Program Analysis for Quantified Information Flow

The 5th CREST Open Workshop

Chunyan Mu

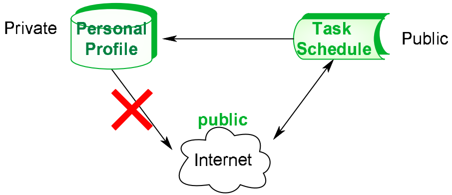
joint work with David Clark

CREST, King's College London

March 31, 2010

Outline

What is secure information flow?



- Information flows between objects of a computing systems, *e.g.*, devices, agents, variables, channels etc.
- Information flow security is concerned with how security information is allowed to flow through a computer system.
- Flow is considered secure if it accepts a specified policy which defines the accessibility of the information.

Example: Secure information flow is violated

Security level

x: HIGH security variable $\qquad$ y: LOW security variable

Assignment

$y := x;$

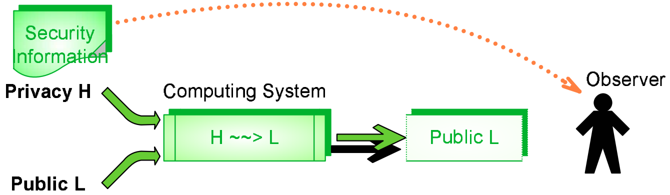
Control flow

$\textit{if } (x \bmod 2 == 0) \textit{ then } y := 0 \textit{ else } y := 1$

Termination behaviour

$y := x;$
$\textit{while}(y \neq 0)\ x := x * x$

Non-interference is too restrictive!



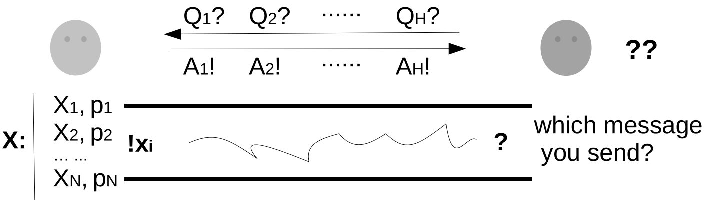
How much information is leaked?

- A new policy to relax the NI
- From quantitative view, the program is secure if the amount of information flow from high to low is small enough.
- Idea: we treat the program as a communication channel, use information theory, consider how much interference?

Outline

Information

An Intuitive Example



- Let H be the average minimum number of questions the receiver needs to guess which symbol you will send:

$$2^H = N \qquad H = \log_2 N$$

$$H = -\log_2 \frac{1}{N} \qquad H = -\log_2 p$$

Information

Information and entropy

- Surprise of an event x_i occurring with probability p_i:

$$-\log_2 p_i$$

- Information (entropy) = expected value of surprise:

$$\mathcal{H} \stackrel{\text{def}}{=} \sum_1^n p_i \log_2 \frac{1}{p_i}$$

- Equivalent to a measurement of uncertainty or variation
- Information is maximised under uniform distribution:

$$\mathcal{H} \leq \log_2 n$$

Random Variables

- A discrete random variable is a surjective function from sample space to observation space:

$$X : D \to \mathcal{R}(D)$$

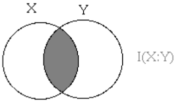
  where D is a finite set with a specified probability distribution, and $\mathcal{R}$ is the finite range of X
- Joint random variable: $\langle X, Y \rangle$
- Random variable X conditioned on $Y = y$: $P(X = x | Y = y)$

Shannon's measure of entropy

Entropy (expected value of surprise when X is observed)

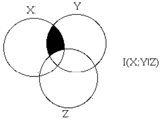
$$\mathcal{H}(X) = \sum_{x \in X} p(x) \log_2 \frac{1}{p(x)} = -\sum_x p(x) \log_2 p(x)$$

Mutual Information (shared information)



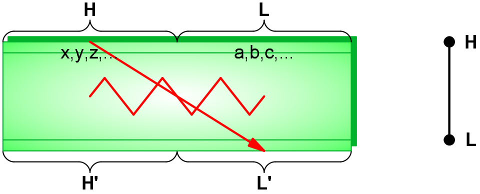
$$\mathcal{I}(X;Y) = \mathcal{H}(X) + \mathcal{H}(Y) - \mathcal{H}(X,Y)$$

Conditional Mutual Information



$$\mathcal{I}(X;Y|Z) = \mathcal{H}(X|Z) + \mathcal{H}(Y|Z) - \mathcal{H}(X,Y|Z)$$

Leakage definition



Leakage Definition for Batch Programs

- $\mathcal{L}(H, L') \triangleq \mathcal{I}(H; L'|L) = \mathcal{H}(L'|L)$ [CHM07]
- Technical considerations allow us to consider $\mathcal{L}(H, L')$ as $\mathcal{H}(L')$ [CHM07]
- How to calculate $\mathcal{H}(L')$??

