ICTAC 2019

October 31, 2019

Hammamet, Tunisia

A Tutorial on Abstract Interpretation

Patrick Cousot

New York University, Courant Institute of Mathematics, Computer Science

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

Part 1

October 31, 2019, 09:00—10:30

Introduction

Static analysis

- A static analyzer
  - inputs the source code of a program in a given programming language
  - always terminates
  - automatically output sound information valid for all possible program executions (*e.g*. runtime errors, data races, *etc*.)

(and this without running the program)

How to design a static analyzer by abstract interpretation

- Define the syntax & semantics of the language
- Define the semantic properties to be analyzed
- Define an abstraction of this semantic properties into an abstract domain (machine representable subset of the semantic properties)
- Design the static analyzer by calculational design of the abstraction of the semantics

How to design a static analyzer by abstract interpretation

- Define the syntax & semantics of the language
- Define the semantic properties to be analyzed
- Define an abstraction of this semantic properties into an abstract domain (machine representable subset of the semantic properties)
- Design the static analyzer by calculational design of the abstraction of the semantics

- This will be illustrated in November 2, 2019 session 9:00—10:30 of ICTAC by the design of a regular model checker
- A this tutorial, we introduce the basic notions of abstract interpretation

Basic notions of abstract interpretation

- structural definitions and structural proofs, program semantics
- property and collecting semantics
- abstraction & Galois connection
- abstract domain
- abstract interpreter

- trace semantics
- fixpoints
- fixpoint abstraction
- fixpoint extrapolation (widening) and interpolation (narrowing)

- a few simple examples of static analyzes

Structural definition and proof, Program semantics

Syntax and semantics of programs

- Syntax: how to write a program (say that compiles correctly)
- Example: $\mathtt{x}, \mathtt{y}, \ldots \in \mathbb{V}$ variables ($\mathbb{V}$ not empty)

  $\mathtt{A} \in \mathbb{A} ::= \mathtt{1} \mid \mathtt{x} \mid \mathtt{A}_1 \mathtt{-} \mathtt{A}_2$ arithmetic expressions[1]

- Semantics: a formal definition of what the program computes

[1] assumed to be left associative that is `1-1-1` is `((1-1)-1)`

Structural definition and proofs

- To define the semantics of programs, we use structural definitions *i.e.* by induction on the program syntax
- Example: $\mathtt{x}, \mathtt{y}, \ldots \in \mathbb{V}$ variables ($\mathbb{V}$ not empty)

  $\mathtt{A} \in \mathbb{A} ::= \mathtt{1} \mid \mathtt{x} \mid \mathtt{A}_1 - \mathtt{A}_2$ arithmetic expressions

- A structural definition of $f \in \mathbb{A} \to S$ where S is a set has the form
  - $f(\mathtt{1})$ and $f(\mathtt{x})$ are defined to be constants (so $f(\mathtt{1}) \triangleq c_{\mathtt{1}}$ and $f(\mathtt{x}) \triangleq c_{\mathtt{x}}$ where $c_{\mathtt{1}}, c_{\mathtt{x}} \in S$);
  - $f(\mathtt{A}_1 - \mathtt{A}_2)$ is a function of $f(\mathtt{A}_1)$ and $f(\mathtt{A}_2)$ (so $f(\mathtt{A}_1 - \mathtt{A}_2) \triangleq F_{-}(f(\mathtt{A}_1), f(\mathtt{A}_2))$ where $F_{-} \in S \times S \to S$).

Environment

- What is the value of expression x?

  0 if x has value 0, 1 if x has value 1, -1 if x has value -1, *etc*.
- An environment formalizes "has value" to avoid considering infinitely many cases
- An environment $\rho \in \mathbb{E}\mathbb{v} \triangleq \mathbb{V} \to \mathbb{Z}$ maps variables $\mathtt{x} \in \mathbb{V}$ to their integer value $\rho(\mathtt{x}) \in \mathbb{Z}$,

Structural definition of the semantics of arithmetic expressions

- The value $\mathscr{A}[\![\mathtt{A}]\!]$ of an arithmetic expression $\mathtt{A} \in \mathbb{A}$ is structurally defined as follows.

$$
\begin{aligned}
\mathscr{A}[\![\mathtt{1}]\!] &\triangleq \boldsymbol{\lambda}\,\rho \centerdot 1 \\
\mathscr{A}[\![\mathtt{x}]\!] &\triangleq \boldsymbol{\lambda}\,\rho \centerdot \rho(\mathtt{x}) \\
\mathscr{A}[\![\mathtt{A}_1 \mathbin{\mathtt{-}} \mathtt{A}_2]\!] &\triangleq \boldsymbol{\lambda}\,\rho \centerdot \mathscr{A}[\![\mathtt{A}_1]\!]\rho - \mathscr{A}[\![\mathtt{A}_2]\!]\rho
\end{aligned}
\tag{3.4}
$$

- $\mathtt{1}$, $\mathtt{x}$, $\mathtt{-}$, and $\mathtt{A}$ are syntactic objects *e.g.* strings of characters.
- 1, ρ, $-$ are (already defined) mathematical objects.
- $\boldsymbol{\lambda}\,x \centerdot f(x)$ is the anonymous function such that $(\boldsymbol{\lambda}\,x \centerdot f(x))\,e = f(e)$.

Proofs by structural induction

- To prove a property of $f \in \mathbb{A} \to \mathcal{S}$ defined by structural induction
  - Prove that the property holds for $f(\mathtt{1})$ and $f(\mathtt{x})$
  - Assuming that the property holds for $f(\mathtt{A}_1)$ and $f(\mathtt{A}_2)$, prove that the property holds for $f(\mathtt{A}_1 - \mathtt{A}_2)$
  - Conclude that $\forall \mathtt{A} \in \mathbb{A} \;.\; f(\mathtt{A})$ has the property.

Proofs by structural induction

- To prove a property of $f \in \mathbb{A} \to \mathbb{S}$ defined by structural induction
  - Prove that the property holds for $f(1)$ and $f(\mathtt{x})$
  - Assuming that the property holds for $f(\mathtt{A}_1)$ and $f(\mathtt{A}_2)$, prove that the property holds for $f(\mathtt{A}_1 - \mathtt{A}_2)$
  - Conclude that $\forall \mathtt{A} \in \mathbb{A} \,.\, f(\mathtt{A})$ has the property.

- Example: prove that $\forall \mathtt{A} \in \mathbb{A} \,.\, \mathcal{A}[\![\mathtt{A}]\!] \in \mathbb{E}\mathbb{v} \to \mathbb{Z}$ where $\mathbb{E}\mathbb{v} \triangleq \mathbb{V} \to \mathbb{Z}$

$$
\begin{aligned}
\mathcal{A}[\![1]\!] &\triangleq \lambda \rho \bullet 1 \\
\mathcal{A}[\![\mathtt{x}]\!] &\triangleq \lambda \rho \bullet \rho(\mathtt{x}) \\
\mathcal{A}[\![\mathtt{A}_1 - \mathtt{A}_2]\!] &\triangleq \lambda \rho \bullet \mathcal{A}[\![\mathtt{A}_1]\!]\rho - \mathcal{A}[\![\mathtt{A}_2]\!]\rho
\end{aligned}
\tag{3.4}
$$

Properties and collecting semantics

Properties

- In computer science properties are often defined using logics[2]
- We use set theory instead
- We define properties as sets (of individuals with this property)
- Examples
  - to be even: $\{2z \mid z \in \mathbb{Z}\}$
  - 0 is even: $0 \in \{2z \mid z \in \mathbb{Z}\}$
  - 1 is not even: $1 \notin \{2z \mid z \in \mathbb{Z}\}$
  - the multiples of 4 are even $\{4z \mid z \in \mathbb{Z}\} \subseteq \{2z \mid z \in \mathbb{Z}\}$ ($\subseteq$ is implication)
  - To be positive or negative $\{z \in \mathbb{Z} \mid z > 0\} \cup \{z \in \mathbb{Z} \mid z < 0\}$ ($\cup$ is disjunction)
  - To be positive and negative $\{z \in \mathbb{Z} \mid z > 0\} \cap \{z \in \mathbb{Z} \mid z < 0\} = \emptyset$ ($\cap$ is conjunction, $\emptyset$ is false)

[2] which have there limitations *e.g.* one cannot define the reflexive transitive closure in first-order logic

