Weak Bisimilarity Coalgebraically

Andrei Popescu

Department of Computer Science

University of Illinois

Context and motivation

Process algebra:

- SOS presentations: one-step behavior
- Process equivalence: **weak bisimilarity**: arbitrarily long sequences of silent (unobservable) actions

Consequence: Modular reasoning difficult

Put in other words: No modular denotational semantics transparent from the syntactic setting

My contribution

- Introduce a coalgebraic semantic domain for weak bisimilarity
- Define a modular fully-abstract denotational semantics for CCS under weak bisimilarity
- Construction quite general – would work for many process algebras

Weak bisimilarity recalled

Labeled Transition System (LTS) over $\mathrm{Act} \cup \{\tau\}$:

$\forall\, \pi, \rho \in \mathrm{Proc}$ – processes

- $a, b \in \mathrm{Act}$ – “loud” (observable) actions
- τ – silent (unobservable) action
- $\alpha \in \mathrm{Act} \cup \{\tau\}$
- For each α, $\quad \xrightarrow{\alpha} \;\subseteq\; \mathrm{Proc} \times \mathrm{Proc}$
- Alternative view: coalgebra for the functor

$X \;\mapsto\; \wp((\mathrm{Act} \cup \{\tau\}) \times X)$

Weak bisimilarity recalled

π and ρ weakly bisimilar iff:

$\forall\, \pi -\tau\rightarrow \pi'$ implies $\rho -\tau^*\rightarrow \rho'$ for some ρ' such that π' and ρ' are weakly bisimilar

$\forall\, \pi -\tau^*\rightarrow \pi' -a\rightarrow \pi'' -\tau^*\rightarrow \pi'''$ implies $\rho -\tau^*\rightarrow \rho' -a\rightarrow \rho'' -\tau^*\rightarrow \rho'''$ for some ρ', ρ'', ρ''' s.t. π''' and ρ''' are weakly bisimilar

- And vice versa
- And so on, indefinitely

Coalgebraic semantic domain for weak bisimilarity

Why coalgebraic?

1. CALCO
2. Alternative: domain theory: problem with infinite branching: breaks compactness – an infinite process/tree no longer determined by its finite subtrees
3. On the “good” side of losing compactness: no need for finiteness/guardedness conditions on syntax

Coalgebraic semantic domain for weak bisimilarity

- For strong bisimilarity: both syntax and semantics form coalgebras
- For weak bisimilarity: structural axioms added:

τ absorbed

- Aczel – Final universes of processes, 1993: τ-system: LTS on $Act \cup \{\tau\}$ s.t., for all processes π, π’, π’’ and action α:

  $\pi \xrightarrow{\tau} \pi$

  $\pi \xrightarrow{\tau} \pi' \xrightarrow{\alpha} \pi''$ implies $\pi \xrightarrow{\alpha} \pi''$

  $\pi \xrightarrow{\alpha} \pi' \xrightarrow{\tau} \pi''$ implies $\pi \xrightarrow{\alpha} \pi''$

- The final τ-system – semantic domain for processes under weak bisimilarity

Coalgebraic semantic domain II

Rephrasing: partial “concatenation” operation, on
$((\mathrm{Act} \cup \{\tau\}) \times \{\tau\}) \cup (\{\tau\} \times (\mathrm{Act} \cup \{\tau\}))$,
defined by $\alpha\,\tau = \tau\,\alpha = \alpha$

τ-system: pair $(A, \rightarrow : (\mathrm{Act} \cup \{\tau\}) \Rightarrow \mathrm{Rel}(A))$,

with $\rightarrow$:

- compatible w.r.t. _ _ versus relation composition
- super-commutes with the identity (i.e., maps τ to a superset of $\mathrm{Diag}(A)$)

Coalgebraic semantic domain III

Problem with this domain:

- describes process in single-step depth only
- hence unnatural for accommodating operations (such as parallel composition) that need to explore processes in more depth

Thus: to know where π | ρ transits to silently (via τ-transitions), need to know where π and ρ transit via arbitrarily long sequences of actions. E,g.:

π –a→ π’ –b⁻ㄹ ㅓ’’ ㅓ ㅃ a⁻ㄹ ㅓ’ ㅃ b→ ρ’’

π | ρ –τ*→ π’’ | ρ’’

Coalgebraic semantic domain IV

Natural improvement of the domain: consider arbitrary sequences (while still absorbing τ), i.e.:

- τ is now the empty sequence, an element of Act^*
- τ-*-system: pair $(A,\rightarrow)$, with $\rightarrow : Act^* \Rightarrow Rel(A)$

1. morphism of semigroups between $(Act^*, \_ \_)$ and $(Rel(A), ;)$
2. again, super-commutes with the identity

The categories of τ-systems and τ-*-systems (regarded as coalgebras) are isomorphic: $\rightarrow$ in a τ-*-system uniquely determined by its restriction to $Act \cup \{\tau\}$ and condition 1

Coalgebraic semantic domain V

Spelling out the above: Act*-coalgebra s.t., for all π, π', π'' and $u,v \in$ Act*:

$\pi \xrightarrow{\tau} \pi$

$\pi \xrightarrow{u} \pi' \xrightarrow{v} \pi''$ implies $\pi \xrightarrow{uv} \pi''$

$\pi \xrightarrow{uv} \pi''$ implies

$\exists\pi'.\ \pi \xrightarrow{u} \pi' \wedge \pi' \xrightarrow{v} \pi''$

Application: denotational semantics for CCS

Syntax:

- $a, b \in Act$ – loud actions
- $\bar{\ } : Act \Rightarrow Act$ involutive bijection
- τ – silent action
- $\alpha \in Act \cup \{\tau\}$
- $X \in Var$, countable set of process variables
- $P \in Proc$, set of (process) terms:

$$P \; ::= \; \ldots \mid X \mid P | Q \mid \mu X.\, P$$

Denotational semantics for CCS II

Transition system:

$$\frac{P \xrightarrow{\alpha} P'}{P \mid Q \xrightarrow{\alpha} P' \mid Q} \qquad \frac{Q \xrightarrow{\alpha} Q'}{P \mid Q \xrightarrow{\alpha} P \mid Q'}$$

$$\frac{P \xrightarrow{a} P' \qquad Q \; {-}\bar{a}\text{ㄹ} \;\; Q'}{P \mid Q \xrightarrow{\tau} P' \mid Q'} \qquad \frac{P[(\mu X.\, P) / X] \xrightarrow{\alpha} Q'}{\mu X.\, P \xrightarrow{\alpha} Q'}$$

Denotational semantics for CCS III

First step: modify transition system to describe behavior along sequences of actions:

$$\frac{P[(\mu X.\, P) / X] \xrightarrow{u} Q'}{\mu X.\, P \xrightarrow{u} Q'} \qquad \frac{P \xrightarrow{u} P' \quad Q \xrightarrow{v} Q'}{P \mid Q \xrightarrow{w} P' \mid Q'}\,[w \in u \mid v]$$

with $\mid : Act^* \times Act^* \Rightarrow \wp(Act^*)$ defined recursively:

- $\tau \mid \tau \;=\; \{\tau\}$
- $(a\ u) \mid (b\ v) \;=\; a\,(u \mid (b\ v)) \cup b\,((a\ u) \mid v)$
  $\qquad\qquad\qquad\qquad \cup\;\; u \mid v, \quad \text{if } b = \bar{a}$

Denotational semantics for CCS IV

Theorem: Weak bisimilarity of the original system coincides with strong bisimilarity of the sequence-based system.