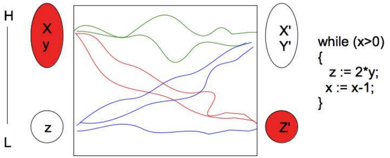
Outline

Current approaches

	description	language	tool	scalability	automatic
Clark,Hunt,Mal	bounds analysis	while	-	$\surd$	$\surd$
Malacaria	partition property	-	-	-	-
McCament,Ernst	dynanmic analysis	C	$\surd$	$\surd$	$\surd$
Backes,Köpf,Ryb	model checking	C	$\surd$	-	$\surd$
Heusser,Mal	model checking	C	$\surd$	-	$\surd$
Lowe	refusal counting	CSP	-	-	-
Boreale	IT in process calculus	CCS	-	-	-

Outline

The idea



- Consider simple imperative programs:
  `skip|ass|if|while|compose`
- Apply probabilistic domain transformer semantics to calculate distribution on outputs given distribution on inputs
- Use information theory to measure flow for a giving input distribution
- Automate the computation of the flows using the semantics

The semantics

$$\begin{array}{ll} \mathcal{M}[\![\texttt{Cmd}]\!] : \Sigma \to \Sigma & \mathcal{M}[\![\texttt{Exp}]\!] : \Sigma \to \mathit{Val} \\ \mathcal{M}[\![\texttt{BExp}]\!] : \Sigma \to \Sigma & \mathit{Val} : X \\ \text{stores } \Sigma : \texttt{Ide} \to \mathit{Val} & \end{array}$$

Figure: Semantics Domains

$$\begin{array}{rcl} f_{[\![x:=e]\!]}(\mu) & \triangleq & \lambda X.\mu(f^{-1}_{[\![x:=e]\!]}(X)) \\ f_{[\![c_1]\!];[\![c_2]\!]}(\mu) & \triangleq & f_{[\![c_2]\!]} \circ f_{[\![c_1]\!]}(\mu) \\ f_{[\![\texttt{if b c}_1\texttt{ c}_2]\!]}(\mu) & \triangleq & f_{[\![c_1]\!]} \circ f_{[\![b]\!]}(\mu) \; + \; f_{[\![c_2]\!]} \circ f_{[\![\neg b]\!]}(\mu) \\ f_{[\![\texttt{while b do c}]\!]}(\mu) & \triangleq & f_{[\![\neg b]\!]}(\lim_{n\to\infty}(\lambda\mu'.\mu + \\ & & f_{[\![c]\!]} \circ f_{[\![b]\!]}(\mu'))^n(\lambda X.\bot)) \\ & & \text{where, } f_{[\![B]\!]}(\mu) = \lambda X.\mu(X \cap B) \end{array}$$

Figure: Probabilistic Denotational Semantics

The leakage definition of loops

Entropy of loops

- We define the leakage for loops up to k^{th} iterations by:

$$\begin{array}{rcl} E & \mapsto & \mathcal{L}_{\texttt{while}}(k) = \widetilde{\mathcal{H}}(\mathcal{P}) + \widetilde{\mathcal{H}}(\mathcal{Q}|\mathcal{P}) \\ & = & \widetilde{\mathcal{H}}(\mathcal{P}_0 \cup \cdots \cup \mathcal{P}_k) + \widetilde{\mathcal{H}}(\mathcal{Q}_0 \cup \cdots \cup \mathcal{Q}_k | \mathcal{P}_0 \cup \cdots \cup \mathcal{P}_k) \end{array}$$

- case $k < n$, we can compute the leakage due to each iteration before the loop terminates with the time of observation
- case $k = n$, this definition has been proved equivalent to Malacaria's leakage definition of loops [Mal07]
- case $k = \infty$, nonterminating loops, $\mathcal{H}(\perp) = 0$