Properties

- In computer science properties are often defined using logics[2]
- We use set theory instead
- We define properties as sets (of individuals with this property)
- Examples
  - to be even: $\{2z \mid z \in \mathbb{Z}\}$
  - 0 is even: $0 \in \{2z \mid z \in \mathbb{Z}\}$
  - 1 is not even: $1 \notin \{2z \mid z \in \mathbb{Z}\}$
  - the multiples of 4 are even $\{4z \mid z \in \mathbb{Z}\} \subseteq \{2z \mid z \in \mathbb{Z}\}$ ($\subseteq$ is implication)
  - To be positive or negative $\{z \in \mathbb{Z} \mid z > 0\} \cup \{z \in \mathbb{Z} \mid z < 0\}$ ($\cup$ is disjunction)
  - To be positive and negative $\{z \in \mathbb{Z} \mid z > 0\} \cap \{z \in \mathbb{Z} \mid z < 0\} = \emptyset$ ($\cap$ is conjunction, $\emptyset$ is false)
- If $\mathbb{U}$ is a universe (a set of individuals/things you are interested in), the properties of the individuals of the universe belong to $\wp(\mathbb{U}) \triangleq \{P \mid P \subseteq \mathbb{U}\}$

[2] which have there limitations *e.g.* one cannot define the reflexive transitive closure in first-order logic

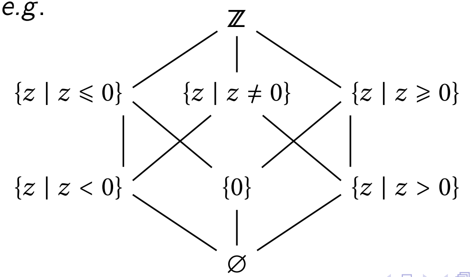
Weaker/stronger properties

- $P \subseteq Q$ is implication
- Example: "to be greater that 42 implies to be positive" is $\{z \in \mathbb{Z} \mid z > 42\} \subseteq \{z \in \mathbb{Z} \mid z \geqslant 0\}$
- P is a stronger/more precise property than Q (less elements satisfy it)
- Q is a weaker/less precise property than P (more elements satisfy it)
- $\varnothing$ (false) is the strongest property of elements of the universe $\mathbb{U}$
- $\mathbb{U}$ (true) is the weakest property
- $\{x\}$ strongest property of element $x \in \mathbb{U}$

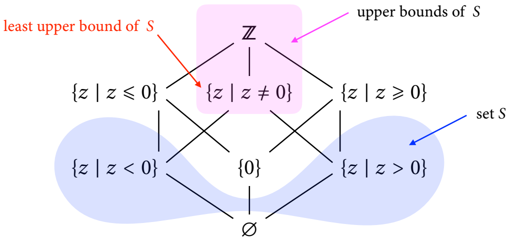
Complete lattice of properties $\langle \wp(\mathbb{U}),\ \subseteq,\ \emptyset,\ \mathbb{U},\ \cup,\ \cap \rangle$

- $\subseteq$ is a partial order (reflexive, antisymmetric, and transitive)
- $\emptyset$ is the infimum (smallest element)
- $\mathbb{U}$ is the infimum (largest element)
- Any set of properties $X \in \wp(\wp(\mathbb{U}))$ has a least upper bound $\bigcup X$
- Any set of properties $X \in \wp(\wp(\mathbb{U}))$ has a greatest lowe bound $\bigcap X$

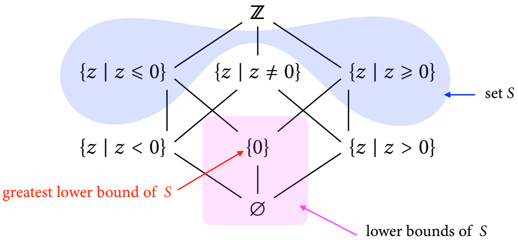
Generalizes to $\langle L,\ \sqsubseteq,\ \bot,\ \top,\ \sqcup,\ \sqcap \rangle$ *e.g.*



Least upper bound



Greatest upper bound



Program properties

- By our definition, a program property is a set of programs
- Example: "to return 1" is

$$
\begin{aligned}
& \{\mathsf{A} \in \mathbb{A} \mid \forall \rho \in \mathbb{E}\mathrm{v} \,.\, \mathcal{A}[\![\mathsf{A}]\!]\rho = 1\} \\
= \; & \{\mathbf{1}, (\mathsf{x} - \mathsf{x}) - ((\mathbf{1} - \mathbf{1}) - \mathbf{1}), \ldots\} \\
\mathbf{1} \; \in \; & \{\mathsf{A} \in \mathbb{A} \mid \forall \rho \in \mathbb{E}\mathrm{v} \,.\, \mathcal{A}[\![\mathsf{A}]\!]\rho = 1\}
\end{aligned}
$$

- We are interested in semantic properties: a set of possible semantics of programs
- Example: "to return 1" is

$$
\begin{aligned}
& \{f \in \mathbb{E}\mathrm{v} \to \mathbb{Z} \mid \forall \rho \in \mathbb{E}\mathrm{v} \,.\, f(\rho) = 1\} \\
\mathcal{A}[\![\mathbf{1}]\!] \; \in \; & \{f \in \mathbb{E}\mathrm{v} \to \mathbb{Z} \mid \forall \rho \in \mathbb{E}\mathrm{v} \,.\, f(\rho) = 1\}
\end{aligned}
$$

Collecting semantics

- The collecting semantics is the strongest property of a program semantics
- $\mathcal{S}^{\mathbb{C}}[\![\mathtt{A}]\!] \triangleq \{\mathcal{A}[\![\mathtt{A}]\!]\}$
- Program $\mathtt{A}$ has property P

  iff $\mathcal{A}[\![\mathtt{A}]\!] \in P$

  iff $\mathcal{S}^{\mathbb{C}}[\![\mathtt{A}]\!] \subseteq P$

  so we can get rid of $\in$ in favor of $\subseteq$ and reason in the complete lattice of properties!

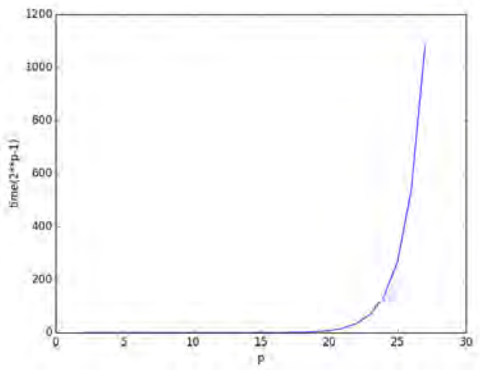
$$\begin{array}{rcl}
\mathcal{S}^{\mathbb{C}}[\![\mathtt{1}]\!] & = & \{\boldsymbol{\lambda}\,\rho \centerdot 1\} \\
\mathcal{S}^{\mathbb{C}}[\![\mathtt{x}]\!] & = & \{\boldsymbol{\lambda}\,\rho \centerdot \rho(\mathtt{x})\} \\
\mathcal{S}^{\mathbb{C}}[\![\mathtt{A}_1 - \mathtt{A}_2]\!] & = & \{\boldsymbol{\lambda}\,\rho \centerdot f_1(\rho) - f_2(\rho) \mid f_1 \in \mathcal{S}^{\mathbb{C}}[\![\mathtt{A}_1]\!] \wedge f_2 \in \mathcal{S}^{\mathbb{C}}[\![\mathtt{A}_2]\!]\}
\end{array}$$

(note: same ρ)

Abstraction & Galois connections

Proving and analyzing programs

- It is not possible to prove program properties by enumerating all possible cases
- *e.g.* Model-checking does not scale
- *e.g.* Prove by enumeration of all cases that $x - x = 0$ where x has integer values encoded on $p = 1,2,3,\ldots,64$ bits



Fully mechanized solutions

- Consider programs with a small number of small executions (model-checking[3])
- Ask for human help (deductive methods using user-provided information and help for theorem-provers or SMT solvers)
- Use sound approximations (static analysis)
  $\longrightarrow$ abstraction formalized by abstract interpretation
- or prove nothing as in unsound static analysis

[3] *e.g.* the model-checker of Scade will almost certainly fail when numerical computations over more than 8 bits have to be taken into account.

