2019 Mooly Fest

April 6th, 2019 —
ETAPS, Prague, Czech Republic

Calculational design of a static dependency analysis

Patrick Cousot

New York University, Courant Institute of Mathematics, Computer Science

pcousot@cs.nyu.edu cs.nyu.edu/~pcousot

Motivation

Dependency

Dependency is prevalent in computer science:

- Non-interference (confidentiality, integrity)
- Security, privacy
- Slicing
- Temporal dependencies in synchronous languages (Lustre, Signal, *etc.*)
- etc.

The existing definitions

- are postulated a priori (par exemple Cheney, Ahmed, and Acar, 2011; D. E. Denning and P. J. Denning, 1977),
- without semantics justifications (except Assaf, Naumann, Signoles, Totel, and Tronel, 2017 ("hyper-collecting semantics"), Urban and Müller, 2018 on program exit uniquely)

We are interested in principles, in soundness proofs, not so much in a new more powerful dependency analysis.

Structural fixpoint trace semantics

Program syntax

- C statements limited to integers, assignments, statement lits, conditionals, iterations
- Programs are labelled to designate program points
  - at⟦S⟧: entry program point of S starts;
  - after⟦S⟧: normal exit program point of S;
  - in⟦S⟧: reachable program points of S (excluding after⟦S⟧);
  - break-to⟦S⟧: breaking point when S contains a **break ;** to exit a loop (then escape⟦S⟧ = tt);

Execution traces

- Program:

$$\ell_1\ \mathtt{x = 0\ ;\ \mathbf{while}}\ \ell_2\ \mathtt{(tt)\ \{}\ \ell_3\ \mathtt{x = x+1\ ;\ \}}\ \ell_4$$

- Infinite execution trace: $\ell_1 \xrightarrow{\mathtt{x} = 0 = 0} \ell_2 \xrightarrow{\mathtt{tt}} \ell_3 \xrightarrow{\mathtt{x} = \mathtt{x} + 1 = 1} \ell_2 \xrightarrow{\mathtt{tt}} \ell_3 \xrightarrow{\mathtt{x} = \mathtt{x} + 1 = 2} \ell_2 \ldots \ell_2 \xrightarrow{\mathtt{tt}} \ell_3 \xrightarrow{\mathtt{x} = \mathtt{x} + 1 = n} \ell_2 \xrightarrow{\mathtt{tt}} \ell_3 \xrightarrow{\mathtt{x} = \mathtt{x} + 1 = n+1} \ell_2 \ldots$

- Trace: finite or infinite sequence of program points separated by action (x = A = *value*, B, ¬B, et **break ;**)

Value of a variable (and an expression)

- The value of a variable x along a trace π is the last assigned value (or 0 at initialization).

$$\begin{aligned}
\varrho(\pi\ell \xrightarrow{\mathtt{x} = \mathtt{E} = v} \ell')\mathtt{x} &\triangleq v \\
\varrho(\pi\ell \xrightarrow{\cdots} \ell')\mathtt{x} &\triangleq \varrho(\pi\ell) \quad \text{otherwise} \\
\varrho(\ell)\mathtt{x} &\triangleq 0
\end{aligned}$$

- Value of an arithmetic expression

$$\begin{aligned}
\mathcal{A}[\![1]\!]\rho &\triangleq 1 \\
\mathcal{A}[\![\mathtt{x}]\!]\rho &\triangleq \rho(\mathtt{x}) \\
\mathcal{A}[\![\mathtt{A}_1 - \mathtt{A}_2]\!]\rho &\triangleq \mathcal{A}[\![\mathtt{A}_1]\!]\rho - \mathcal{A}[\![\mathtt{A}_2]\!]\rho
\end{aligned}$$

- Same for boolean expressions.

Structural fixpoint prefix/maximal trace semantics $\widehat{\mathcal{S}}^*[\![\mathtt{S}]\!]$

- The prefix trace semantics $\widehat{\mathcal{S}}^*[\![\mathtt{S}]\!]$ is a relation between
  - an initialization trace $\pi_0$at⟦S⟧ arriving at⟦S⟧, and
  - the prefix execution traces at⟦S⟧π continuing this initialization by zero or more execution steps
- The maximal trace semantics $\widehat{\mathcal{S}}^{+\infty}[\![\mathtt{S}]\!]$ collects the maximal finite traces and the infinite traces obtained as limits of their prefixes.

Structural fixpoint definition of the prefix trace semantics (I)

- Assignment S ::= $^{\ell}$ x = A ; (where at⟦S⟧ = ℓ)

$$\mathcal{S}^*[\![\mathtt{S}]\!] \triangleq \{\langle \pi\ell,\ \ell\rangle \mid \pi\ell \in \mathbb{T}^+\} \cup \{\langle \pi\ell,\ \ell \xrightarrow{\mathtt{x}\,=\,\mathtt{A}\,=\,v} \mathsf{after}[\![\mathtt{S}]\!]\rangle \mid \pi\ell \in \mathbb{T}^+ \wedge v = \mathcal{A}[\![\mathtt{A}]\!]\varrho(\pi\ell)\}$$

Structural fixpoint definition of the prefix trace semantics (II)