Leakage Analysis by Probabilistic Semantics: Example

Example: A terminating loop

```
l:=0; while(l<h) l:=l+1;
```

- Assume h is 3-bit *high* security variable with distribution:
  $\left[\ 0 \ w.p.\ \frac{7}{8} \quad 1 \ w.p.\ \frac{1}{56} \quad \ldots \quad 7 \ w.p.\ \frac{1}{56}\ \right]$
- l is *low* security variable
- Consider the decompositions $\mathcal{P}_i$ and $\mathcal{Q}_i$ due to event b^i:

$$
\begin{array}{ll}
\mathcal{P}_0 = \{\mu(b^0)\} = \{\frac{7}{8}\} & \qquad \mathcal{Q}_0 = \{\mu_l(0)\} = \{\frac{7}{8}\} \\
\mathcal{P}_1 = \{\mu(b^1)\} = \{\frac{1}{56}\} & \qquad \mathcal{Q}_1 = \{\mu_l(1)\} = \{\frac{1}{56}\} \\
\ldots & \qquad \ldots \\
\mathcal{P}_7 = \{\mu(b^7)\} = \{\frac{1}{56}\} & \qquad \mathcal{Q}_7 = \{\mu_l(7)\} = \{\frac{1}{56}\}
\end{array}
$$

Leakage Analysis by Probabilistic Semantics

Example: A terminating loop

- Note that $q_i = p_i$, hence $\widetilde{\mathcal{H}}(\mathcal{Q}|\mathcal{P}) = 0$, i.e., the information flow within body is 0
- The leakage computation due to each iteration:

$$
\begin{aligned}
\mathcal{L}_{\texttt{while}-0} &= \widetilde{\mathcal{H}}(\mathcal{P}_0) = 0.192645 \\
\mathcal{L}_{\texttt{while}-1} &= \widetilde{\mathcal{H}}(\mathcal{P}_0 \cup \mathcal{P}_1) = 0.304939275 \\
\mathcal{L}_{\texttt{while}-2} &= \widetilde{\mathcal{H}}(\mathcal{P}_0 \cup \mathcal{P}_1 \cup \mathcal{P}_2) = 0.412829778 \\
\mathcal{L}_{\texttt{while}-3} &= \widetilde{\mathcal{H}}(\mathcal{P}_0 \cup \mathcal{P}_1 \cup \mathcal{P}_2 \cup \mathcal{P}_3) = 0.516570646 \\
\mathcal{L}_{\texttt{while}-4} &= \widetilde{\mathcal{H}}(\mathcal{P}_0 \cup \mathcal{P}_1 \cup \cdots \cup \mathcal{P}_4) = 0.616396764 \\
\mathcal{L}_{\texttt{while}-5} &= \widetilde{\mathcal{H}}(\mathcal{P}_0 \cup \mathcal{P}_1 \cup \cdots \cup \mathcal{P}_5) = 0.71252562 \\
\mathcal{L}_{\texttt{while}-6} &= \widetilde{\mathcal{H}}(\mathcal{P}_0 \cup \mathcal{P}_1 \cup \cdots \cup \mathcal{P}_6) = 0.805158879 \\
\mathcal{L}_{\texttt{while}-7} &= \widetilde{\mathcal{H}}(\mathcal{P}_0 \cup \mathcal{P}_1 \cup \cdots \cup \mathcal{P}_7) = 0.894483808
\end{aligned}
$$

Outline

The idea

- define an abstraction on the measure space
  - concrte lattice
  - abstract lattice
  - Galois connection
- abstract semantic operations are applied to the abstract space
  - soundness and correctness of the abstraction
- estimate the abstract spaces to provide safe bounds on the entropy computation

Measurable partitions and abstract domain

Concrete lattice

- the σ-algebra $\mathcal{B}$ of a finite measure space X forms a complete lattice
- we define a partial order on $\mathcal{B}$ as follows:

$$\forall x_1, x_2 \in \mathcal{B}, x_1 < x_2 \textit{ iff } \mathcal{H}(x_1) \leq \mathcal{H}(x_2)$$