Abstraction and abstract properties

- Do not consider all possible properties of the semantics (*e.g.* all properties of the semantics of an arithmetic expression)
- Abstraction consists in considering a subset pertinent to what you want to prove (*e.g.* the sign of an arithmetic expression knowing the sign of its arguments)
- Abstract properties are a computer representation of these properties of interest

$$\mathbb{P}^{\pm} = \begin{tikzcd}[arrows=dash]
 & \top_{\pm} \arrow[dl] \arrow[d] \arrow[dr] & \\
\leqslant 0 \arrow[d] \arrow[dr] & \neq 0 \arrow[dl] \arrow[dr] & \geqslant 0 \arrow[dl] \arrow[d] \\
< 0 \arrow[dr] & = 0 \arrow[d] & > 0 \arrow[dl] \\
 & \bot_{\pm} &
\end{tikzcd}$$

Abstract domain

- Abstract domain $=$ abstract properties $+$ operations on abstract properties
- Lattice operations $\sqsubseteq_\pm, \bot_\pm, \top_\pm, \sqcup_\pm, \sqcap_\pm$
- Example of operation on sign abstract properties[4]

$s_1 -_\pm s_2$	s_2: $\bot_\pm$	<0	$=0$	>0	$\leqslant 0$	$\neq 0$	$\geqslant 0$	$\top_\pm$
s_1: $\bot_\pm$	$\bot_\pm$	$\bot_\pm$	$\bot_\pm$	$\bot_\pm$	$\bot_\pm$	$\bot_\pm$	$\bot_\pm$	$\bot_\pm$
<0	$\bot_\pm$	$\top_\pm$	<0	<0	$\top_\pm$	$\top_\pm$	<0	$\top_\pm$
$=0$	$\bot_\pm$	>0	$=0$	<0	$\geqslant 0$	$\neq 0$	$\leqslant 0$	$\top_\pm$
>0	$\bot_\pm$	>0	>0	$\top_\pm$	>0	$\top_\pm$	$\top_\pm$	$\top_\pm$
$\leqslant 0$	$\bot_\pm$	$\top_\pm$	$\leqslant 0$	<0	$\top_\pm$	$\top_\pm$	$\leqslant 0$	$\top_\pm$
$\neq 0$	$\bot_\pm$	$\top_\pm$	$\neq 0$	$\top_\pm$	$\top_\pm$	$\top_\pm$	$\top_\pm$	$\top_\pm$
$\geqslant 0$	$\bot_\pm$	>0	$\geqslant 0$	$\top_\pm$	$\geqslant 0$	$\top_\pm$	$\top_\pm$	$\top_\pm$
$\top_\pm$	$\bot_\pm$	$\top_\pm$	$\top_\pm$	$\top_\pm$	$\top_\pm$	$\top_\pm$	$\top_\pm$	$\top_\pm$

[4] Observe the loss of information

Correspondance between abstract and concrete properties

- Concretization function γ
- Example, sign concretization

$$
\begin{array}{rclrcl}
\gamma_{\pm}(\bot_{\pm}) & \triangleq & \emptyset & \gamma_{\pm}(\leqslant 0) & \triangleq & \{z \in \mathbb{Z} \mid z \leqslant 0\} \\
\gamma_{\pm}(<0) & \triangleq & \{z \in \mathbb{Z} \mid z < 0\} & \gamma_{\pm}(\neq 0) & \triangleq & \{z \in \mathbb{Z} \mid z \neq 0\} \\
\gamma_{\pm}(=0) & \triangleq & \{0\} & \gamma_{\pm}(\geqslant 0) & \triangleq & \{z \in \mathbb{Z} \mid z \geqslant 0\} \\
\gamma_{\pm}(>0) & \triangleq & \{z \in \mathbb{Z} \mid z > 0\} & \gamma_{\pm}(\top_{\pm}) & \triangleq & \mathbb{Z}
\end{array} \tag{3.23}
$$

Correspondance between concrete and abstract properties

- Abstraction function α
- Example, sign abstraction

$$
\begin{aligned}
\alpha_{\pm}(P) \triangleq \llparenthesis\, & P \subseteq \emptyset \; ? \; \bot_{\pm} \\
\| \; & P \subseteq \{z \mid z < 0\} \; ? \; {<}0 \\
\| \; & P \subseteq \{0\} \; ? \; {=}0 \\
\| \; & P \subseteq \{z \mid z > 0\} \; ? \; {>}0 \\
\| \; & P \subseteq \{z \mid z \leqslant 0\} \; ? \; {\leqslant}0 \\
\| \; & P \subseteq \{z \mid z \neq 0\} \; ? \; {\neq}0 \\
\| \; & P \subseteq \{z \mid z \geqslant 0\} \; ? \; {\geqslant}0 \\
\text{\textdiv} \; & \top_{\pm} \,\rrparenthesis
\end{aligned}
\tag{3.30}
$$

Best approximation

- $\alpha_\pm(P)$ is the best over-approximation of $P \in \wp(\mathbb{Z})$ in $\mathbb{P}^\pm$ since
  - $P \subseteq \gamma_\pm(\alpha_\pm(P))$ *i.e.* $\alpha_\pm(P)$ is an over-approximation/sound abstraction of P;

    e.g. $\gamma_\pm(\alpha_\pm(\{z \in \mathbb{Z} \mid z \geqslant 42\})) = \gamma_\pm(> 0) = \{z \in \mathbb{Z} \mid z > 0\}$

  - if $\overline{P} \in \mathbb{P}^\pm$ and $P \subseteq \gamma_\pm(\overline{P})$ then $\alpha_\pm(P) \sqsubseteq_\pm \overline{P}$
    i.e. $\alpha_\pm(P)$ is more precise than any other over-approximation/sound abstraction of P.

    e.g. $\{z \in \mathbb{Z} \mid z \geqslant 42\} \subseteq \gamma_\pm(>0), \gamma_\pm(\geqslant 0), \gamma_\pm(\top_\pm)$ and
    $\alpha_\pm(\{z \in \mathbb{Z} \mid z \geqslant 42\}) = {>0} \sqsubseteq_\pm {>0} \sqsubset_\pm {\geqslant 0} \sqsubset_\pm \top_\pm$

$$\mathbb{P}^{\pm} = \begin{tikzcd}[arrows=dash] & \top_\pm \arrow[dl] \arrow[d] \arrow[dr] & \\ \leqslant 0 \arrow[d] \arrow[dr] & \neq 0 \arrow[dl] \arrow[dr] & \geqslant 0 \arrow[dl] \arrow[d] \\ <0 \arrow[dr] & =0 \arrow[d] & >0 \arrow[dl] \\ & \bot_\pm & \end{tikzcd}$$

Galois connection

- The pair $\langle \alpha_\pm, \gamma_\pm \rangle$ is a Galois connection, *i.e.*

$$\forall P \in \wp(\mathbb{Z}) \;.\; \forall \overline{P} \in \mathbb{P}^\pm \;.\; \alpha_\pm(P) \sqsubseteq_\pm Q \quad \text{iff} \quad P \subseteq \gamma_\pm(Q)$$

  - if $\alpha_\pm(P) \sqsubseteq_\pm Q$ then Q is a sound over-approximation of P (including $Q = \alpha_\pm(P)$)
  - if Q is a sound over-approximation of P (*i.e.* $P \subseteq \gamma_\pm(Q)$) then $\alpha_\pm(P)$ is better/more precise than Q (so $\alpha_\pm(P)$ is the best sound abstraction of P)

- Notation: $\langle \wp(\mathbb{Z}), \subseteq \rangle \xrightleftharpoons[\alpha_\pm]{\gamma_\pm} \langle \mathbb{P}^\pm, \sqsubseteq_\pm \rangle$

Properties of Galois connection $\langle \wp(\mathbb{Z}), \subseteq \rangle \xleftarrow[\alpha_\pm]{\gamma_\pm} \hspace{-1em}\rightarrow \langle \mathbb{P}^\pm, \sqsubseteq \rangle$

- Essential properties
  - $\alpha_\pm$ and $\gamma_\pm$ are increasing
  - $\forall P \in \wp(\mathbb{Z}) \;.\; P \subseteq \gamma_\pm(\alpha_\pm(P))$
  - $\forall Q \in \mathbb{P}^\pm \;.\; \alpha_\pm(\gamma_\pm(Q)) \sqsubseteq Q$
  - $\alpha_\pm$ preserves least upper bounds, $\gamma_\pm$ preserves greatest lower bounds
  - $\forall Q \in \mathbb{P}^\pm \;.\; \alpha_\pm(\gamma_\pm(Q)) = Q$ iff $\alpha_\pm$ is surjective iff $\gamma_\pm$ is injective
  - One function uniquely determines the other (for the given orders)

