WHAT YOU LOSE IS WHAT YOU LEAK

INFORMATION LEAKAGE IN DECLASSIFICATION POLICIES

A. Banerjee, R. Giacobazzi and I. Mastroeni

Kansas State University
Manhattan (KS), USA

Università di Verona
Verona, Italy

MFPS 2007

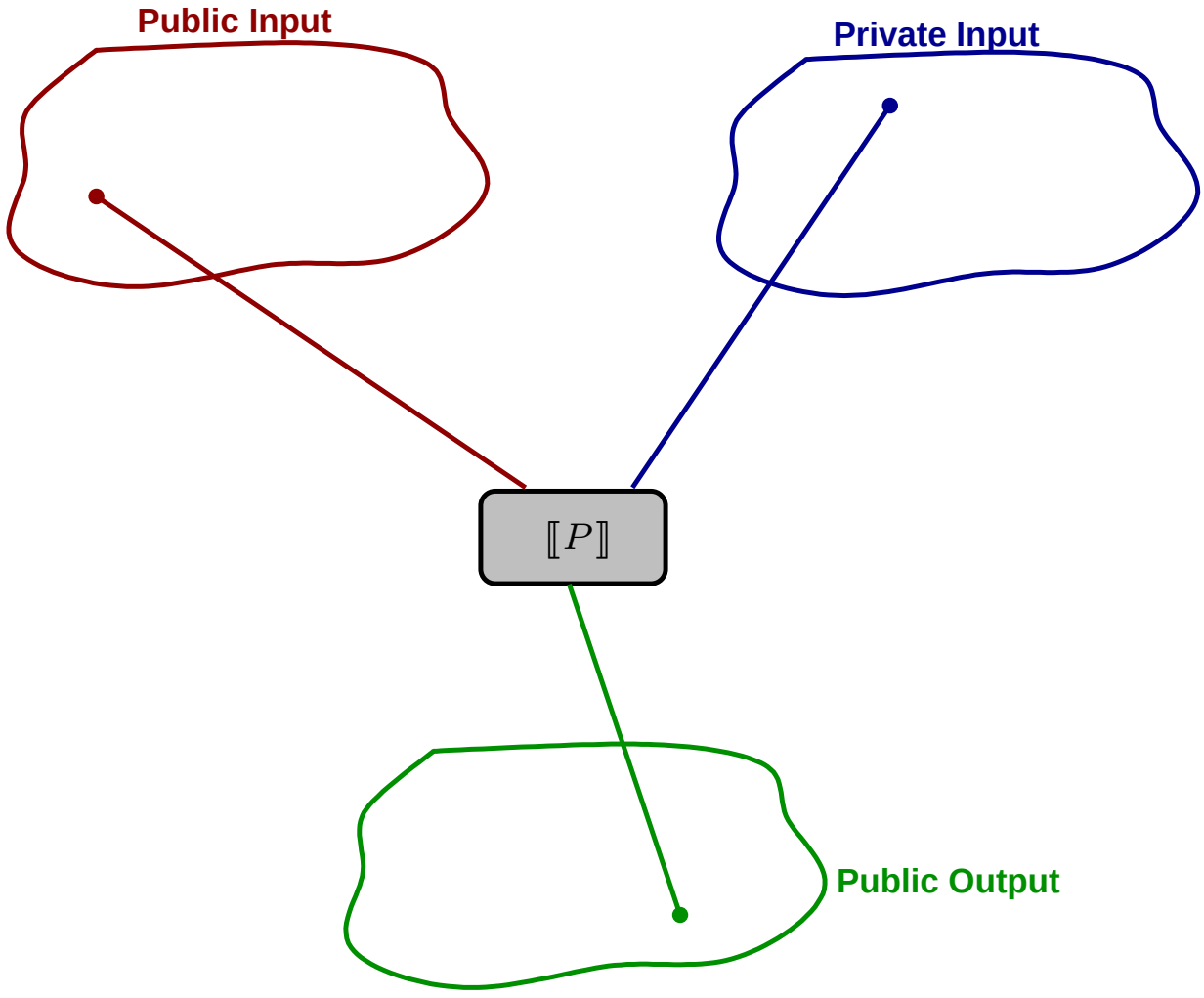
Overview

- By exploiting the strong relation between completeness and non-interference we can obtain the following results:
  - Model declassification as a forward completeness problem for the weakest precondition semantics;
  - Derive counterexamples to a given declassification policy;
  - Refine a given declassification policy;

Overview

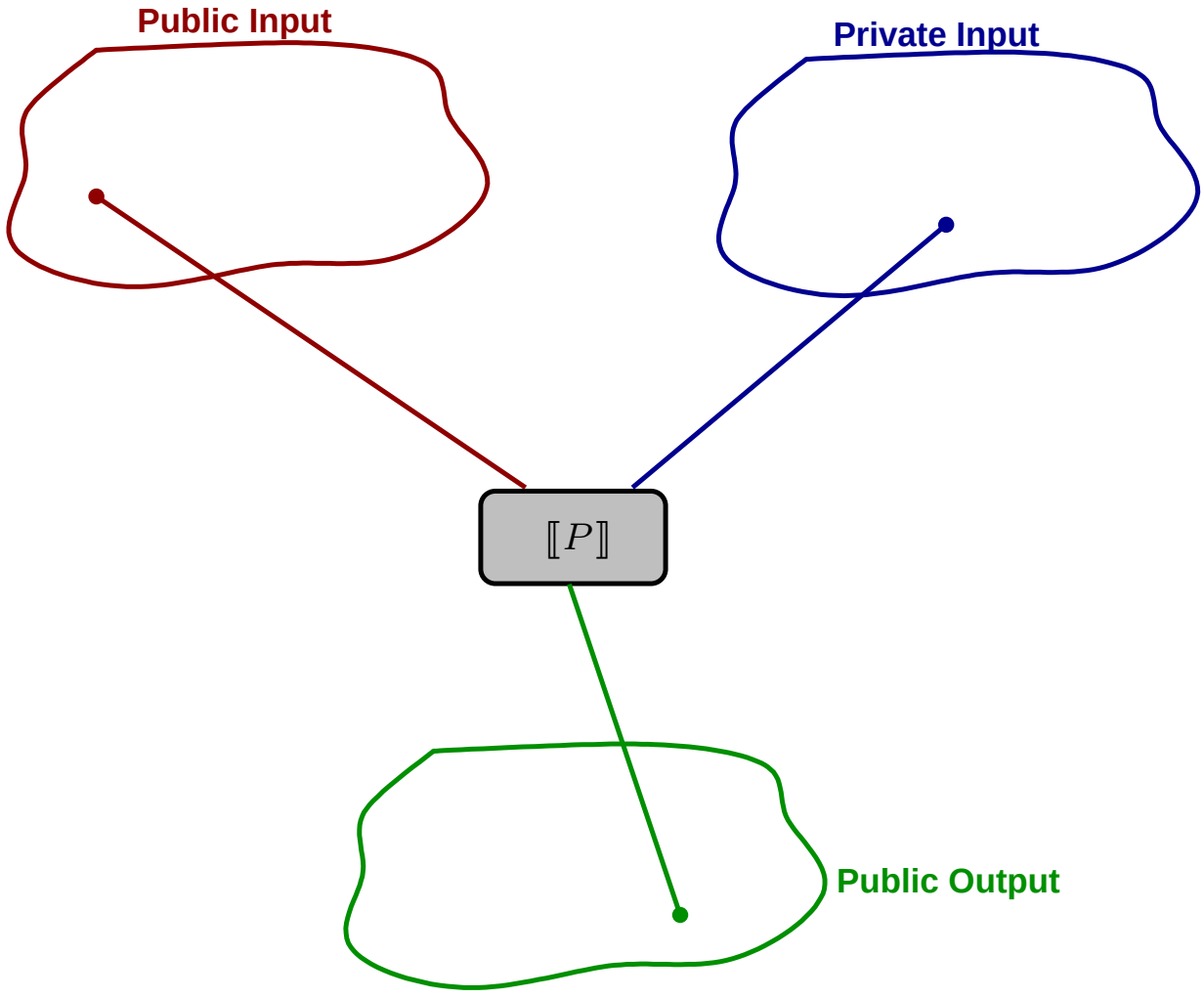
- By exploiting the strong relation between completeness and non-interference we can obtain the following results:
  - Model declassification as a forward completeness problem for the weakest precondition semantics;
  - Derive counterexamples to a given declassification policy;
  - Refine a given declassification policy;
- We can model declassification as a model checking problem (see the relation with robust declassification)

Standard Non-Interference



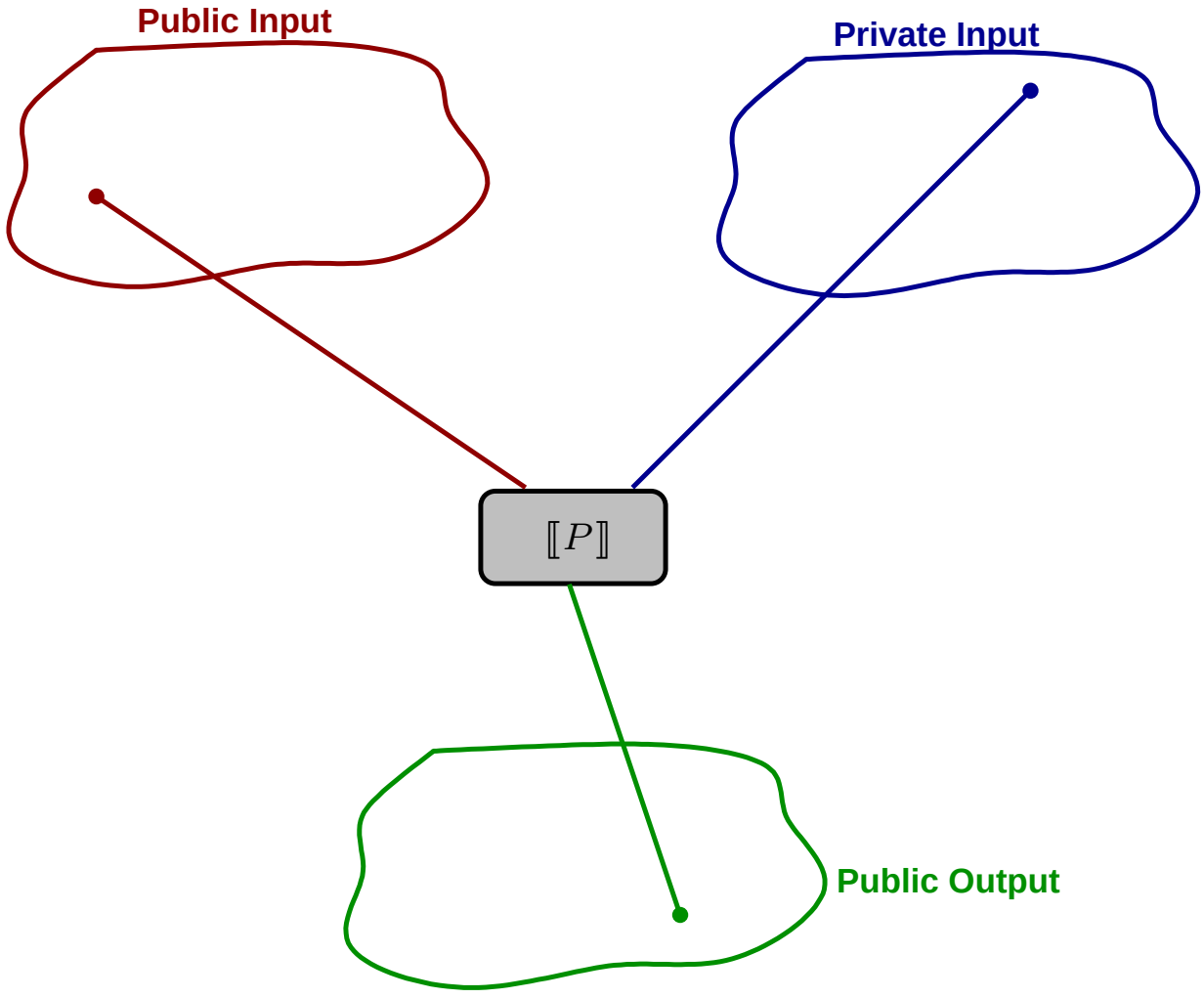
$$\forall l : \mathsf{L}, \forall h_1, h_2 : \mathsf{H}.\ [\![P]\!](h_1, l)^{\mathsf{L}} = [\![P]\!](h_2, l)^{\mathsf{L}}$$