- Iteration S ::= **while** $^{\ell}$ (B) $\mathtt{S}_b$ (where at⟦S⟧ = ℓ):

$$\mathcal{S}^*[\![\mathtt{S}]\!] = \mathsf{lfp}^{\subseteq}\, \mathcal{F}^*[\![\mathtt{S}]\!]$$

$$\mathcal{F}^*[\![\mathtt{while}\ ^{\ell}\ (\mathtt{B})\ \mathtt{S}_b]\!](X) \triangleq \{\langle \pi_1\ell',\ \ell'\rangle \mid \pi_1\ell' \in \mathbb{T}^+ \wedge \ell' = \ell\} \quad \text{(a)}$$

$$\cup \{\langle \pi_1\ell',\ \ell'\pi_2\ell' \xrightarrow{\neg(\mathtt{B})} \mathsf{after}[\![\mathtt{S}]\!]\rangle \mid \langle \pi_1\ell',\ \ell'\pi_2\ell'\rangle \in X \wedge \mathcal{B}[\![\mathtt{B}]\!]\varrho(\pi_1\ell'\pi_2\ell') = \mathrm{ff} \wedge \ell' = \ell\} \quad \text{(b)}$$

$$\cup \{\langle \pi_1\ell',\ \ell'\pi_2\ell' \xrightarrow{\mathtt{B}} \mathsf{at}[\![\mathtt{S}_b]\!] \frown \pi_3\rangle \mid \langle \pi_1\ell',\ \ell'\pi_2\ell'\rangle \in X \wedge \mathcal{B}[\![\mathtt{B}]\!]\varrho(\pi_1\ell'\pi_2\ell') = \mathrm{tt} \wedge \langle \pi_1\ell'\pi_2\ell' \xrightarrow{\mathtt{B}} \mathsf{at}[\![\mathtt{S}_b]\!],\ \pi_3\rangle \in \mathcal{S}^*[\![\mathtt{S}_b]\!] \wedge \ell' = \ell\} \quad \text{(c)}$$

A definition of the form $d(\vec{x}) \triangleq \{f(\vec{x}') \mid P(\vec{x}', \vec{x})\}$ has the variables $\vec{x}'$ in $P(\vec{x}', \vec{x})$ bound to those of $f(\vec{x}')$ whereas $\vec{x}$ is free in $P(\vec{x}', \vec{x})$ since it appears neither in $f(\vec{x}')$ nor (by assumption) under quantifiers in $P(\vec{x}', \vec{x})$. The $\vec{x}$ of $P(\vec{x}', \vec{x})$ is therefore bound to the $\vec{x}$ of $d(\vec{x})$.

Properties

Property

- A property is represented by a set of elements (those elements which have the property)
- Even intergers: $2\mathbb{Z} \triangleq \{2k \mid k \in \mathbb{Z}\}$
- x has property P is $x \in P$
- Implication is $P_1 \subseteq P_2$

Semantic property

- The prefix trace semantics belongs to $\wp(\mathbb{T}^{+} \times \mathbb{T}^{+\infty})$
- A semantics property belongs to $\wp(\wp(\mathbb{T}^{+} \times \mathbb{T}^{+\infty}))$
- The abstraction

$$\langle \wp(\wp(\mathbb{T}^{+} \times \mathbb{T}^{+\infty})), \subseteq\rangle \xleftarrow[\lambda P \cdot \cup P]{\lambda Q \cdot \wp(Q)} \langle \wp(\mathbb{T}^{+} \times \mathbb{T}^{+\infty}), \subseteq\rangle$$

provides trace properties (*e.g.*safety, liveness, *etc.*)

Dependency, informally

Dependency, informally

- At program point ℓ, the variable y depends upon the initial value x_0 of variable x
  iff
  changing only x_0 will change the non-empty sequences of values $y_0, y_1, \ldots$ of y observed at ℓ whenever control reaches ℓ
- Example: ℓ_0 `if (x=0) { y=x;` ℓ_1 `}` ℓ_2
  - y does not depend on x neither at ℓ_0 nor at ℓ_1
  - y depends on x at ℓ_2
- No need to distinguish between explicit and implicit dependencies
- Absence of observation is not an observation
- No timing channels

Dependency, formally

Observation of the sequence of values of a variable at a program point

- non-empty initialization trace $\pi_0 \in \mathbb{T}^+$
- non-empty continuation trace $\pi \in \mathbb{T}^{+\infty}$
- $\mathsf{seqval}[\![\mathsf{y}]\!]\ell(\pi_0,\ \pi)$ is the sequence of values of the variable y at program point ℓ along the trace π continuing π_0

$$
\begin{aligned}
\mathsf{seqval}[\![\mathsf{y}]\!]\ell(\pi_0, \ell) &\triangleq \varrho(\pi_0)\mathsf{y} \\
\mathsf{seqval}[\![\mathsf{y}]\!]\ell(\pi_0, \ell') &\triangleq \ni \\
\mathsf{seqval}[\![\mathsf{y}]\!]\ell(\pi_0, \ell \xrightarrow{a} \ell''\pi) &\triangleq \varrho(\pi_0)\mathsf{y} \cdot \mathsf{seqval}[\![\mathsf{y}]\!]\ell(\pi_0 \smallfrown \ell \xrightarrow{a} \ell'', \ell''\pi) \\
\mathsf{seqval}[\![\mathsf{y}]\!]\ell(\pi_0, \ell' \xrightarrow{a} \ell''\pi) &\triangleq \mathsf{seqval}[\![\mathsf{y}]\!]\ell(\pi_0 \smallfrown \ell' \xrightarrow{a} \ell'', \ell''\pi)
\end{aligned}
$$

- $\mathsf{seqval}[\![\mathsf{y}]\!]\ell(\pi_0, \pi)$ is the empty sequence $\ni$ if ℓ never appears in π

  (co-inductive definition for infinite traces).

