Complete Lattices and Up-to Techniques

Damien Pous - ENS Lyon, France

APLAS'07
December 1st
Singapore

Outline

1. A concrete toy example to get started,
2. General and abstract theory of up-to techniques,
3. A new method for validating up-to context techniques.

Bisimulation, up-to techniques

- **Bisimilarity**: a behavioural equivalence associated with a proof technique: bisimulation.

  “To prove that $p$ and $q$ are bisimilar, it suffices to show that $p$ and $q$ are related by a relation which is a bisimulation”.

- Up-to techniques, “bisimulation up-to”:

  “To prove that $p$ and $q$ are bisimilar, it suffices to show that $p$ and $q$ are related by a relation which is almost a bisimulation”.

- Not only for $\pi$-calculists: growing interest in bisimulation proof methods for extensions of the $\lambda$-calculus.

A typical bisimulation proof

- Bisimilarity is the largest symmetric relation such that the following diagram holds:

  $\begin{array}{c}
p \\
\alpha \downarrow \\
p' \sim q'
\end{array}$

- Suppose we have an LTS, with a replication operator ($!$), defined by the following rule:

  $\begin{array}{c}
p \xrightarrow{\alpha} p' \\
\hline
!p \xrightarrow{\alpha} !p | p'
\end{array}$

  let’s prove that bisimilarity is preserved by this operator:

  $p \sim q \Rightarrow !p \sim !q$
A typical bisimulation proof, cont.

Assuming that $p \sim q$, we have to find a relation that contains $⟨!p, !q⟩$ and satisfies the previous diagram.

\[ R_1 \triangleq \{⟨!p, !q⟩\} \]

\[ \begin{array}{ccc}
P & R_1 & Q \\
α & \uparrow & α \\
!p | p' & ? & !q | q' \\
\end{array} \]

$R_2 \triangleq R_1 \cup \{⟨!p | p_1, !q | q_1⟩ \mid \forall p_1, q_1, p_1 \sim q_1\}$

\[ \begin{array}{ccc}
P & R_2 & Q \\
α & \uparrow & α \\
!p | p_1 & ? & !q | q_1 \\
\end{array} \]

\[ (⟨!p | p_2⟩ | p_1) \sim (⟨!q | q_2⟩ | q_1) \]

\[ \ldots \]

\[ R_∞ \triangleq \{⟨(\ldots (⟨!p | p_n⟩ | p_{n−1}) \ldots ) | p_1, (\ldots (⟨!q | q_n⟩ | q_{n−1}) \ldots ) | q_1⟩ \mid \forall n \in \mathbb{N}, \forall i \leq n, p_i \sim q_i\} \]

A typical bisimulation up-to proof

- Start again with the singleton relation: $R_1 \triangleq \{⟨!p, !q⟩\}$

\[ \begin{array}{ccc}
P & R_1 & Q \\
α & \uparrow & α \\
!p | p' & ? & !q | q' \\
\end{array} \]

\[ R_3 \triangleq (C(R_1) \sim !q | q' \sim !p | p') \]

\[ \begin{array}{ccc}
P & R_1 & Q \\
α & \uparrow & α \\
!p | p' & ? & !q | q' \\
\end{array} \]

- $R_1$ is a bisimulation up-to the map $R \mapsto C(R) \sim$.

- This up-to technique is actually correct, we have proved that $R_1 \subseteq \sim$.

A Theory of Up-to Techniques

- Other examples of up-to techniques:
  - based on diagram chasing arguments:
    - $R \mapsto \sim R \sim$ (more modular proofs)
    - $R \mapsto R^*$ (small and local candidates)
    - $R \mapsto \preceq R \approx$ (patch for the weak case)
  - based on the structure of processes:
    - $\mathcal{C}$ (keep small processes)
    - $\sigma$ “injective substitution” (work with fewer names)

- Some of these techniques them may be delicate to obtain.

- We often want to combine these techniques, to obtain a powerful one.

Challenge 1: modularity
Challenge 2: abstraction

▶ Several kinds of bisimilarity:
   1. several kinds of transition systems (π, λ…)
   2. strong (∼), weak (≈), expansion (≿), coupled…
   3. labelled, barbed, hedged, typed, with environments…

▶ The notion of LTS give the first level of abstraction: we don’t need to fix the set of processes.

▶ The theory of up-to techniques can be defined at the abstract level of complete lattices; In doing this, we will reason about coinduction in general, rather than about a specific form of bisimilarity.