Standard Non-Interference



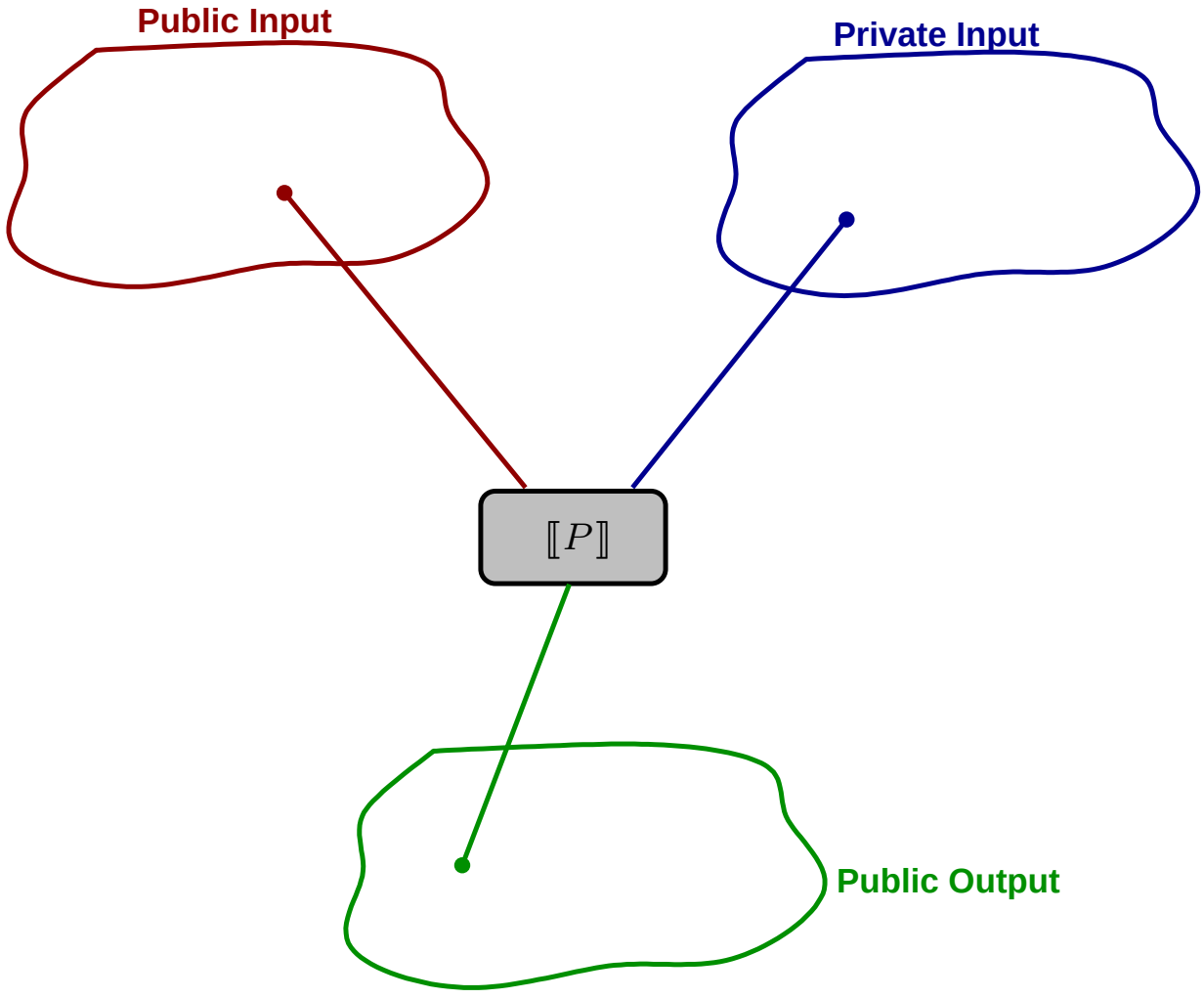
$$\forall l : \mathsf{L}, \forall h_1, h_2 : \mathsf{H}.\ [\![P]\!](h_1, l)^{\mathsf{L}} = [\![P]\!](h_2, l)^{\mathsf{L}}$$

Standard Non-Interference



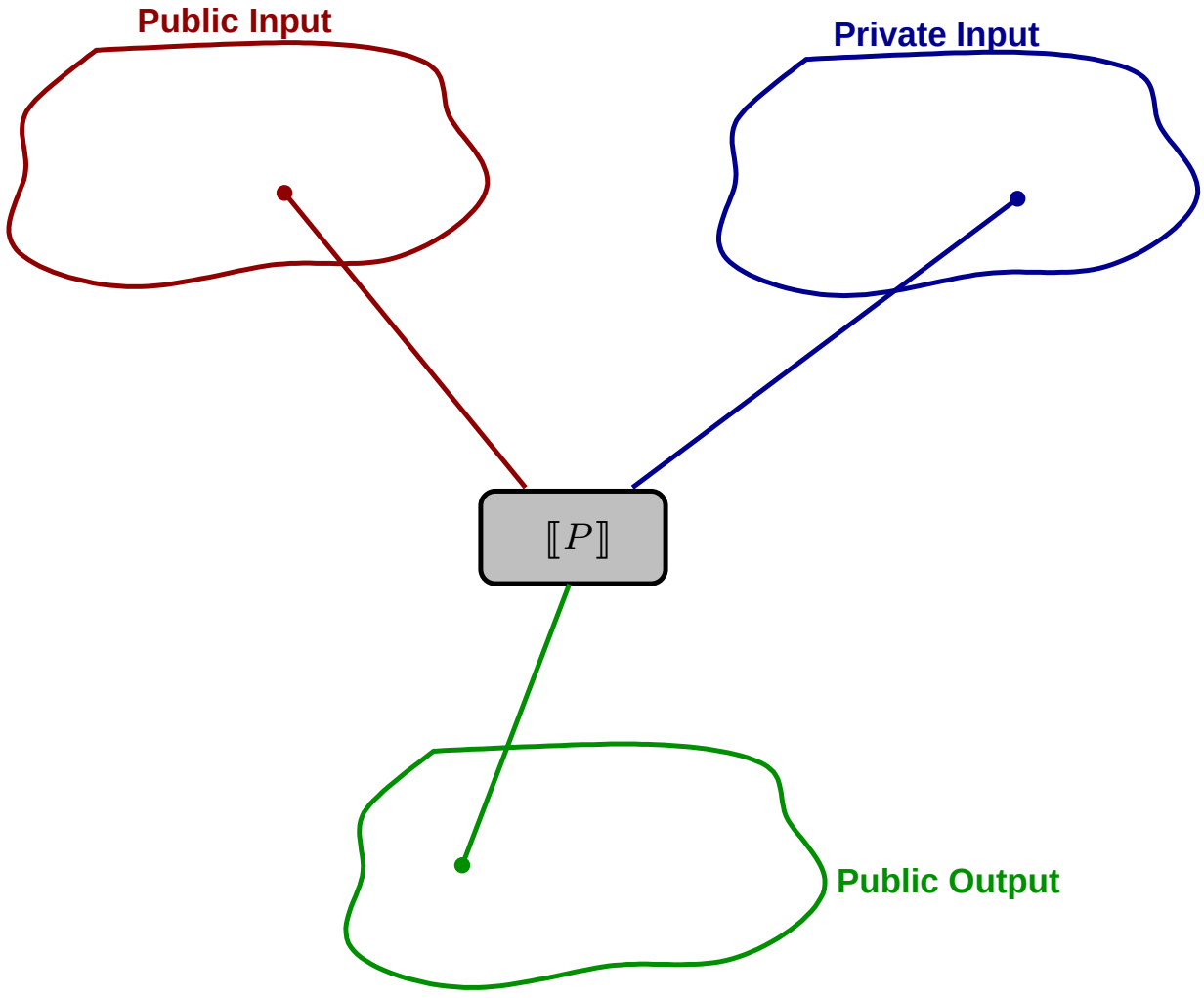
$$\forall l : \mathsf{L}, \forall h_1, h_2 : \mathsf{H}.\ [\![P]\!](h_1, l)^{\mathsf{L}} = [\![P]\!](h_2, l)^{\mathsf{L}}$$

Standard Non-Interference



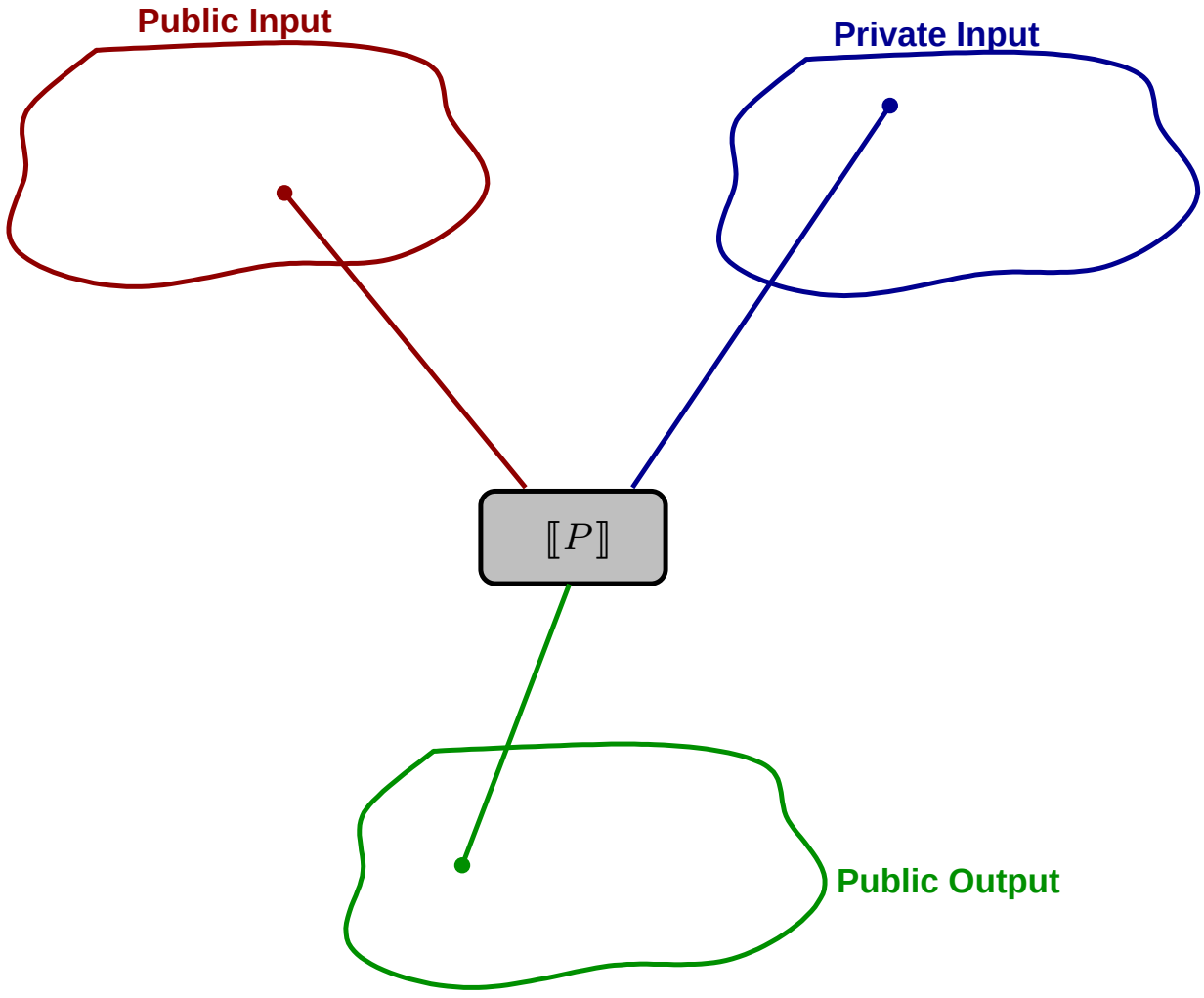
$$\forall l : \mathsf{L}, \forall h_1, h_2 : \mathsf{H}.\ [\![P]\!](h_1, l)^{\mathsf{L}} = [\![P]\!](h_2, l)^{\mathsf{L}}$$

Standard Non-Interference



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

Standard Non-Interference



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

NI: A completeness problem

Recall that [Joshi & Leino'00]

P is *secure* iff $\texttt{HH} \,;\, P;\, \texttt{HH} \doteq P \,;\, \texttt{HH}$

NI: A completeness problem

Recall that [Joshi & Leino'00]

P is *secure* iff $\mathtt{HH}\,;\, P;\, \mathtt{HH} \doteq P\,;\, \mathtt{HH}$

Let $X = \langle X^{\mathsf{H}}, X^{\mathsf{L}} \rangle \Rightarrow \mathcal{H}(X) \stackrel{\text{def}}{=} \langle \top^{\mathsf{H}}, X^{\mathsf{L}} \rangle \in \mathit{uco}(\wp(\mathbb{V}))$

$$\begin{array}{ccc} \mathtt{HH}\,;\, P;\, \mathtt{HH} & \doteq & P\,;\, \mathtt{HH} \\ & \Downarrow & \\ \mathcal{H} \circ [\![P]\!] \circ \mathcal{H} & = & \mathcal{H} \circ [\![P]\!] \end{array}$$

NI: A completeness problem

Recall that [Joshi & Leino'00]

