# PROVING ABSTRACT NON-INTERFERENCE

**Roberto Giacobazzi and Isabella Mastroeni**

Dipartimento di Informatica

Università di Verona

Italy

CSL'04

Karpacz, September 21st, 2004

# The Problem

- The problem: Protect data confidentiality from erroneous/malicious attacks while data are processed
  - Typing of data (and variables) in *private* (H) and *public* (L);
  - *Non-Interference*: to prevent the results of the computation from leaking even partial information about private inputs!
    - *Explicit flow*: caused by directly passing private data to a public variable: $\mathtt{l} := 2 * \mathtt{h}$;
    - *Implicit flow*: arise from control structure of the program: **while** $\mathtt{h}$ **do** $\mathtt{l} := \mathtt{l} + 1; \mathtt{h} := \mathtt{h} - 1$.

# The Problem

- The problem: Protect data confidentiality from erroneous/malicious attacks while data are processed

- Goal: Automatically generate certificates about secure information flows
  - Design of accurate security polices
  - $\Rightarrow$ Static program analysis & verification techniques (types,CFA,DFA,...)

# The Problem

- The problem: Protect data confidentiality from erroneous/malicious attacks while data are processed
- Goal: Automatically generate certificates about secure information flows
- State of the art: Standard non-interference
  - No sensitive information can be disclosed
  - Any change upon confidential data has not to be revealed by public ones
  - Rigid security policy: L can flow into H but H cannot flow into L [Denning and Denning '77]

# The Problem

- The problem: Protect data confidentiality from erroneous/malicious attacks while data are processed
- Goal: Automatically generate certificates about secure information flows
- State of the art: Standard non-interference and abstract non-interference [Giacobazzi and Mastroeni, POPL'04]
  - Characterization of the secrecy degree of a program
  - H can flow into L unless a given property of H is disclosed
  - Weakening of standard non-interference

# AI: Lattice of Abstractions

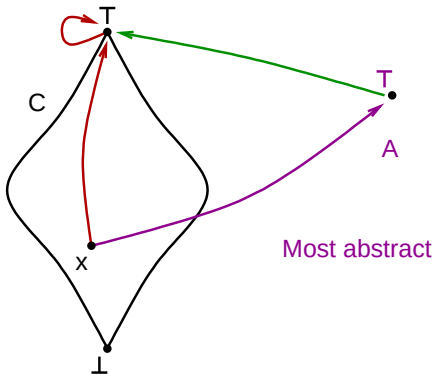
The concrete domain $< C, \leq, \wedge, \vee, \bot, \top >$ [Cousot & Cousot '79]

Lattice of abstract domains $\equiv Abs(C)$

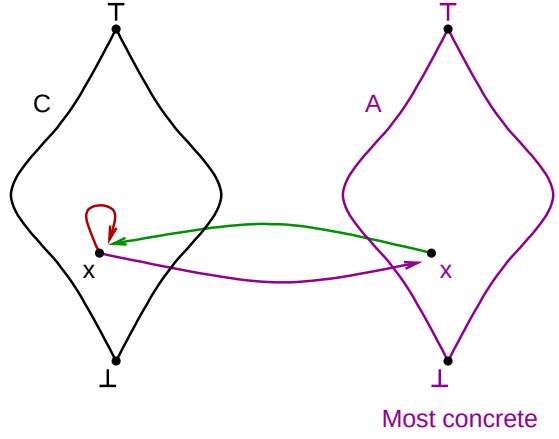
$< Abs(C), \sqsubseteq, \sqcap, \sqcup, \top, C >$

$A_1 \sqsubseteq A_2 \quad \Leftrightarrow \quad A_2 \subseteq A_1$ ($A_1$ more precise than $A_2$)

Top:




Bottom:



# Standard non-interference

*"One group of users [...] is noninterfering with another group of users if what the first group does [...] has no effect on what the second group of users can see"* [Goguen & Meseguer '82]

Standard non-interference

$$\forall l : \mathsf{L}, \forall h_1, h_2 : \mathsf{H}.\ [\![P]\!](h_1, l)^{\mathsf{L}} = [\![P]\!](h_2, l)^{\mathsf{L}}$$

# Standard non-interference

**Standard non-interference**

$$\forall l : \mathsf{L}, \forall h_1, h_2 : \mathsf{H}.\ [\![P]\!](h_1, l)^{\mathsf{L}} = [\![P]\!](h_2, l)^{\mathsf{L}}$$

EXAMPLE:

$$\textbf{while } h \textbf{ do } (l := l + 2;\ h := h - 1).$$

# Standard non-interference

**Standard non-interference**

$$\forall l : \mathsf{L}, \forall h_1, h_2 : \mathsf{H}.\ [\![P]\!](h_1, l)^{\mathsf{L}} = [\![P]\!](h_2, l)^{\mathsf{L}}$$

EXAMPLE:

**while** $h$ **do** $(l := l + 2;\ h := h - 1)$.

$$h = 0,\ l = 1 \ \rightsquigarrow\ l = 1$$
$$h = 1,\ l = 1 \ \rightsquigarrow\ l = 3$$
$$h = n,\ l = 1 \ \rightsquigarrow\ l = 1 + 2n$$

# Standard non-interference

**Standard non-interference**

$$\forall l : \mathsf{L}, \forall h_1, h_2 : \mathsf{H}.\ [\![P]\!](h_1, l)^{\mathsf{L}} = [\![P]\!](h_2, l)^{\mathsf{L}}$$

EXAMPLE:

**while** $h$ **do** $(l := l + 2;\ h := h - 1)$.

$$h = 0,\ l = 1 \ \rightsquigarrow\ l = 1$$
$$h = 1,\ l = 1 \ \rightsquigarrow\ l = 3$$
$$h = n,\ l = 1 \ \rightsquigarrow\ l = 1 + 2n$$

If $l$ is unchanged then $h$ is $0$!

$\rightsquigarrow$ There is an information flow from $h$ into $l$.