(we will actually use this gain of generality)

▶ We enrich the complete lattice with a monoid structure to define bisimilarity, and diagram based techniques

▶ For up-to context techniques, we need to work with concrete relations, however.

Up-to techniques

▶ A diagram becomes a simple inclusion:

\[
egin{array}{ccc}
  p & R & q \\
  \alpha & \downarrow & \alpha \\
  p' & R & q' \\
  R \subseteq s(R)
\end{array}
\quad
\begin{array}{ccc}
  p & R & q \\
  \alpha & \downarrow & \alpha \\
  p' & f(R) & q' \\
  R \subseteq s(f(R))
\end{array}
\]

▶ A map \( f \) is correct if any simulation up to \( f \) is contained in similarity, or equivalently, if

\[
\nu(s \circ f) \subseteq \nu s
\]

▶ Different maps may generate the same similarity, and some of them are easier to work with; we would like to find the “largest” one.

▶ Problem with correct maps: they are not always preserved by composition or lubs.

Compatible maps

▶ An order-preserving map \( f \) is compatible (with \( s \)) if

\[
f \circ s \subseteq s \circ f
\]

Proposition.

▶ Compatible maps are correct maps.

▶ Compatible maps are closed under composition and lubs.

▶ In the case of strong/weak bisimilarity, all known up-to techniques can be expressed by means of compatible maps [San98]…

▶ except for recent ones, that go beyond the “up to expansion” technique by using termination hypotheses [Pou05].

Coinduction

▶ Assume a complete lattice \( (X, \subseteq, \bigcup) \) (elements of \( X \) \( (R, S \ldots) \) intuitively represent binary relations)

▶ Knaster-Tarski theorem ensures that any order preserving map \( s : X \rightarrow X \) has a greatest fixpoint, obtained as the lub of its post-fixpoints:

\[
\nu s \triangleq \bigcup \{ R \in X \mid R \subseteq s(R) \}
\]

▶ We introduce the following terminology:

▶ \( \nu s \) is called the similarity,

▶ a simulation is an element \( R \) s.t. \( R \subseteq s(R) \),

“similarity is the greatest simulation”

(almost all notions of bisimilarity can be defined in this way)

▶ a simulation up to \( f \) is an element \( R \) s.t. \( R \subseteq s(f(R)) \),
Combining correct and compatible maps

These recent techniques being only correct, they could not be combined for free, even with standard (compatible) techniques.

The following theorem gives a sufficient condition for such a combination to remain correct:

**Composition Theorem.**
Let $g$ be correct and $f$ be compatible.
If $f$ is compatible with $g$, then $g \circ f$ is correct.
(surprisingly, the sufficient condition is a compatibility property)

**An application of the composition theorem**

**Complex technique.** Let $\succ$ be a relation. If $\succ^+ \cdot \rightarrow^+$ is strongly normalising, then the following map is correct:
$$t_\succ : R \mapsto (R \cap \succ)^* \cdot R.$$ 

**Standard technique.** The following map is compatible
$$f : R \mapsto C(R)^= \cdot \approx.$$ 

To combine both techniques, it suffices that $f$ be compatible with $t_\succ$. A sufficient condition for that is $C(\succ) \subseteq \succ$.

**Combined, scary technique.** If $\succ^+ \cdot \rightarrow^+$ is strongly normalising, and $C(\succ) \subseteq \succ$, then any symmetric relation satisfying the following diagram is contained in $\approx$:

\[
\begin{array}{ccc}
p & R & q \\
\alpha \downarrow & \Downarrow & \alpha \downarrow \\
p' & ((C(R) \cup \approx) \cap \succ)^* \cdot C(R)^= \cdot \approx & q'
\end{array}
\]

**Up to context techniques**

- The following situation may appear in a bisimulation game:

\[
\begin{array}{ccc}
p & R & q \\
\alpha \downarrow & \Downarrow & \alpha \downarrow \\
c[p'] & C(R) & c[q'] \\
q' & R & q'
\end{array}
\]

- In this case, we would like to reason “up to context”, and just remove the context part of the processes.