P is *secure* iff $\mathtt{HH}\,;\,P;\,\mathtt{HH} \doteq P\,;\,\mathtt{HH}$

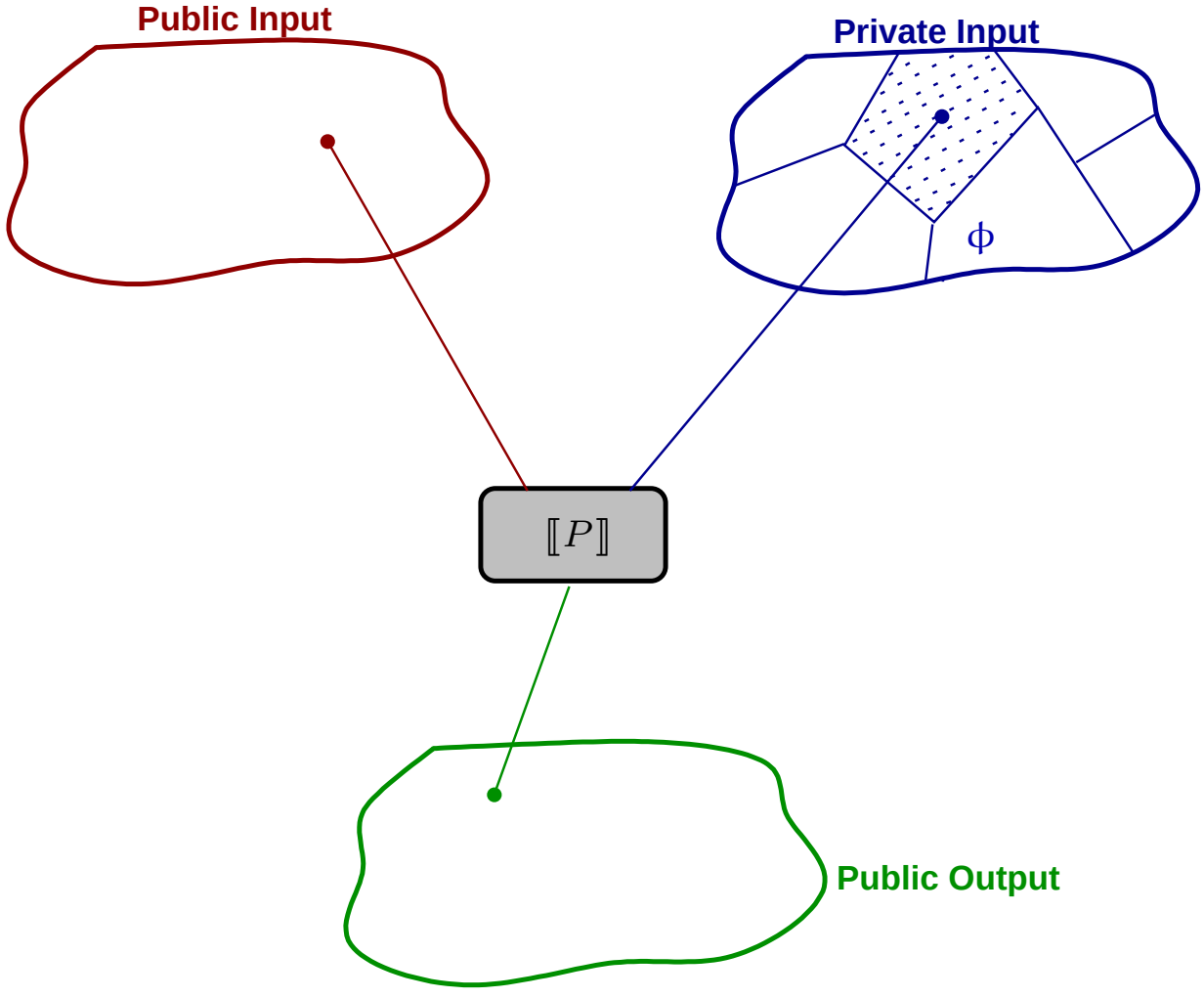
Let $X = \langle X^{\mathsf{H}}, X^{\mathsf{L}}\rangle \Rightarrow \mathcal{H}(X) \stackrel{\text{def}}{=} \langle \top^{\mathsf{H}}, X^{\mathsf{L}}\rangle \in \mathit{uco}(\wp(\mathbb{V}))$

$$\begin{array}{ccc} \mathtt{HH}\,;\,P;\,\mathtt{HH} & \doteq & P\,;\,\mathtt{HH} \\ & \Downarrow & \\ \mathcal{H}\circ[\![P]\!]\circ\mathcal{H} & = & \mathcal{H}\circ[\![P]\!] \end{array}$$

$\Rightarrow$ A COMPLETENESS PROBLEM

[Giacobazzi& Mastroeni '05]

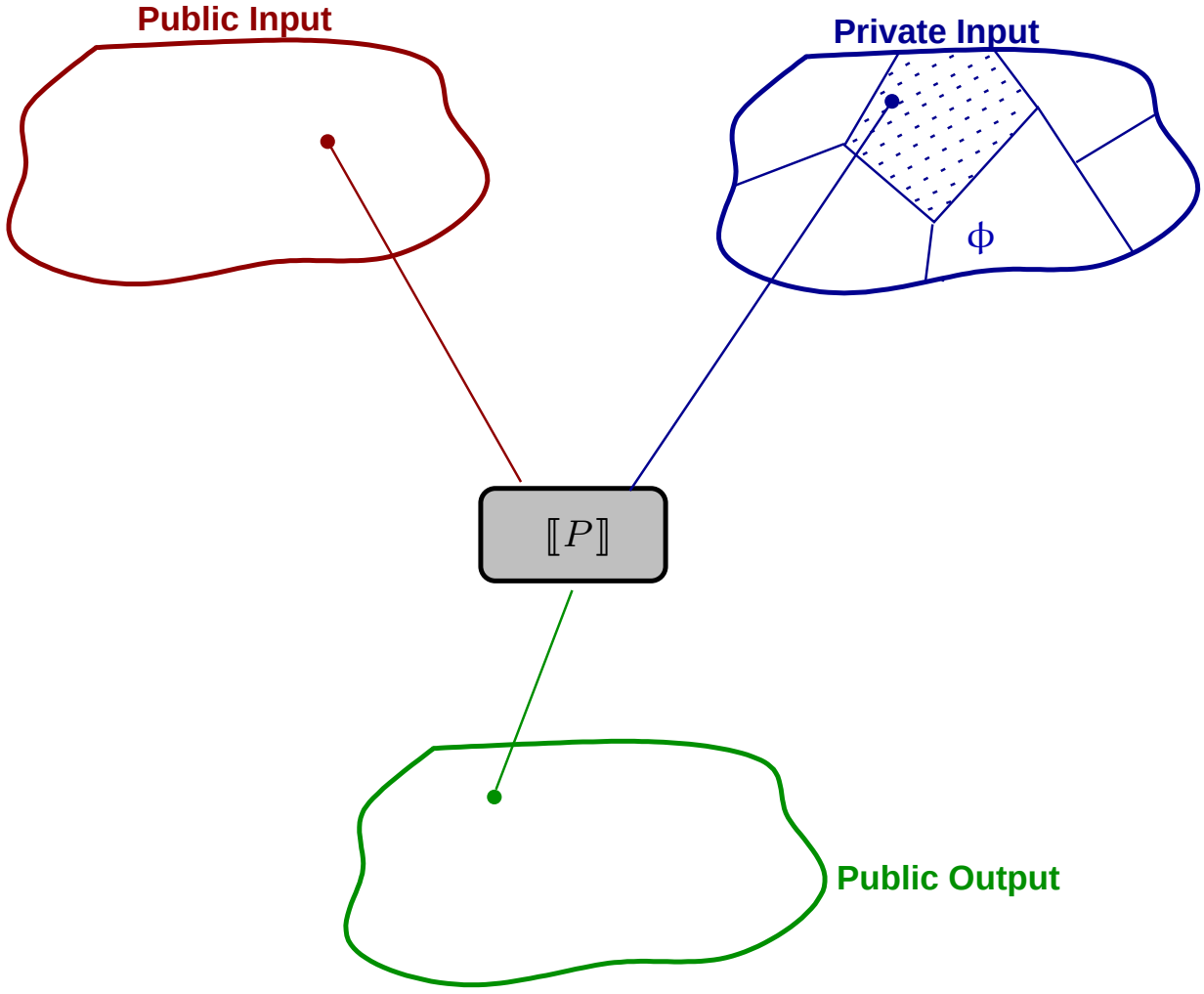
Declassified NI



[Mastroeni '05]

$\phi \in \mathit{Abs}(\wp(\mathbb{V}^{\mathsf{H}}))$: $\phi(h_1) = \phi(h_2) \Rightarrow [\![P]\!](h_1, l)^{\mathsf{L}} = [\![P]\!](h_2, l)^{\mathsf{L}}$

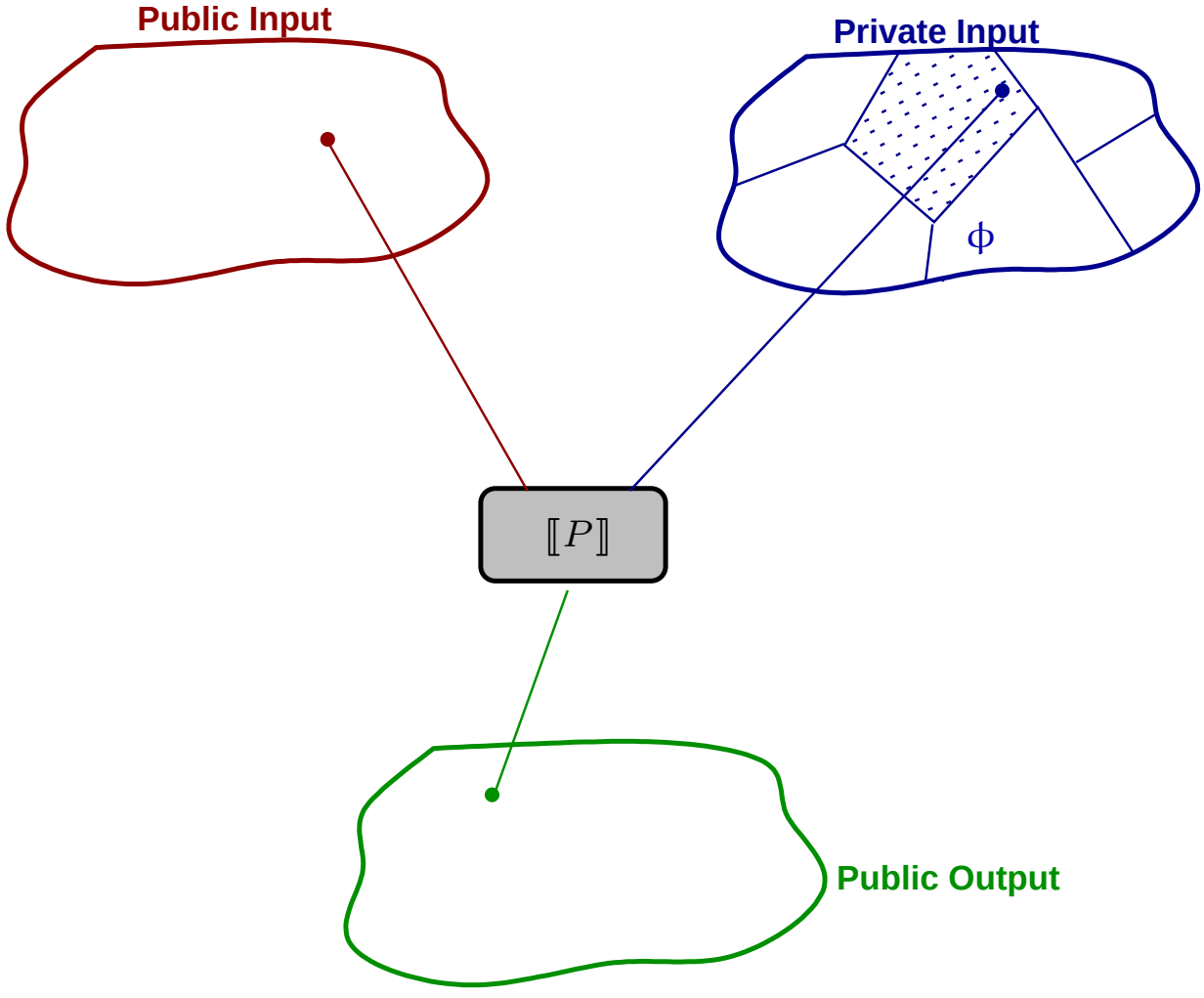
Declassified NI



[Mastroeni '05]

$\phi \in \mathit{Abs}(\wp(\mathbb{V}^{\mathsf{H}}))$: $\phi(h_1) = \phi(h_2) \Rightarrow [\![P]\!](h_1, l)^{\mathsf{L}} = [\![P]\!](h_2, l)^{\mathsf{L}}$

Declassified NI



[Mastroeni '05]

$\phi \in \mathit{Abs}(\wp(\mathbb{V}^{\mathsf{H}}))$: $\phi(h_1) = \phi(h_2) \Rightarrow [\![P]\!](h_1, l)^{\mathsf{L}} = [\![P]\!](h_2, l)^{\mathsf{L}}$

Modelling declassification: A running example

Let $\phi = \text{Parity} \stackrel{\text{def}}{=} \{\top, \mathit{Even}, \mathit{Odd}, \varnothing\}$,

$$P = \left[\begin{array}{l} h := |h|; \\ \textbf{while } (h > 0) \textbf{ do } (h := h - 1; l := h) \textbf{ endw} \end{array} \right.$$

Modelling declassification: A running example

Let $\phi = \text{Parity} \stackrel{\text{def}}{=} \{\top, \mathit{Even}, \mathit{Odd}, \varnothing\}$,

$$
\begin{array}{l}
\quad \{h \in \mathbb{Z}\} \\
h := |h|; \\
\quad \{(h = 0 \wedge l = 0) \ \vee h > 0\} \\
\textbf{while } (h > 0) \textbf{ do } (h := h - 1; l := h) \textbf{ endw} \\
\quad \{l = 0\}
\end{array}
$$

$\mathbb{Z} \in \phi \Rightarrow \phi$ is ok!

Modelling declassification: A running example

Let $\phi = \text{Parity} \stackrel{\text{def}}{=} \{\top, \mathit{Even}, \mathit{Odd}, \varnothing\}$,

$$
\begin{array}{l}
\qquad \{h = 0\} \\
h := |h|; \\
\qquad \{h = 0 \wedge l = a\} \\
\textbf{while } (h > 0) \textbf{ do } (h := h - 1; l := h) \textbf{ endw} \\
\qquad \{l = a \neq 0\}
\end{array}
$$

$\{0\} \notin \phi \Rightarrow \phi$ is not ok!