Abstracting properties of functions

- Abstracting properties of environments

$$\dot{\alpha}_{\pm}(P) \triangleq \boldsymbol{\lambda}\, \mathsf{x} \centerdot \alpha_{\pm}(\{\rho(\mathsf{x}) \mid \rho \in P\}) \tag{3.33}$$

$$\langle \wp(\mathbb{V} \to \mathbb{Z}), \subseteq \rangle \xrightleftharpoons[\dot{\alpha}_{\pm}]{\dot{\gamma}_{\pm}} \langle \mathbb{V} \to \mathbb{P}^{\pm}, \dot{\sqsubseteq}_{\pm} \rangle$$ [5]

- Abstracting properties of expression semantics

$$\ddot{\alpha}_{\pm}(P) \triangleq \boldsymbol{\lambda}\, \overset{\pm}{\rho} \centerdot \alpha_{\pm}(\{\mathcal{S}(\rho) \mid \mathcal{S} \in P \wedge \rho \in \dot{\gamma}_{\pm}(\overset{\pm}{\rho})\}) \tag{3.34}$$

$$\langle \wp((\mathbb{V} \to \mathbb{Z}) \to \mathbb{Z}), \subseteq \rangle \xrightleftharpoons[\ddot{\alpha}_{\pm}]{\ddot{\gamma}_{\pm}} \langle ((\mathbb{V} \to \mathbb{P}^{\pm}) \to \mathbb{P}^{\pm}), \ddot{\sqsubseteq}_{\pm} \rangle$$

[5] pointwise ordering: $f \mathrel{\dot{\sqsubseteq}} g$ iff $\forall x \,.\, f(x) \sqsubseteq g(x)$, $F \mathrel{\ddot{\sqsubseteq}} G$ iff $\forall f \,.\, \forall x \,.\, F(f)x \sqsubseteq G(f)x$

Sign analysis

Sign analysis

- Sign analysis $\mathcal{S}^{\pm}[\![\mathsf{A}]\!]$ is the abstraction of the collecting semantics $\mathcal{S}^{\mathbb{C}}[\![\mathsf{A}]\!]$ of arithmetic expressions A

$$\ddot{\alpha}_{\pm}(\mathcal{S}^{\mathbb{C}}[\![\mathsf{A}]\!]) \stackrel{..}{\sqsubseteq}_{\pm} \mathcal{S}^{\pm}[\![\mathsf{A}]\!]$$

- Sound approximation (can be $\stackrel{..}{\sqsubset}_{\pm}$)
- $\mathcal{S}^{\pm}[\![\mathsf{A}]\!]$ can be formally derived form the definition of $\mathcal{S}^{\mathbb{C}}[\![\mathsf{A}]\!]$ by calculus

Calculational design of the sign analysis

Sign analysis

- By calculus (to be shown after that slide), we get the structural sign semantics $\mathcal{S}^{\pm}[\![\mathtt{A}]\!] \in (\mathbb{V} \to \mathbb{P}^{\pm}) \to \mathbb{P}^{\pm}$ defined as follows

$$\begin{aligned}
\mathcal{S}^{\pm}[\![\mathtt{1}]\!] &= \boldsymbol{\lambda}\, \dot{\rho}^{\pm} \centerdot >0 \\
\mathcal{S}^{\pm}[\![\mathtt{x}]\!] &= \boldsymbol{\lambda}\, \dot{\rho}^{\pm} \centerdot \dot{\rho}^{\pm}(\mathtt{x}) \\
\mathcal{S}^{\pm}[\![\mathtt{A}_1 - \mathtt{A}_2]\!] &= \boldsymbol{\lambda}\, \dot{\rho}^{\pm} \centerdot (\mathcal{S}^{\pm}[\![\mathtt{A}_1]\!]\dot{\rho}^{\pm}) -_{\pm} (\mathcal{S}^{\pm}[\![\mathtt{A}_2]\!]\dot{\rho}^{\pm})
\end{aligned}$$

- Strategy
  - by structural induction
  - develop and simplify the definitions
  - make approximations to get rid of concrete semantic computations

Constants

- Assume $\dot{\gamma}_\pm(\overset{\pm}{\rho}) \neq \emptyset$ is not empty

$$
\begin{aligned}
& \mathcal{S}^{\pm}[\![1]\!]\overset{\pm}{\rho} \\
\triangleq\ & \ddot{\alpha}_\pm(\mathcal{S}^{\mathbb{C}}[\![1]\!])\overset{\pm}{\rho} && \text{(def. abstraction)} \\
=\ & \alpha_\pm(\{\mathcal{S}(\rho) \mid \mathcal{S} \in \mathcal{S}^{\mathbb{C}}[\![1]\!] \wedge \rho \in \dot{\gamma}_\pm(\overset{\pm}{\rho})\}) && \text{(def. (3.34) of } \ddot{\alpha}_\pm\text{)} \\
=\ & \alpha_\pm(\{\mathcal{A}[\![1]\!](\rho) \mid \rho \in \dot{\gamma}_\pm(\overset{\pm}{\rho})\}) && \text{(def. (3.13) of } \mathcal{S}^{\mathbb{C}}[\![1]\!]\text{)} \\
=\ & \alpha_\pm(\{1\}) && \text{(}\dot{\gamma}_\pm(\overset{\pm}{\rho}) \text{ is not empty and def. (3.4) of } \mathcal{A}[\![1]\!]\text{)} \\
=\ & {>}0 && \text{(def. (3.30) of } \alpha_\pm\text{)}
\end{aligned}
$$

- Otherwise $\dot{\gamma}_\pm(\overset{\pm}{\rho}) = \emptyset$ is empty

$$
\begin{aligned}
& \mathcal{S}^{\pm}[\![\mathsf{A}]\!]\overset{\pm}{\rho} \\
=\ & \alpha_\pm(\{\mathcal{A}[\![\mathsf{A}]\!](\rho) \mid \rho \in \dot{\gamma}_\pm(\overset{\pm}{\rho})\}) \quad = \quad \alpha_\pm(\emptyset) && \text{(def. } \mathcal{S}^{\pm}[\![\mathsf{A}]\!] \text{ with } \dot{\gamma}_\pm(\overset{\pm}{\rho}) = \emptyset\text{)} \\
=\ & \bot_\pm && \text{(def. } \alpha_\pm\text{)}
\end{aligned}
$$

Variable (when $\gamma_\pm(\overset{\pm}{\rho}(\mathsf{y}))$ is not empty)

$$
\begin{aligned}
& \mathcal{S}^{\pm}[\![\mathsf{x}]\!]\overset{\pm}{\rho} \\
= \;& \ddot{\alpha}_\pm(\mathcal{S}^{\mathbb{C}}[\![\mathsf{x}]\!])\overset{\pm}{\rho} \\
= \;& \alpha_\pm(\{\mathcal{S}(\rho) \mid \mathcal{S} \in \mathcal{S}^{\mathbb{C}}[\![\mathsf{x}]\!] \wedge \rho \in \dot{\gamma}_\pm(\overset{\pm}{\rho})\}) && \text{(def. (3.34) of } \ddot{\alpha}_\pm\text{)} \\
= \;& \alpha_\pm(\{\mathcal{A}[\![\mathsf{x}]\!](\rho) \mid \rho \in \dot{\gamma}_\pm(\overset{\pm}{\rho})\}) && \text{(def. (3.13) of } \mathcal{S}^{\mathbb{C}}[\![\mathsf{x}]\!]\text{)} \\
= \;& \alpha_\pm(\{\rho(\mathsf{x}) \mid \rho \in \dot{\gamma}_\pm(\overset{\pm}{\rho})\}) && \text{(def. (3.4) of } \mathcal{A}[\![\mathsf{x}]\!]\text{)} \\
= \;& \alpha_\pm(\{\rho(\mathsf{x}) \mid \forall \mathsf{y} \in \mathbb{V} \,.\, \rho(\mathsf{y}) \in \gamma_\pm(\overset{\pm}{\rho}(\mathsf{y}))\}) && \text{(def. (3.24) of } \dot{\gamma}_\pm\text{)} \\
= \;& \alpha_\pm(\{\rho(\mathsf{x}) \mid \rho(\mathsf{x}) \in \gamma_\pm(\overset{\pm}{\rho}(\mathsf{x}))\})
\end{aligned}
$$