- These techniques generally fall in the scope of compatible maps; however, proving this usually requires us:
  - in most cases, to consider polyadic contexts;
  - to reason by induction on the structure of these contexts.
Standard definition of context closure

Consider the case of CCS:

\[ p, q ::= 0 \mid (\nu a)p \mid \alpha.p \mid p \mid q \mid !p \]

- A polyadic context \( c \) is a process whose occurrences of \( 0 \) are numbered; \( c[p_1 \ldots p_n] \) is the term obtained by replacing numbered occurrences; we associate to each context the following map over relations:

\[ \lfloor c \rfloor : R \mapsto \{ (c[p_1, \ldots, p_n], c[q_1, \ldots, q_n]) \mid \forall i, \ p_i R q_i \} \]

- The context closure is the map \( C \triangleq \bigcup_c \lfloor c \rfloor \).

- Proving the compatibility of \( C \) directly requires a tedious structural induction.

Characterisation by means of initial contexts

- Define the following initial contexts:

\[ 0 : \emptyset \mapsto 0 \]
\[ (\nu a) : p \mapsto (\nu a)p \]
\[ \alpha. : p \mapsto \alpha.p \]
\[ | : p, q \mapsto p \mid q \]
\[ ! : p \mapsto !p \]

- By iterating over these contexts, we can reach the previous definition of context closure:

**Proposition.**

\[ C = \left( \text{id} \cup \bigcup_{c \text{ initial}} \lfloor c \rfloor \right)^\omega. \]

- Therefore, it should suffice to prove that maps \( \lfloor c \rfloor \) are compatible, where \( c \) is initial. **Unfortunately**, \( \lfloor ! \rfloor \) is not compatible by itself (\( C \) is, don’t worry...).

Up-to techniques for compatibility

- Recall that \( f \) is compatible if \( f \circ s \subseteq s \circ f \).  
- This property can be defined coinductively, by working in the function space (which is a complete lattice): there exist a second-order map \( \varphi \) s.t.:

\[ f \subseteq \varphi(g) \iff f \circ s \subseteq s \circ g \quad (\text{notation: } f \stackrel{s}{\rightarrow} g). \]

- We have a theory of up-to techniques for compatibility!

**Theorem.** If \( f \stackrel{s}{\rightarrow} f^\omega \), then \( f^\omega \) is compatible.

**Theorem.** If \( g \) is compatible, and \( f \stackrel{s}{\rightarrow} g \circ f^\omega \), then \( g \circ f^\omega \) is compatible.

(in both cases, under some uninteresting technical conditions on \( f \) and \( g \), easily satisfied in practise)

The initial contexts method

- We want to prove that \( C \) is compatible, i.e., that \( C \stackrel{s}{\rightarrow} C \).
- Proving \( \lfloor c \rfloor \stackrel{s}{\rightarrow} \lfloor c \rfloor \) for any initial context is sufficient, but may not always be possible;
- Thanks to the “up to iteration” technique, it suffices to prove \( \lfloor c \rfloor \stackrel{s}{\rightarrow} C \) for any initial context. This amounts to checking a simple condition between each syntactic construction of the language and the map that generates the bisimilarity we consider.

- This method is complete; in CCS, in the strong case, we have:

\[ \lfloor ! \rfloor \stackrel{s}{\rightarrow} \lfloor ! \rfloor^\omega \circ (\lfloor ! \rfloor \cup \text{id}) \subseteq C \]

- in the weak case, we found a mistake in the standard proof [SW01]: \( C \) itself is not compatible, we have to reason modulo unfolding of replications.
Summing up

▶ A general theory of up to techniques for coinduction: compatible and correct maps that can be composed.
▶ A theory of up to techniques for compatibility,
▶ used to define a method for validating up to context techniques in an easy way (initial contexts)
▶ More in the paper:
  ▶ going from one-sided games to two-sided games, at the abstract level.
  ▶ proof of the scary technique based on termination guarantees;
  ▶ detailed proofs for up to context in CCS,
  ▶ a counter-example for an invalid combination of up to context and a restricted form of up to transitivity.

Remarks & Future work

▶ We cannot encompass the recent “logical bisimulations” [SKS07a], but it seems that we can analyse the even more recent “environment bisimulations” [SKS07b].
▶ Up-to techniques relying on termination hypotheses can be proved at the abstract (point-free) level [DBvdW97].
▶ Parts of the theory presented here are formalised in the Coq proof assistant; in the long term, we would like to define a framework in which bisimulation proofs could be done formally and easily, in a semi-automatic way.
▶ Can SOS rule formats for congruence (tyft/tyxt, panth...) be turned into rule formats for up-to context techniques?

Thanks!