# Standard non-interference

**Standard non-interference**

$$\forall l : \mathsf{L}, \forall h_1, h_2 : \mathsf{H}.\ [\![P]\!](h_1, l)^{\mathsf{L}} = [\![P]\!](h_2, l)^{\mathsf{L}}$$

EXAMPLE:

**while** $h$ **do** $(l := l + 2;\ h := h - 1)$.

$$h = 0,\ l = 1 \rightsquigarrow l = 1$$
$$h = 1,\ l = 1 \rightsquigarrow l = 3$$
$$h = n,\ l = 1 \rightsquigarrow l = 1 + 2n$$

If $l$ is unchanged then $h$ is $0$!

$\rightsquigarrow$ There is an information flow from $h$ into $l$.

$\Rightarrow$ Note that if the input $l$ is even/odd then the output $l$ is even/odd!

# Abstracting non-interference I

Standard non-interference

$$\forall l : \mathsf{L}, \forall h_1, h_2 : \mathsf{H}.\ [\![P]\!](h_1, l)^{\mathsf{L}} = [\![P]\!](h_2, l)^{\mathsf{L}}$$

Consider $\alpha, \eta \in \mathit{Abs}(\wp(\mathbb{V}^{\mathsf{L}}))$:

# Abstracting non-interference I

Standard non-interference

$$\forall l : \mathsf{L}, \forall h_1, h_2 : \mathsf{H}.\ [\![P]\!](h_1, l)^{\mathsf{L}} = [\![P]\!](h_2, l)^{\mathsf{L}}$$

Consider $\alpha, \eta \in \mathit{Abs}(\wp(\mathbb{V}^{\mathsf{L}}))$:

*Narrow (abstract) non-interference* $[\eta]P(\alpha)$:

$$\eta(l_1) = \eta(l_2) \Rightarrow \alpha([\![P]\!](h_1, l_1)^{\mathsf{L}}) = \alpha([\![P]\!](h_2, l_2)^{\mathsf{L}})$$

- No change of H values and $\eta$-equivalent L values may affect the $\alpha$ abstraction of L outputs.
- Possible deceptive interference due to $\eta$-undistinguished L values!
- The more $\eta$ is precise the less deceptive interference appears

# Abstracting non-interference I

Standard non-interference

$$\forall l : \mathsf{L}, \forall h_1, h_2 : \mathsf{H}.\ [\![P]\!](h_1, l)^{\mathsf{L}} = [\![P]\!](h_2, l)^{\mathsf{L}}$$

Consider $\alpha, \eta \in \mathit{Abs}(\wp(\mathbb{V}^{\mathsf{L}}))$:

*Narrow (abstract) non-interference* $[\eta]P(\alpha)$:

$$\eta(l_1) = \eta(l_2) \Rightarrow \alpha([\![P]\!](h_1, l_1)^{\mathsf{L}}) = \alpha([\![P]\!](h_2, l_2)^{\mathsf{L}})$$

EXAMPLE: $[\mathtt{id}]P(\mathit{Par})$

$$P = \textbf{while } h \textbf{ do } (l := l + 2;\ h := h - 1).$$

$$h = 0,\ l = 1 \rightsquigarrow \mathit{Par}(l) = \mathsf{odd}$$
$$h = 1,\ l = 1 \rightsquigarrow \mathit{Par}(l) = \mathsf{odd}$$
$$h = n,\ l = 1 \rightsquigarrow \mathit{Par}(l) = \mathsf{odd}$$

# Abstracting non-interference I

Standard non-interference

$$\forall l : \mathsf{L}, \forall h_1, h_2 : \mathsf{H}.\ [\![P]\!](h_1, l)^{\mathsf{L}} = [\![P]\!](h_2, l)^{\mathsf{L}}$$

Consider $\alpha, \eta \in \mathit{Abs}(\wp(\mathbb{V}^{\mathsf{L}}))$:

*Narrow (abstract) non-interference* $[\eta]P(\alpha)$:

$$\eta(l_1) = \eta(l_2) \Rightarrow \alpha([\![P]\!](h_1, l_1)^{\mathsf{L}}) = \alpha([\![P]\!](h_2, l_2)^{\mathsf{L}})$$

EXAMPLE: $[\mathtt{id}]P(\mathit{Par})$

$$P = \textbf{while } h \textbf{ do } (l := l + 2;\ h := h - 1).$$

$$h = 0,\ l = 1 \rightsquigarrow \mathit{Par}(l) = \text{odd}$$
$$h = 1,\ l = 1 \rightsquigarrow \mathit{Par}(l) = \text{odd}$$
$$h = n,\ l = 1 \rightsquigarrow \mathit{Par}(l) = \text{odd}$$

If $l$ is odd/even then, independently from $h$, after the execution $l$ is odd/even!

$\rightsquigarrow$ There is not an information flow from $h$ into the parity of $l$.

# Abstracting non-interference I

Standard non-interference

$$\forall l : \mathsf{L}, \forall h_1, h_2 : \mathsf{H}.\ [\![P]\!](h_1, l)^{\mathsf{L}} = [\![P]\!](h_2, l)^{\mathsf{L}}$$

Consider $\alpha, \eta \in \mathit{Abs}(\wp(\mathbb{V}^{\mathsf{L}}))$:

*Narrow (abstract) non-interference* $[\eta]P(\alpha)$:

$$\eta(l_1) = \eta(l_2) \Rightarrow \alpha([\![P]\!](h_1, l_1)^{\mathsf{L}}) = \alpha([\![P]\!](h_2, l_2)^{\mathsf{L}})$$

EXAMPLE II: $[\mathit{Par}]P(\mathit{Sign})$

$$P = \quad l := 2 * l * h^2.$$

$$h = -3,\ l = -2\ (\mathit{Par}(-2) = \mathsf{even}) \rightsquigarrow \mathit{Sign}(l) = -$$
$$h = 1,\ l = -4\ (\mathit{Par}(-4) = \mathsf{even}) \rightsquigarrow \mathit{Sign}(l) = -$$

# Abstracting non-interference I

Standard non-interference

$$\forall l : \mathsf{L}, \forall h_1, h_2 : \mathsf{H}.\ [\![P]\!](h_1, l)^{\mathsf{L}} = [\![P]\!](h_2, l)^{\mathsf{L}}$$