(when $\gamma_\pm(\overset{\pm}{\rho}(\mathsf{y}))$ is not empty so for $\mathsf{y} \neq \mathsf{x}$, $\rho(\mathsf{y})$ can be chosen arbitrarily to satisfy $\rho(\mathsf{y}) \in \gamma_\pm(\overset{\pm}{\rho}(\mathsf{y}))$)

$$
\begin{aligned}
= \;& \alpha_\pm(\{x \mid x \in \gamma_\pm(\overset{\pm}{\rho}(\mathsf{x}))\}) && \text{(letting } x = \rho(\mathsf{x})\text{)} \\
= \;& \alpha_\pm(\gamma_\pm(\overset{\pm}{\rho}(\mathsf{x}))) && \text{(since } S = \{x \mid z \in S\} \text{ for any set } S\text{)} \\
= \;& \overset{\pm}{\rho}(\mathsf{x}) && \text{(by (3.37), } \alpha_\pm \circ \gamma_\pm \text{ is the identity)}
\end{aligned}
$$

Difference (when $\dot{\gamma}_\pm(\overset{\pm}{\rho})$ is not empty)

We assume, by structural induction hypothesis, that $\ddot{\alpha}_\pm(\mathcal{S}^{\mathbb{C}}[\![\mathsf{A}_1]\!]) \mathrel{\dot{\sqsubseteq}_\pm} \mathcal{S}^{\pm}[\![\mathsf{A}_1]\!]$ and $\ddot{\alpha}_\pm(\mathcal{S}^{\mathbb{C}}[\![\mathsf{A}_2]\!]) \mathrel{\dot{\sqsubseteq}_\pm} \mathcal{S}^{\pm}[\![\mathsf{A}_2]\!]$

$$
\begin{aligned}
&\ddot{\alpha}_\pm(\mathcal{S}^{\mathbb{C}}[\![\mathsf{A}_1 - \mathsf{A}_2]\!])\overset{\pm}{\rho} \\
= \;& \alpha_\pm(\{\mathcal{S}(\rho) \mid \mathcal{S} \in \mathcal{S}^{\mathbb{C}}[\![\mathsf{A}_1 - \mathsf{A}_2]\!] \wedge \rho \in \dot{\gamma}_\pm(\overset{\pm}{\rho})\}) && \text{(def. (3.34) of } \ddot{\alpha}_\pm\text{)} \\
= \;& \alpha_\pm(\{\mathcal{A}[\![\mathsf{A}_1 - \mathsf{A}_2]\!](\rho) \mid \rho \in \dot{\gamma}_\pm(\overset{\pm}{\rho})\}) && \text{(def. (3.13) of } \mathcal{S}^{\mathbb{C}}[\![\mathsf{A}_1 - \mathsf{A}_2]\!]\text{)} \\
= \;& \alpha_\pm(\{\mathcal{A}[\![\mathsf{A}_1]\!](\rho) - \mathcal{A}[\![\mathsf{A}_2]\!](\rho) \mid \rho \in \dot{\gamma}_\pm(\overset{\pm}{\rho})\}) && \text{(def. (3.4) of } \mathcal{A}\text{)} \\
\sqsubseteq_\pm \;& \alpha_\pm(\{x - y \mid x \in \{\mathcal{A}[\![\mathsf{A}_1]\!](\rho') \mid \rho' \in \dot{\gamma}_\pm(\overset{\pm}{\rho})\} \wedge y \in \{\mathcal{A}[\![\mathsf{A}_2]\!](\rho'') \mid \rho'' \in \dot{\gamma}_\pm(\overset{\pm}{\rho})\}\}) \\
& \quad \text{(}\{f(\rho) - g(\rho) \mid \rho \in R\} \subseteq \{x - y \mid x \in \{f(\rho') \mid \rho' \in R\} \wedge y \in \{g(\rho'') \mid \rho'' \in R\}\} \text{ and } \alpha_\pm \text{ is increasing.[6])}
\end{aligned}
$$

[6] This over-approximation allows for A_1 and A_2 to be evaluated in the concrete with different environments ρ' and ρ'' with the same sign of variables but possibly different values of variables. This accounts for the fact that the rule of signs does not take relationships between values of variables into account. For example the sign of x - x is not =0 in general.

$$\sqsubseteq_\pm \; \alpha_\pm(\{x - y \mid x \in \gamma_\pm(\alpha_\pm(\{\mathscr{A}[\![\mathsf{A}_1]\!](\rho) \mid \rho \in \dot{\gamma}_\pm(\overset{\pm}{\rho})\}) \wedge y \in \gamma_\pm(\alpha_\pm(\{\mathscr{A}[\![\mathsf{A}_2]\!](\rho) \mid \rho \in \dot{\gamma}_\pm(\overset{\pm}{\rho})\})\})$$

$\wr\{x - y \mid x \in P \wedge y \in Q\} \subseteq \{x - y \mid x \in \gamma_\pm(\alpha_\pm(P)) \wedge y \in \gamma_\pm(\alpha_\pm(Q))\}$ since $\gamma_\pm \circ \alpha_\pm$ is. extensive and $\alpha_\pm$ is increasing[7]$\wr$

$$= \; \alpha_\pm(\{\mathscr{A}[\![\mathsf{A}_1]\!](\rho) \mid \rho \in \dot{\gamma}_\pm(\overset{\pm}{\rho})\}) \mathbin{-_\pm} \alpha_\pm(\{\mathscr{A}[\![\mathsf{A}_2]\!](\rho) \mid \rho \in \dot{\gamma}_\pm(\overset{\pm}{\rho})\})$$

$\wr s_1 \mathbin{-_\pm} s_2 = \alpha_\pm(\{x - y \mid x \in \gamma_\pm(s_1) \wedge y \in \gamma_\pm(s_2)\})\wr$

$$= \; \alpha_\pm(\{\mathcal{S}(\rho) \mid \mathcal{S} \in \mathcal{S}^{\mathbb{C}}[\![\mathsf{A}_1]\!] \wedge \rho \in \dot{\gamma}_\pm(\overset{\pm}{\rho})\}) \mathbin{-_\pm} \alpha_\pm(\{\mathcal{S}(\rho) \mid \mathcal{S} \in \mathcal{S}^{\mathbb{C}}[\![\mathsf{A}_2]\!] \wedge \rho \in \dot{\gamma}_\pm(\overset{\pm}{\rho})\}) \quad \wr\text{def. } \mathcal{S}^{\mathbb{C}}\wr$$

$$= \; \ddot{\alpha}_\pm(\mathcal{S}^{\mathbb{C}}[\![\mathsf{A}_1]\!])\overset{\pm}{\rho} \mathbin{-_\pm} \ddot{\alpha}_\pm(\mathcal{S}^{\mathbb{C}}[\![\mathsf{A}_2]\!])\overset{\pm}{\rho} \quad \wr\text{def. } \ddot{\alpha}_\pm\wr$$

$$= \; \ddot{\alpha}_\pm(\mathcal{S}^{\mathbb{C}}[\![\mathsf{A}_1]\!])\overset{\pm}{\rho} \mathbin{-_\pm} \ddot{\alpha}_\pm(\mathcal{S}^{\mathbb{C}}[\![\mathsf{A}_2]\!])\overset{\pm}{\rho} \quad \wr\text{def. } \ddot{\alpha}_\pm\wr$$

$$\sqsubseteq_\pm \; (\mathcal{S}^{\pm}[\![\mathsf{A}_1]\!]\overset{\pm}{\rho}) \mathbin{-_\pm} (\mathcal{S}^{\pm}[\![\mathsf{A}_2]\!]\overset{\pm}{\rho})$$

$\wr$induction hypothesis and $-_\pm$ is increasing in both parameters$\wr$

$$\triangleq \; \mathcal{S}^{\pm}[\![\mathsf{A}_1 - \mathsf{A}_2]\!]\overset{\pm}{\rho} \quad \wr\text{def. } \mathcal{S}^{\pm}[\![\mathsf{A}_1 - \mathsf{A}_2]\!] \text{ when } \forall \mathsf{y} \in \mathbb{V} \,.\, \overset{\pm}{\rho}(\mathsf{y}) \neq \bot_\pm\wr \qquad \square$$

[7]This over-approximation allows for the evaluation of the sign to be performed in the abstract with $-_\pm$ instead of the concrete.