Difference between sequences of values ω and ω'

- Sequences that differ may have a common prefix but must eventually have a different value at some position in the sequences.

$$\mathsf{diff}(\omega, \omega') \triangleq \exists \omega_0, \omega_1, \omega_1', \nu, \nu' \ .\ \omega = \omega_0 \cdot \nu \cdot \omega_1 \wedge \omega' = \omega_0 \cdot \nu' \cdot \omega_1' \wedge \nu \neq \nu'$$

Dependency, formally

- Dependency property:

$$
\begin{aligned}
\mathcal{D}_{\mathsf{diff}}{}^{\ell}\langle \mathsf{x},\ \mathsf{y}\rangle \triangleq \{\Pi \in \wp(\mathbb{T}^+ \times \mathbb{T}^{+\infty}) \mid\ & \exists \langle \pi_0,\ \pi_1\rangle, \langle \pi_0',\ \pi_1'\rangle \in \Pi\ . \\
& (\forall \mathsf{z} \in \mathbb{V} \setminus \{\mathsf{x}\}\ .\ \varrho(\pi_0)\mathsf{z} = \varrho(\pi_0')\mathsf{z}) \wedge \\
& \mathsf{diff}(\mathsf{seqval}[\![\mathsf{y}]\!]\ell(\pi_0,\ \pi_1), \mathsf{seqval}[\![\mathsf{y}]\!]\ell(\pi_0',\ \pi_1'))\}
\end{aligned}
$$

- y depends on the initial value of x at program point ℓ in program P is:

$$\widehat{\mathcal{S}}^{+\infty}[\![\mathsf{P}]\!] \in \mathcal{D}_{\mathsf{diff}}{}^{\ell}\langle \mathsf{x},\ \mathsf{y}\rangle$$

- Lemma

$$\widehat{\mathcal{S}}^{+\infty}[\![\mathsf{P}]\!] \in \mathcal{D}_{\mathsf{diff}}{}^{\ell}\langle \mathsf{x},\ \mathsf{y}\rangle \Leftrightarrow \widehat{\mathcal{S}}^{*}[\![\mathsf{P}]\!] \in \mathcal{D}_{\mathsf{diff}}{}^{\ell}\langle \mathsf{x},\ \mathsf{y}\rangle$$

Value dependency abstraction

Abstraction en dépendance de données

- The abstraction of a semantic property $\mathcal{S} \in \wp(\wp(\mathbb{T}^+ \times \mathbb{T}^{+\infty}))$ into a value dependency property $\alpha^{\mathrm{d}}(\mathcal{S}) \in \mathbb{L} \to \wp(\mathbb{V} \times \mathbb{V})$ is:

$$\alpha^{\mathrm{d}}(\mathcal{S})\ell \quad \triangleq \quad \{\langle \mathsf{x},\ \mathsf{y}\rangle \mid \mathcal{S} \in \mathcal{D}_{\mathrm{diff}}{}^{\ell}\langle \mathsf{x},\ \mathsf{y}\rangle\}$$

- This is a Galois connection:

Lemma 1 $\langle \wp(\wp(\mathbb{T}^+ \times \mathbb{T}^{+\infty})),\ \subseteq\rangle \xrightleftharpoons[\alpha^{\mathrm{d}}]{\gamma^{\mathrm{d}}} \langle \mathbb{L} \to \wp(\mathbb{V} \times \mathbb{V}),\ \dot{\supseteq}\rangle$ where the concretization of a dependency property $\mathbf{D} \in \mathbb{L} \to \wp(\mathbb{V} \times \mathbb{V})$ is:

$$\gamma^{\mathrm{d}}(\mathbf{D}) \quad \triangleq \quad \bigcap_{\ell \in \mathbb{L}} \ \bigcap_{\langle \mathsf{x},\ \mathsf{y}\rangle \in \mathbf{D}(\ell)} \mathcal{D}_{\mathrm{diff}}{}^{\ell}\langle \mathsf{x},\ \mathsf{y}\rangle$$

(the more semantics, the less common dependencies)

Static dependency analysis

Potential dependency

- $\alpha^{\mathrm{d}}(\{\boldsymbol{\mathcal{S}}^*[\![\mathsf{s}]\!]\})$ is not computable (Rice theorem)
- We design an over-approximation:

Abstract potential dependency semantics $\widehat{\boldsymbol{\mathcal{S}}}^{\mathrm{diff}}_{\exists}$ *:*

$$\alpha^{\mathrm{d}}(\{\boldsymbol{\mathcal{S}}^{+\infty}[\![\mathsf{s}]\!]\}) \quad \dot{\subseteq} \quad \widehat{\boldsymbol{\mathcal{S}}}^{\mathrm{diff}}_{\exists}[\![\mathsf{s}]\!]$$

- The abstraction in D. E. Denning and P. J. Denning, 1977 is purely syntactic;
- We do a little better by taking the semantics is a simple way.

Calculation design

- $\widehat{\boldsymbol{\mathcal{S}}}^{\mathrm{diff}}_{\exists}[\![\mathsf{s}]\!]$ is designed by calculus (in principle can be checked in Coq as Jourdan, Laporte, Blazy, Leroy, and Pichardie, 2015);
- By structural induction on the program syntax;
- By fixpoint approximation for iteration:

Theorem (fixpoint over-approximation) If $\langle \mathcal{C},\ \sqsubseteq,\ \bot,\ \top,\ \sqcup,\ \sqcap\rangle$ and $\langle \mathcal{A},\ \preccurlyeq,\ 0,\ 1,\ \curlyvee,\ \curlywedge\rangle$ are complete lattices, $\langle \mathcal{C},\ \sqsubseteq\rangle \xrightleftharpoons[\alpha]{\gamma} \langle \mathcal{A},\ \preccurlyeq\rangle$ is a Galois connection, $f \in \mathcal{C} \xrightarrow{\nearrow} \mathcal{C}$ and $\overline{f} \in \mathcal{A} \xrightarrow{\nearrow} \mathcal{A}$ are monotonally increasing and $\alpha \circ f \ \dot{\preccurlyeq}\ \overline{f} \circ \alpha$ (*semi-commutation*) then $\mathsf{lfp}^{\sqsubseteq} f \sqsubseteq \gamma(\mathsf{lfp}^{\preccurlyeq} \overline{f})$.