Consider $\alpha, \eta \in \mathit{Abs}(\wp(\mathbb{V}^{\mathsf{L}}))$:

*Narrow (abstract) non-interference* $[\eta]P(\alpha)$:

$$\eta(l_1) = \eta(l_2) \Rightarrow \alpha([\![P]\!](h_1, l_1)^{\mathsf{L}}) = \alpha([\![P]\!](h_2, l_2)^{\mathsf{L}})$$

EXAMPLE II: $[\mathit{Par}]P(\mathit{Sign})$

$$P = \quad l := 2 * l * h^2.$$

$$h = 1,\ l = 4\ (\mathit{Par}(4) = \mathsf{even}) \rightsquigarrow \mathit{Sign}(l) = +$$
$$h = 1,\ l = -4\ (\mathit{Par}(-4) = \mathsf{even}) \rightsquigarrow \mathit{Sign}(l) = -$$

The sign of the output $l$ depends on the sign of the input $l$!

$\rightsquigarrow$ There is a DECEPTIVE FLOW!

# Abstracting non-interference I

Standard non-interference

$$\forall l : \mathsf{L}, \forall h_1, h_2 : \mathsf{H}.\ [\![P]\!](h_1, l)^{\mathsf{L}} = [\![P]\!](h_2, l)^{\mathsf{L}}$$

Consider $\alpha, \eta \in \mathit{Abs}(\wp(\mathbb{V}^{\mathsf{L}}))$:

*Narrow (abstract) non-interference* $[\eta]P(\alpha)$:

$$\eta(l_1) = \eta(l_2) \Rightarrow \alpha([\![P]\!](h_1, l_1)^{\mathsf{L}}) = \alpha([\![P]\!](h_2, l_2)^{\mathsf{L}})$$

EXAMPLE II: $[\mathit{Par}]P(\mathit{Sign})$

$$P = \quad l := 2 * l * h^2.$$

$$h = 1,\ l = 4\ (\mathit{Par}(4) = \text{even}) \rightsquigarrow \mathit{Sign}(l) = +$$
$$h = 1,\ l = -4\ (\mathit{Par}(-4) = \text{even}) \rightsquigarrow \mathit{Sign}(l) = -$$

The sign of the output $l$ depends on the sign of the input $l$!

$\rightsquigarrow$ There is a DECEPTIVE FLOW!

$\Rightarrow$ We compute the semantics on the concrete value of the input $l$!

# Abstracting non-interference II

Consider $\alpha, \eta \in \mathit{Abs}(\wp(\mathbb{V}^{\mathsf{L}}))$:

*Narrow (abstract) non-interference* $[\eta]\mathrm{P}(\alpha)$:
$\eta(l_1) = \eta(l_2) \Rightarrow \alpha([\![\mathrm{P}]\!](h_1, l_1)^{\mathsf{L}}) = \alpha([\![\mathrm{P}]\!](h_2, l_2)^{\mathsf{L}})$

# Abstracting non-interference II

Consider $\alpha, \eta \in \mathit{Abs}(\wp(\mathbb{V}^{\mathsf{L}}))$:

*Narrow (abstract) non-interference* $[\eta]\mathrm{P}(\alpha)$:
$\eta(l_1) = \eta(l_2) \Rightarrow \alpha([\![\mathrm{P}]\!](h_1, l_1)^{\mathsf{L}}) = \alpha([\![\mathrm{P}]\!](h_2, l_2)^{\mathsf{L}})$

*Abstract non-interference* $(\eta)\mathrm{P}(\alpha)$:
$\eta(l_1) = \eta(l_2) \Rightarrow \alpha([\![\mathrm{P}]\!](h_1, \eta(l_1))^{\mathsf{L}}) = \alpha([\![\mathrm{P}]\!](h_2, \eta(l_2))^{\mathsf{L}})$

- No change of H values may affect the $\alpha$ abstraction of L outputs.
- No deceptive interference due to L data

# Abstracting non-interference II

Consider $\alpha, \eta \in \mathit{Abs}(\wp(\mathbb{V}^{\mathsf{L}}))$:

*Narrow (abstract) non-interference* $[\eta]\mathrm{P}(\alpha)$:
$\eta(l_1) = \eta(l_2) \Rightarrow \alpha([\![\mathrm{P}]\!](h_1, l_1)^{\mathsf{L}}) = \alpha([\![\mathrm{P}]\!](h_2, l_2)^{\mathsf{L}})$

*Abstract non-interference* $(\eta)\mathrm{P}(\alpha)$:
$\eta(l_1) = \eta(l_2) \Rightarrow \alpha([\![\mathrm{P}]\!](h_1, \eta(l_1))^{\mathsf{L}}) = \alpha([\![\mathrm{P}]\!](h_2, \eta(l_2))^{\mathsf{L}})$

EXAMPLE: $(\mathit{Par})\mathrm{P}(\mathit{Sign})$

$$\mathrm{P} = \quad l := 2 * l * h^2.$$

$h = -3,\ \mathit{Par}(l) = \text{even} \rightsquigarrow \mathit{Sign}(l) = \text{I don't know}$
$h = 1,\ \mathit{Par}(l) = \text{even} \rightsquigarrow \mathit{Sign}(l) = \text{I don't know}$

# Abstracting non-interference II

Consider $\alpha, \eta \in \mathit{Abs}(\wp(\mathbb{V}^{\mathsf{L}}))$:

*Narrow (abstract) non-interference* $[\eta]\mathrm{P}(\alpha)$:
$\eta(l_1) = \eta(l_2) \Rightarrow \alpha([\![\mathrm{P}]\!](h_1, l_1)^{\mathsf{L}}) = \alpha([\![\mathrm{P}]\!](h_2, l_2)^{\mathsf{L}})$

*Abstract non-interference* $(\eta)\mathrm{P}(\alpha)$:
$\eta(l_1) = \eta(l_2) \Rightarrow \alpha([\![\mathrm{P}]\!](h_1, \eta(l_1))^{\mathsf{L}}) = \alpha([\![\mathrm{P}]\!](h_2, \eta(l_2))^{\mathsf{L}})$

EXAMPLE: $(\mathit{Par})\mathrm{P}(\mathit{Sign})$

$$\mathrm{P} = \quad l := 2 * l * h^2.$$

$h = -3$, $\mathit{Par}(l) = \text{even} \rightsquigarrow \mathit{Sign}(l) = \text{I don't know}$
$h = 1$, $\mathit{Par}(l) = \text{even} \rightsquigarrow \mathit{Sign}(l) = \text{I don't know}$

$\rightsquigarrow$ There is not information flow from $h$ into the sign of $l$.

