Verifying Temporal Properties using Explicit Approximants

–

Completeness for Context-free Processes

Ulrich Schöpp and Alex Simpson

Laboratory for Foundations of Computer Science

Division of Informatics

University of Edinburgh

Verifying Temporal Properties

Does process p satisfy the temporal property φ?

$$\boxed{p \models \varphi}$$

The temporal property φ might be:

- p can eventually enter a critical section
- p can always engage in a communication
- p can only do infinitely many pop-actions if it also does infinitely many push-actions

Verifying Temporal Properties

Finite State:

$$\begin{tikzcd} & Q \arrow[dr, bend right, "b"'] & \\ P \arrow[ur, "a"] & & R \arrow[ul, bend right, "a"'] \end{tikzcd} \models \varphi$$

- Model Checking
- *automatic* methods based on global state exploration

Infinite State:

$$\begin{tikzcd} P \arrow[r, bend left, "\text{push}"] & PP \arrow[l, bend left, "\text{pop}"] \arrow[r, bend left, "\text{push}"] & PPP \arrow[l, bend left, "\text{pop}"] \arrow[r, bend left, "\text{push}"] & \cdots \arrow[l, bend left, "\text{pop}"] \end{tikzcd} \models \varphi$$

- arises very naturally, e.g. unbounded stacks
- decidable for restricted classes such as Context-free Processes or Pushdown Processes [Muller, Schupp 85]
- in general undecidable

Verifying Infinite State Processes

Despite undecidability one still wants a formal system for verifying infinite state systems.

$\Rightarrow$ Need for user interaction

$\Rightarrow$ Requirements on a formal system:

- Structural reasoning ("natural" proofs)
- Compositionality
  - Modularity
  - Proof reuse
  - Parameterised verification

Verification as Provability of Sequents

Consider sequents $\Gamma \vdash \Delta$ containing *verification assertions* $p : \varphi$.

- Verification goals $\qquad \vdash \mathsf{P} : \varphi$
- Parameterised goals $\qquad x : \psi_1,\ y : \psi_2 \vdash x|y : \varphi$
- Compositional reasoning

$$\frac{\vdash p|q : \varphi}{\vdash p : \psi_1 \qquad \vdash q : \psi_2 \qquad x : \psi_1,\ y : \psi_2 \vdash x|y : \varphi}$$

- General purpose: Use of *transition assertions* $p \xrightarrow{a} q$ allows to derive process rules from operational semantics. [Simpson 95]
- The main difficulty is how to incorporate temporal reasoning. This is addressed in [Dam 95,96,98], [Dam,Gurov 00], . . .

Limitations

Any finitary system for verifying general infinite state systems is necessarily incomplete: There are true statements $p \models \varphi$ which the system cannot prove, i.e. $\not\vdash\ p : \varphi$.

Ways of assessing the power of such systems:

- Case studies [Dam], [Gurov], [Fredlund], . . .
- Finite-State completeness [Dam 95]
- Completeness for restricted classes of infinite state systems whose verification problem is decidable.

  The present paper: Completeness for Context-Free Processes.
- Completeness with respect to parametric goals [Future Work]

Verifying Temporal Properties using Explicit Approximants

Consider sequents $\Gamma \;\vdash\; \Delta$ containing verification assertions $p : \varphi$.

φ is given in the modal μ-calculus in positive normal form:

$$\varphi \;::=\; X \mid \varphi_1 \vee \varphi_2 \mid \varphi_1 \wedge \varphi_2 \mid \langle a \rangle \varphi \mid [a]\varphi \mid \mu X.\,\varphi \mid \nu X.\,\varphi$$

Least fixed points

$$\emptyset \subseteq \varphi(\emptyset) \subseteq \varphi^2(\emptyset) \subseteq \ldots \subseteq \varphi^\omega(\emptyset) \subseteq \ldots = \mu X.\,\varphi$$

Greatest fixed points

$$T \supseteq \varphi(T) \supseteq \varphi^2(T) \supseteq \ldots \supseteq \varphi^\omega(T) \supseteq \ldots = \nu X.\,\varphi$$

Free variable X in a sequent represents a fixed-point approximant.

Extended Syntax

It is useful to express more than just “Is a fixed point approximant.”

We extend the syntax with two constructs:

Explicit approximant declarations: $[X \geqslant \varphi]\,\Phi$

The explicit declaration $[X \geqslant \varphi]\,\Phi$ expresses that Φ holds for all approximants X of $\nu X.\,\varphi$.

Think of it as

$$\forall X \in \mathcal{A}^{\nu X.\,\varphi}.\,\Phi$$

Note that $[X \geqslant \varphi]$ is a binder for X.

Example: For any ordinary μ-calculus formula ψ, the formula

$[X \geqslant \varphi]\,\psi$ is equivalent to $\psi[\nu X.\,\varphi \,/\, X]$.

Extended Syntax

Modification modalities: $[+X]\,\Phi$

$\boxed{\text{If } X \text{ is an approximant of } \nu X.\,\varphi \text{ then } [+X]\,\Phi \text{ expresses that for all approximants } X' \text{ of } \nu X.\,\varphi \text{ with } X' \supset X \text{ it holds that } \Phi[X'/X].}$

Think of it as

$$\forall (X' \sqsupset X).\, \Phi[X'/X]$$

where $X' \sqsupset X$ means that X' approximates the *same* fixed-point as X and that $X' \supset X$ holds.

