Interface Abstraction for Compositional Verification

Dilian Gurov

Royal Institute of Technology, Stockholm

Marieke Huisman

INRIA Sophia-Antipolis

SEFM 05, Koblenz

September 9, 2005

Overview

1. (A Rather Lengthy) Motivation
2. Interface Behaviour
3. The Inlining Transformation
4. A Compositional Verification Method
5. Conclusions

Smart Cards and Security

Smart cards

- store privacy–sensitive data
- require strong guarantees of security: formal verification

Multiple interacting applets (e.g. JavaCard applets)

- communication via method invocation over shared interfaces
- example: electronic purse applet and several loyalties

Post-issuance loading

- ability to load new applets after the card has been issued to the user
- requires compositional verification

Compositional Verification

Compositional Verification Principle

$$\frac{\models A : \psi \qquad X : \psi \models X \otimes B : \phi}{\models A \otimes B : \phi}$$

premises: local property of A and correctness of decomposition

Scenarios for secure post-issuance loading

1. card issuer specifies ϕ and ψ and checks property decomposition;
   pre-load check of $\models A : \psi$
2. card issuer provides only ϕ, applet provider specifies ψ;
   pre-load check of $\models A : \psi$ and property decomposition

Maximal Models

In certain setups:

- property preserving simulation preorder
- for any formula ψ, the set of models for ψ has a maximal element $Max(\psi)$ wrt. the preorder: maximal model
- simulation preorder preserved by composition $\otimes$

Maximal Model Principle [Grumberg & Long '94]

$$\frac{\models Max(\psi) \otimes B : \phi}{X : \psi \models X \otimes B : \phi}$$

Compositional Verification Principle

$$\frac{\models A : \psi \qquad \models Max(\psi) \otimes B : \phi}{\models A \otimes B : \phi}$$

Previous Work

Theory [Sprenger, Huisman, Gurov: MEMOCODE'04]

- formal framework
- maximal model construction

Case Study [Huisman, Gurov, Sprenger, Chugunov: FASE'04]

- electronic purse with loyalty programmes
- by smart card provider Gemplus
- verified absence of illicit applet interactions

Models, Simulation and Logic

Applets unified treatment of structure and behaviour, control–flow based

Model Labelled transition system + Valuation

Simulation Preorder $\leq$ standard

Simulation Logic modal logic with box modalities and gfp recursion:

$$\phi ::= p \mid \neg p \mid X \mid \phi_1 \wedge \phi_2 \mid \phi_1 \vee \phi_2 \mid [a]\,\phi \mid \nu X.\phi$$

Maximal Models $Max(\psi)$

- exist
- exponential construction, lazy

Applet Structure

Applet $\mathcal{A}$

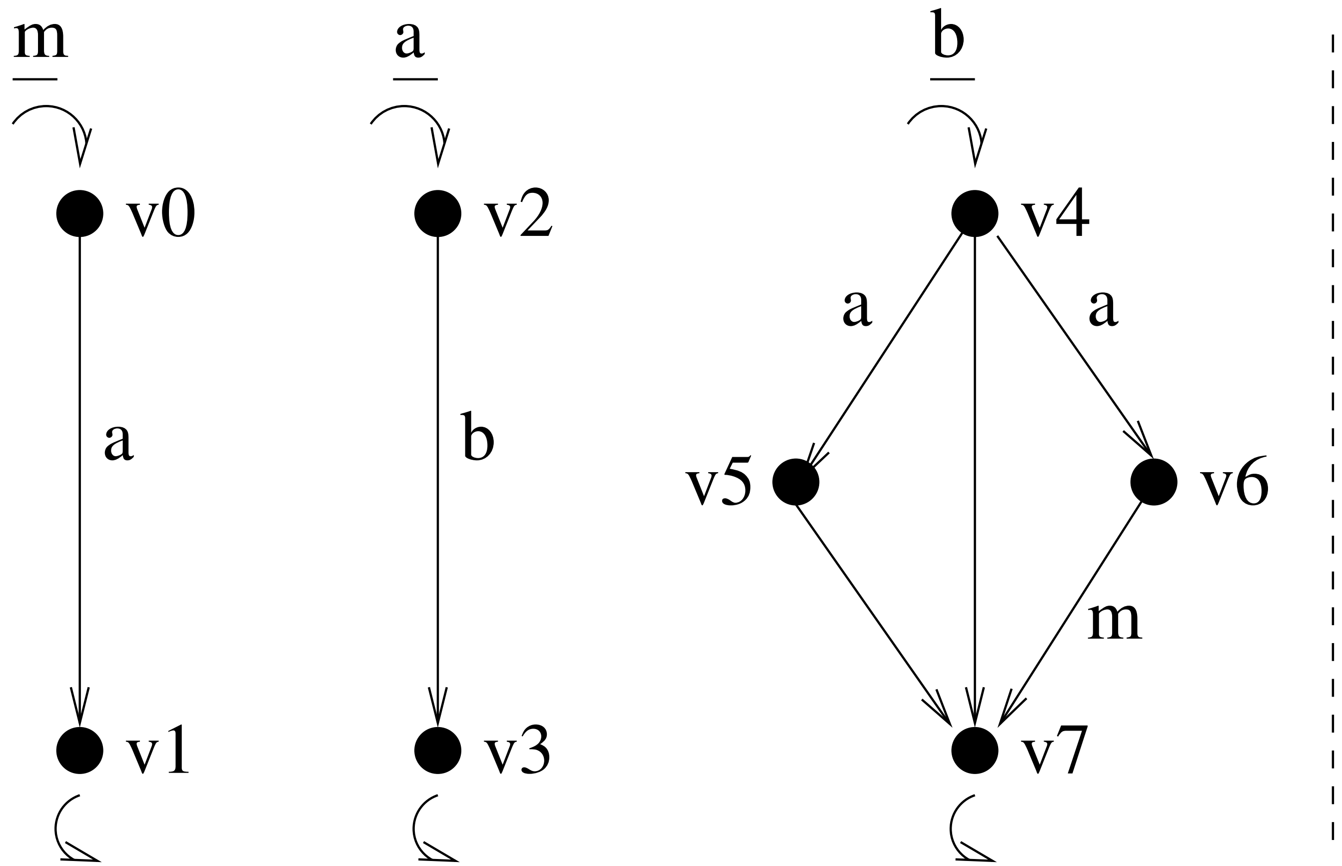
- control–flow graph represented as model
- applet composition $\uplus$
- structural simulation and properties