Modelling declassification: A running example

Let ϕ = Parity $\stackrel{\text{def}}{=} \{\top, \mathit{Even}, \mathit{Odd}, \varnothing\}$,

$H_a = \left\{ \langle h, l \rangle \;\middle|\; h \in \mathbb{Z}, l = a \right\}$ (a value observed in output).

$$P = \left[\begin{array}{l} h := |h|; \\ \textbf{while } (h > 0) \textbf{ do } (h := h - 1; l := h) \textbf{ endw} \end{array} \right.$$

$$\mathit{Wlp} : \left\{ \begin{array}{lcll} H_0 & \mapsto & \{\langle h, l \rangle \mid h \neq 0, l \in \mathbb{Z}\} \cup \{\langle 0, 0 \rangle\} & \\ H_a & \mapsto & \{\langle 0, a \rangle\} & (a \neq 0) \end{array} \right.$$

P secure with ϕ declassified $\Leftrightarrow$ $\mathcal{H}^{\Phi} \circ \mathit{Wlp}_P \circ \mathcal{H} = \mathit{Wlp}_P \circ \mathcal{H}$

DNI: A completeness problem (1)

Let $\mathcal{H}^{\Phi}$ the abstract domain declassifying the property ϕ of the private *input*:

$$\mathcal{H} \circ [\![P]\!] \circ \mathcal{H}^{\Phi} = \mathcal{H} \circ [\![P]\!] \Leftrightarrow \mathcal{H}^{\Phi} \circ \mathit{Wlp}_P \circ \mathcal{H} = \mathit{Wlp}_P \circ \mathcal{H}$$

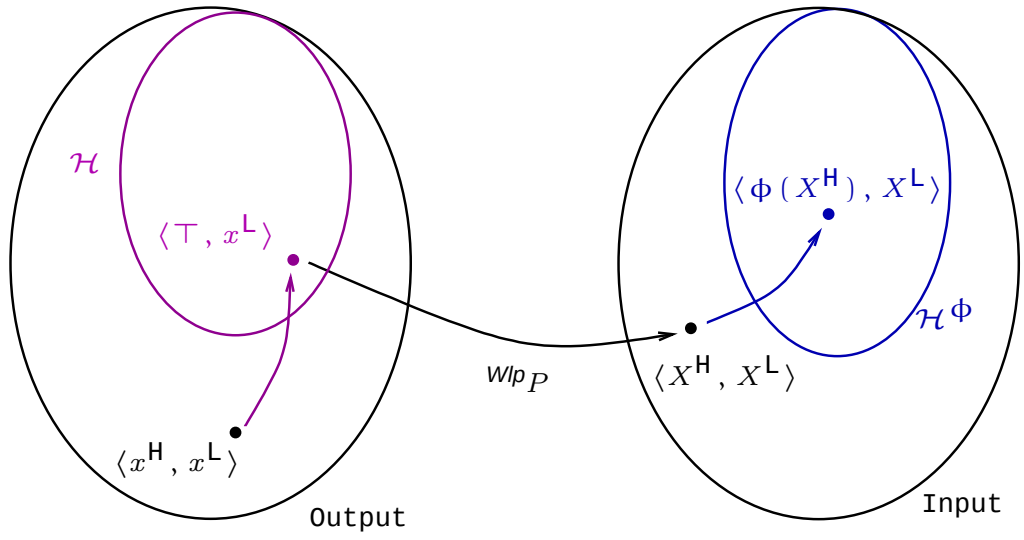
$\Downarrow$

To release ϕ *means* to distinguish between elements in ϕ!

DNI: A completeness problem (1)

Let $\mathcal{H}^{\Phi}$ the abstract domain declassifying the property ϕ of the private *input*:

$$\mathcal{H} \circ [\![P]\!] \circ \mathcal{H}^{\Phi} = \mathcal{H} \circ [\![P]\!] \Leftrightarrow \mathcal{H}^{\Phi} \circ \mathit{Wlp}_P \circ \mathcal{H} = \mathit{Wlp}_P \circ \mathcal{H}$$



Deriving counterexamples: A running example

Consider again $\phi = \text{Parity} \stackrel{\text{def}}{=} \{\top, \mathit{Even}, \mathit{Odd}, \varnothing\}$,

$$P = \left[\quad h := |h|; \quad \textbf{while } (h > 0) \textbf{ do } (h := h - 1; l := h) \textbf{ endw} \right.$$

Deriving counterexamples: A running example

Consider again $\phi = \text{Parity} \stackrel{\text{def}}{=} \{\top, \mathit{Even}, \mathit{Odd}, \varnothing\}$,

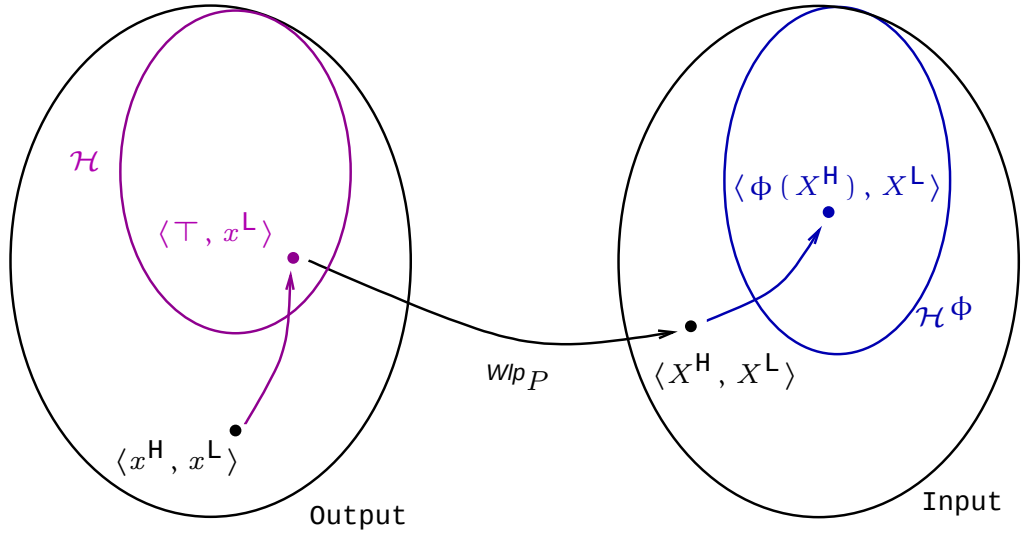
$$\{h = 0\} \qquad \Rightarrow \mathit{Even} \text{ split in } \{0\} \text{ and } \mathit{Even} \setminus \{0\}$$
$$h := |h|;$$
$$\{h = 0 \wedge l = a\}$$
$$\textbf{while } (h > 0) \textbf{ do } (h := h - 1; l := h) \textbf{ endw}$$
$$\{l = a \neq 0\}$$

Let $l = 5$, $h_1 = 0 \in \mathit{Even}$ and $h_2 = 2 \in \mathit{Even}$:
$$[\![P]\!](\langle 0, 5\rangle) = \langle 0, 5\rangle \neq \langle 0, 0\rangle = [\![P]\!](\langle 2, 5\rangle)$$

DNI: A completeness problem (2)

Let $\mathcal{H}^{\Phi}$ the abstract domain declassifying the property ϕ of the private *input*:

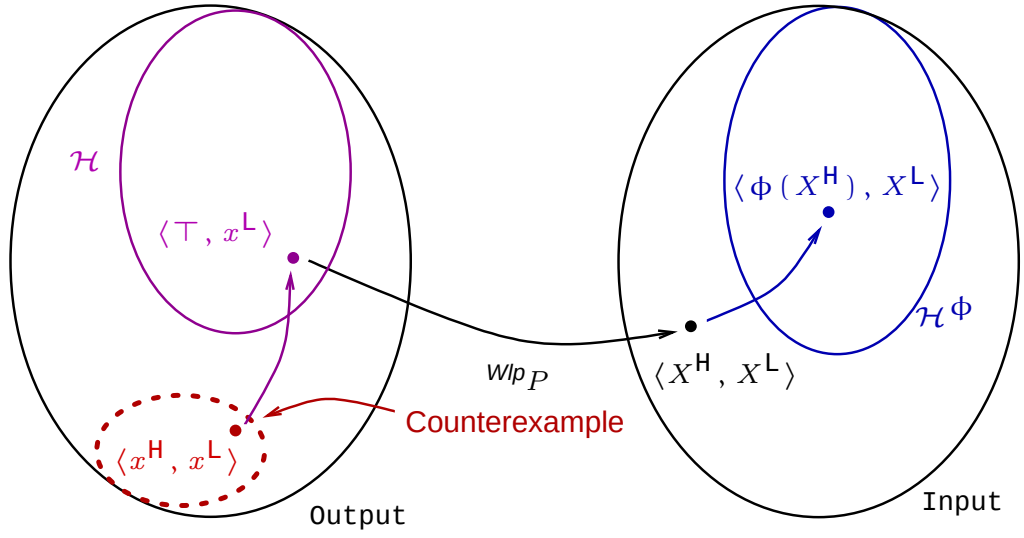
$$\mathcal{H} \circ [\![P]\!] \circ \mathcal{H}^{\Phi} = \mathcal{H} \circ [\![P]\!] \Leftrightarrow \mathcal{H}^{\Phi} \circ \mathit{Wlp}_P \circ \mathcal{H} = \mathit{Wlp}_P \circ \mathcal{H}$$



DNI: A completeness problem (2)

Let $\mathcal{H}^{\Phi}$ the abstract domain declassifying the property ϕ of the private *input*:

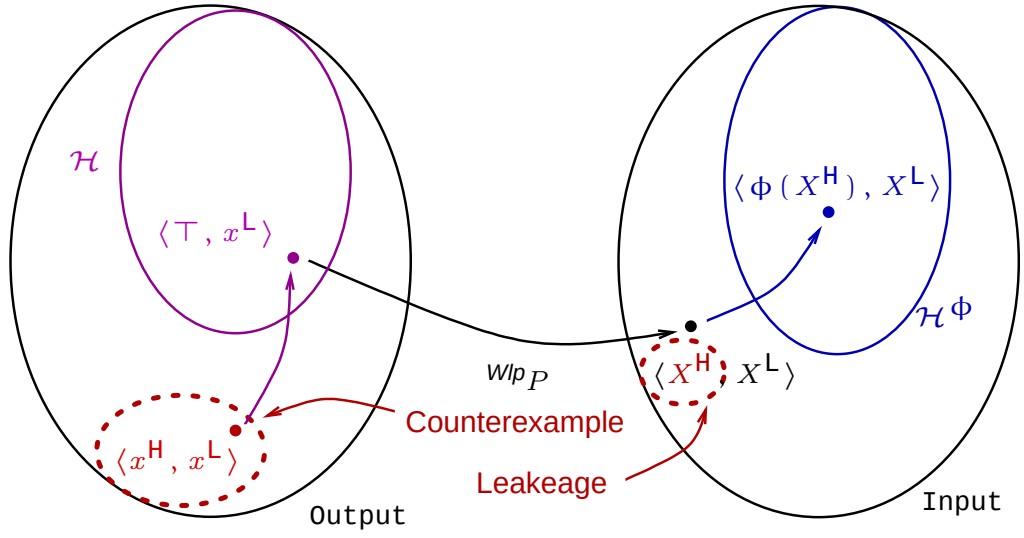
$$\mathcal{H} \circ [\![P]\!] \circ \mathcal{H}^{\Phi} = \mathcal{H} \circ [\![P]\!] \Leftrightarrow \mathcal{H}^{\Phi} \circ \mathit{Wlp}_P \circ \mathcal{H} = \mathit{Wlp}_P \circ \mathcal{H}$$



DNI: A completeness problem (2)

Let $\mathcal{H}^{\Phi}$ the abstract domain declassifying the property ϕ of the private *input*:

$$\mathcal{H} \circ [\![P]\!] \circ \mathcal{H}^{\Phi} = \mathcal{H} \circ [\![P]\!] \Leftrightarrow \mathcal{H}^{\Phi} \circ \mathit{Wlp}_P \circ \mathcal{H} = \mathit{Wlp}_P \circ \mathcal{H}$$



Refining policies: An example

Consider $\phi = \left\{ \left\{ \langle h_1, h_2, \ldots, h_n \rangle \;\middle|\; (h_1 + h_2 + \ldots + h_n)/n = a \right\} \;\middle|\; a \in \mathbb{Z} \right\}$

$$P = \left[\begin{array}{l} h_1 := h_1;\ h_2 := h_2; \ldots;\ h_n = h_n \\ avg := \mathsf{declassify}((h_1 + h_2 + \ldots + h_n)/n); \end{array} \right.$$

Refining policies: An example

Consider $\phi = \left\{ \left\{ \langle h_1, h_2, \ldots, h_n \rangle \,\middle|\, (h_1 + h_2 + \ldots + h_n)/n = a \right\} \,\middle|\, a \in \mathbb{Z} \right\}$

$$
\begin{array}{l}
\quad \{h_1 = a\} \\
h_1 := h_1;\ h_2 := h_2; \ldots;\ h_n = h_n \\
\quad \{(h_1 + h_2 + \ldots + h_n)/n = a\} \\
avg := \mathsf{declassify}((h_1 + h_2 + \ldots + h_n)/n); \\
\quad \{avg = a\}
\end{array}
$$

$\left\{ \langle h_1, h_2, \ldots, h_n \rangle \,\middle|\, (h_1 + h_2 + \ldots + h_n)/n = a, h_1 = a \right\} \notin \phi \Rightarrow \phi$ is not ok!

