Types and Layered Logics for Program Verification

O. Shkaravska

Institute of Cybernetics
at Tallinn University of Technology

TYPES-2006, Nottingham

Outline

Strongest Assertions

Uustalu, Saabas – “Compositional Type Systems for Stack-Based Low-Level Languages”.
The low-level language with an operand stack

- **Val – bool**, **int.**
- a **state**:
  - labels, ℓ (in a program counter *pc*)
  - an operand stack *os*: **List**(**Val**)
  - a storage *st*: **Var** $\longrightarrow$ **Val**

Strongest assertions mirror operational semantics

$$\{pc = \ell \ \wedge \ os = n :: \zeta \ \wedge \ st = \sigma\}$$
$$\textbf{store } x$$
$$\{pc = \ell + 1 \ \wedge \ os = \zeta \ \wedge \ st = \sigma[x := n]\}$$

Strongest Assertions

Uustalu, Saabas – "Compositional Type Systems for Stack-Based Low-Level Languages".
The low-level language with an operand stack

- **Val** – **bool**, **int**.
- a **state**:
  - labels, ℓ (in a program counter *pc*)
  - an operand stack *os*: $\mathbf{List}(\mathbf{Val})$
  - a storage *st*: $\mathbf{Var} \longrightarrow \mathbf{Val}$

Strongest assertions mirror operational semantics

$$\{pc = \ell \ \wedge \ os = n :: \zeta \ \wedge \ st = \sigma\}$$
$$\textbf{store}\ x$$
$$\{pc = \ell + 1 \ \wedge \ os = \zeta \ \wedge \ st = \sigma[x := n]\}$$

Specialised (Abstracted) Assertions

Specify the property we are interested in.
Abstract from irrelevant details.
For instance: the special property of interest: stack error freedom.
The abstraction and its meaning:

- $abstr(3) = \mathbf{int}$, meaning of the abstraction $(|\mathbf{int}|) = \{\mathbf{int}\}$,
- $abstr(3 :: \zeta) = \mathbf{int} :: abstr(\zeta)$,
- $abstr(\zeta) = \star$, meaning $(|\star|) = \{\mathbf{int},\ \mathbf{bool}\}^{\star}$.

Specialised Assertion

$\{pc = \ell \ \wedge \ os = \tau :: \Psi\}\ \mathbf{store}\ x\ \{pc = \ell + 1 \ \wedge \ os = \Psi\}$

Specialised (Abstracted) Assertions

Specify the property we are interested in.
Abstract from irrelevant details.
For instance: the special property of interest: stack error freedom.
The abstraction and its meaning:

- $\mathit{abstr}\,(3) = \mathbf{int}$, meaning of the abstraction $(|\mathbf{int}|) = \{\mathbf{int}\}$,
- $\mathit{abstr}\,(3 :: \zeta) = \mathbf{int} :: \mathit{abstr}\,(\zeta)$,
- $\mathit{abstr}\,(\zeta) = \star$, meaning $(|\star|) = \{\mathbf{int},\ \mathbf{bool}\}^{\star}$.

Specialised Assertion

$\{pc = \ell \ \wedge \ os = \tau :: \Psi\}$ **store** x $\{pc = \ell + 1 \ \wedge \ os = \Psi\}$

Specialised (Abstracted) Assertions

Specify the property we are interested in.
Abstract from irrelevant details.
For instance: the special property of interest: stack error freedom.
The abstraction and its meaning:

- $abstr\,(3) = \mathbf{int}$, meaning of the abstraction $(|\mathbf{int}|) = \{\mathbf{int}\}$,
- $abstr\,(3 :: \zeta) = \mathbf{int} :: abstr\,(\zeta)$,
- $abstr\,(\zeta) = \star$, meaning $(|\star|) = \{\mathbf{int},\ \mathbf{bool}\}^{\star}$.

Specialised Assertion

$\{pc = \ell \ \wedge \ os = \tau :: \Psi\}$ **store** x $\{pc = \ell + 1 \ \wedge \ os = \Psi\}$

What is Soundness

What is soundness

$\{A\}\ c\ \{B\}$ is sound iff
it is provable from the logic of the strongest specifications together with the rule of consequence

$$\frac{\{A\}\ c\ \{B\} \qquad (A \longrightarrow B) \longrightarrow (A' \longrightarrow B')}{\{A'\}\ c\ \{B'\}}$$

Yet another definition – via abstract operational semantics?

If typing appears from abstract interpretation –

- $\{(\mid abstrA \mid)\}\ c\ \{(\mid abstrB \mid)\}$
- the *preservation of evaluation* principle:
  $(\ell, \zeta, \sigma),\ c \rightsquigarrow (\ell', \zeta', \sigma')$ implies
  $(\ell, abstr(\zeta),),\ c \rightsquigarrow (\ell', abstr(\zeta'),)$.

What is Soundness

What is soundness

$\{A\}\ c\ \{B\}$ is sound iff
it is provable from the logic of the strongest specifications together with the rule of consequence

$$\frac{\{A\}\ c\ \{B\} \qquad (A \longrightarrow B) \longrightarrow (A' \longrightarrow B')}{\{A'\}\ c\ \{B'\}}$$

Yet another definition – via abstract operational semantics?