- Finite domain, no need for widening

Abstract potential dependency semantics of assignment S ::= x = A ;

$$\widehat{\mathcal{S}}^{\mathrm{diff}}_{\exists}\llbracket \mathtt{S}\rrbracket\,\ell \;=\; (\!|\, \ell = \mathsf{at}\llbracket \mathtt{S}\rrbracket \;?\; \{\langle \mathtt{y},\,\mathtt{y}\rangle \mid \mathtt{y}\in\mathbb{V}\}$$
$$\|\; \ell = \mathsf{after}\llbracket \mathtt{S}\rrbracket \;?\; \{\langle \mathtt{y},\,\mathtt{x}\rangle \mid \mathtt{y}\in \widehat{\mathcal{S}}^{\mathrm{diff}}_{\exists}\llbracket \mathtt{A}\rrbracket\} \cup \{\langle \mathtt{y},\,\mathtt{y}\rangle \mid \mathtt{y}\neq\mathtt{x}\}$$
$$\mathbin{⁏}\; \varnothing \,|\!)$$
$$\widehat{\mathcal{S}}^{\mathrm{diff}}_{\exists}\llbracket \mathtt{A}\rrbracket \;\triangleq\; \{\mathtt{y} \mid \exists\rho\in\mathbb{E}\mathrm{v}\,.\,\exists\nu\in\mathbb{V}\,.\,\mathscr{E}\llbracket \mathtt{A}\rrbracket\rho \neq \mathscr{E}\llbracket \mathtt{A}\rrbracket\rho[\mathtt{y}\leftarrow\nu]\}$$

$\widehat{\mathcal{S}}^{\mathrm{diff}}_{\exists}\llbracket 1\rrbracket \triangleq \varnothing$ $\quad\widehat{\mathcal{S}}^{\mathrm{diff}}_{\exists}\llbracket \mathtt{x}\rrbracket \triangleq \{\mathtt{x}\}$ $\quad\widehat{\mathcal{S}}^{\mathrm{diff}}_{\exists}\llbracket \mathtt{A}_1 - \mathtt{A}_2\rrbracket \triangleq \{\mathtt{y}\in \mathbb{vars}\llbracket \mathtt{A}_1\rrbracket \cup \mathbb{vars}\llbracket \mathtt{A}_2\rrbracket \mid \mathtt{A}_1 \neq \mathtt{A}_2\}$

$\widehat{\mathcal{S}}^{\mathrm{diff}}_{\exists}\llbracket \mathtt{A}\rrbracket \subseteq \mathbb{vars}\llbracket \mathtt{A}\rrbracket$

Examples:

- after x = y - y ;, x does not depends on y.
- after x = y ; x = y - x ;, x depends on the initial value of x and y (to be more precise information of values of variables must be kept such as y - x = 0 by symbolic constant analysis)

Proof I

The case $\ell = \mathsf{at}\llbracket \mathtt{S}\rrbracket$ was handled in (44.39). Assume $\ell = \mathsf{after}\llbracket \mathtt{S}\rrbracket$.

$\alpha^{\mathrm{d}}(\{\mathcal{S}^{+\infty}\llbracket \mathtt{S}\rrbracket\})\,\mathsf{after}\llbracket \mathtt{S}\rrbracket$

$=\alpha^{\mathrm{d}}(\{\mathcal{S}^{+}\llbracket \mathtt{S}\rrbracket\})\,\mathsf{after}\llbracket \mathtt{S}\rrbracket$ ⦃def. (7.6) of $\mathcal{S}^{+\infty}\llbracket \mathtt{S}\rrbracket$ since the assignment S has only finite prefix traces⦄

$=\{\langle \mathtt{x}',\,\mathtt{y}\rangle \mid \mathcal{S}^{+}\llbracket \mathtt{S}\rrbracket \in \mathcal{D}_{\mathrm{diff}}(\mathsf{after}\llbracket \mathtt{S}\rrbracket)\langle \mathtt{x}',\,\mathtt{y}\rangle\}$ ⦃def. (44.23) of α^{d} and def. $\subseteq$⦄

$=\{\langle \mathtt{x}',\,\mathtt{y}\rangle \mid \exists\langle\pi_0,\,\pi_1\rangle,\langle\pi_0',\,\pi_1'\rangle \in \mathcal{S}^{+}\llbracket \mathtt{S}\rrbracket\,.\,(\forall \mathtt{z}\in\mathbb{V}\setminus\{\mathtt{x}'\}\,.\,\varrho(\pi_0)\mathtt{z} = \varrho(\pi_0')\mathtt{z}) \wedge \mathsf{diff}(\mathsf{seqval}\llbracket \mathtt{y}\rrbracket(\mathsf{at}\llbracket \mathtt{S}\rrbracket)(\pi_0,\pi_1), \mathsf{seqval}\llbracket \mathtt{y}\rrbracket(\mathsf{at}\llbracket \mathtt{S}\rrbracket)(\pi_0',\pi_1'))\}$ ⦃def. (44.18) of $\mathcal{D}_{\mathrm{diff}}\,\ell\langle \mathtt{x}',\,\mathtt{y}\rangle$⦄

