACP

Correctness and Completeness of Models

- Any of the set of axioms considered above induces an *equality relation*, denoted by $=$.
- A *model* for an axiomatization is a pair $\langle \mathcal{M}, \phi \rangle$, where $\mathcal{M}$ is a set and ϕ is a function that associates elements of $\mathcal{M}$ to ACP terms. We then have
    1. $\langle \mathcal{M}, \phi \rangle$ is *correct* if $s = t$ implies $\phi(s) = \phi(t)$
    2. $\langle \mathcal{M}, \phi \rangle$ is *complete* if $\phi(s) = \phi(t)$ implies $s = t$, for every pair of terms s and t.

1. Any model of (A1)-(A5) is a BPA;
2. Any model of (A1)-(A5) plus (M1), (LM2)-(LM4), (CM5)-(CM10) is a CPA;
3. Any models of ALL the axioms seen above is an ACP.

Models o BPA

Initial Models

- The simplest model for BPA has as elements the equivalence classes induced by $=$, i.e. all BPA terms obtained starting from atomic action, sequentialization and nondeterministic composition and mapping each term t to its equivalence class $[\![t]\!]$ as determined by $=$.
- This model is correct and complete and is known as *initial model* for the axiomatization.

Other, more complex models can be obtained by using LTS and factorizing them via bisimulation.

Operational Models for BPA

BPA operational semantics is defined by a *doubly labelled transition system* $\langle BPA, \Lambda, \rightarrow, \surd v \rangle$ where

- BPA is the set of terms generated by the corresponding syntax;
- Λ is the actions alphabet;
- $\rightarrow : BPA \times \Lambda \times BPA$ is the transition relation;
- $\surd v$ is an auxiliary predicate indicating that a process can terminate after executing action $\surd v$.

$$(\textsc{Self}) \quad \frac{}{v \surd v}$$

$$(\textsc{Alt1}) \quad \frac{x \surd v}{x + y \surd v} \qquad (\textsc{Alt2}) \quad \frac{x \xrightarrow{v} x'}{x + y \xrightarrow{v} x'} \qquad (\textsc{Alt3}) \quad \frac{y \surd v}{x + y \surd v}$$

$$(\textsc{Alt4}) \quad \frac{y \xrightarrow{v} y'}{x + y \xrightarrow{v} y'} \qquad (\textsc{Seq1}) \quad \frac{x \surd v}{x \cdot y \xrightarrow{v} y} \qquad (\textsc{Seq2}) \quad \frac{x \xrightarrow{v} x'}{x \cdot y \xrightarrow{v} x' \cdot y'}$$

Axioms and Bisimilarity

Correspondence between Axiomatic and Operational Semantics

- Equality $=$ as induced by (A1)-(A5) is *correct* relatively to bisimilarity $\sim$, i.e., if $p = q$ then $\mathcal{LTS}(p) \sim \mathcal{LTS}(q)$;

- Equality $=$ as induced by (A1)-(A5) is *complete* relatively to bisimilarity $\sim$, i.e., if $\mathcal{LTS}(p) \sim \mathcal{LTS}(q)$ then $p = q$.

TCSP: Theoretical Communicating Sequential Processes

Brookes-Hoare-Roscoe - 1984

The set of actions is a set Λ, and the syntax is

$$E \quad ::= \quad \mathit{Stop} \;\Big|\; \mathit{skip} \;\Big|\; a \rightarrow E \;\Big|\; E_1 \sqcap E_2 \;\Big|\; E_1 \,\Box\, E_2 \;\Big|\; E_1 \,|[L]|\, E_2 \;\Big|\; E/a$$

where

- $a \in \Lambda$, $L \subseteq \Lambda$, $f : \Lambda \rightarrow \Lambda$,
- the operators $\sqcap$ and $\Box$ denote internal and external choice respectively;
- the operator $\rightarrow$ denotes action prefixing

CSP has been studied with Failure Semantics - a variant of Testing Sem.

Failure Sets

1. $\langle s, V\rangle \in F \implies V$ finite.
2. $\langle \epsilon, \emptyset\rangle \in F$, where ϵ denotes the empty sequence and $\emptyset$ the empty set
   Refusal-set are not-empty.
3. $\langle st, \emptyset\rangle \in F \implies \langle s, \emptyset\rangle \in F$.
   The set of traces needs o be prefix-closed.
4. $V \subseteq W$ e $\langle s, W\rangle \in F \implies \langle s, V\rangle \in F$.
   Refusal sets are downwards closed.
5. If $U = \{a \,|\, \langle sa, \emptyset\rangle \in F\}$ and $W \subseteq_f (A - U)$ then
   $\langle s, V\rangle \in F \implies \langle s, V \cup W\rangle \in F$.
   If from a state reacheable via trace s an action a cannot be performed then after s there must be a refusal set containing a, i.e., if $\langle sa, \emptyset\rangle \notin F$ and $\langle s, V\rangle \in F$ then $\langle s, V \cup \{a\}\rangle \in F$.

Failure Semantics for TCSP

- $\mathcal{F}[\![Stop]\!] = \{\langle \epsilon, V\rangle \mid V \subseteq A\}$
- $\mathcal{F}[\![skip]\!] = \{\langle \epsilon, V\rangle \mid V \subseteq A\} \ \cup \ \{\langle \surd, V\rangle \mid V \subseteq A\}$
- $\mathcal{F}[\![a \to P]\!] = \{\langle \epsilon, V\rangle \mid V \subseteq A - \{a\}\} \cup \{\langle as, W\rangle \mid \langle s, W\rangle \in \mathcal{F}[\![P]\!]\}$
- $\mathcal{F}[\![P_1 \,\Box\, P_2]\!] = \{\langle \epsilon, V\rangle \mid \langle \epsilon, V\rangle \in \mathcal{F}[\![P_1]\!] \cap \mathcal{F}[\![P_2]\!]\} \cup \{\langle s, W\rangle \mid \langle s, W\rangle \in \mathcal{F}[\![P_1]\!] \cup \mathcal{F}[\![P_2]\!]$ and s is a non empty sequence of actions$\}$
- $\mathcal{F}[\![P_1 \sqcap P_2]\!] = \mathcal{F}[\![P_1]\!] \cup \mathcal{F}[\![P_2]\!]$
- $\mathcal{F}[\![P_1 \;|[L]|\; P_2]\!] = \{\langle u, V \cup W\rangle \mid V - L = W - L \wedge \langle s, V\rangle \in \mathcal{F}[\![P_1]\!] \wedge \langle t, W\rangle \in \mathcal{F}[\![P_2]\!] \wedge u \in \|_L(s, t)\}$ - $\|_L(s, t)$ denotes the merging of s and t considering synchronization of actions in L.
- $\mathcal{F}[\![P/a]\!] = \{\langle s/a, V\rangle \mid \langle s, V \cup \{a\}\rangle \in \mathcal{F}[\![P]\!]\}$, - s/a denotes the sequence obtained from s by removing all occurrences of a.

Testing and Failures for CSP

Correspondence between Denotational and Operational Semantics

- $\mathcal{F}[\![P]\!] = \mathcal{F}[\![Q]\!]$ if and only if $\mathcal{LTS}(P) \simeq_{test} \mathcal{LTS}(Q)$;