- define an equivalence relation on $\mathcal{B}$:

$$x_1 \simeq x_2 \textit{ iff } \mathcal{H}(x_1) = \mathcal{H}(x_2)$$

Measurable partitions and abstract domain

Abstract space

- An element of the abstract domain $x_i^\sharp \in X^\sharp$ is defined as a pair $(\mu_i, [E_i])$, where μ_i is the weight on the element
- Adjust the concrete space to be sorted
- Make the partition: $\xi = \{E_i | 1 \leq i \leq n\}$
- Lift to interval-based partition:

$$[E_i] : \langle I_1^i, I_2^i, \ldots, I_k^i \rangle \rightarrow \mu_i$$

Measurable partitions and abstract domain

The Galois connection

- the abstraction function α is a mapping from concrete space X to the sets of interval-based partitions $X^\sharp$: $X \longrightarrow [X/\xi]$, where $[X/\xi] = \{(\mu_i, [E_i]) | 0 < i \leq n\}$
- the concretisation function γ is a mapping: $X^\sharp \rightarrow \bigcup\{x | x \in [E_i]/\eta\}$, where the $[E_i]$ are the blocks of the abstract object $X^\sharp$, η is a sub-partition on each block under uniform distribution

Entropy of Measurable Partition and Leakage Computation

- Uniformalisation: a transformation of each block of space of a variable into a space with uniform distribution on each block
- let $[\![\cdot]\!]\xi = \xi'$, the leakage upper bound

$$
\begin{aligned}
U_v &= \mathcal{H}(\xi'\eta) = \mathcal{H}(\xi') + \mathcal{H}(\eta|\xi') \\
&= \mathcal{H}(\mu_1, \ldots, \mu_n) + \sum_{i=1}^{n} \mu_i \mathcal{H}(\frac{\mu_i/N_i}{\mu_i}, \ldots, \frac{\mu_i/N_i}{\mu_i}) \\
&= \mathcal{H}(\mu_1, \ldots, \mu_n) + \sum_{i=1}^{n} \mu_i \log_2(N_i)
\end{aligned}
$$

where N_i is the size of the partition E_i

Example

$[\![$`l:=0; while(l<h) do l++;`$]\!]$

- initial distribution $\mu_h \mapsto \begin{pmatrix} 0 \ w.p.\ 0.1, & 1 \ w.p.\ 0.1 \\ 2 \ w.p.\ 0.1, & 3 \ w.p.\ 0.1 \\ 4 \ w.p.\ 0.2, & 5 \ w.p.\ 0.2 \\ 6 \ w.p.\ 0.1, & 7 \ w.p.\ 0.1 \end{pmatrix}$
- Consider the partitions ξ:

$$\left\{ \begin{array}{l} E_1\langle [0,3]_h, [0,0]_l \rangle \rightarrow 0.4, \\ E_2\langle [4,7]_h, [0,0]_l \rangle \rightarrow 0.6 \end{array} \right\}$$

Example

$[\![$`l:=0; while(l<h) do l++;`$]\!]$

- A fixpoint is reached at the end.
- Concentrate on the low variable, do uniformalisation on each block to concretise the final space, and have:

$$\left\{\begin{array}{l} [0,3]_l \rightarrow 0.4, \\ [4,7]_l \rightarrow 0.6 \end{array}\right\} \xRightarrow{\texttt{Uniformalisation}} \mu_l \mapsto \left(\begin{array}{cc} 0 \rightarrow 0.4/4 & 1 \rightarrow 0.4/4 \\ 2 \rightarrow 0.4/4 & 3 \rightarrow 0.4/4 \\ ----- & ----- \\ 4 \rightarrow 0.6/4 & 5 \rightarrow 0.6/4 \\ 6 \rightarrow 0.6/4 & 7 \rightarrow 0.6/4 \end{array}\right)$$

- leakage upper bound:
  $U_l = \mathcal{H}(0.4, 0.6) + 0.4 * \log_2 4 + 0.6 * \log_2 4 = 2.97$
- exact leakage: $\mathcal{L} = 2.92$

Outline

The idea

- Consider the quantity of information flow in reactive processes by looking at the different behaviours of a high user from a low user's observations.
- The reactive system is modelled by using Probabilistic Labelled Transition System (PLTS)
- The observation records the history traces of behaviours from the view of low users in a way of distributions.
- Introduce a transformation on the process tree.
- A metric space is built upon the transformation tree, and the information flow is measured via metrics.