$=\{\langle \mathtt{x}',\,\mathtt{y}\rangle \mid \exists\langle\pi_0,\,\pi_1\rangle,\langle\pi_0',\,\pi_1'\rangle \in \{\langle\pi\mathsf{at}\llbracket \mathtt{S}\rrbracket,\,\mathsf{at}\llbracket \mathtt{S}\rrbracket \xrightarrow{\mathtt{x}=\mathscr{E}\llbracket \mathtt{A}\rrbracket\boldsymbol{\varrho}(\pi\mathsf{at}\llbracket \mathtt{S}\rrbracket)} \mathsf{after}\llbracket \mathtt{S}\rrbracket\rangle \mid \pi\mathsf{at}\llbracket \mathtt{S}\rrbracket \in \mathbb{T}^{+}\}\,.\,(\forall \mathtt{z}\in\mathbb{V}\setminus\{\mathtt{x}'\}\,.\,\varrho(\pi_0)\mathtt{z} = \varrho(\pi_0')\mathtt{z}) \wedge \mathsf{diff}(\mathsf{seqval}\llbracket \mathtt{y}\rrbracket(\mathsf{at}\llbracket \mathtt{S}\rrbracket)(\pi_0,\pi_1), \mathsf{seqval}\llbracket \mathtt{y}\rrbracket(\mathsf{at}\llbracket \mathtt{S}\rrbracket)(\pi_0',\pi_1'))\}$

⦃def. maximal finite trace semantics in Section **6.4** and (6.13)⦄

$=\{\langle \mathtt{x}',\,\mathtt{y}\rangle \mid \exists\langle\pi_0\mathsf{at}\llbracket \mathtt{S}\rrbracket,\,\mathsf{at}\llbracket \mathtt{S}\rrbracket \xrightarrow{\mathtt{x}=\mathscr{E}\llbracket \mathtt{A}\rrbracket\boldsymbol{\varrho}(\pi_0\mathsf{at}\llbracket \mathtt{S}\rrbracket)} \mathsf{after}\llbracket \mathtt{S}\rrbracket\rangle, \langle\pi_0'\mathsf{at}\llbracket \mathtt{S}\rrbracket,\,\mathsf{at}\llbracket \mathtt{S}\rrbracket \xrightarrow{\mathtt{x}=\mathscr{E}\llbracket \mathtt{A}\rrbracket\boldsymbol{\varrho}(\pi_0'\mathsf{at}\llbracket \mathtt{S}\rrbracket)} \mathsf{after}\llbracket \mathtt{S}\rrbracket\rangle\,.\,(\forall \mathtt{z}\in\mathbb{V}\setminus\{\mathtt{x}'\}\,.\,\varrho(\pi_0\mathsf{at}\llbracket \mathtt{S}\rrbracket)\mathtt{z} = \varrho(\pi_0'\mathsf{at}\llbracket \mathtt{S}\rrbracket)\mathtt{z}) \wedge \mathsf{diff}(\mathsf{seqval}\llbracket \mathtt{y}\rrbracket\mathsf{after}\llbracket \mathtt{S}\rrbracket(\pi_0\mathsf{at}\llbracket \mathtt{S}\rrbracket \xrightarrow{\mathtt{x}=\mathscr{E}\llbracket \mathtt{A}\rrbracket\boldsymbol{\varrho}(\pi_0\mathsf{at}\llbracket \mathtt{S}\rrbracket)} \mathsf{after}\llbracket \mathtt{S}\rrbracket,\,\mathsf{after}\llbracket \mathtt{S}\rrbracket), \mathsf{seqval}\llbracket \mathtt{y}\rrbracket\mathsf{after}\llbracket \mathtt{S}\rrbracket(\pi_0'\mathsf{at}\llbracket \mathtt{S}\rrbracket \xrightarrow{\mathtt{x}=\mathscr{E}\llbracket \mathtt{A}\rrbracket\boldsymbol{\varrho}(\pi_0'\mathsf{at}\llbracket \mathtt{S}\rrbracket)} \mathsf{after}\llbracket \mathtt{S}\rrbracket,\,\mathsf{after}\llbracket \mathtt{S}\rrbracket))\}$ ⦃def. $\in$⦄

$=\{\langle \mathtt{x}',\,\mathtt{y}\rangle \mid \exists\langle\pi_0\mathsf{at}\llbracket \mathtt{S}\rrbracket,\,\mathsf{at}\llbracket \mathtt{S}\rrbracket \xrightarrow{\mathtt{x}=\mathscr{E}\llbracket \mathtt{A}\rrbracket\boldsymbol{\varrho}(\pi_0\mathsf{at}\llbracket \mathtt{S}\rrbracket)} \mathsf{after}\llbracket \mathtt{S}\rrbracket\rangle, \langle\pi_0'\mathsf{at}\llbracket \mathtt{S}\rrbracket,\,\mathsf{at}\llbracket \mathtt{S}\rrbracket \xrightarrow{\mathtt{x}=\mathscr{E}\llbracket \mathtt{A}\rrbracket\boldsymbol{\varrho}(\pi_0'\mathsf{at}\llbracket \mathtt{S}\rrbracket)} \mathsf{after}\llbracket \mathtt{S}\rrbracket\rangle\,.\,(\forall \mathtt{z}\in\mathbb{V}\setminus\{\mathtt{x}'\}\,.\,\varrho(\pi_0\mathsf{at}\llbracket \mathtt{S}\rrbracket)\mathtt{z} = \varrho(\pi_0'\mathsf{at}\llbracket \mathtt{S}\rrbracket)\mathtt{z}) \wedge \mathsf{diff}(\varrho(\pi_0\mathsf{at}\llbracket \mathtt{S}\rrbracket)\mathtt{y}\cdot\varrho(\pi_0\mathsf{at}\llbracket \mathtt{S}\rrbracket \xrightarrow{\mathtt{x}=\mathscr{E}\llbracket \mathtt{A}\rrbracket\boldsymbol{\varrho}(\pi_0\mathsf{at}\llbracket \mathtt{S}\rrbracket)} \mathsf{after}\llbracket \mathtt{S}\rrbracket)\mathtt{y},\ \varrho(\pi_0'\mathsf{at}\llbracket \mathtt{S}\rrbracket)\mathtt{y}\cdot\varrho(\pi_0'\mathsf{at}\llbracket \mathtt{S}\rrbracket \xrightarrow{\mathtt{x}=\mathscr{E}\llbracket \mathtt{A}\rrbracket\boldsymbol{\varrho}(\pi_0'\mathsf{at}\llbracket \mathtt{S}\rrbracket)} \mathsf{after}\llbracket \mathtt{S}\rrbracket)\mathtt{y})\}$ ⦃def. (44.15) of $\mathsf{seqval}\llbracket \mathtt{y}\rrbracket$⦄