Refining policies: An example

Consider $\phi = \left\{ \left\{ \langle h_1, h_2, \ldots, h_n \rangle \;\middle|\; (h_1 + h_2 + \ldots + h_n)/n = a \right\} \;\middle|\; a \in \mathbb{Z} \right\}$

$H_a = \left\{ \langle h_1, \ldots, h_n, avg \rangle \;\middle|\; avg_{h_i} = a \right\}$ (a value observed in output).

$$P = \left[\begin{array}{l} h_1 := h_1;\ h_2 := h_2; \ldots;\ h_n = h_n \\ avg := \text{declassify}((h_1 + h_2 + \ldots + h_n)/n); \end{array} \right.$$

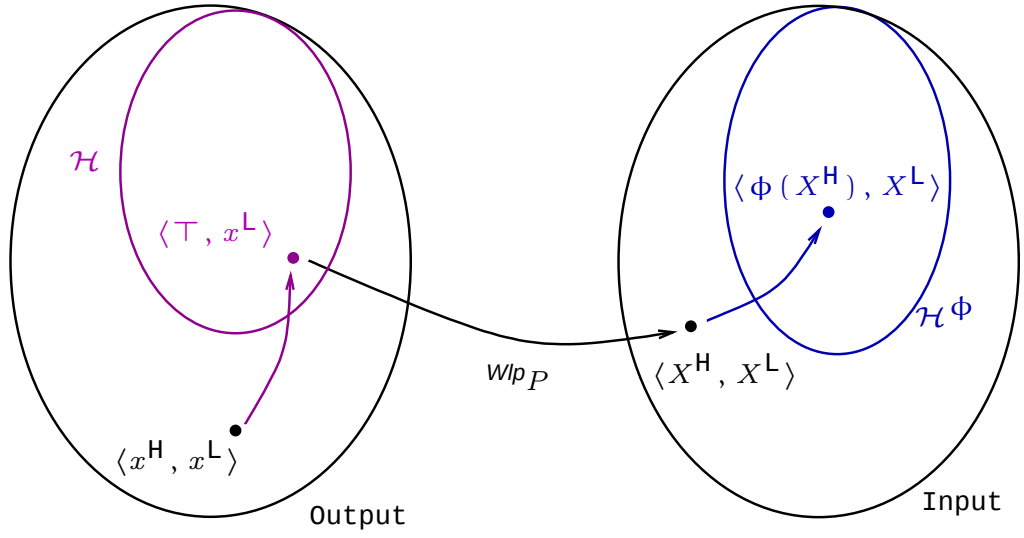
$$\mathit{Wlp}: \quad H_a \quad \mapsto \quad \left\{ \langle a, h_2, \ldots, h_n, a \rangle \;\middle|\; avg = a \right\}$$

P secure with ϕ' declassified $\Leftrightarrow \quad \phi' = \phi \sqcap \left\{ \mathit{Wlp}(H_a) \;\middle|\; a \in \mathbb{Z} \right\}$

DNI: A completeness problem (3)

Let $\mathcal{H}^{\Phi}$ the abstract domain declassifying the property ϕ of the private *input*:

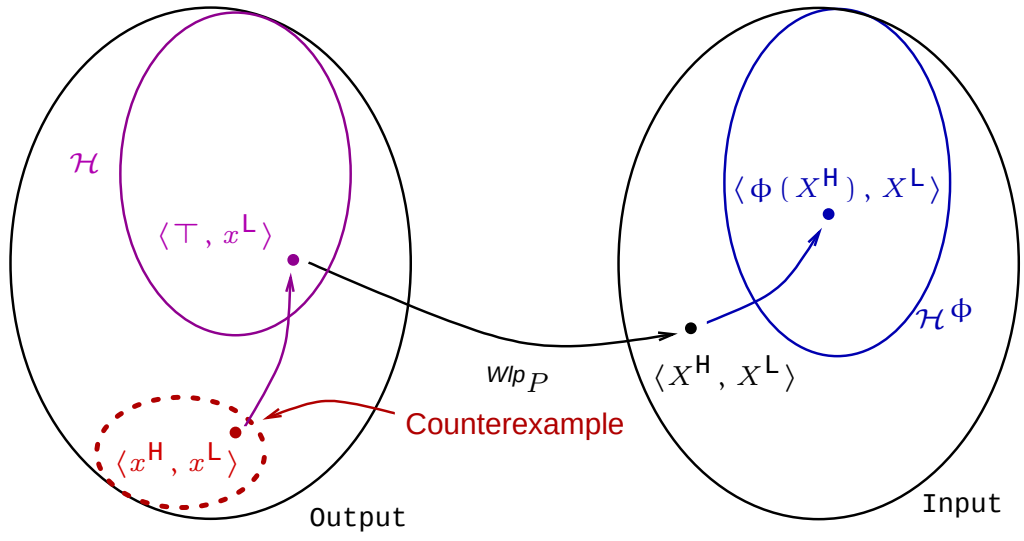
$$\mathcal{H} \circ [\![P]\!] \circ \mathcal{H}^{\Phi} = \mathcal{H} \circ [\![P]\!] \Leftrightarrow \mathcal{H}^{\Phi} \circ \mathit{Wlp}_P \circ \mathcal{H} = \mathit{Wlp}_P \circ \mathcal{H}$$



DNI: A completeness problem (3)

Let $\mathcal{H}^{\Phi}$ the abstract domain declassifying the property ϕ of the private *input*:

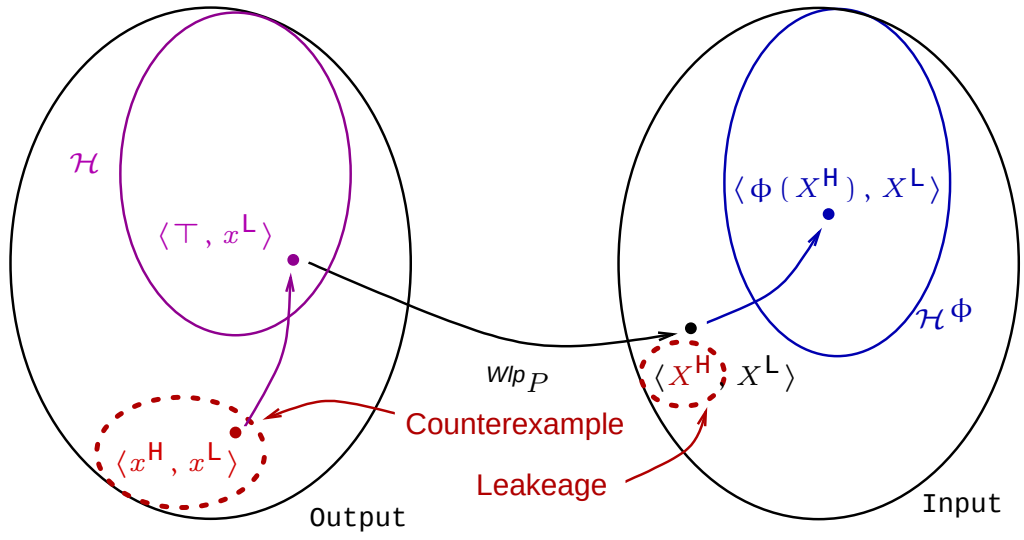
$$\mathcal{H} \circ [\![P]\!] \circ \mathcal{H}^{\Phi} = \mathcal{H} \circ [\![P]\!] \Leftrightarrow \mathcal{H}^{\Phi} \circ \mathit{Wlp}_P \circ \mathcal{H} = \mathit{Wlp}_P \circ \mathcal{H}$$



DNI: A completeness problem (3)

Let $\mathcal{H}^{\Phi}$ the abstract domain declassifying the property ϕ of the private *input*:

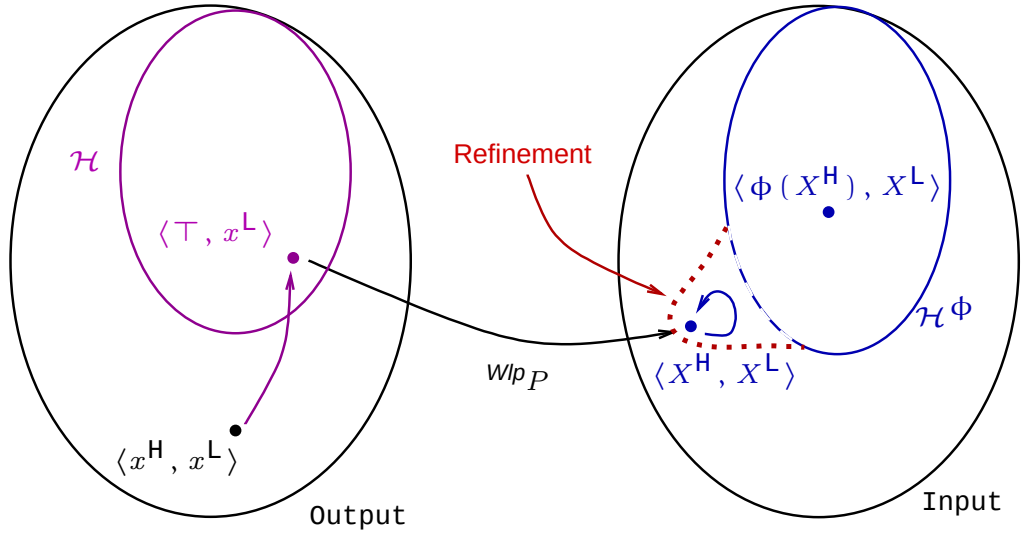
$$\mathcal{H} \circ [\![P]\!] \circ \mathcal{H}^{\Phi} = \mathcal{H} \circ [\![P]\!] \Leftrightarrow \mathcal{H}^{\Phi} \circ \mathit{Wlp}_P \circ \mathcal{H} = \mathit{Wlp}_P \circ \mathcal{H}$$



DNI: A completeness problem (3)

Let $\mathcal{H}^{\Phi}$ the abstract domain declassifying the property ϕ of the private *input*:

$$\mathcal{H} \circ [\![P]\!] \circ \mathcal{H}^{\Phi} = \mathcal{H} \circ [\![P]\!] \Leftrightarrow \mathcal{H}^{\Phi} \circ \mathit{Wlp}_P \circ \mathcal{H} = \mathit{Wlp}_P \circ \mathcal{H}$$



Example: Oblivious Transfer Protocol

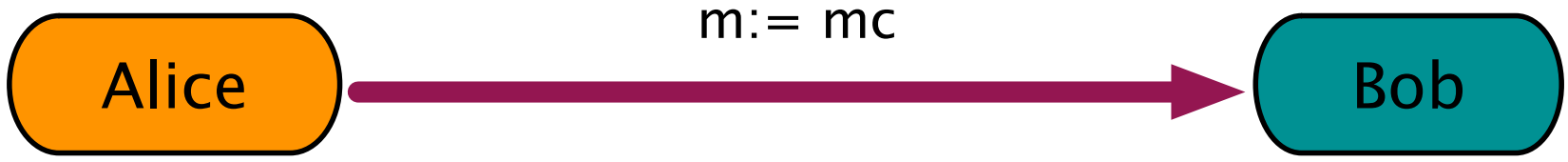
[C. Morgan]

Alice has two messages;
Bob knows their "names",
but not their contents.

Bob asks for one,
by name.

m0, m1: M

c: {0,1}



Alice sends the message...

...Bob receives it

PROBLEM: Alice is not to know which message Bob asked for
Bob is not to know the other message

Example: Oblivious Transfer Protocol

[C. Morgan]
Ted: Trusted party

	Alice	Bob
Hid:	$r; d; c \in \{0,1\}; m$	$m_0; m_1; r_0; r_1$
Vis:	$m_0; m_1; r_0; r_1; f_0; f_1; e$	$c; m; d; r; f_0; f_1; e$

$$P \stackrel{\text{def}}{=} \left[\begin{array}{l} r_0, r_1 :\in M;\ d :\in \{0,1\}; \\ r := r_d; \\ e := c \oplus d; \\ f_0, f_1 := m_0 \oplus r_e,\ m_1 \oplus r_{1\oplus e}; \\ m := f_c \oplus r; \end{array}\right.$$

Example: Oblivious Transfer Protocol

[C. Morgan]
Ted: Trusted party

	Alice	Bob
Hid:	$r; d; c \in \{0,1\}; m$	$m_0; m_1; r_0; r_1$
Vis:	$m_0; m_1; r_0; r_1; f_0; f_1; e$	$c; m; d; r; f_0; f_1; e$

Bob's point of view: He has not to see $m_{1 \oplus c}$

$r_0, r_1 :\in M;\ d :\in \{0,1\};$
$r := r_d;$
$\quad \{(c = d, f_0 = m_0 \oplus r_0, f_1 = m_1 \oplus r_1) \vee (c \neq d, f_0 = m_0 \oplus r_1, f_1 = m_1 \oplus r_0)\}$
$e := c \oplus d;$
$\quad \{f_0 = m_0 \oplus r_e, f_1 = m_1 \oplus r_{1 \oplus e}\}$
$f_0, f_1 := m_0 \oplus r_e,\ m_1 \oplus r_{1 \oplus e};$
$m := f_c \oplus r;$
$\quad \{f_0; f_1; m\}$

Example: Oblivious Transfer Protocol

[C. Morgan]
Ted: Trusted party

	Alice	Bob
Hid:	$r; d; c \in \{0,1\}; m$	$m_0; m_1; r_0; r_1$
Vis:	$m_0; m_1; r_0; r_1; f_0; f_1; e$	$c; m; d; r; f_0; f_1; e$