Maximal Model for property ψ is not necessarily a legal applet structure!

- interface $I = (I^+, I^-)$ of provided and required methods
- formula ϕ_I axiomatizing applets with interface I

Maximal Applet $Max_I(\psi)$

- is the maximal model $Max(\phi_I \wedge \psi)$


m
a
b
v0
v2
v4
a
b
a
a
v5
v6
m
v1
v3
v7

Applet Behaviour

- Applet structure $\mathcal{A}$ induces applet behaviour $b(\mathcal{A})$

  - configurations: pairs (v, σ) of control point and call stack

  - labels: ε, $m_1\,call\,m_2$, $m_2\,ret\,m_1$

  - transitions: standard, induced in a context–free manner

- Behavioural simulation and properties

  - applet interaction properties

- Applet behaviour is not axiomatizable within the logic...

  ...but at least structural simulation implies behavioural simulation!

Operational Semantics

(call)
$$\frac{m_1, m_2 \in I^+ \qquad v_1 \xrightarrow{m_2}_{m_1} v_1' \qquad v_2 \models m_2 \wedge e}{(v_1, \sigma) \xrightarrow{m_1 \text{call } m_2} (v_2, v_1' \cdot \sigma)}$$

(return)
$$\frac{m_1, m_2 \in I^+ \qquad v_2 \models m_2 \wedge r \qquad v_1 \models m_1}{(v_2, v_1 \cdot \sigma) \xrightarrow{m_2 \text{ ret } m_1} (v_1, \sigma)}$$

(transfer)
$$\frac{m \in I^+ \qquad v \rightarrow_m v'}{(v, \sigma) \xrightarrow{\varepsilon} (v', \sigma)}$$

Verification Method

Compositional Verification Principle

$$\frac{\mathcal{A} \models_s \sigma \qquad \mathcal{M}ax_{I_\mathcal{A}}(\sigma) \uplus \mathcal{B} \models_b \psi}{\mathcal{A} \uplus \mathcal{B} \models_b \psi} \ \mathcal{A} : I_\mathcal{A}$$

1. a) Specify global property ψ as a behavioural property

   b) For applet $\mathcal{A}$, specify local property σ as a structural property

2. Verify the correctness of the property decomposition:

   a) compute maximal applet $\mathcal{M}ax_{I_\mathcal{A}}(\sigma)$

   b) model check $\mathcal{M}ax_{I_\mathcal{A}}(\sigma) \uplus \mathcal{B} \models_b \psi$

3. When implementation of $\mathcal{A}$ available, verify $\mathcal{A} \models_s \sigma$

Main Shortcomings

1. Requires knowledge of the complete interface, but in a truly compositional setting we can only assume knowledge of the names of the public methods

2. Interfaces are significantly larger than public ones, which is critical for the applicability of the (exponential) maximal model construction

Present Paper

Public and Private Methods M a set of public methods

Transformation transforms applet $\mathcal{A}$ with interface (I^+, I^-) into a simulating applet $\alpha_M(\mathcal{A})$ with interface $(M, I^- - (I^+ - M))$

Modified CVP

$$\frac{\alpha_M(\mathcal{A}) \models_s \sigma \qquad \mathcal{M}ax_{I_{\alpha_M(\mathcal{A})}}(\sigma) \uplus \mathcal{B} \models_b \psi}{\mathcal{A} \uplus \mathcal{B} \models_b^{M \cup I_{\mathcal{B}}^+} \psi}$$

- simulation: w.r.t. public behaviour, or interface behaviour
- transformation: inlining of private methods