Proof II

$\subseteq\{\langle \mathtt{x}',\,\mathtt{y}\rangle \mid \exists\langle\pi_0\mathsf{at}\llbracket \mathtt{S}\rrbracket,\,\mathsf{at}\llbracket \mathtt{S}\rrbracket \xrightarrow{\mathtt{x}=\mathscr{E}\llbracket \mathtt{A}\rrbracket\boldsymbol{\varrho}(\pi_0\mathsf{at}\llbracket \mathtt{S}\rrbracket)} \mathsf{after}\llbracket \mathtt{S}\rrbracket\rangle, \langle\pi_0'\mathsf{at}\llbracket \mathtt{S}\rrbracket,\,\mathsf{at}\llbracket \mathtt{S}\rrbracket \xrightarrow{\mathtt{x}=\mathscr{E}\llbracket \mathtt{A}\rrbracket\boldsymbol{\varrho}(\pi_0'\mathsf{at}\llbracket \mathtt{S}\rrbracket)} \mathsf{after}\llbracket \mathtt{S}\rrbracket\rangle\,.\,(\forall \mathtt{z}\in\mathbb{V}\setminus\{\mathtt{x}'\}\,.\,\varrho(\pi_0\mathsf{at}\llbracket \mathtt{S}\rrbracket)\mathtt{z} = \varrho(\pi_0'\mathsf{at}\llbracket \mathtt{S}\rrbracket)\mathtt{z}) \wedge ((\varrho(\pi_0\mathsf{at}\llbracket \mathtt{S}\rrbracket)\mathtt{y} \neq \varrho(\pi_0'\mathsf{at}\llbracket \mathtt{S}\rrbracket)\mathtt{y}\cdot) \vee (\varrho(\pi_0\mathsf{at}\llbracket \mathtt{S}\rrbracket)\mathtt{y} = \varrho(\pi_0'\mathsf{at}\llbracket \mathtt{S}\rrbracket)\mathtt{y}\cdot) \wedge \varrho(\pi_0\mathsf{at}\llbracket \mathtt{S}\rrbracket \xrightarrow{\mathtt{x}=\mathscr{E}\llbracket \mathtt{A}\rrbracket\boldsymbol{\varrho}(\pi_0\mathsf{at}\llbracket \mathtt{S}\rrbracket)} \mathsf{after}\llbracket \mathtt{S}\rrbracket)\mathtt{y} \neq \varrho(\pi_0'\mathsf{at}\llbracket \mathtt{S}\rrbracket \xrightarrow{\mathtt{x}=\mathscr{E}\llbracket \mathtt{A}\rrbracket\boldsymbol{\varrho}(\pi_0'\mathsf{at}\llbracket \mathtt{S}\rrbracket)} \mathsf{after}\llbracket \mathtt{S}\rrbracket)\mathtt{y}))\}$ ⦃(44.17) so that $\mathsf{diff}(a\cdot b,\ c\cdot d)$ if and only if (1) $a\neq c$ or (2) $a = c \wedge b \neq d$.⦄

$\subseteq\{\langle \mathtt{x}',\,\mathtt{y}\rangle \mid \exists\langle\pi_0\mathsf{at}\llbracket \mathtt{S}\rrbracket,\,\mathsf{at}\llbracket \mathtt{S}\rrbracket \xrightarrow{\mathtt{x}=\mathscr{E}\llbracket \mathtt{A}\rrbracket\boldsymbol{\varrho}(\pi_0\mathsf{at}\llbracket \mathtt{S}\rrbracket)} \mathsf{after}\llbracket \mathtt{S}\rrbracket\rangle, \langle\pi_0'\mathsf{at}\llbracket \mathtt{S}\rrbracket,\,\mathsf{at}\llbracket \mathtt{S}\rrbracket \xrightarrow{\mathtt{x}=\mathscr{E}\llbracket \mathtt{A}\rrbracket\boldsymbol{\varrho}(\pi_0'\mathsf{at}\llbracket \mathtt{S}\rrbracket)} \mathsf{after}\llbracket \mathtt{S}\rrbracket\rangle\,.\,(\forall \mathtt{z}\in\mathbb{V}\setminus\{\mathtt{x}'\}\,.\,\varrho(\pi_0\mathsf{at}\llbracket \mathtt{S}\rrbracket)\mathtt{z} = \varrho(\pi_0'\mathsf{at}\llbracket \mathtt{S}\rrbracket)\mathtt{z}) \wedge ((\mathtt{y}=\mathtt{x}') \vee (\mathtt{y}=\mathtt{x} \wedge \mathscr{E}\llbracket \mathtt{A}\rrbracket\varrho(\pi_0\mathsf{at}\llbracket \mathtt{S}\rrbracket) \neq \mathscr{E}\llbracket \mathtt{A}\rrbracket\varrho(\pi_0'\mathsf{at}\llbracket \mathtt{S}\rrbracket))))\}$ ⦃def. (6.3) of ϱ⦄