If typing appears from abstract interpretation –

- $\{(\mid \mathit{abstrA} \mid)\}\ c\ \{(\mid \mathit{abstrB} \mid)\}$
- the *preservation of evaluation* principle:
  $(\ell,\ \zeta,\ \sigma),\ c\ \rightsquigarrow\ (\ell',\ \zeta',\ \sigma')$ implies
  $(\ell,\ \mathit{abstr}(\zeta),\ ),\ c\ \rightsquigarrow\ (\ell',\ \mathit{abstr}(\zeta'),\ ).$

What is Soundness

What is soundness

$\{A\}\ c\ \{B\}$ is sound iff
it is provable from the logic of the strongest specifications together with the rule of consequence

$$\frac{\{A\}\ c\ \{B\} \qquad (A \longrightarrow B) \longrightarrow (A' \longrightarrow B')}{\{A'\}\ c\ \{B'\}}$$

Yet another definition – via abstract operational semantics?

If typing appears from abstract interpretation –

- $\{(\mid \mathit{abstrA} \mid)\}\ c\ \{(\mid \mathit{abstrB} \mid)\}$
- the *preservation of evaluation* principle:
  $(\ell,\ \zeta,\ \sigma),\ c\ \rightsquigarrow\ (\ell',\ \zeta',\ \sigma')$ implies
  $(\ell,\ \mathit{abstr}(\zeta),\ ),\ c\ \rightsquigarrow\ (\ell',\ \mathit{abstr}(\zeta'),\ )$.

A "consequence" may be difficult to prove

An expression free subgoal

Consequence: $$\frac{\{A\}\ c\ \{B\} \qquad (A \longrightarrow B) \longrightarrow (A' \longrightarrow B')}{\{A'\}\ c\ \{B'\}}$$

The subgoal $(A \longrightarrow B) \longrightarrow (A' \longrightarrow B')$ may be difficult to prove.

The case of composition

$$\frac{\begin{array}{ll}\{A_1\}\ c_1\ \{B_1\} \qquad \{A_2\}\ c_2\ \{B_2\} & \\ A \longrightarrow A_1 & \text{OK} \\ B_1 \longrightarrow A_2 & \text{may be too strong} \\ B_2 \longrightarrow B & \text{may be too strong}\end{array}}{\{A\}\ c_1; c_2\ \{B\}}$$

A “consequence” may be difficult to prove

An expression free subgoal

Consequence: $\dfrac{\{A\}\ c\ \{B\} \qquad (A \longrightarrow B) \longrightarrow (A' \longrightarrow B')}{\{A'\}\ c\ \{B'\}}$

The subgoal $(A \longrightarrow B) \longrightarrow (A' \longrightarrow B')$ may be difficult to prove.

The case of composition

$$\dfrac{\begin{array}{l} \{A_1\}\ c_1\ \{B_1\} \qquad \{A_2\}\ c_2\ \{B_2\} \\ A \longrightarrow A_1 \quad \text{OK} \\ B_1 \longrightarrow A_2 \quad \text{may be too strong} \\ B_2 \longrightarrow B \quad \text{may be too strong} \end{array}}{\{A\}\ c_1; c_2\ \{B\}}$$

Modularised Subgoal

$$\frac{\{A_1\}\ c_1\ \{B_1\} \qquad \{A_2\}\ c_2\ \{B_2\} \\ A \longrightarrow A_1 \\ A \longrightarrow A_1 \longrightarrow B_1 \longrightarrow A_2 \\ A \longrightarrow A_1 \longrightarrow B_1 \longrightarrow A_2 \longrightarrow B_2 \longrightarrow B}{\{A\}\ c_1; c_2\ \{B\}}$$

It is sound!

Parametric Specialised Assertions

In fact,

$$\frac{\begin{array}{l}\{?A_1\}\ c_1\ \{?B_1\} \qquad \{?A_2\}\ c_2\ \{?B_2\} \\ ?A \longrightarrow ?A_1 \\ ?A \longrightarrow ?A_1 \longrightarrow ?B_1 \longrightarrow ?A_2 \\ ?A \longrightarrow ?A_1 \longrightarrow ?B_1 \longrightarrow ?A_2 \longrightarrow ?B_2 \longrightarrow ?B\end{array}}{\{?A\}\ c_1; c_2\ \{?B\}}$$

is sound!
That is we have a “template lemma”,
where parameters may be instantiated by arbitrary assertions ...

Summary

- The proof-of-the-concept template specialised logic have been designed.
- It helps in soundness proving for type systems.
- Future work
  - Testing ...
  - How do the **preservation of evaluation** for an abstract operational semantics and ***this* soundness** interplay?

Summary

- The proof-of-the-concept template specialised logic have been designed.
- It helps in soundness proving for type systems.

- Future work
  - Testing ...
  - How do the **preservation of evaluation** for an abstract operational semantics and ***this* soundness** interplay?

Summary

- The proof-of-the-concept template specialised logic have been designed.
- It helps in soundness proving for type systems.

- Future work
  - Testing ...
  - How do the **preservation of evaluation** for an abstract operational semantics and ***this* soundness** interplay?