The Probabilistic Model

Probabilistic Labelled Transition Systems

- The PLTS is given as a triple $PLTS = (T, \Sigma, \mu)$
- Specifically, $\mu_{p,a} : T \rightarrow [0, 1]$, $\mu_p : \Sigma \rightarrow T \rightarrow [0, 1]$, where for any $a \in \Sigma$ and p is a state that can perform the action a, indicating the possible next states and their probabilities after p has performed a.
- Furthermore, $\forall p \in T$ and can perform action a, $\sum_{p' \in T} \mu_{a,p}(p') = 1$, *i.e.*, $\mu_{p,a}$ is a probability distribution.

The Language and its Semantics

Syntax

$$F ::= \bot \mid x \mid \sum_{i \in I} a_i.p_i.F_i \mid \sqcap S \mid F_1 \| F_2 \mid \mu x.F$$

Operational Semantics

Act $$\frac{E \xrightarrow{a_i}_{p_i} E_i}{E \xrightarrow{a}_{\pi} \sum_{i=1}^{n} p_i.a_i.E_i}$$
$$\pi : \{p_i | 1 \leq i \leq n\},\ a = \{a_i | 1 \leq i \leq n\}$$

Par $$\frac{E_1 \xrightarrow{\tau} E_1'}{E_1 \| E_2 \xrightarrow{\tau} E_1' \| E_2} \qquad \frac{E_2 \xrightarrow{\tau} E_2'}{E_1 \| E_2 \xrightarrow{\tau} E_1 \| E_2'}$$
$$\frac{E_1 \xrightarrow{a}_{\pi_1} E_1' \quad E_2 \xrightarrow{a}_{\pi_2} E_2'}{E_1 \| E_2 \xrightarrow{a}_{\pi_1, \pi_2} \pi_1 \pi_2 . a.(E_1' \| E_2')} (a \neq \tau)$$

Rec $$\frac{}{\mu x.E \xrightarrow{\tau} E[\mu x.E/x]}$$

Observation on traces

- A set of traces can be extracted from the process tree built by the semantics structure.
- Consider the observation as the sum of the low projection on such traces.
- Information on the projection of the high inputs from the trace can be deduced from these observations.
- Under repeated observation on traces, we can deduce probability distributions on the possible traces.

Probabilistic Low Bi-simulation $\sim_L$

- an extension to the concept of bisimulation
- an equivalence relation on the set of processes $\mathcal{R}$ produced by the PLTS, such that, whenever

  $$E_i \sim_L E_j$$

  the following holds:

  $$\forall S \in \mathcal{R}/\sim_L . E_i \xrightarrow{L}_{\mu} S \Leftrightarrow E_j \xrightarrow{L}_{\mu} S$$

  where $\mathcal{R}/\sim_L$ denotes the set of bisimilar classes of $\mathcal{R}$ under $\sim_L$ and $E_i \xrightarrow{L}_{\mu} S$ if and only if $\mu = \sum\{\mu' | E_i' \in S\}$ and $E_i \xrightarrow{L}_{\mu'} E_i'$.

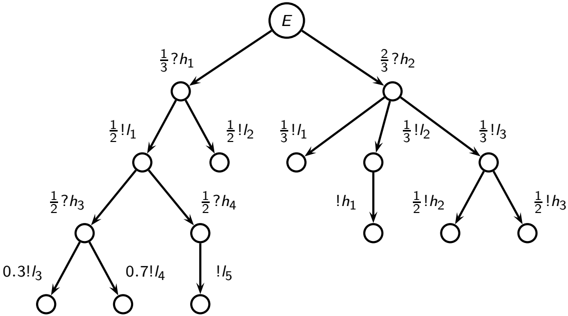
A Transformation on the Process Tree

Interaction unit

Define a (high) interaction unit (step) as a subtree of the process tree whose

- root is labelled by a high input action
- includes every branch terminated by a high input action or $\perp$.