$\subseteq\{\langle \mathtt{x}',\,\mathtt{y}\rangle \mid ((\mathtt{y}=\mathtt{x}') \vee (\mathtt{y}=\mathtt{x} \wedge \exists\rho,\nu\,.\,\mathscr{E}\llbracket \mathtt{A}\rrbracket\rho \neq \mathscr{E}\llbracket \mathtt{A}\rrbracket\rho[\mathtt{x}'\leftarrow\nu]))\}$

⦃letting $\rho = \varrho(\pi_0\mathsf{at}\llbracket \mathtt{S}\rrbracket)$ and $\nu = \varrho(\pi_0'\mathsf{at}\llbracket \mathtt{S}\rrbracket)(\mathtt{x}')$ so that $\forall \mathtt{z}\in\mathbb{V}\setminus\{\mathtt{x}'\}\,.\,\varrho(\pi_0\mathsf{at}\llbracket \mathtt{S}\rrbracket)\mathtt{z} = \varrho(\pi_0'\mathsf{at}\llbracket \mathtt{S}\rrbracket)\mathtt{z}$ implies that $\varrho(\pi_0'\mathsf{at}\llbracket \mathtt{S}\rrbracket) = \rho[\mathtt{x}'\leftarrow\nu]$⦄

$\subseteq\{\langle \mathtt{x}',\,\mathtt{x}'\rangle \mid \mathtt{x}'\neq\mathtt{x}\} \cup \{\langle \mathtt{x}',\,\mathtt{x}\rangle \mid \exists\rho,\nu\,.\,\mathscr{E}\llbracket \mathtt{A}\rrbracket\rho \neq \mathscr{E}\llbracket \mathtt{A}\rrbracket\rho[\mathtt{x}'\leftarrow\nu]\}$ ⦃case analysis⦄

$=\{\langle \mathtt{x}',\,\mathtt{x}'\rangle \mid \mathtt{x}'\neq\mathtt{x}\} \cup \{\langle \mathtt{x}',\,\mathtt{x}\rangle \mid \mathtt{x}'\in\widehat{\mathcal{S}}^{\mathrm{diff}}_{\exists}\llbracket \mathtt{A}\rrbracket\}$

⦃by defining the functional dependency of an expression A as $\widehat{\mathcal{S}}^{\mathrm{diff}}_{\exists}\llbracket \mathtt{A}\rrbracket \triangleq \{\mathtt{x}' \mid \exists\rho,\nu\,.\,\mathscr{E}\llbracket \mathtt{A}\rrbracket\rho \neq \mathscr{E}\llbracket \mathtt{A}\rrbracket\rho[\mathtt{x}'\leftarrow\nu]\}$⦄ □

□

Abstract potential dependency semantics of the iteration S ::= while ℓ (B) S$_b$

$$\widehat{\mathcal{S}}^{\mathrm{diff}}_{\exists}\llbracket \mathtt{S}\rrbracket\,\ell' \;=\; (\mathsf{lfp}^{\dot\subseteq}\,\mathcal{F}^{\mathrm{d}}\llbracket \mathtt{while}\ \ell\ (\mathtt{B})\ \mathtt{S}_b\rrbracket)\,\ell'$$
$$\mathcal{F}^{\mathrm{d}}\llbracket \mathtt{while}\ \ell\ (\mathtt{B})\ \mathtt{S}_b\rrbracket\,X\,\ell' \;=$$
$$(\!|\, \ell' = \ell \;?\; \mathbb{1}_{\mathbb{V}} \cup X(\ell) \cup (X(\ell) \mathbin{⁏} \widehat{\mathcal{S}}^{\mathrm{diff}}_{\exists}\llbracket \mathtt{S}_b\rrbracket\,\ell)$$
$$\|\; \ell' \in \mathsf{in}\llbracket \mathtt{S}\rrbracket \cup (\!|\, \mathsf{escape}\llbracket \mathtt{S}\rrbracket \;?\; \{\mathsf{break\text{-}to}\llbracket \mathtt{S}\rrbracket\} \mathbin{⁏} \varnothing \,|\!) \;?\; X(\ell') \cup (X(\ell) \mathbin{⁏} \widehat{\mathcal{S}}^{\mathrm{diff}}_{\exists}\llbracket \mathtt{S}_b\rrbracket\,\ell')$$
$$\|\; \ell' = \mathsf{after}\llbracket \mathtt{S}\rrbracket \;?\; X(\ell) \cup \{\langle \mathtt{x}',\,\mathtt{y}\rangle \mid \mathtt{x}' \in \mathbb{vars}\llbracket \mathtt{B}\rrbracket \wedge \mathtt{y}\in\mathsf{mod}\llbracket \mathtt{S}_b\rrbracket\}$$
$$\mathbin{⁏}\; \varnothing \,|\!)$$

- Can be refined by taking test determinacy into account (*e.g.* after test x == 1, x can only have value 1 so nothing can depend on x afterwards).