Transformation seems to work not only for CCS, but for a general class of process algebras, as in

van Glabbeek – On cool congruence formats for weak bisimulations, 2005 (building on previous work by B. Bloom)

Denotational semantics for CCS V

Second step: denotational semantics for the sequence-based system into our sequence-based domain (the final τ-*-system)

- Almost falls under general theory:
  - Rutten – Processes as terms: Non-well-founded models for bisimulation, 1992
  - Turi, Plotkin – Towards a mathematical operational semantics, 1997
- E.g., SOS rule for parallel composition transliterates into

$$\mathrm{Unfold}(\pi \mid \rho) = \{(w, \pi' \mid \rho').\ \exists\, u, v.\ (u, \pi') \in \mathrm{Unfold}(\pi) \ \wedge\ (v, \rho') \in \mathrm{Unfold}(\rho) \ \wedge\ w \in u \mid v\}$$

Denotational semantics for CCS VI

Recursion rule

$$\frac{P[(\mu\ X.\ P)\ /\ X] \ -u\!\rightarrow\ Q'}{\mu\ X.\ P \ -u\!\rightarrow\ Q'}$$

Further modified into an equivalent “well-founded” rule:

$$\frac{P[P\ /\ X]^{n} \ -u\!\rightarrow\ Q'}{\mu\ X.\ P \ -u\!\rightarrow\ Q'[(\mu\ X.\ P)\ /\ X]}\,[n \in N]$$

Corresponding second-order semantic operator on the final τ-*-system: $\mathrm{Rec} : (\mathrm{Proc} \Rightarrow \mathrm{Proc}) \Rightarrow \mathrm{Proc}$,

$\mathrm{Unfold}(\mathrm{Rec}\ F) = \{(u,\ G(\mathrm{Rec}\ F)).$

$\exists n{\geq}1.\forall\pi.\ (u,\ G\ \pi) \in\ \mathrm{Unfold}(F^{n}\ \pi)\}$

Denotational semantics for CCS VII

- Thus: we have semantic operators corresponding to the syntactic constructs
- $P \mapsto [[P]]$ denotes the standard interpretation of terms in the semantic domain via environments

Theorem (Full abstraction): The following are equivalent:

- $[[P]] = [[Q]]$
- P and Q are strongly bisimilar in the sequence-based system
- P and Q are weakly bisimilar in the original system

Denotational semantics for CCS (parenthesis)

- Alternative to using numbers when defining semantic recursion: Peter Aczel’s approach from “Final universes of processes”:
  - no semantic operator for recursion
  - instead: give recursion a special treatment, integrating it globally into the semantics

Theorem: There exists a unique “least non-deterministic” map

[[_]] from terms to processes such that:

- [[_]] satisfies the transliterated semantic equations for all operators except μ
- $[[\ \mu X.\ P\ ]] = [[\ P[(\mu X.\ P) / X]\ ]]$

Future work

- Employ the sequence-based semantics for weak bisimilarity in modular theorem proving:
  - knowledge of behavior along arbitrary traces necessary for knowledge about silent-step behavior,
  - thus having the former knowledge explicitly represented seems helpful
- Prove, for systems in a general SOS format, also incorporating syntax with bindings / substitution
  - soundness of the one-step to multi-step transformation
  - the full abstraction theorem

Future work and more related work

Cover issues such as name-passing and scope extrusion (i.e., systems in the π-calculus family)

- Much existing work on compositional semantics for π under strong bisimilarity:
  - Domain-theoretic: Stark 1996; Fiore, Moggi, Sangiorgi 1996; Staton – Ph.D. thesis, 2007
  - Coalgebraic: Honsell, Lenisa, Montanari, Pistore, 1998, Lenisa – Ph.D. thesis, 1998.
- For weak bisimilarity: Popescu – Tech. report, 2009: employ the same technique as for CCS + parameterize parallel composition with all the dynamic topological information:
  - semantics is compositional and fully abstract
  - but technically too complicated, hence not very useful for modular reasoning

Future work and more related work

More insightful approach for π-like calculi:

- Shall be based on levels of information, as in, e.g., Stark 1996 and Fiore et al. 1996: a process at level n knows n channel names

- Challenge: define the appropriate categorical structure for an index-free treatment
  - Objects: natural numbers
  - “Vertical” morphisms: $m \xrightarrow{\sigma} n$ – as before, σ map between m and n treated as finite sets (intuition: renaming)
  - “Horizontal” morphisms: $n \xrightarrow{w} n + p$ iff the sequence of actions w increases the number of known channels from n to $n + p$
  - Domain: Functor from this category into the category Rel, of sets and relations
  - Hopefully: Syntax – initial domain; semantics – final domain

Thank you!