Note that X is *free* in $[+X]\,\Phi$.

Example: X is equivalent to $[+X]\,\varphi$. (Unfolding)

Context-free Processes

Process definition:

$$\mathsf{P} \xrightarrow{a} \mathsf{PP}, \qquad \mathsf{P} \xrightarrow{b} \varepsilon$$

Transition system:

$$\begin{tikzcd}
\varepsilon & \mathsf{P} \arrow[l, bend left, "b"] \arrow[r, bend left, "a"] & \mathsf{PP} \arrow[l, bend left, "b"] \arrow[r, bend left, "a"] & \mathsf{PPP} \arrow[l, bend left, "b"] \arrow[r, bend left, "a"] & \cdots \arrow[l, bend left, "b"]
\end{tikzcd}$$

A context-free process is given by rules of the form

$$\mathsf{P} \xrightarrow{a} \mathsf{P}_1\mathsf{P}_2 \ldots \mathsf{P}_n$$

where $\mathsf{P}, \mathsf{P}_1, \ldots, \mathsf{P}_n$ are taken from a finite set of nonterminals.

The transitions induced are: $\mathsf{P}\,\mathsf{q} \xrightarrow{a} \mathsf{P}_1\mathsf{P}_2 \ldots \mathsf{P}_n\,\mathsf{q}$

Global Rule

Assuming the root sequent is false, we get an infinite chain of approximants interpreting X:

$$\mathcal{X} \supset \mathcal{X} \supseteq \mathcal{X} \supseteq \mathcal{X} \supseteq \mathcal{X} \supseteq \mathcal{X}$$

$$
\begin{array}{c}
\vdash\ \mathsf{P} : \nu X.\, \mu Y.\, [a]X \wedge [b]Y \\
\hline
\vdots \\
\hline
x : \varphi \vdash \mathsf{P}\, x : \varphi \\
\hline
x : \varphi \vdash \mathsf{P}\, x : [a]X \wedge [b]\varphi \\
\hline
x : \varphi \vdash \mathsf{P}\, x : [a]X \qquad \ddots \\
\hline
x : \varphi \vdash \mathsf{PP}\, x : X \\
\hline
x : \varphi \vdash \mathsf{P}\, x : [+X\,]\,\varphi \qquad\qquad x : [+X\,]\,\varphi \vdash \mathsf{P}\, x : X \quad (\mathsf{Cut})
\end{array}
$$

$$
\frac{x : \varphi \vdash \mathsf{P}\, x : [+X\,]\,\varphi}{x : \varphi \vdash \mathsf{P}\, x : \varphi}\,(+X)
\qquad
\frac{\dfrac{x : [+X\,]\,\varphi \vdash \mathsf{P}\, x : X}{x : [+X\,]\,\varphi \vdash \mathsf{P}\, x : [+X\,]\,\varphi}}{x : \varphi \vdash \mathsf{P}\, x : \varphi}\,(+X)
$$

Why Modifiers?

Consider the last example without the use of a modifier in (Cut):

$$\mathcal{X} \supseteq \mathcal{X} \supseteq \mathcal{X} \supseteq \mathcal{X} \supseteq \mathcal{X}$$

$$\dfrac{\dfrac{\dfrac{\dfrac{x:\varphi \vdash \mathsf{P}\,x:\varphi}{x:\varphi \vdash \mathsf{P}\,x:[a]X \wedge [b]\varphi}\quad \ddots}{x:\varphi \vdash \mathsf{P}\,x:[a]X}}{x:\varphi \vdash \mathsf{PP}\,x:X}}{x:\varphi \vdash \mathsf{P}\,x:\varphi \qquad \dfrac{\dfrac{x:\varphi \vdash \mathsf{P}\,x:X}{x:\varphi \vdash \mathsf{P}\,x:[+X]\,\varphi}}{x:\varphi \vdash \mathsf{P}\,x:\varphi}\,(+X)}\,(\mathsf{Cut})$$

No strict inclusion! This derivation cannot count as a proof.

Completeness

Theorem: The calculus is complete for context-free processes.

- The proof works by building a derivation using property-checking games [Stirling 01].
- Finiteness of the derivation is ensured by an eager decomposition of processes.
- Modifiers are crucial to deal with problems introduced by the decomposition.

Related Work

Sequent calculus in [Dam, Gurov 00]:

- Explicit approximants are written as $\mu^\kappa X.\,\varphi$.
- $[X \geqslant \varphi]$ and $[+X]$ can be added as $\forall\kappa$ and $\forall(\kappa' < \kappa)$.
- Independently, [Fredlund 01] added these constructs.

Other methods of verifying context-free processes:

- [Burkart, Steffen 92]: Iterative algorithm
- [Walukiewicz 95]: Game based algorithm
- [Hungar, Steffen 94]: Tableau for the alternation-free μ-calculus

Conclusions and Future Work

We consider explicit approximants and modifiers and show how they can be used to overcome problems in compositional proof systems for temporal properties.

Further work:

- Investigation of the extended logic
- Completeness for richer classes of processes
- Completeness with respect to parameterised verification goals