Bob's point of view: He has not to see $m_{1 \oplus c}$

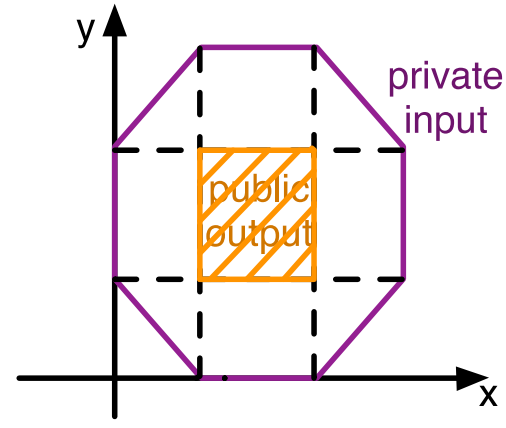
Soundness guarantees that Bob knows $m = m_c, f_c, r_d$
Wlp guarantees that Bob knows only $f_c = m_c \oplus r_d$ and $f_{1 \oplus c} = m_{1 \oplus c} \oplus r_{1 \oplus d}$

$\Downarrow$

$f_{1 \oplus c}$ tells *almost nothing* of the secret $m_{1 \oplus c}$

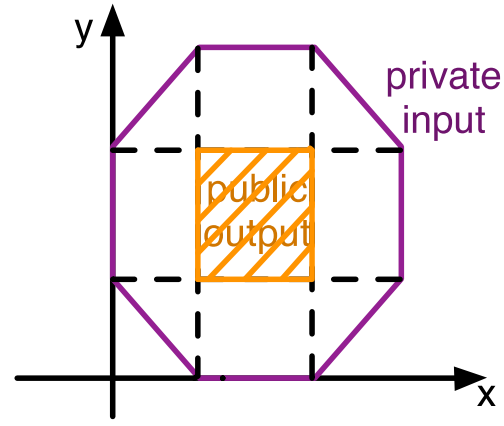
Declassified *Abstract* non-interference

$$P \stackrel{\text{def}}{=} \left[\begin{array}{l}
\textbf{if}(d \leq x+y \leq d+d_x+d_y \ \wedge\ -d_y \leq x-y \leq d_x)\ \textbf{then} \\
\quad \textbf{if}(x \geq 0 \ \wedge x \leq d)\ \textbf{then}\ x_L := d; \\
\quad \textbf{if}(x > d \ \wedge\ x \leq d_x)\ \textbf{then}\ \ x_L := x; \\
\quad \textbf{if}(x > d_x \ \wedge\ x \leq d_x + d)\ \textbf{then}\ x_L := d_x; \\
\quad \textbf{if}(y \geq 0 \ \wedge y \leq d)\ \textbf{then}\ y_L := d; \\
\quad \textbf{if}(y > d \ \wedge\ y \leq d_y)\ \textbf{then}\ \ y_L := y; \\
\quad \textbf{if}(y > d_y \ \wedge\ y \leq d_y + d)\ \textbf{then}\ y_L := d_y;
\end{array}\right.$$



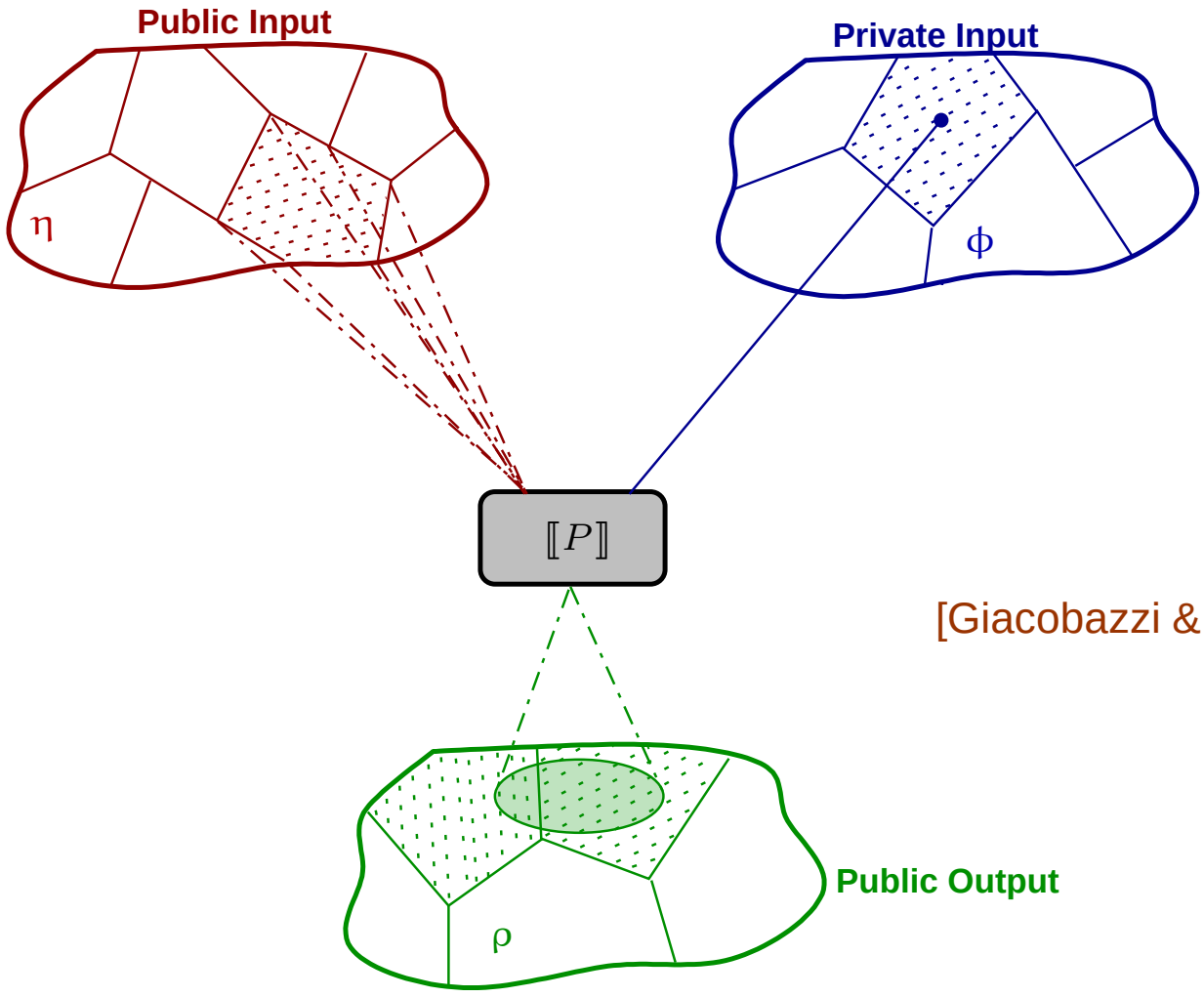
Declassified *Abstract* non-interference

$$P \stackrel{\text{def}}{=} \left[\begin{array}{l}
\textbf{if}(d \le x+y \le d+d_x+d_y \ \wedge\ -d_y \le x-y \le d_x)\ \textbf{then} \\
\quad \textbf{if}(x \ge 0 \ \wedge x \le d)\ \textbf{then}\ x_L := d; \\
\quad \textbf{if}(x > d \ \wedge\ x \le d_x)\ \textbf{then}\ \ x_L := x; \\
\quad \textbf{if}(x > d_x \ \wedge\ x \le d_x + d)\ \textbf{then}\ x_L := d_x; \\
\quad \textbf{if}(y \ge 0 \ \wedge y \le d)\ \textbf{then}\ y_L := d; \\
\quad \textbf{if}(y > d \ \wedge\ y \le d_y)\ \textbf{then}\ \ y_L := y; \\
\quad \textbf{if}(y > d_y \ \wedge\ y \le d_y + d)\ \textbf{then}\ y_L := d_y;
\end{array}\right.$$



$$\mathcal{H}^{\Phi}{}_{\eta} \circ \mathit{Wlp}_P \circ \mathcal{H}_{\rho} = \mathit{Wlp}_P \circ \mathcal{H}_{\rho}$$

Declassified *Abstract* non-interference

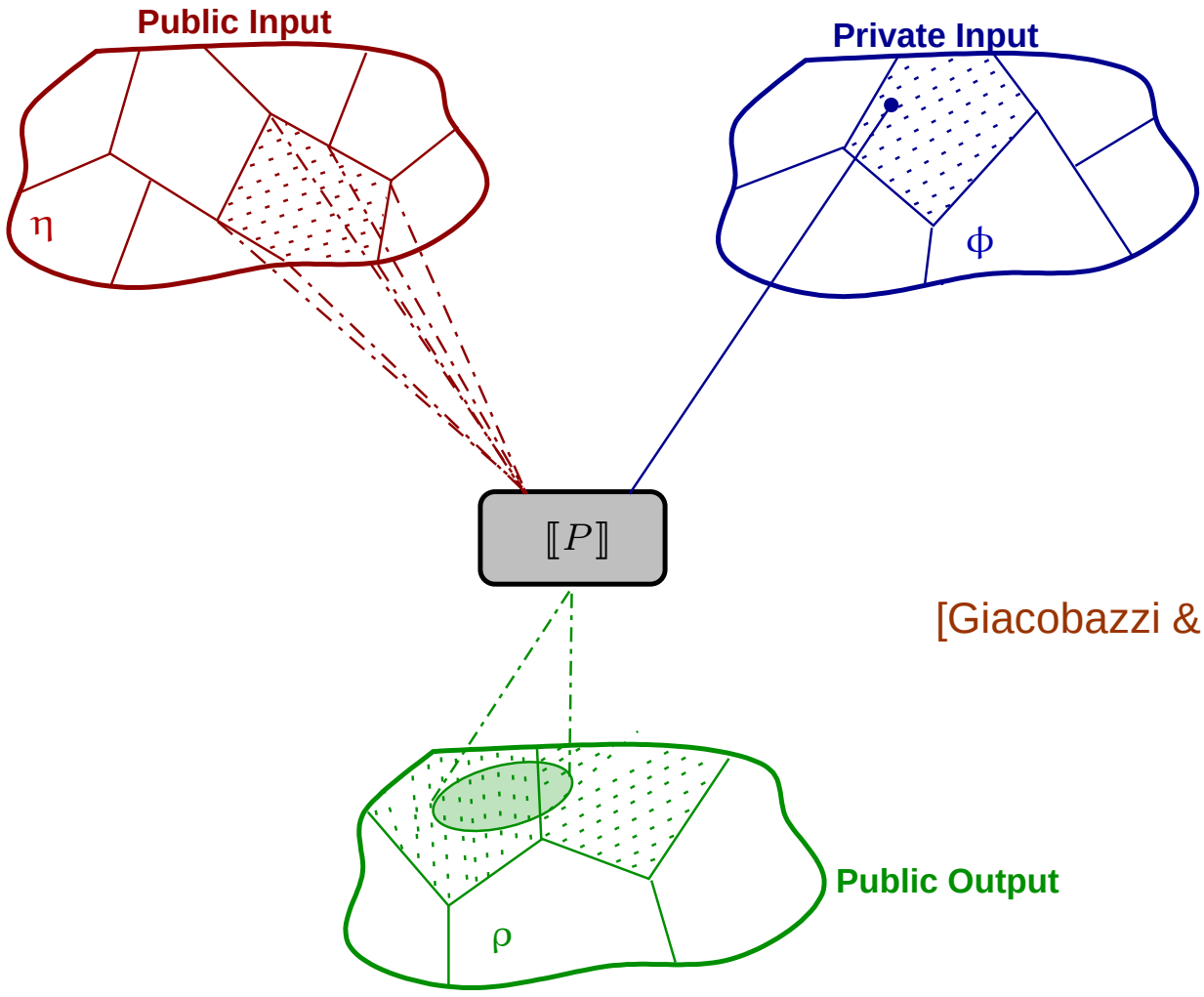


[Giacobazzi & Mastroeni '04]

$\rho, \eta \in \mathit{Abs}(\wp(\mathbb{V}^{\mathsf{L}})), \phi \in \mathit{Abs}(\wp(\mathbb{V}^{\mathsf{H}}))$: $(\eta)P(\phi \Rightarrow \rho)$:

$\eta(l_1) = \eta(l_2)$ and $\phi(h_1) = \phi(h_2) \Rightarrow \rho(\llbracket P \rrbracket(h_1, \eta(l_1))^{\mathsf{L}}) = \rho(\llbracket P \rrbracket(h_2, \eta(l_2))^{\mathsf{L}})$