Abstract interpreter

Abstract interpreter

- The calculational design can be generalized to any abstract domain

$$\mathbb{D}^{¤} \triangleq \langle \mathbb{P}^{¤}, \sqsubseteq^{¤}, \perp^{¤}, \sqcup^{¤}, 1^{¤}, \ominus^{¤} \rangle$$

  such that

  - $\langle \wp(\mathbb{Z}), \subseteq \rangle \xleftarrow[\alpha]{\gamma} \langle \mathbb{P}^{¤}, \sqsubseteq^{¤} \rangle$
  - $\{1\} \subseteq \gamma(1^{¤})$
  - $\forall \overline{P}_1, \overline{P}_2 \in \mathbb{P}^{¤} \,.\, \{x - y \mid x \in \gamma(\overline{P}_1) \wedge y \in \gamma(\overline{P}_2)\} \subseteq \gamma(\overline{P}_1 \ominus^{¤} \overline{P}_2)$

- Then the abstract interpreter

$$\begin{aligned} \mathcal{S}^{¤}[\![1]\!] &= \boldsymbol{\lambda} \overline{\rho} \bullet 1^{¤} \\ \mathcal{S}^{¤}[\![\mathsf{x}]\!] &= \boldsymbol{\lambda} \overline{\rho} \bullet \overline{\rho}(\mathsf{x}) \\ \mathcal{S}^{¤}[\![\mathsf{A}_1 - \mathsf{A}_2]\!] &= \boldsymbol{\lambda} \overline{\rho} \bullet (\mathcal{S}^{¤}[\![\mathsf{A}_1]\!]\overline{\rho}) \ominus^{¤} (\mathcal{S}^{¤}[\![\mathsf{A}_2]\!]\overline{\rho}) \end{aligned}$$

  is sound $\forall \mathsf{A} \in \mathcal{A} \,.\, \mathcal{S}^{\mathbb{C}}[\![\mathsf{A}]\!] \subseteq \ddot{\gamma}(\mathcal{S}^{¤}[\![\mathsf{A}]\!])$ *i.e.* $\mathcal{A}[\![\mathsf{A}]\!] \in \ddot{\gamma}(\mathcal{S}^{¤}[\![\mathsf{A}]\!])$

Parity analysis

- Abstract domain:

$$\mathbb{P}^2 = \begin{tikzcd}[column sep=small] & \mathbb{Z} \arrow[dl, dash] \arrow[dr, dash] & \\ \{2z \mid z \in \mathbb{Z}\} \arrow[dr, dash] & & \{2z+1 \mid z \in \mathbb{Z}\} \arrow[dl, dash] \\ & \varnothing & \end{tikzcd} \qquad \overline{\mathbb{P}^2} = \begin{tikzcd}[column sep=small] & \top^2 \arrow[dl, dash] \arrow[dr, dash] & \\ \mathbb{e} \arrow[dr, dash] & & \mathbb{o} \arrow[dl, dash] \\ & \bot^2 & \end{tikzcd}$$

- Constant 1: $1^2 \triangleq \mathbb{o}$
- Difference:

x	$\mathbb{e}$	$\mathbb{e}$	$\mathbb{o}$	$\mathbb{o}$	$\_$	$\bot^2/\top^2$
y	$\mathbb{e}$	$\mathbb{o}$	$\mathbb{e}$	$\mathbb{o}$	$\bot^2/\top^2$	$\_$
$x \ominus^2 y$	$\mathbb{e}$	$\mathbb{o}$	$\mathbb{o}$	$\mathbb{e}$	$\bot^2/\top^2$	$\bot^2/\top^2$

Part 2

October 31, 2019, 11:00—12:00

Introduction

Great but what about iteration (and recursion)

- trace semantics
- semantics of `while` iteration
- fixpoints
- fixpoint extrapolation (widening) and interpolation (narrowing)

Traces

Hand computation of

$$\underbrace{\underbrace{\underbrace{(1-1)}_{a}-1}_{b} < \underbrace{(1-1)}_{a}}_{c}$$

is

$$
\begin{aligned}
a &= 1 - 1\\
a &= 0\\
b &= a - 1\\
b &= 0 - 1\\
b &= -1\\
c &= b < a\\
c &= -1 < 0\\
c &= tt
\end{aligned}
$$

read from

partial trace

maximal finite trace

Traces

Finite traces of a program: P

- Program (notice the labelling):

$$\ell_1\ \texttt{x = x + 1 ;}\quad \texttt{while}\ \ell_2\ (\mathrm{tt})\ \{\ \ell_3\ \texttt{x = x + 1 ;}\quad \texttt{if}\ \ell_4\ (\texttt{x > 2})\ \ell_5\ \texttt{break ;}\}\ell_6\texttt{;}\ell_7 \tag{4.4}$$

- Prefix traces (from ℓ_1, initially $\mathtt{x} = 0$):
  - ℓ_1
  - $\ell_1 \xrightarrow{\mathtt{x}=1} \ell_2 \xrightarrow{\mathrm{tt}} \ell_3 \xrightarrow{\mathtt{x}=2} \ell_4 \xrightarrow{\neg(\mathtt{x}>2)} \ell_2 \xrightarrow{\mathrm{tt}} \ell_3$

- Finite (maximal) traces:
  - $\ell_1 \xrightarrow{\mathtt{x}=1} \ell_2 \xrightarrow{\mathrm{tt}} \ell_3 \xrightarrow{\mathtt{x}=2} \ell_4 \xrightarrow{\neg(\mathtt{x}>2)} \ell_2 \xrightarrow{\mathrm{tt}} \ell_3 \xrightarrow{\mathtt{x}=3} \ell_4 \xrightarrow{\mathtt{x}>2} \ell_5 \xrightarrow{\texttt{break}}$
    $\ell_6 \xrightarrow{\mathsf{skip}} \ell_7$

Infinite traces of a program: P

- Program:

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

- Infinite trace:

$$\ell_1 \xrightarrow{\mathtt{x}\,=\,0} \ell_2 \xrightarrow{\mathrm{tt}} \ell_3 \xrightarrow{\mathtt{x}\,=\,1} \ell_2 \xrightarrow{\mathrm{tt}} \ell_3 \xrightarrow{\mathtt{x}\,=\,2} \ell_2 \ldots \ell_2 \xrightarrow{\mathrm{tt}} \ell_3 \xrightarrow{\mathtt{x}\,=\,n} \ell_2 \xrightarrow{\mathrm{tt}} \ell_3 \xrightarrow{\mathtt{x}\,=\,n+1} \ell_2 \ldots$$

Traces

- $\mathbb{T}^+$: the set of all finite traces,
- $\mathbb{T}^\infty$: the set of all infinite traces,
- $\mathbb{T}^{+\infty}$: the set of all finite or infinite traces.
- Conventions:
  - we write $\pi = {}^\ell\pi'$ to make clear that the trace π is assumed to start with the program label ℓ (although π' is not itself a properly formed trace),
  - we write $\pi = \pi'^\ell$ when assuming that the trace π is finite and ends with label ℓ (although, again, π' is not itself a properly formed trace).

Trace concatenation $\frown$

- Definition:

$$
\begin{array}{rcll}
\pi_1\ell_1 \frown \ell_2\pi_2 & & & \text{undefined if } \ell_1 \neq \ell_2 \\
\pi_1\ell_1 \frown \ell_1\pi_2 & \triangleq & \pi_1\ell_1\pi_2 & \text{if } \pi_1 \text{ is finite} \\
\pi_1 \frown \pi_2 & \triangleq & \pi_1 & \text{if } \pi_1 \text{ is infinite}
\end{array}
$$

- In pattern matching, we sometimes need the empty trace $\ni$. For example $\ell\pi\ell' = \ell$ then $\pi = \ni$ and $\ell = \ell'$.