# Deriving output attackers

Abstract interpretation provides advanced methods for designing abstractions (refinement, simplification, compression ...) [Giacobazzi & Ranzato ’97]

Designing abstractions = designing attackers

# Deriving output attackers

Abstract interpretation provides advanced methods for designing abstractions (refinement, simplification, compression ...) [Giacobazzi & Ranzato '97]

Designing abstractions = designing attackers

- Characterize the most concrete $\alpha$ such that $(\eta)\mathrm{P}(\alpha)$

  [The most powerful *output* attacker]

# Deriving output attackers

The following theorems hold: Consider $\eta \in \mathit{Abs}(\wp(\mathbb{V}^{\mathsf{L}}))$:

- We characterize the function $\lambda\eta.\ [\eta][\![\mathrm{P}]\!](\mathtt{id})$ whose result is $\bigsqcap \left\{ \beta \;\middle|\; [\eta]\mathrm{P}(\beta) \right\}$.

# Deriving output attackers

The following theorems hold: Consider $\eta \in \mathit{Abs}(\wp(\mathbb{V}^{\mathsf{L}}))$:

- We characterize the function $\lambda\eta.\ [\eta][\![\mathrm{P}]\!](\mathtt{id})$ whose result is $\bigsqcap \left\{ \beta \;\middle|\; [\eta]\mathrm{P}(\beta) \right\}$.
- We characterize the function $\lambda\eta.\ (\eta)[\![\mathrm{P}]\!](\mathtt{id})$ whose result is $\bigsqcap \left\{ \beta \;\middle|\; (\eta)\mathrm{P}(\beta) \right\}$.

# Deriving output attackers

The following theorems hold: Consider $\eta \in \mathit{Abs}(\wp(\mathbb{V}^{\mathsf{L}}))$:

- We characterize the function $\lambda\eta.\ [\eta][\![\mathrm{P}]\!](\mathtt{id})$ whose result is $\bigsqcap \left\{ \beta \;\middle|\; [\eta]\mathrm{P}(\beta) \right\}$.
- We characterize the function $\lambda\eta.\ (\eta)[\![\mathrm{P}]\!](\mathtt{id})$ whose result is $\bigsqcap \left\{ \beta \;\middle|\; (\eta)\mathrm{P}(\beta) \right\}$.

$\Rightarrow$ This would provide a certificate for security with a fixed input observation.

# Proving Abstract Non-Interference

- We introduce a compositional proof-system for certifying abstract non-interference;

# Proving Abstract Non-Interference

- We introduce a compositional proof-system for certifying abstract non-interference;
  - PROOF-SYSTEM OF INVARIANTS $\mathcal{I}$: $\{\rho\}_L\ c\ \{\rho\}_L$ means that $c$ is $\rho$-observably equivalent to the statement **nil**:

$$\{\rho\}_L\ c\ \{\rho\}_L \text{ iff } \rho([\![c]\!](h,l)^{\downarrow_L}) = \rho(l)$$

# Proving Abstract Non-Interference

- We introduce a compositional proof-system for certifying abstract non-interference;
  - PROOF-SYSTEM OF INVARIANTS $\mathcal{I}$: $\{\rho\}_{\mathsf{L}}\ c\ \{\rho\}_{\mathsf{L}}$ means that $c$ is $\rho$-observably equivalent to the statement **nil**:

$$\{\rho\}_{\mathsf{L}}\ c\ \{\rho\}_{\mathsf{L}} \text{ iff } \rho([\![c]\!](h, l)^{\mathsf{L}}) = \rho(l)$$

  - PROOF-SYSTEM FOR NARROW NON-INTERFERENCE $\mathcal{N}$: Syntax-driven certification of narrow non-interference for deterministic languages;

# Proving Abstract Non-Interference

- We introduce a compositional proof-system for certifying abstract non-interference;
  - PROOF-SYSTEM OF INVARIANTS $\mathcal{I}$: $\{\rho\}_L\ c\ \{\rho\}_L$ means that $c$ is $\rho$-observably equivalent to the statement **nil**:

$$\{\rho\}_L\ c\ \{\rho\}_L \text{ iff } \rho([\![c]\!](h,l)^{\downarrow_L}) = \rho(l)$$

  - PROOF-SYSTEM FOR NARROW NON-INTERFERENCE $\mathcal{N}$: Syntax-driven certification of narrow non-interference for deterministic languages;
  - PROOF-SYSTEM FOR ABSTRACT NON-INTERFERENCE $\mathcal{A}$: Syntax-driven certification of abstract non-interference for deterministic languages.

# The invariants proof-system $\mathcal{I}$

**I1:** $\{\top\}_L\ c\ \{\top\}_L$

**I2:** $\{\rho\}_L$ **nil** $\{\rho\}_L$

**I3:**
$$\frac{x : H}{\{\rho\}_L\ x := e\ \{\rho\}_L}$$

**I4:**
$$\frac{\{\rho\}\ \langle e, x\rangle\ \{\rho\},\ x : L}{\{\rho\}_L\ x := e\ \{\rho\}_L}$$

**I5:**
$$\frac{\{\rho\}_L\ c_1\ \{\rho\}_L\,,\ \{\rho\}_L\ c_2\ \{\rho\}_L}{\{\rho\}_L\ c_1\,;c_2\ \{\rho\}_L}$$

**I6:**
$$\frac{\{\rho\}_L\ c\ \{\rho\}_L}{\{\rho\}_L\ \textbf{while}\ x\ \textbf{do}\ c\ \textbf{endw}\ \{\rho\}_L}$$

**I7:**
$$\frac{\{\rho'\}_L\ c\ \{\rho'\}_L\,,\ \rho' \sqsubseteq \rho}{\{\rho\}_L\ c\ \{\rho\}_L}$$

Next Table

# The assignment

$$\frac{\{\rho\}\,\langle e, x\rangle\,\{\rho\},\ x : L}{\{\rho\}_L\ x := e\ \{\rho\}_L}$$

where

$$\{\rho\}\,\langle e, x\rangle\,\{\rho\} \text{ iff } \rho(\mathcal{E}[\![e]\!](h, l)) = \rho(l_{|_x}).$$