Declassified *Abstract* non-interference

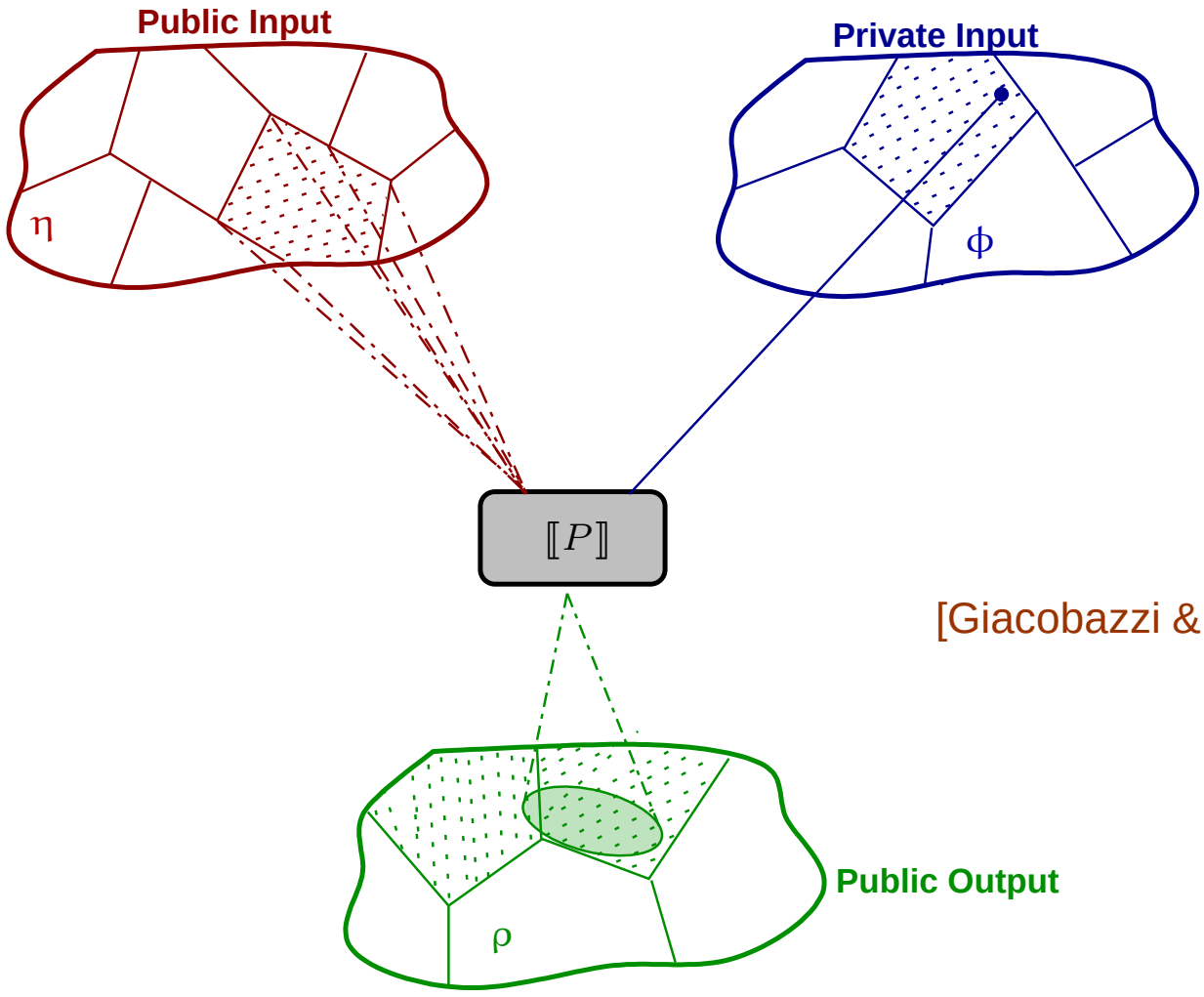


[Giacobazzi & Mastroeni '04]

$\rho, \eta \in \mathit{Abs}(\wp(\mathbb{V}^{\mathsf{L}})), \phi \in \mathit{Abs}(\wp(\mathbb{V}^{\mathsf{H}}))$: $(\eta)P(\phi \Rightarrow \rho)$:

$\eta(l_1) = \eta(l_2)$ and $\phi(h_1) = \phi(h_2) \Rightarrow \rho(\llbracket P \rrbracket(h_1, \eta(l_1))^{\mathsf{L}}) = \rho(\llbracket P \rrbracket(h_2, \eta(l_2))^{\mathsf{L}})$

Declassified *Abstract* non-interference



[Giacobazzi & Mastroeni '04]

$\rho, \eta \in \mathit{Abs}(\wp(\mathbb{V}^{\mathsf{L}})), \phi \in \mathit{Abs}(\wp(\mathbb{V}^{\mathsf{H}}))$: $(\eta)P(\phi \Rightarrow \rho)$:

$\eta(l_1) = \eta(l_2)$ and $\phi(h_1) = \phi(h_2) \Rightarrow \rho(\llbracket P \rrbracket(h_1, \eta(l_1))^{\mathsf{L}}) = \rho(\llbracket P \rrbracket(h_2, \eta(l_2))^{\mathsf{L}})$

Abstract Model checking DNI

- Robust declassification transforms the attacker observational capability [Zdancewic & Myers '01]:

$$\forall \sigma, \sigma' \in \Sigma \,.\, \langle \sigma, \sigma' \rangle \in S[\approx] \;\Leftrightarrow\; Obs_{\sigma}(S, \approx) \equiv Obs_{\sigma'}(S, \approx)$$

Abstract Model checking DNI

- Robust declassification transforms the attacker observational capability [Zdancewic & Myers '01]:

$$\forall \sigma, \sigma' \in \Sigma \,.\, \langle \sigma, \sigma' \rangle \in S[\approx] \;\Leftrightarrow\; Obs_{\sigma}(S, \approx) \equiv Obs_{\sigma'}(S, \approx)$$

- $S[\approx] = \approx$ iff $\approx$ backward complete for *post*

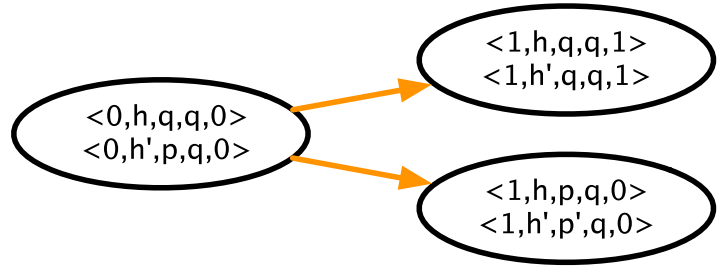
Abstract Model checking DNI

- Robust declassification transforms the attacker observational capability [Zdancewic & Myers '01]:

$$\forall \sigma, \sigma' \in \Sigma \,.\, \langle \sigma, \sigma' \rangle \in S[\approx] \;\Leftrightarrow\; Obs_{\sigma}(S, \approx) \equiv Obs_{\sigma'}(S, \approx)$$

- Example:

$$\begin{array}{lcll}
\langle t, h, p, q, r \rangle & \mapsto & \langle t, h, p, q, r \rangle & \\
\langle 0, h, q, q, 0 \rangle & \mapsto & \langle 1, h, q, q, 1 \rangle & \\
\langle 0, h, q, q, 1 \rangle & \mapsto & \langle 1, h, q, q, 0 \rangle & \\
\langle 0, h, p, q, 0 \rangle & \mapsto & \langle 1, h, p, q, 0 \rangle & p \neq q \\
\langle 0, h, p, q, 1 \rangle & \mapsto & \langle 1, h, p, q, 1 \rangle & p \neq q
\end{array}$$



The public variables are t, q, r, hence the partition induced by $\mathcal{H}$ is:

$$\langle t, h, p, q, r \rangle \;\equiv\; \langle t', h', p', q', r' \rangle \text{ iff } t = t' \;\wedge\; q = q' \;\wedge\; r = r'$$

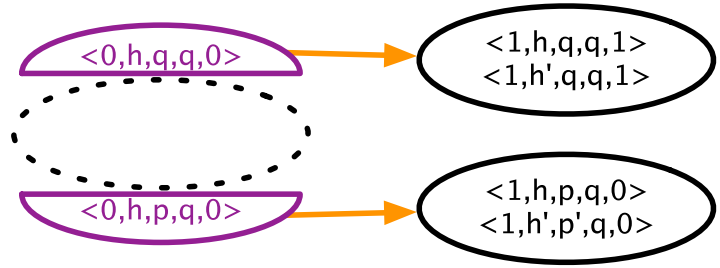
Abstract Model checking DNI

- Robust declassification transforms the attacker observational capability [Zdancewic & Myers '01]:

$$\forall \sigma, \sigma' \in \Sigma \,.\, \langle \sigma, \sigma' \rangle \in S[\approx] \;\Leftrightarrow\; Obs_{\sigma}(S, \approx) \equiv Obs_{\sigma'}(S, \approx)$$

- Example:

$$\begin{array}{lcll}
\langle t, h, p, q, r \rangle & \mapsto & \langle t, h, p, q, r \rangle & \\
\langle 0, h, q, q, 0 \rangle & \mapsto & \langle 1, h, q, q, 1 \rangle & \\
\langle 0, h, q, q, 1 \rangle & \mapsto & \langle 1, h, q, q, 0 \rangle & \\
\langle 0, h, p, q, 0 \rangle & \mapsto & \langle 1, h, p, q, 0 \rangle & p \neq q \\
\langle 0, h, p, q, 1 \rangle & \mapsto & \langle 1, h, p, q, 1 \rangle & p \neq q
\end{array}$$



$$\langle 0, h, p, q, 0 \rangle \mapsto \left\{ \begin{array}{l} \langle 1, h, q, q, 1 \rangle \\ \langle 1, h, p, q, 0 \rangle \end{array} \right. \quad \widetilde{pre}_P : \left\{ \begin{array}{l} \langle 1, h, q, q, 1 \rangle \mapsto \langle 0, h, q, q, 0 \rangle \\ \langle 1, h, p, q, 0 \rangle \mapsto \langle 0, h, p, q, 0 \rangle \end{array} \right. \quad p \neq q$$

Discussion

- What already exists: *Several studies about declassification, derivation of counterexamples and refinements... nothing that combines all together*
  - Several approaches for modelling and checking declassification policies: PER model [Sabelfeld and Sands], dynamic logic [Darvas et al.], robust declassification [Zdancewic and Myers], Delimited release [Sabelfeld and Myers], Relaxed non-interference [Li and Zdancewic],...;
  - Derivation of counterexamples of secure information flows (without declassification) [Unno et al.];
  - Preservation of secrecy under refinement [Alur et al.];

Discussion

- What already exists: *Several studies about declassification, derivation of counterexamples and refinements... nothing that combines all together*
- What we have done: Modelling declassification as a completeness problem;
  - We analyze the accuracy of a declassification policy;
  - We associate with each public observation the corresponding information released;
  - We can refine the accuracy of the policy;
  - We create a connection with abstract model checking;

Discussion

- What already exists: *Several studies about declassification, derivation of counterexamples and refinements... nothing that combines all together*
- What we have done: Modelling declassification as a completeness problem;
  - We analyze the accuracy of a declassification policy;
  - We associate with each public observation the corresponding information released;
  - We can refine the accuracy of the policy;
  - We create a connection with abstract model checking;
- What we have to do: We are interested in...
  - ...extending our approach to more complex systems;
  - ...exploiting this connection for implementing our approach;

Abstract Interpretation

Consider the complete lattice $< C, \leq, \wedge, \vee, \bot, \top >$, $A_i \in \mathit{uco}(C)$

Lattice of Abstract Domains $\equiv$ Lattice *uco*

$\mathsf{A} \equiv \wp(C)$

$< \mathit{uco}(C), \sqsubseteq, \sqcap, \sqcup, \lambda x.\ \top, \lambda x.\ x >$

Abstract Interpretation

Consider the complete lattice $< C, \leq, \wedge, \vee, \bot, \top >$, $A_i \in \mathit{uco}(C)$

Lattice of Abstract Domains $\equiv$ Lattice *uco*

A $\equiv \wp(C)$

$< \mathit{uco}(C), \sqsubseteq, \sqcap, \sqcup, \lambda x.\ \top, \lambda x.\ x >$

$$A_1 \sqsubseteq A_2 \quad \Leftrightarrow \quad A_2 \subseteq A_1$$

Abstract Interpretation

Consider the complete lattice $< C, \leq, \wedge, \vee, \bot, \top >$, $A_i \in \mathit{uco}(C)$

Lattice of Abstract Domains $\equiv$ Lattice *uco*

A $\equiv \wp(C)$

$< \mathit{uco}(C), \sqsubseteq, \sqcap, \sqcup, \lambda x.\ \top, \lambda x.\ x >$

$$A_1 \sqsubseteq A_2 \quad \Leftrightarrow \quad A_2 \subseteq A_1$$

$$\sqcap_i A_i = \mathcal{M}(\cup_i A_i)$$

Abstract Interpretation

Consider the complete lattice $< C, \leq, \wedge, \vee, \bot, \top >$, $A_i \in \mathit{uco}(C)$

Lattice of Abstract Domains $\equiv$ Lattice *uco*

A $\equiv \wp(C)$

$< \mathit{uco}(C), \sqsubseteq, \sqcap, \sqcup, \lambda x.\ \top, \lambda x.\ x >$

$$A_1 \sqsubseteq A_2 \quad \Leftrightarrow \quad A_2 \subseteq A_1$$

$$\sqcap_i A_i = \mathcal{M}(\cup_i A_i)$$

$$\sqcup_i A_i = \cap_i A_i$$

Abstract Interpretation

Consider the complete lattice $< C, \leq, \wedge, \vee, \bot, \top >$, $A_i \in \mathit{uco}(C)$

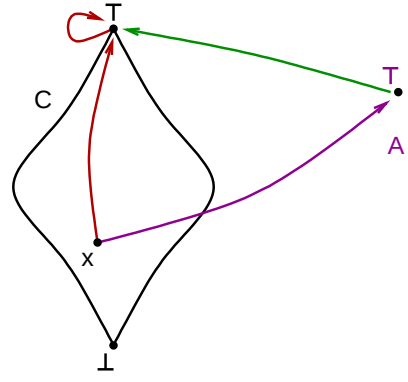
Lattice of Abstract Domains $\equiv$ Lattice *uco*

$\mathsf{A} \equiv \rho(C)$

$< \mathit{uco}(C), \sqsubseteq, \sqcap, \sqcup, \lambda x.\ \top, \lambda x.\ x >$

$$A_1 \sqsubseteq A_2 \quad \Leftrightarrow \quad A_2 \subseteq A_1$$
$$\sqcap_i A_i = \mathcal{M}(\cup_i A_i)$$
$$\sqcup_i A_i = \cap_i A_i$$

Top:



Abstract Interpretation

Consider the complete lattice $< C, \leq, \wedge, \vee, \bot, \top >$, $A_i \in \mathit{uco}(C)$