Example process tree



Transformation trees on interaction units

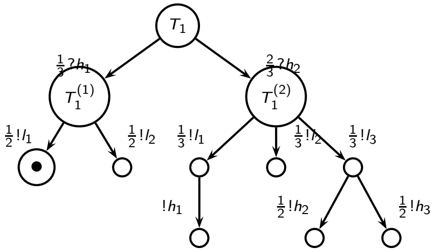


Figure: Transformation tree on the first interaction step T_1

we obtain two subtrees due to the two atomic actions $?h_1$ and $?h_2$ in $?H_0$ as:

$$
\begin{aligned}
T_1^{(1)} &= \left(\frac{1}{2}!l_1.\bot + \frac{1}{2}!l_2.\bot\right) \rightarrow \frac{1}{3} \\
T_1^{(2)} &= \left(\frac{1}{3}!l_1.!h_1.\bot + \frac{1}{3}!l_2.\bot + \frac{1}{6}!l_3.!h_2.\bot + \frac{1}{6}!l_3.!h_3.\bot\right) \rightarrow \frac{2}{3}
\end{aligned}
$$

Transformation trees on interaction units

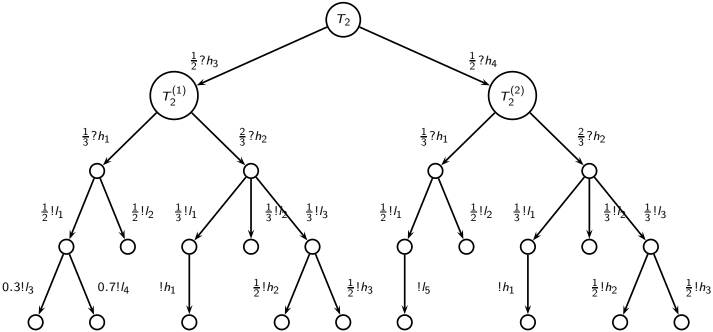


Figure: Transformation tree on the second interaction step T_2

Transformation trees on interaction units

we obtain two subtrees due to the two atomic action $?h_1$ and $?h_2$ in $?H_1$ as:

$$
\begin{aligned}
T_2^{(1)} &= (\frac{0.3}{6}?h_1.!l_1.!l_3.\perp + \frac{0.7}{6}?h_1.!l_1.!l_4.\perp + \frac{1}{6}?h_1.!l_2.\perp + \\
&\quad \frac{2}{9}?h_2.!l_1.!h_1.\perp + \frac{2}{9}?h_2.!l_2.\perp + \frac{1}{9}?h_2.!l_3.!h_2.\perp + \frac{1}{9}?h_2.!l_3.!h_3.\perp) \\
&\quad \rightarrow \frac{1}{2} \\
T_2^{(2)} &= (\frac{1}{6}?h_1.!l_1.!l_5.\perp + \frac{1}{6}?h_1.!l_2.\perp + \frac{2}{9}?h_2.!l_1.!h_1.\perp + \\
&\quad \frac{2}{9}?h_2.!l_2.\perp + \frac{1}{9}?h_2.!l_3.!h_2.\perp + \frac{1}{9}?h_2.!l_3.!h_3.\perp) \rightarrow \frac{1}{2}
\end{aligned}
$$

Observation on the transformation tree

- the observation due to the first interaction unit:

$$\begin{aligned} \mathcal{O}(T_1^{(1)}) &= (\frac{1}{2}!l_1 + \frac{1}{2}!l_2) \to \frac{1}{3} \\ \mathcal{O}(T_1^{(2)}) &= (\frac{1}{3}!l_1 + \frac{1}{3}!l_2 + \frac{1}{3}!l_3) \to \frac{2}{3} \end{aligned}$$

- the observation due to the second interaction unit:

$$\begin{aligned} \mathcal{O}(T_2^{(1)}) &= (\frac{0.3}{6}?h_1.!l_1.!l_3.\perp + \frac{0.7}{6}?h_1.!l_1.!l_4.\perp + \frac{1}{6}?h_1.!l_2.\perp + \frac{2}{9}?h_2.!l_1.\perp + \\ &\quad \frac{2}{9}?h_2.!l_2.\perp + \frac{2}{9}?h_2.!l_3.\perp) \to \frac{1}{2} \\ \mathcal{O}(T_2^{(2)}) &= (\frac{1}{6}?h_1.!l_1.!l_5.\perp + \frac{1}{6}?h_1.!l_2.\perp + \frac{2}{9}?h_2.!l_1.\perp + \frac{2}{9}?h_2.!l_2.\perp + \\ &\quad \frac{2}{9}?h_2.!l_3.\perp) \to \frac{1}{2} \end{aligned}$$

Information Flow Measurement

Jensen-Shannon Divergence (JSD)