# The assignment

$$\frac{\{\rho\}\,\langle e, x\rangle\,\{\rho\},\ x : L}{\{\rho\}_L\ x := e\ \{\rho\}_L}$$

where

$$\{\rho\}\,\langle e, x\rangle\,\{\rho\} \text{ iff } \rho(\mathcal{E}[\![e]\!](h, l)) = \rho(l_{|_x}).$$

EXAMPLE:

Let $e = l + 2$. Then $\not\models \{Sign\}\,\langle e, l\rangle\,\{Sign\}$ since if $l = -1$

$$Sign(l + 2) = Sign(1) = + \neq Sign(l) = -$$

# The assignment

$$\frac{\{\rho\}\,\langle e, x\rangle\,\{\rho\},\ x : \mathsf{L}}{\{\rho\}_{\mathsf{L}}\ x := e\ \{\rho\}_{\mathsf{L}}}$$

where

$$\{\rho\}\,\langle e, x\rangle\,\{\rho\} \text{ iff } \rho(\mathcal{E}[\![e]\!](h, l)) = \rho(l_{|_x}).$$

EXAMPLE:

Let $e = l + 2$. We have $\models \{Par\}\,\langle e, l\rangle\,\{Par\}$.
Consider $c = \ l := l + 2$, we obtain that

$$\{Par\}_{\mathsf{L}}\ l := l + 2\ \{Par\}_{\mathsf{L}}$$

Return

# The sequential composition

$$\frac{\{\rho\}_L\ c_1\ \{\rho\}_L,\ \{\rho\}_L\ c_2\ \{\rho\}_L}{\{\rho\}_L\ c_1;c_2\ \{\rho\}_L}$$

# The sequential composition

$$\frac{\{\rho\}_L\ c_1\ \{\rho\}_L,\ \{\rho\}_L\ c_2\ \{\rho\}_L}{\{\rho\}_L\ c_1; c_2\ \{\rho\}_L}$$

EXAMPLE:

Let $c = \ l := l + 2;\ h := h + 1$.

$$\Rightarrow \begin{cases} \models \{Par\}_L\ l := l + 2\ \{Par\}_L \text{ by Rule I4} \\ \models \{Par\}_L\ h := h + 1\ \{Par\}_L \text{ by Rule I3} \end{cases}$$

# The sequential composition

$$\frac{\{\rho\}_L\ c_1\ \{\rho\}_L,\ \{\rho\}_L\ c_2\ \{\rho\}_L}{\{\rho\}_L\ c_1; c_2\ \{\rho\}_L}$$

EXAMPLE:

Let $c = \ l := l + 2;\ h := h + 1$.

$$\Rightarrow \begin{cases} \models \{Par\}_L\ l := l + 2\ \{Par\}_L \text{ by Rule I4} \\ \models \{Par\}_L\ h := h + 1\ \{Par\}_L \text{ by Rule I3} \end{cases}$$

$$\Downarrow$$

$$\models \{Par\}_L\ l := l + 2;\ h := h + 1\ \{Par\}_L$$

Return

# The proof-system $\mathcal{N}$

$$\textbf{N0:}\ \frac{[\eta][\![c]\!](\mathtt{id}) \sqsubseteq \rho}{[\eta]c(\rho)} \qquad \textbf{N1:}\ [\eta]c(\top) \qquad \textbf{N2:}\ \frac{\Pi(\eta) \sqsubseteq \Pi(\rho)}{[\eta]\textbf{nil}(\rho)}$$

$$\textbf{N3:}\ \frac{[\eta]e(\rho),\ [\Pi(\eta) \sqsubseteq \Pi(\rho)],\ x : \mathtt{L}}{[\eta]x := e(\rho)} \qquad \textbf{N4:}\ \frac{x : \mathtt{H},\ \Pi(\eta) \sqsubseteq \Pi(\rho)}{[\eta]x := e(\rho)}$$

$$\textbf{N5:}\ \frac{[\eta]c_1(\rho),\ [\rho]c_2(\beta)}{[\eta]c_1 ; c_2(\beta)} \qquad \textbf{N6:}\ \frac{\{\rho\}_{\mathtt{L}}\ c\ \{\rho\}_{\mathtt{L}}}{[\rho]\textbf{while } x \textbf{ do } c \textbf{ endw}(\rho)} \qquad \textbf{N7:}\ \frac{\forall i \in I\,.\,[\eta]c(\rho_i)}{[\eta]c(\bigsqcup_{i\in I} \rho_i)}$$

$$\textbf{N8:}\ \frac{\forall i \in I\,.\,[\eta]c(\rho_i)}{[\eta]c(\bigsqcap_{i\in I} \rho_i)} \qquad \textbf{N9:}\ \frac{[\eta']c(\rho'),\ \eta \sqsubseteq \eta',\ \rho' \sqsubseteq \rho}{[\eta]c(\rho)}$$

Next Table

# The low assignment

$$\frac{[\eta]e(\rho),\ [\Pi(\eta) \sqsubseteq \Pi(\rho)],\ x : \mathsf{L}}{[\eta]x := e(\rho)}$$

where

$$[\eta]e(\rho) \text{ iff } \rho(\mathcal{E}[\![e]\!](h_1, l_1)) = \rho(\mathcal{E}[\![e]\!](h_2, l_2)).$$