Values of variables at the end of a trace

- the value $\varrho(\pi)\mathtt{x}$ of variable $\mathtt{x}$ *at the end* of trace π is the last value assigned to $\mathtt{x}$ (or 0 at initialization).

$$
\begin{aligned}
\varrho(\pi^{\ell} \xrightarrow{\mathtt{x} = \upsilon} \ell')\mathtt{x} &\triangleq \upsilon \\
\varrho(\pi^{\ell} \xrightarrow{\cdots} \ell')\mathtt{x} &\triangleq \varrho(\pi^{\ell}) \quad \text{otherwise} \\
\varrho(\ell)\mathtt{x} &\triangleq 0
\end{aligned}
\tag{6.4}
$$

Chapter 0

Prefix trace semantics

Prefix trace semantics of the assignment statement

Prefix traces of an assignment statement $\mathsf{S} ::= {}^{\ell}\ \mathtt{x = A\ ;}$

$$\widehat{\mathcal{S}}^{*}[\![\mathsf{S}]\!] = \{\langle \pi\ell',\ \ell'\rangle \mid \ell' = \ell\} \cup \{\ell' \xrightarrow{\mathtt{x = A} = v} \mathsf{after}[\![\mathsf{S}]\!] \mid \ell' = \ell \wedge v = \mathcal{A}[\![\mathsf{A}]\!]\varrho(\pi\ell)\} \qquad (15.1)$$

- $\mathsf{after}[\![\mathsf{S}]\!]$ is the program label reached on termination of program component S
- $\mathsf{at}[\![\mathsf{S}]\!]$ is the program label where the execution of S starts
- $\varrho(\pi^{\ell})$ is the environment assigning a value to variables at the end of the trace π^{ℓ}
- The semantics of a program component S is a set of pairs $\langle \pi^{\ell},\ {}^{\ell}\pi' \rangle$ where the initialization π^{ℓ} is a computation arriving $\mathsf{at}[\![\mathsf{S}]\!] = \ell$ and the continuation ${}^{\ell}\pi'$ describes zero or more computation steps of S after reaching $\mathsf{at}[\![\mathsf{S}]\!] = \ell$

Prefix trace semantics of a statement list

Prefix traces of a statement list $\texttt{Sl} ::= \texttt{Sl}'\ \texttt{S}$

$$\widehat{\mathcal{S}}^*[\![\texttt{Sl}]\!] = \widehat{\mathcal{S}}^*[\![\texttt{Sl}']\!] \cup \{\langle\pi_1,\ \pi_2 \frown \pi_3\rangle \mid \langle\pi_1,\ \pi_2\rangle \in \widehat{\mathcal{S}}^*[\![\texttt{Sl}']\!] \wedge \langle\pi_1 \frown \pi_2,\ \pi_3\rangle \in \widehat{\mathcal{S}}^*[\![\texttt{S}]\!]\} \tag{15.2}$$

- π_3 starts $\text{at}[\![\texttt{S}]\!] = \text{after}[\![\texttt{Sl}']\!]$ so π_2 must necessarily terminate $\text{after}[\![\texttt{Sl}']\!] = \text{at}[\![\texttt{S}]\!]$ *i.e.* the execution of $\texttt{Sl}'$ must necessarily terminate for that of $\texttt{S}$ to start
- The values of variables on π_1, π_2, and π_3 are necessarily compatible

$\underbrace{\ldots}_{\pi_1} \ell_1 \underbrace{\xrightarrow{\texttt{x = 0} = 0}}_{\pi_2} \ell_2 \underbrace{\xrightarrow{\texttt{x = x - 1} = 42}}_{\pi_3} \ell_3$ is impossible

Prefix trace semantics of the conditional statement

Prefix traces of a conditional statement $\texttt{S} ::= \texttt{if}\ ^{\ell}\ \texttt{(B)}\ \texttt{S}_t$

$$\widehat{\mathcal{S}}^*[\![\texttt{S}]\!] = \{\langle\pi_1\ell,\ \ell \xrightarrow{\neg(\texttt{B})} \text{after}[\![\texttt{S}]\!]\rangle \mid \mathcal{B}[\![\texttt{B}]\!]\varrho(\pi_1\ell) = \text{ff}\} \cup \tag{6.16}$$

$$\{\langle\pi_1\ell,\ \ell \xrightarrow{\texttt{B}} \text{at}[\![\texttt{S}_t]\!] \frown \pi_2\rangle \mid \mathcal{B}[\![\texttt{B}]\!]\varrho(\pi_1\ell) = \text{tt} \wedge \tag{6.17}$$

$$\langle\pi_1\ell \xrightarrow{\texttt{B}} \text{at}[\![\texttt{S}_t]\!],\ \pi_2\rangle \in \widehat{\mathcal{S}}^*[\![\texttt{S}_t]\!]\}$$

- This includes the case when the true alternative $\texttt{S}_t$ terminates $\text{after}[\![\texttt{S}_t]\!] = \text{after}[\![\texttt{S}]\!]$

Prefix trace semantics of the while iteration

- The prefix trace semantics $\widehat{\mathcal{S}}^*[\![\texttt{while}\ \ell\ \texttt{(B)}\ \texttt{S}_b]\!]$ of an iteration $\texttt{while}\ \ell\ \texttt{(B)}\ \texttt{S}_b$ with loop body $\texttt{S}_b$ define traces after 0, 1, 2, ... iterations
- $\texttt{while (B) S}_b \equiv \texttt{if (B) \{S}_b\texttt{;while (B) S}_b\texttt{\}}$
- or $X \equiv \texttt{if (B) \{S}_b\texttt{;}X\texttt{\}}$ where $X \equiv \texttt{while (B) S}_b$
- So the prefix trace semantics $\widehat{\mathcal{S}}^*[\![\texttt{while}\ \ell\ \texttt{(B)}\ \texttt{S}_b]\!]$ is defined recursively

$$
\begin{array}{rcl}
\widehat{\mathcal{S}}^*[\![\texttt{while}\ \ell\ \texttt{(B)}\ \texttt{S}_b]\!] & = & \mathcal{F}^*[\![\texttt{while}\ \ell\ \texttt{(B)}\ \texttt{S}_b]\!](\widehat{\mathcal{S}}^*[\![\texttt{while}\ \ell\ \texttt{(B)}\ \texttt{S}_b]\!]) \\
\text{or} \quad X & = & \mathcal{F}^*[\![\texttt{while}\ \ell\ \texttt{(B)}\ \texttt{S}_b]\!](X)
\end{array}
$$

- $\mathcal{F}^*[\![\texttt{while}\ \ell\ \texttt{(B)}\ \texttt{S}_b]\!]X$ describes the effect of one iteration $\texttt{if (B) \{S}_b\texttt{;}X\texttt{\}}$
- Technically, $\widehat{\mathcal{S}}^*[\![\texttt{while}\ \ell\ \texttt{(B)}\ \texttt{S}_b]\!]$ is the least fixpoint of $\mathcal{F}^*[\![\texttt{while}\ \ell\ \texttt{(B)}\ \texttt{S}_b]\!]$

Prefix trace semantics of the `while` iteration (cont'd)

Prefix traces of an iteration statement $\mathtt{S} ::= \mathtt{while}\ ^{\ell}\ (\mathtt{B})\ \mathtt{S}_b$

$$\widehat{\mathcal{S}}^*[\![\mathtt{while}\ ^{\ell}\ (\mathtt{B})\ \mathtt{S}_b]\!] \quad = \quad \mathsf{lfp}^{\dot{\subseteq}}\ \mathcal{F}^*[\![\mathtt{while}\ ^{\ell}\ (\mathtt{B})\ \mathtt{S}_b]\!] \tag{15.3}$$

$$\mathcal{F}^*[\![\mathtt{while}\ ^{\ell}\ (\mathtt{B})\ \mathtt{S}_b]\!](X) \quad \triangleq \quad \{\langle\pi_1\ell,\ \ell\rangle\} \tag{a}$$

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

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

- $\mathcal{F}^*[\![\mathtt{while}\ ^{\ell}\ (\mathtt{B})\ \mathtt{S}_b]\!](X)(\pi_1\ell') \quad = \quad \varnothing$ when $\ell' \neq \ell$

Example

Consider $\texttt{S} = \textbf{while}\ ^{\ell}\ (\text{tt})\ ^{\ell'}\texttt{x = x + 1 ;}$ so that $\texttt{S}_b = {}^{\ell'}\texttt{x = x + 1 ;}$. We have

$$\begin{aligned}\mathscr{F}^*[\![\texttt{S}]\!](X) \quad \triangleq \quad & \{\langle\pi_1\ell,\ \ell\rangle\} \cup \{\langle\pi_1\ell,\ \ell\pi_2\ell \xrightarrow{\text{tt}} \ell'\rangle \mid \langle\pi_1\ell,\ \ell\pi_2\ell\rangle \in X\} \cup \\ & \{\langle\pi_1\ell,\ \ell\pi_2\ell \xrightarrow{\text{tt}} \ell' \xrightarrow{\texttt{x = x + 1} = v} \ell\rangle \mid \langle\pi_1\ell,\ \ell\pi_2\ell\rangle \in X \land v = \varrho(\pi_1\ell\pi_2\ell) + 1\}\end{aligned}$$