- Consider m distributions $P^{(1)}, P^{(2)}, \ldots, P^{(m)}$.
- The JSD between the m distributions $P^{(1)}, \ldots, P^{(m)}$ with weights $w^{(1)}, \ldots, w^{(m)}$ is given by

$$D_{\mathrm{JS}}(P^{(1)}, P^{(2)}, \ldots, P^{(m)}) = \mathcal{H}(\sum_{j=1}^{m} w^{(j)} P^{(j)}) - \sum_{j=1}^{m} w^{(j)} \mathcal{H}(P^{(j)})$$

Information Flow Measurement

The Metric

For a set of processes $f_1, \ldots, f_m \in \mathcal{R},\ d_\mu(f_1, \ldots, f_m)$ is defined as:

$$d_\mu(f_1, \ldots, f_m) = \sqrt{\mathcal{H}(\sum_{j=1}^{m} w^{(j)} P^{(j)}) - \sum_{j=1}^{m} w^{(j)} \mathcal{H}(P^{(j)})}$$

Proposition 2.

For any processes $f_1, \ldots, f_m \in \mathcal{R},\ d(f_1, \ldots, f_m),\ d(f_1, \ldots, f_m) = 0$ iff $f_1 \sim_L \cdots \sim_L f_m$.

Build the metric spaces

- Consider all the interaction units, we build a collection of metric spaces $(\bigcup T_i, \bigcup d_i),\ (i = 0, 1, \dots)$:

$$T_0 = \{p_0\},\ T_1 =?H_0 \prec T_0,\ \dots,\ T_{n+1} =?H_n \prec T_n$$

- Clearly, for $T_i^{(1)}, \dots, T_i^{(m_i)} \in T_i$,
  - if $P_i^{(1)} = \cdots = P_i^{(m_i)},\ d_i = 0$;
  - otherwise $d_i = \sqrt{D_{\mathrm{JS}}(P_i^{(1)}, \dots, P_i^{(m_i)})}$ is the metric between the distributions extracted from the subtree set due to the interaction step i.

Quantity of the information flow

Definition of leakage

- For each interaction unit started by

$$?H_{i-1} = \{?h_{i-1,1} \to w_{i-1,1}, \ldots, ?h_{i-1,m_{i-1}} \to w_{i-1,m_{i-1}}\}$$

we have built a metric space $(T_i, d_i)_{i \geq 1}$, where

$$d_i = \sqrt{D_{\mathrm{JS}}(P_i^{(1)}, \ldots, P_i^{(m_{i-1})})}$$

- The leakage upper bound is defined as the square of the sum:

$$(\sum_{i=1}^{n} d_i)^2$$

Quantity of the information flow

Example

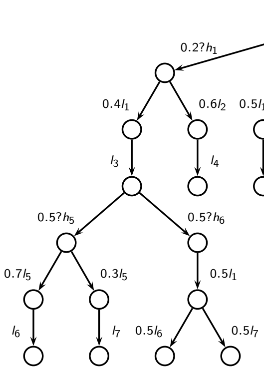


Figure: The example process tree

Quantity of the information flow

Example

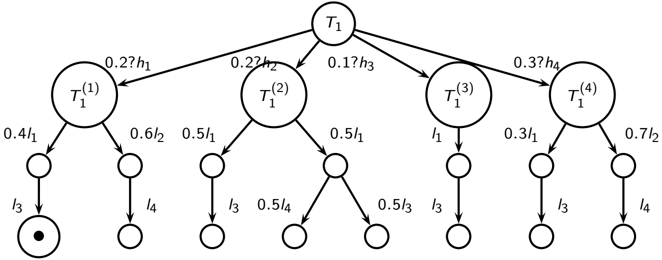


Figure: Transformation on the first interaction unit: T_1

Quantity of the information flow

Example

- the observations:

$$\begin{aligned}
\mathcal{O}(P_0^{(1)}) &= 0.4 \cdot l_1.l_3.\perp + 0.6 \cdot l_2.l_4.\perp \\
\mathcal{O}(P_0^{(2)}) &= 0.5 \cdot l_1.l_3.\perp + 0.25 \cdot l_2.l_4.\perp + 0.25 \cdot l_2.l_3.\perp \\
\mathcal{O}(P_0^{(3)}) &= l_1.l_3.\perp \\
\mathcal{O}(P_0^{(4)}) &= 0.3 \cdot l_1.l_3.\perp + 0.7 \cdot l_2.l_4.\perp
\end{aligned}$$