# The low assignment

$$\frac{[\eta]e(\rho),\ [\Pi(\eta) \sqsubseteq \Pi(\rho)],\ x : L}{[\eta]x := e(\rho)}$$

where

$$[\eta]e(\rho) \text{ iff } \rho(\mathcal{E}[\![e]\!](h_1, l_1)) = \rho(\mathcal{E}[\![e]\!](h_2, l_2)).$$

EXAMPLE:

Let $c = \ l_1 := 2 * h * l_2$.
Then $\not\models [\top] l_1 := 2 * h * l_2\,(\mathit{Par})$ since

$$\mathit{Par}([\![l_1 := 2 * h * l_2]\!](h, \langle l_1, 3\rangle)^L) = \langle \text{even}, \text{odd}\rangle \neq$$
$$\mathit{Par}([\![l_1 := 2 * h * l_2]\!](h, \langle l_1, 2\rangle)^L) = \langle \text{even}, \text{even}\rangle$$

This because $\top(2) = \top(3)$ but $\mathit{Par}(2) \neq \mathit{Par}(3)$.

# The low assignment

$$\frac{[\eta]e(\rho),\ [\Pi(\eta) \sqsubseteq \Pi(\rho)],\ x : L}{[\eta]x := e(\rho)}$$

where

$$[\eta]e(\rho) \text{ iff } \rho(\mathcal{E}[\![e]\!](h_1, l_1)) = \rho(\mathcal{E}[\![e]\!](h_2, l_2)).$$

NOTE: If there's only one low variable the condition $\Pi(\eta) \sqsubseteq \Pi(\rho)$ is not necessary.

EXAMPLE:

Consider $c = \ l := 2 * h$

$$[\top]2 * h(\mathit{Par}) \quad \Rightarrow \quad [\top]l := 2 * h(\mathit{Par})$$

Return

# The high assignment

$$\frac{x : \mathsf{H},\ \Pi(\eta) \sqsubseteq \Pi(\rho)}{[\eta]x := e(\rho)}$$

# The high assignment

$$\frac{x : H,\ \Pi(\eta) \sqsubseteq \Pi(\rho)}{[\eta]x := e(\rho)}$$

EXAMPLE:

Let $c = \ h := h + 1$. Then

$$\rho([\![h := h + 1]\!](h_1, l_1)^{\downarrow}) = \rho(l_1)$$
$$\rho([\![h := h + 1]\!](h_2, l_2)^{\downarrow}) = \rho(l_2)$$

Therefore

$$[\eta]h := h + 1(\rho) \quad \Leftrightarrow \quad (\eta(l_1) = \eta(l_2) \ \Rightarrow \ \rho(l_1) = \rho(l_2))$$
$$\Leftrightarrow \quad \Pi(\eta) \sqsubseteq \Pi(\rho)$$

Return

# The proof-system $\mathcal{A}$

**A0:** $\dfrac{(\eta)[\![c]\!](\mathtt{id}) \sqsubseteq \rho}{(\eta)c(\rho)}$

**A1:** $(\eta)c(\top)$

**A2:** $(\eta)\mathbf{nil}(\rho)$

**A3:** $\dfrac{(\eta)e(\rho),\ x : \mathsf{L}}{(\eta)x := e(\rho)}$

**A4:** $\dfrac{x : \mathsf{H}}{(\eta)x := e(\rho)}$

**A5:** $\dfrac{(\eta)c_1(\curlyvee(\rho)),\ [\rho]c_2(\curlyvee(\beta))}{(\eta)c_1;c_2(\curlyvee(\beta))}$

**A6:** $\dfrac{\{\rho\}_{\mathsf{L}}\ c\ \{\rho\}_{\mathsf{L}},\ x : \mathsf{H}}{(\rho)\mathbf{while}\ x\ \mathbf{do}\ c\ \mathbf{endw}(\rho)}$

**A7:** $\dfrac{(\eta)c(\rho),\ x : \mathsf{L}}{(\eta)\mathbf{while}\ x\ \mathbf{do}\ c\ \mathbf{endw}(\rho)}$

**A8:** $\dfrac{(\eta)c(\rho'),\ \rho' \sqsubseteq \rho}{(\eta)c(\rho)}$

**A9:** $\dfrac{\forall i \in I\,.\,(\eta)c(\rho_i)}{(\eta)c(\bigsqcup_{i\in I}\rho_i)}$

**A10:** $\dfrac{\forall i \in I\,.\,(\eta)c(\rho_i)}{(\eta)c(\bigsqcap_{i\in I}\rho_i)}$

# The high assignment

$$\frac{x : H}{(\eta)x := e(\rho)}$$

# The high assignment

$$\frac{x : H}{(\eta)x := e(\rho)}$$

EXAMPLE:

Let $c = \ h := h + 1$. Then

$$\rho([\![h := h + 1]\!](h_1, \eta(l_1))^{\downarrow}) = \rho(\eta(l_1))$$
$$\rho([\![h := h + 1]\!](h_2, \eta(l_2))^{\downarrow}) = \rho(\eta(l_2))$$

Therefore

$$[\eta]h := h + 1(\rho) \quad \Leftrightarrow \quad (\eta(l_1) = \eta(l_2) \ \Rightarrow \ \rho(\eta(l_1)) = \rho(\eta(l_2)))$$

Return

# The concatenation

$$\frac{(\eta)c_1(\Upsilon(\rho)),\ [\rho]c_2(\Upsilon(\beta))}{(\eta)c_1;c_2(\Upsilon(\beta))}$$

# The concatenation

$$\frac{(\eta)c_1(\Upsilon(\rho)),\ [\rho]c_2(\Upsilon(\beta))}{(\eta)c_1;c_2(\Upsilon(\beta))}$$

EXAMPLE:

Let $\rho = \{\mathbb{Z}, 4\mathbb{Z}, 4\mathbb{Z}+1, 4\mathbb{Z}+2, 4\mathbb{Z}+3, \varnothing\}$ and

$$c = c_1;\ c_2 = \left[\begin{array}{l} l := (h \bmod 2)(2l \bmod 4) + (1-(h \bmod 2))(l \bmod 2 + 1); \\ l := (l \bmod 2) * 4h + (1-(l \bmod 2)) * (4h+1) \end{array}\right.$$

then

$$h \in 2\mathbb{Z} \ \Rightarrow\ \rho([\![c_1]\!](h,\mathbb{Z})^{\mathsf{L}}) = \rho(\{1,2\}) = \mathbb{Z}$$
$$h \in 2\mathbb{Z}+1 \ \Rightarrow\ \rho([\![c_1]\!](h,\mathbb{Z})^{\mathsf{L}}) = \rho(\{0,2\}) = \mathbb{Z}$$

$$\Rightarrow\ (\top)c_1(\rho)$$

# The concatenation

$$\frac{(\eta)c_1(\gamma(\rho)),\ [\rho]c_2(\gamma(\beta))}{(\eta)c_1;c_2(\gamma(\beta))}$$

EXAMPLE:

Let $\rho = \{\mathbb{Z}, 4\mathbb{Z}, 4\mathbb{Z}+1, 4\mathbb{Z}+2, 4\mathbb{Z}+3, \varnothing\}$ and

$$c = c_1;\ c_2 = \left[\begin{array}{l} l := (h \bmod 2)(2l \bmod 4) + (1-(h \bmod 2))(l \bmod 2 + 1); \\ l := (l \bmod 2) * 4h + (1-(l \bmod 2)) * (4h+1) \end{array}\right.$$

$(\top)c_1(\rho)$ and

$$l \in 2\mathbb{Z} \ \Rightarrow\ \rho([\![c_2]\!](h,l)^{\downarrow}) = \rho(\{4h+1\}) = 4\mathbb{Z}+1$$
$$l \in 2\mathbb{Z}+1 \ \Rightarrow\ \rho([\![c_2]\!](h,l)^{\downarrow}) = \rho(\{4h\}) = 4\mathbb{Z}$$

$$\Rightarrow\ [\rho]c_2(\rho)$$

# The concatenation

$$\frac{(\eta)c_1(\Upsilon(\rho)),\ [\rho]c_2(\Upsilon(\beta))}{(\eta)c_1;c_2(\Upsilon(\beta))}$$

EXAMPLE:

Let $\rho = \{\mathbb{Z}, 4\mathbb{Z}, 4\mathbb{Z}+1, 4\mathbb{Z}+2, 4\mathbb{Z}+3, \varnothing\}$ and

$$c = c_1;\ c_2 = \left[\begin{array}{l} l := (h \bmod 2)(2l \bmod 4) + (1-(h \bmod 2))(l \bmod 2 + 1); \\ l := (l \bmod 2) * 4h + (1-(l \bmod 2)) * (4h+1) \end{array}\right.$$

$(\top)c_1(\rho)$ and $[\rho]c_2(\rho)$, but

$$h \in 2\mathbb{Z} \ \Rightarrow\ \rho([\![c_1;c_2]\!](h,\mathbb{Z})^{\downarrow}) = \rho(\{4h, 4h+1\}) = \mathbb{Z}$$
$$h \in 2\mathbb{Z}+1 \ \Rightarrow\ \rho([\![c_1;c_2]\!](h,\mathbb{Z})^{\downarrow}) = \rho(\{4h+1\}) = 4\mathbb{Z}+1$$

$$\Rightarrow \not\models (\top)c(\rho)$$

Return

# Theorems

$\mathcal{T}1$) The system $\mathcal{I}$ is correct.

# Theorems

$\mathcal{T}1$) The system $\mathcal{I}$ is correct.

$\mathcal{T}2$) The system $\mathcal{N}$ is complete but $\mathcal{N} \setminus \left\{ \dfrac{[\eta][\![c]\!](\mathtt{id}) \sqsubseteq \rho}{[\eta]c(\rho)} \right\}$ is correct.

EXAMPLE:

Let $\rho = \{2\mathbb{Z}\} \cup \left\{ \{n\} \,\middle|\, n \in 2\mathbb{Z}+1 \right\}$ and

$$P = \ l := 4 * h^2 + 4;\ c$$

where $c =$ **while** $h$ **do** $l := l \bmod 4;\ h := 0$ **endw**.

We have $[\top]l := 4 * h^2 + 4(\rho)$ and $[\top]P(\rho)$
But $\not\models [\rho]$**while** $h$ **do** $l := l \bmod 4;\ h := 0$ **endw**$(\rho)$:

$$\rho([\![c]\!](0,5)^{\mathsf{L}}) = 5 \neq \rho([\![c]\!](1,5)^{\mathsf{L}}) = 1$$

# Theorems

$\mathcal{T}1$) The system $\mathcal{I}$ is correct.

$\mathcal{T}2$) The system $\mathcal{N}$ is complete but $\mathcal{N} \setminus \left\{ \dfrac{[\eta][\![c]\!](\mathtt{id}) \sqsubseteq \rho}{[\eta]c(\rho)} \right\}$ is correct.

$\mathcal{T}3$) The system $\mathcal{A}$ is complete but $\mathcal{A} \setminus \left\{ \dfrac{(\eta)[\![c]\!](\mathtt{id}) \sqsubseteq \rho}{(\eta)c(\rho)} \right\}$ is correct.

EXAMPLE:

Let $\rho = \{\mathbb{Z}, 2\mathbb{Z}, 4\mathbb{Z}, \varnothing\}$ and

$$P = \textbf{ while } h \textbf{ do } l := (l \bmod 4) * (l \div 4);\ h := 0 \textbf{ endw}$$

Then $(\rho)P(\rho)$: $\rho([\![P]\!](h, 2\mathbb{Z})^{\mathsf{L}}) = 2\mathbb{Z}$; But $\not\models \{\rho\}_{\mathsf{L}}\ P\ \{\rho\}_{\mathsf{L}}$:

$$\rho([\![c]\!](1,2)^{\mathsf{L}}) = \rho(0) = 4\mathbb{Z} \neq \rho(2) = 2\mathbb{Z}$$

# Theorems

$\mathcal{T}1$) The system $\mathcal{I}$ is correct.