No structural compositionality

In the following statement, x and y at ℓ_1 depend on x at ℓ_0.

```
                    /* x = x0, y = y0 */
ℓ0 y = x ;
ℓ1                  /* x = x0, y = x0 */
```

In the following statement, x and y at ℓ_2 depend on x at ℓ_1.

```
                    /* x = x0, y = y0 */
ℓ1 y = y-x ;
ℓ2                  /* x = x0, y = y0 - x0 */
```

In the sequential composition of the two statements

```
                    /* x = x0, y = y0 */
ℓ0 y = x ;          /* x = x0, y = x0 */
ℓ1 y = y-x ;        /* x = x0, y = 0 */
ℓ2
```

y at ℓ_2 depends on x at ℓ_1 which depends on x at ℓ_0 so, by composition, y at ℓ_2 depends on x at ℓ_0.

However, $y = 0$ at ℓ_2 so y at ℓ_2 does not depend on x at ℓ_0.

Improving precision

- To improve prcision one must take values of variables into account;
- Reduced product with a reachability analsyis (*e.g.* Cortesi, Ferrara, Halder, and Zanioli, 2018; Zanioli and Cortesi, 2011)

Conclusion

Dependency analysis is an abstract interrpetation

- No need for a generalized theory (as proposed by Assaf, Naumann, Signoles, Totel, and Tronel, 2017; Urban and Müller, 2018)
- This includes further abstractions, dye analysis, taint analysis, *etc*.
- Many possible variants (*e.g.* by changing diff to = we get timing channel dependency).
- Data dependency analysis to detect parallelism in sequential codes Padua and Wolfe, 1986 is also an abstract interpretation Tzolovski, 1997, Tzolovski, 2002, Ch. 5.

Bibliographie

References I

Assaf, Mounir, David A. Naumann, Julien Signoles, Eric Totel, and Frédéric Tronel (2017). "Hypercollecting semantics and its application to static analysis of information flow". In: *POPL*. ACM, pp. 874–887 (53, 3).

Barthe, Gilles, Benjamin Grégoire, and Vincent Laporte (2017). "Provably secure compilation of side-channel countermeasures". *IACR Cryptology ePrint Archive* 2017, p. 1233 (53, 33).

Cheney, James, Amal Ahmed, and Umut A. Acar (2011). "Provenance as dependency analysis". *Mathematical Structures in Computer Science* 21.6, pp. 1301–1337 (3, 51).

Cortesi, Agostino, Pietro Ferrara, Raju Halder, and Matteo Zanioli (2018). "Combining Symbolic and Numerical Domains for Information Leakage Analysis". *Trans. Computational Science* 31, pp. 98–135 (53, 30).

Cousot, Patrick and Radhia Cousot (2009). "Bi-inductive structural semantics". *Inf. Comput.* 207.2, pp. 258–283 (5, 11, 3, 8).

References II

Denning, Dorothy E. and Peter J. Denning (1977). "Certification of Programs for Secure Information Flow". *Commun. ACM* 20.7, pp. 504–513 (1, 3–5, 7, 11, 13, 52, 23).

Giacobazzi, Roberto and Isabella Mastroeni (2018). "Abstract Non-Interference: A Unifying Framework for Weakening Information-flow". *ACM Trans. Priv. Secur.* 21.2, 9:1–9:31 (53, 33).

Goguen, Joseph A. and José Meseguer (1982). "Security Policies and Security Models". In: *IEEE Symposium on Security and Privacy*. IEEE Computer Society, pp. 11–20 (1, 3, 51, 52).

– (1984). "Unwinding and Inference Control". In: *IEEE Symposium on Security and Privacy*. IEEE Computer Society, pp. 75–87 (1, 3, 51, 52).

Jourdan, Jacques-Henri, Vincent Laporte, Sandrine Blazy, Xavier Leroy, and David Pichardie (2015). "A Formally-Verified C Static Analyzer". In: *POPL*. ACM, pp. 247–259 (18, 17, 13, 16, 5, 24).

References III

Lampson, Butler W. (1973). "A Note on the Confinement Problem". *Commun. ACM* 16.10, pp. 613–615 (5).

Mulder, Elke De, Thomas Eisenbarth, and Patrick Schaumont (2018). "Identifying and Eliminating Side-Channel Leaks in Programmable Systems". *IEEE Design & Test* 35.1, pp. 74–89 (5).

Padua, David A. and Michael Wolfe (1986). "Advanced Compiler Optimizations for Supercomputers". *Commun. ACM* 29.12, pp. 1184–1201 (53, 32).

Russo, Alejandro, John Hughes, David A. Naumann, and Andrei Sabelfeld (2006). "Closing Internal Timing Channels by Transformation". In: *ASIAN*. Vol. 4435. Lecture Notes in Computer Science. Springer, pp. 120–135 (5).

Sabelfeld, Andrei and Andrew C. Myers (2003). "Language-based information-flow security". *IEEE Journal on Selected Areas in Communications* 21.1, pp. 5–19 (5).

Tzolovski, Stanislav (1997). "Data Dependence as Abstract Interpretations". In: *SAS*. Vol. 1302. Lecture Notes in Computer Science. Springer, p. 366 (53, 32).

References IV

Tzolovski, Stanislav (15 June 2002). "Raffinement d'analyses par interprétation abstraite". Thèse de doctorat. Palaiseau, France: École polytechnique (53, 32).

Urban, Caterina and Peter Müller (2018). "An Abstract Interpretation Framework for Input Data Usage". In: *ESOP*. Vol. 10801. Lecture Notes in Computer Science. Springer, pp. 683–710 (21, 3, 53).

Zanioli, Matteo and Agostino Cortesi (2011). "Information Leakage Analysis by Abstract Interpretation". In: *SOFSEM*. Vol. 6543. Lecture Notes in Computer Science. Springer, pp. 545–557 (53, 30).

The End, Thank you
Happy sixties Mooly!