- the metric:

$$\begin{aligned}
d_1 &= \sqrt{\mathcal{H}(\sum_{i=1}^{4} w_0^{(i)} P_0^{(i)}) - \sum_{i=1}^{4} w_0^{(i)} \mathcal{H}(P_0^{(i)})} \\
&= \mathcal{H}(0.47, 0.43, 0.1) - (0.2\mathcal{H}(0.4, 0.6) + 0.4\mathcal{H}(0.5, 0.25, 0.25) + \\
&\quad 0.1 * 0 + 0.3\mathcal{H}(0.3, 0.7)) = 0.557
\end{aligned}$$

Quantity of the information flow

Example

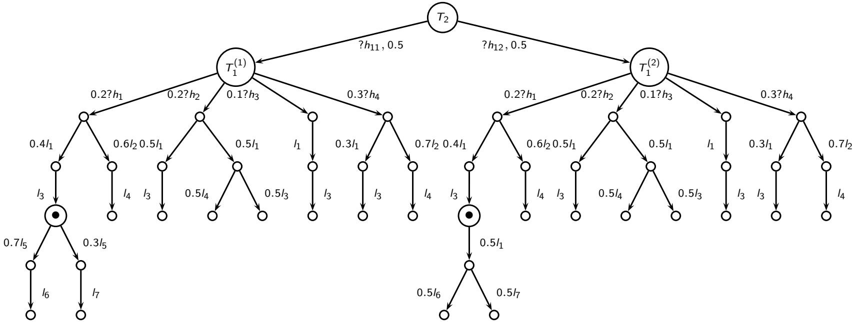


Figure: Transformation on the second interaction: T_2

Example: quantity of the information flow

- observations:

$$
\begin{aligned}
\mathcal{O}(P_1^{(1)}) &= 0.004?h_1.l_1.l_3.l_5.l_6.\perp + 0.04?h_1.l_1.l_3.l_5.l_7.\perp \\
&\quad 0.12?h_1.l_2.l_4.\perp + 0.2?h_2.l_1.l_3.\perp + 0.1?h_2.l_2.l_4.\perp + 0.1?h_2.l_2.l_3.\perp \\
&\quad 0.1?h_3.l_1.l_3.\perp + 0.09?h_4.l_1.l_3 + 0.21?h_4.l_2.l_4
\end{aligned}
$$

$$
\begin{aligned}
\mathcal{O}(P_1^{(2)}) &= 0.056?h_1.l_1.l_3.l_5.l_6.\perp + 0.024?h_1.l_1.l_3.l_5.l_7.\perp \\
&\quad 0.12?h_1.l_2.l_4.\perp + 0.2?h_2.l_1.l_3.\perp + 0.1?h_2.l_2.l_4.\perp + 0.1?h_2.l_2.l_3.\perp \\
&\quad 0.1?h_3.l_1.l_3.\perp + 0.09?h_4.l_1.l_3 + 0.21?h_4.l_2.l_4
\end{aligned}
$$

- the metric:

$$
\begin{aligned}
d_2 &= \sqrt{\mathcal{H}(\sum_{i=1}^{2} w_1^{(i)} P_1^{(i)}) - \sum_{i=1}^{2} w_1^{(i)} \mathcal{H}(P_1^{(i)})} \\
&= (\mathcal{H}(0.048, 0.032, 0.12, 0.2, 0.1, 0.1, 0.1, 0.09, 0.21) - \\
&\quad 0.5\mathcal{H}(0.056, 0.024, 0.12, 0.2, 0.1, 0.1, 0.1, 0.09, 0.21) - \\
&\quad 0.5\mathcal{H}(0.04, 0.04, 0.12, 0.2, 0.1, 0.1, 0.1, 0.09, 0.21))^{\frac{1}{2}} \doteq 0.049
\end{aligned}
$$

- The leakage upper bound:

$$\mathcal{L} \leq (d_1 + d_2)^2 \doteq 0.36$$

Outline

Conclusions

- We present an automatic analysis for measuring information flow within software systems.
- We quantify leakage in terms of information theory and incorporate this computation into probabilistic semantics.
- An abstraction on the exact leakage analysis
- An approach for leakage analysis in reactive processes