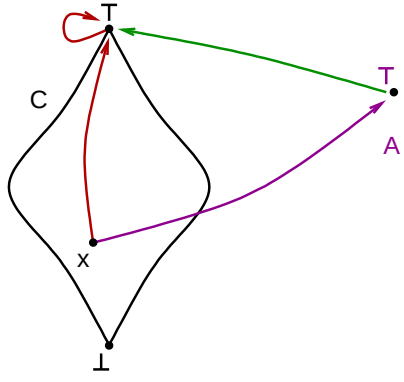
Lattice of Abstract Domains $\equiv$ Lattice *uco*

A $\equiv \rho(C)$

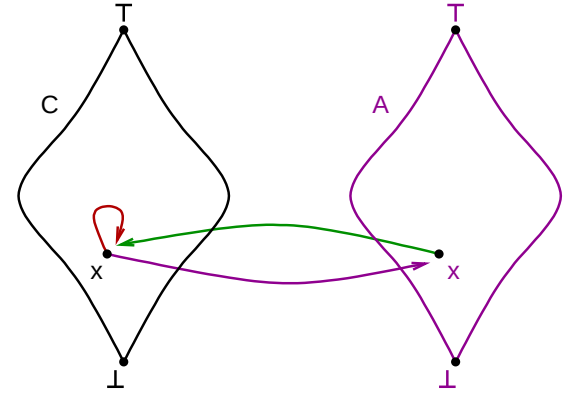
$< \mathit{uco}(C), \sqsubseteq, \sqcap, \sqcup, \lambda x.\ \top, \lambda x.\ x >$

$$A_1 \sqsubseteq A_2 \quad \Leftrightarrow \quad A_2 \subseteq A_1$$
$$\sqcap_i A_i = \mathcal{M}(\cup_i A_i)$$
$$\sqcup_i A_i = \cap_i A_i$$

Top:



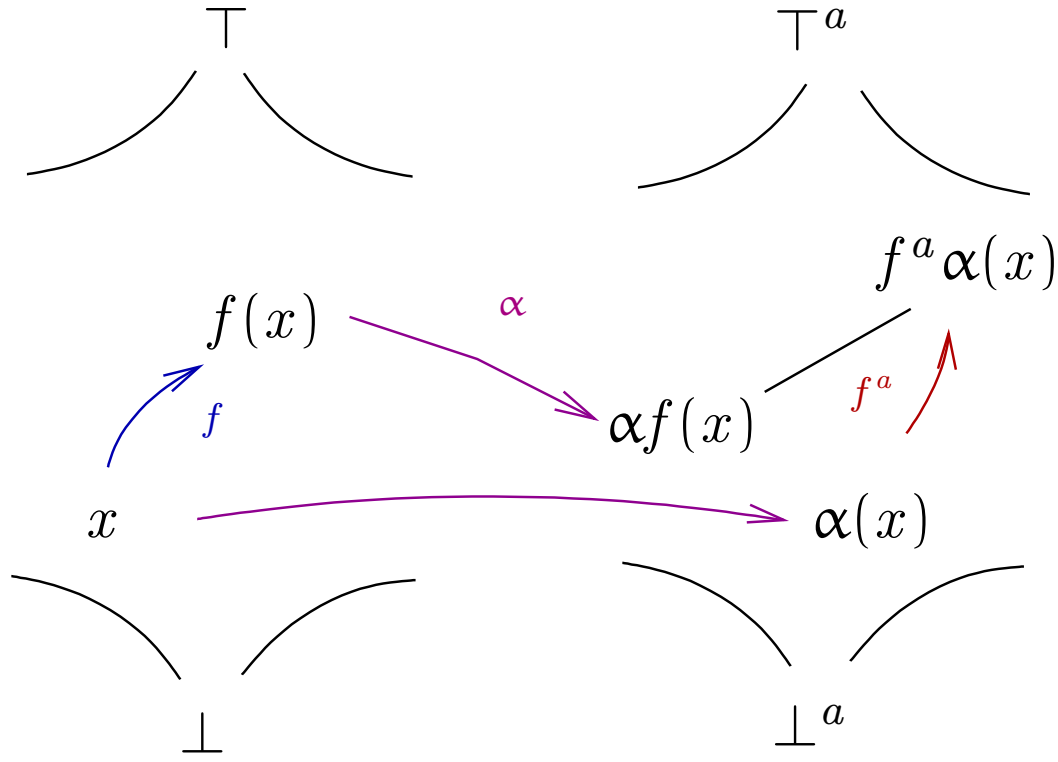
Bottom:



Abstract domain *backward* completeness

Let $< A, \alpha, \gamma, C >$ a Galois insertion. [Cousot & Cousot '77,'79]
$f : C \longrightarrow C$, $f^a = \alpha \circ f \circ \gamma : A \longrightarrow A$ (b.c.a. of f) and $\rho = \gamma \circ \alpha$

ρ correct for f



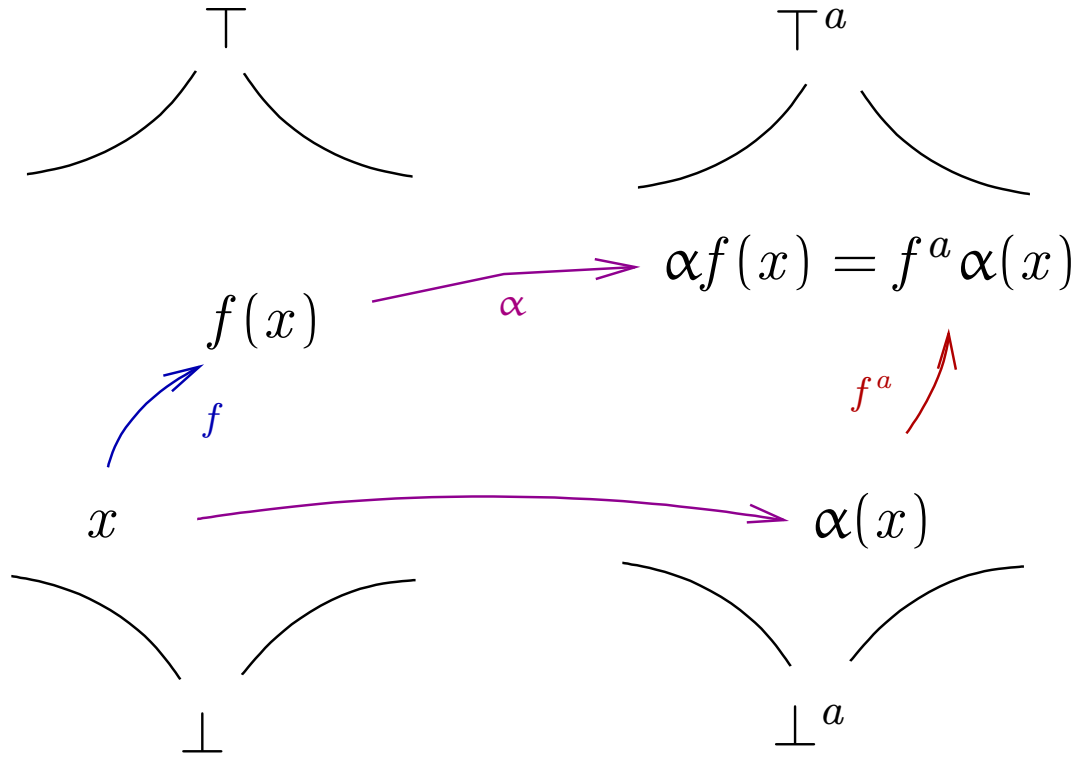
Abstract domain *backward* completeness

Let $< A, \alpha, \gamma, C >$ a Galois insertion. [Cousot & Cousot '77,'79]
$f : C \longrightarrow C$, $f^a = \alpha \circ f \circ \gamma : A \longrightarrow A$ (b.c.a. of f) and $\rho = \gamma \circ \alpha$

ρ complete for f

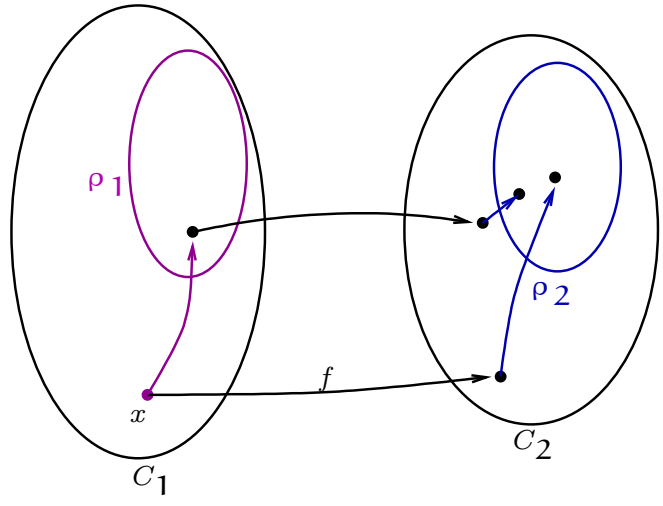
$\Updownarrow$

$\rho f \rho = \rho f$



Making *backward* complete

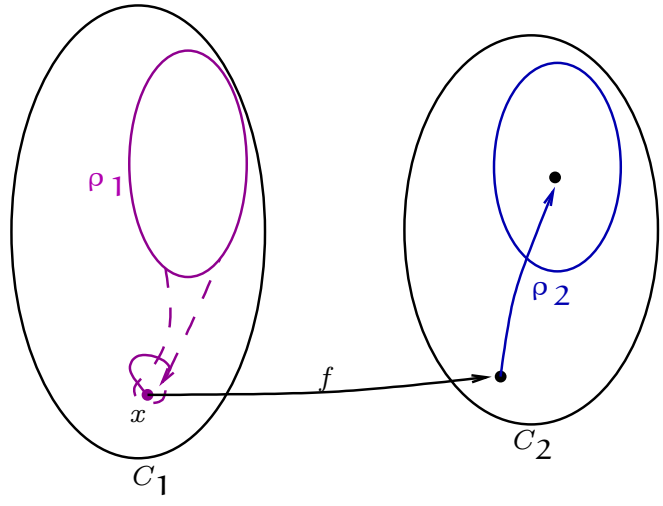
Giacobazzi et al. ‘00



$$\rho_2 f \rho_1 = \rho_2 f$$

Making *backward* complete

Giacobazzi et al. ‘00

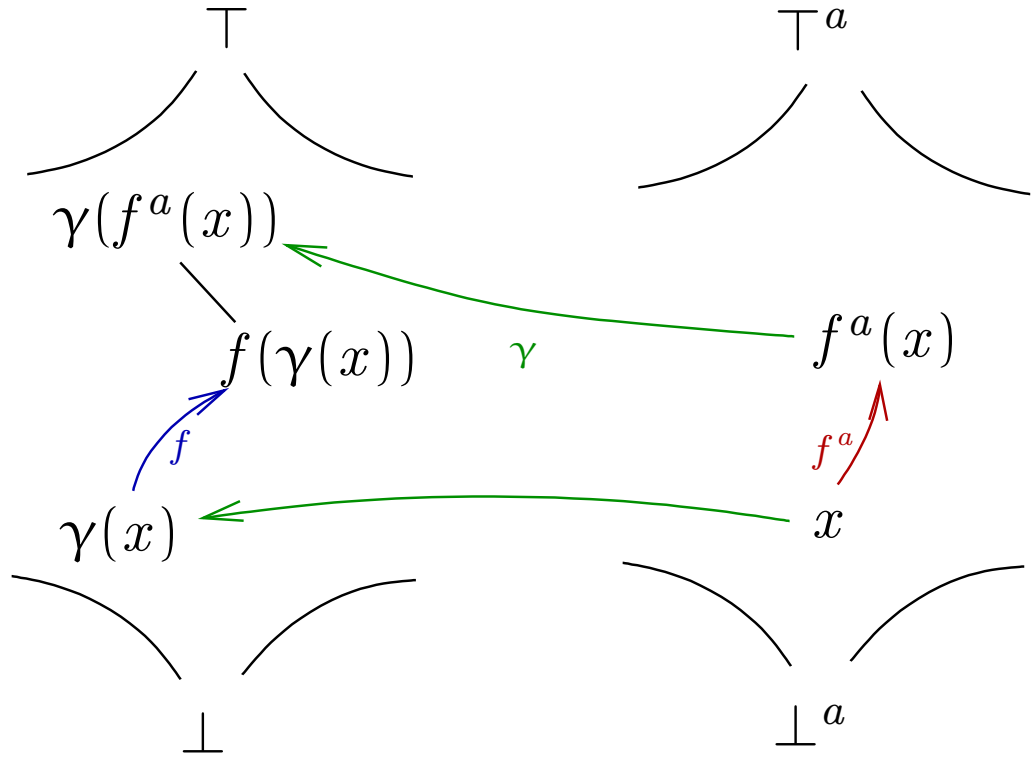


$$\rho_2 f \rho_1 = \rho_2 f$$

Abstract domain *forward* completeness

Let $< A, \alpha, \gamma, C >$ a Galois insertion. [Cousot & Cousot '77,'79]
$f : C \longrightarrow C$, $f^a = \alpha \circ f \circ \gamma : A \longrightarrow A$ (b.c.a. of f) and ρ=$\gamma \circ \alpha$

ρ correct for f



Abstract domain *forward* completeness

Let $< A, \alpha, \gamma, C >$ a Galois insertion. [Cousot & Cousot '77,'79]
$f : C \longrightarrow C$, $f^a = \alpha \circ f \circ \gamma : A \longrightarrow A$ (b.c.a. of f) and $\rho = \gamma \circ \alpha$

ρ complete for f

$\Updownarrow$

$\rho f \rho = f \rho$