$\mathcal{T}2$) The system $\mathcal{N}$ is complete but $\mathcal{N} \setminus \left\{ \dfrac{[\eta][\![c]\!](\mathtt{id}) \sqsubseteq \rho}{[\eta]c(\rho)} \right\}$ is correct.

$\mathcal{T}3$) The system $\mathcal{A}$ is complete but $\mathcal{A} \setminus \left\{ \dfrac{(\eta)[\![c]\!](\mathtt{id}) \sqsubseteq \rho}{(\eta)c(\rho)} \right\}$ is correct.

$\mathcal{T}4$) The system $\mathcal{N}$ is stronger than $\mathcal{A}$.

EXAMPLE:

$$P = \ h := h + 1;\ l := 2 * h$$

Then $\models [\mathit{Sign}]P(\mathit{Par})$ but $\not\vdash_{\mathcal{N}} [\mathit{Sign}]P(\mathit{Par})$

# Theorems

$\mathcal{T}1$) The system $\mathcal{I}$ is correct.

$\mathcal{T}2$) The system $\mathcal{N}$ is complete but $\mathcal{N} \setminus \left\{ \dfrac{[\eta][\![c]\!](\mathtt{id}) \sqsubseteq \rho}{[\eta]c(\rho)} \right\}$ is correct.

$\mathcal{T}3$) The system $\mathcal{A}$ is complete but $\mathcal{A} \setminus \left\{ \dfrac{(\eta)[\![c]\!](\mathtt{id}) \sqsubseteq \rho}{(\eta)c(\rho)} \right\}$ is correct.

$\mathcal{T}4$) The system $\mathcal{N}$ is stronger than $\mathcal{A}$.

EXAMPLE:

$$P = \ h := h + 1;\ l := 2 * h$$

Then $\models [Sign]P(Par)$ but $\not\vdash_{\mathcal{N}} [Sign]P(Par)$ :

$Sign(2) = Sign(3)$ and $\quad Par([\![h := h + 1]\!](h, 3)^{\mathsf{L}}) = Par(3) = \text{odd} \neq$
$Par([\![h := h + 1]\!](h, 2)^{\mathsf{L}}) = Par(2) = \text{even}$

# Theorems

$\mathcal{T}1$) The system $\mathcal{I}$ is correct.

$\mathcal{T}2$) The system $\mathcal{N}$ is complete but $\mathcal{N} \setminus \left\{ \dfrac{[\eta][\![c]\!](\mathtt{id}) \sqsubseteq \rho}{[\eta]c(\rho)} \right\}$ is correct.

$\mathcal{T}3$) The system $\mathcal{A}$ is complete but $\mathcal{A} \setminus \left\{ \dfrac{(\eta)[\![c]\!](\mathtt{id}) \sqsubseteq \rho}{(\eta)c(\rho)} \right\}$ is correct.

$\mathcal{T}4$) The system $\mathcal{N}$ is stronger than $\mathcal{A}$.

EXAMPLE:

$$P = \ h := h + 1;\ l := 2 * h$$

Then $\models [Sign]P(Par)$ but $\not\vdash_{\mathcal{N}} [Sign]P(Par)$ while $\vdash_{\mathcal{A}} (Sign)P(Par)$

# Discussion

- We map security of programs into the lattice of abstract interpretations:
  - systematic methods for designing attackers and certificates
  - security degrees compared in the lattice
  - checking abstract non-interference by static program analysis

# Discussion

- We map security of programs into the lattice of abstract interpretations:
  - systematic methods for designing attackers and certificates
  - security degrees compared in the lattice
  - checking abstract non-interference by static program analysis
- Abstract non-interference is a semantics property
  - the method is language independent (as any abstract interpretation)
  - refined semantics may refine security!

# Discussion

- We map security of programs into the lattice of abstract interpretations:
  - systematic methods for designing attackers and certificates
  - security degrees compared in the lattice
  - checking abstract non-interference by static program analysis
- Abstract non-interference is a semantics property
  - the method is language independent (as any abstract interpretation)
  - refined semantics may refine security!
- We introduced a sound proof-system for both narrow and abstract non interference.
  - Checking abstract non-interference can be easily mechanized;
  - Generating basic certificates for simple program fragments;
  - The interest in this technology is mostly related with its use *a la proof carrying code* verification of abstract non-interference.

# Further research

- On going works:
  - Concurrent and multi-threaded systems
    - Abstract non-interference for concurrent systems;
    - Abstract non-interference through bisimulation of abstract systems;

# Further research

- On going works:
  - Concurrent and multi-threaded systems
  - Abstract non-interference for covert channels: A semantic problem
    - Non-termination;
    - Timing channels;
    - Probabilistic channels;

# Further research

- On going works:
  - Concurrent and multi-threaded systems
  - Abstract non-interference for covert channels: A semantic problem
  - Proof-carrying code for abstract non-interference
    - Abstract non-interference in Logical Frameworks;
    - Carrying proofs of abstract non-interference;

# Further research

- On going works:
  - Concurrent and multi-threaded systems
  - Abstract non-interference for covert channels: A semantic problem
  - Proof-carrying code for abstract non-interference
  - Checking abstract non-interference by program slicing;
    - Non-interference as slice of a program;
    - Abstract program slicing;

# Further research

- On going works:
  - Concurrent and multi-threaded systems
  - Abstract non-interference for covert channels: A semantic problem
  - Proof-carrying code for abstract non-interference
  - Checking abstract non-interference by program slicing;
- Future works:
  - Abstract non-interference with active attackers;

# Further research

- On going works:
  - Concurrent and multi-threaded systems
  - Abstract non-interference for covert channels: A semantic problem
  - Proof-carrying code for abstract non-interference
  - Checking abstract non-interference by program slicing;
- Future works:
  - Abstract non-interference with active attackers;
  - Abstract non-interference for security protocols;

# Further research

- On going works:
  - Concurrent and multi-threaded systems
  - Abstract non-interference for covert channels: A semantic problem
  - Proof-carrying code for abstract non-interference
  - Checking abstract non-interference by program slicing;
- Future works:
  - Abstract non-interference with active attackers;
  - Abstract non-interference for security protocols;
  - Abstract non-interference for mobile code;