2. Interface Behaviour

An abstraction on applet behaviour wrt. $M \subseteq I_{\mathcal{A}}^{+}$

- keep configurations unchanged
- relabel configurations

  - current control is in the top–most public method of $v \cdot \sigma$
- relabel transitions accordingly

  - configuration–dependent relabelling
- denoted $b^{M}(\mathcal{A})$

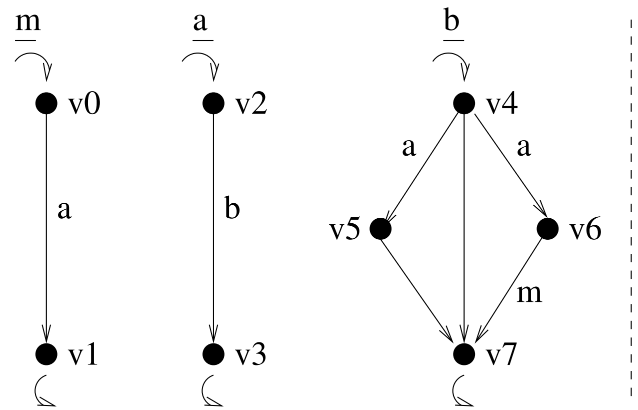
3. The Inlining Transformation

Inlining replace method call by method body

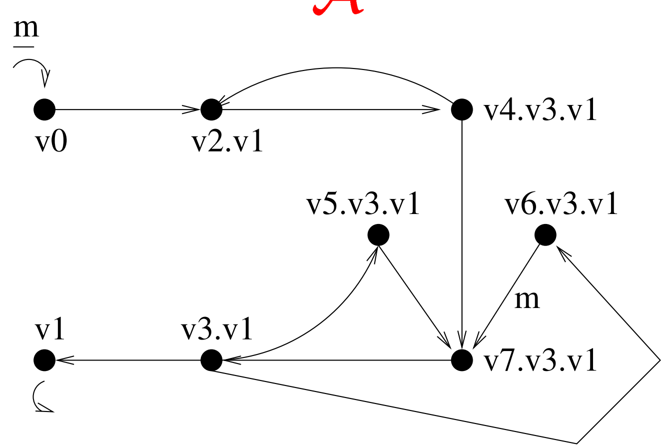
- need to: guarantee termination, prove simulation

Transformation For each (public) method $m \in M$

- execute m so that:
  - label local calls and returns by ε
  - treat calls to public methods as local transfer, but keep label
  - replace recursion by iteration
- result denoted $\alpha_M(\mathcal{A})$



$\mathcal{A}$



$\alpha_M(\mathcal{A})$

- introduces more interface behaviour!

Simulation Results

Theorem Let $\mathcal{A} : I$ and $M \subseteq I^+$.

Then $b^M(\mathcal{A}) \leq b(\alpha_M(\mathcal{A})) = b^M(\alpha_M(\mathcal{A}))$.

Last–call recursion call edges are followed by transfer edges only

Theorem Let $\mathcal{A} : I$ be last-call recursive, and $M \subseteq I^+$.

Then $b^M(\mathcal{A}) \equiv_w b(\alpha_M(\mathcal{A})) = b^M(\alpha_M(\mathcal{A}))$.

4. A Compositional Verification Method

Modified CVP

$$\frac{\alpha_M(\mathcal{A}) \models_s \sigma \qquad \mathcal{M}ax_{I_{\alpha_M(\mathcal{A})}}(\sigma) \uplus \mathcal{B} \models_b \psi}{\mathcal{A} \uplus \mathcal{B} \models_b^{M \cup I_{\mathcal{B}}^{+}} \psi}$$

1. a) Specify global property ψ as an interface behavioural property

   b) For applet $\mathcal{A}$, specify local property σ as a structural property of $\alpha_M(\mathcal{A})$

2. Verify the correctness of the property decomposition:

   a) compute maximal applet $\mathcal{M}ax_{I_{\alpha_M(\mathcal{A})}}(\sigma)$

   b) model check $\mathcal{M}ax_{I_{\alpha_M(\mathcal{A})}}(\sigma) \uplus \mathcal{B} \models_b \psi$

3. When implementation of $\mathcal{A}$ available:

   a) compute $\alpha_M(\mathcal{A})$

   b) verify $\alpha_M(\mathcal{A}) \models_s \sigma$

Practical Impact of Inlining

- Knowledge of public interfaces suffices for applying the verification method
- Reconsider the case study from [Huisman, Gurov, Sprenger, Chugunov: FASE'04]

	$\mathcal{M}ax(\sigma_L)$	in [HGSC'04]	$\mathcal{M}ax(\sigma_P)$	in [HGSC'04]
#nodes	8	474	8	2786
#edges	120	277 700	88	603 128
constr. time	0.05 s.	25 min.	0.05 s.	13 hrs.

- Some natural structural properties are only expressible as properties of the inlined applet

5. Conclusions

We presented

- Notion of interface behaviour
- Inlining transformation which
  - reduces applet interfaces to public interfaces
  - extends/preserves interface behaviour
  - supports compositional verification

Future work

- multi–threaded applets

New Slide

- blah