The iterates $\langle\mathscr{F}^{*n},\ n \in \mathbb{N}\rangle$ of $\mathscr{F}^*[\![\texttt{S}]\!]$ from $\varnothing$ are

$$\begin{aligned}\mathscr{F}^{*0} \quad = \quad & \varnothing \\ \mathscr{F}^{*1} \quad = \quad & \{\langle\pi_1\ell,\ \ell\rangle\} \\ \mathscr{F}^{*2} \quad = \quad & \{\langle\pi_1\ell,\ \ell\rangle, \langle\pi_1\ell,\ \ell \xrightarrow{\text{tt}} \ell'\rangle, \langle\pi_1\ell,\ \ell \xrightarrow{\text{tt}} \ell' \xrightarrow{\texttt{x = x + 1} = v} \ell\rangle \mid v = \varrho(\pi_1\ell) + 1\} \\ \mathscr{F}^{*3} \quad = \quad & \{\langle\pi_1\ell,\ \ell\rangle, \langle\pi_1\ell,\ \ell \xrightarrow{\text{tt}} \ell'\rangle, \langle\pi_1\ell,\ \ell \xrightarrow{\text{tt}} \ell' \xrightarrow{\texttt{x = x + 1} = v(1)} \ell\rangle, \langle\pi_1\ell,\ \ell \xrightarrow{\text{tt}} \\ & \ell' \xrightarrow{\texttt{x = x + 1} = v(1)} \ell \xrightarrow{\text{tt}} \ell'\rangle, \langle\pi_1\ell,\ \ell \xrightarrow{\text{tt}} \ell' \xrightarrow{\texttt{x = x + 1} = v(1)} \ell \xrightarrow{\text{tt}} \\ & \ell' \xrightarrow{\texttt{x = x + 1} = v(2)} \ell\rangle \mid \forall i \in [1, 2]\ .\ v(i) = \varrho(\pi_1\ell) + i\}\end{aligned}$$

...

$$\mathcal{F}^{*n} = \Big\{\langle\pi_1\ell,\ \Big(\ell \xrightarrow{\text{tt}} \ell' \xrightarrow{\mathtt{x = x + 1} = v(i)} \ell\Big)_{i=0}^{k}\rangle, \langle\pi_1\ell,\ \Big(\ell \xrightarrow{\text{tt}} \ell' \xrightarrow{\mathtt{x = x + 1} = v(i)} \ell\Big)_{i=0}^{k'} \frown \ell \xrightarrow{\text{tt}} \ell'\rangle \ \Big|\ k \in [0, n[\wedge k' \in [0, n-2] \wedge \forall i \in [1, n-1]\ .\ v(i) = \varrho(\pi_1\ell) + i\Big\}$$

$$\wr\text{ind. hyp. with } \Big(\ell \ldots \ell\Big)^0 = \ell\,\S$$

$$\mathcal{F}^{*n+1} = \mathcal{F}^*[\![\mathtt{S}]\!](\mathcal{F}^{*n}) \qquad \wr\text{def. iterates}\S$$

... $\ldots$ $\wr$develop and simplify$\S$

$$= \Big\{\langle\pi_1\ell,\ \Big(\ell \xrightarrow{\text{tt}} \ell' \xrightarrow{\mathtt{x = x + 1} = v(i)} \ell\Big)_{i=0}^{k}\rangle, \langle\pi_1\ell,\ \Big(\ell \xrightarrow{\text{tt}} \ell' \xrightarrow{\mathtt{x = x + 1} = v(i)} \ell\Big)_{i=0}^{k'} \frown \ell \xrightarrow{\text{tt}} \ell'\rangle \ \Big|\ k \in [0, n] \wedge k' \in [0, n-1] \wedge \forall i \in [1, n]\ .\ v(i) = \varrho(\pi_1\ell) + i\Big\}$$

...

$$
\begin{aligned}
\widehat{\mathcal{S}}^{*}[\![\mathsf{s}]\!] &= \mathsf{lfp}^{\dot{\subseteq}}\,\mathscr{F}^{*}[\![\mathsf{s}]\!] \\
&= \mathscr{F}^{*\,\omega} \\
&= \bigcup_{n\in\mathbb{N}} \mathscr{F}^{*\,n} \\
&= \Big\{\langle \pi_1\ell,\ \Big(\ell \xrightarrow{\ \mathtt{tt}\ } \ell' \xrightarrow{\ \mathsf{x = x + 1} = v(i)\ } \ell\Big)_{i=0}^{k}\rangle, \langle \pi_1\ell,\ \Big(\ell \xrightarrow{\ \mathtt{tt}\ } \ell' \xrightarrow{\ \mathsf{x = x + 1} = v(i)\ } \\
&\qquad \ell\Big)_{i=0}^{k} \frown \ell \xrightarrow{\ \mathtt{tt}\ } \ell'\rangle \;\Big|\; k \in \mathbb{N} \wedge \forall i \in \mathbb{N}\,.\, v(i) = \varrho(\pi_1\ell) + i\Big\}
\end{aligned}
\qquad \square
$$

Fixpoints

Iteration

- We have seen that the (partial trace) semantics of an iteration is defined as

$$\mathcal{S} \quad = \quad \mathsf{lfp}^{\sqsubseteq} \mathscr{F}$$

that is the $\sqsubseteq$-least solution/fixpoint of the equation

$$X = \mathscr{F}(X)$$

on a partial order $\langle \mathcal{D}, \sqsubseteq \rangle$

- Kleene/Tarski/Scott theorems ensure the existence of this $\sqsubseteq$-least solution/fixpoint

Kleene/Tarski/Scott fixpoint iteration theorem

If

- $\langle \mathcal{D}, \sqsubseteq, \perp, \sqcup \rangle$ is a poset with infimum $\perp$ and (partially defined) least upper bound $\sqcup$
- $\mathscr{F} \in \mathcal{D} \xrightarrow{uc} \mathcal{D}$ is upper-continuous

  i.e. if the increasing chain $x_0 \sqsubseteq x_1 \sqsubseteq \ldots \sqsubseteq x_n \sqsubseteq \ldots$ of elements of $\mathcal{D}$ has a least upper bound $\bigsqcup_{n \in \mathbb{N}} x_n \in \mathcal{D}$ then $\mathscr{F}(\bigsqcup_{n \in \mathbb{N}} x_n) = \bigsqcup_{n \in \mathbb{N}} \mathscr{F}(x_n)$
- The iterates $\mathscr{F}^0 = \perp$, ..., $\mathscr{F}^{n+1} = \mathscr{F}(\mathscr{F}^n)$ have a least upper bound in $\mathcal{D}$

then

$X = \mathscr{F}(X)$ has a least solution $\mathsf{lfp}^{\sqsubseteq}\, \mathscr{F} = \bigsqcup_{n \in \mathbb{N}} \mathscr{F}^n$

i.e. $\mathsf{lfp}^{\sqsubseteq}\, \mathscr{F} = \mathscr{F}(\mathsf{lfp}^{\sqsubseteq}\, \mathscr{F})$

& if $X = \mathscr{F}(X)$ then $\mathsf{lfp}^{\sqsubseteq}\, \mathscr{F} \sqsubseteq X$

Fixpoint abstraction

Exact fixpoint abstraction

If

- $\langle \mathcal{D}, \sqsubseteq, \bot, \sqcup \rangle$ is a poset with infimum $\bot$ and (partially defined) least upper bound $\sqcup$
- $\mathcal{F} \in \mathcal{D} \xrightarrow{uc} \mathcal{D}$ is upper-continuous
- The iterates $\mathcal{F}^0 = \bot$, ..., $\mathcal{F}^{n+1} = \mathcal{F}(\mathcal{F}^n)$ have a least upper bound in $\mathcal{D}$
- $\langle \mathcal{D}, \sqsubseteq \rangle \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} \langle \mathbb{P}^{¤}, \sqsubseteq^{¤} \rangle$, α surjective

then

$$\mathsf{lfp}^{\sqsubseteq} \mathcal{F} \sqsubseteq \gamma(\mathsf{lfp}^{\sqsubseteq^{¤}} \alpha \circ \mathcal{F} \circ \gamma)$$

Fixpoint over-approximation

If

- $\langle \mathcal{D}, \sqsubseteq, \perp, \sqcup \rangle$ is a poset with infimum $\perp$ and (partially defined) least upper bound $\sqcup$
- $\mathscr{F} \in \mathcal{D} \xrightarrow{uc} \mathcal{D}$ is upper-continuous
- The iterates $\mathscr{F}^0 = \perp$, ..., $\mathscr{F}^{n+1} = \mathscr{F}(\mathscr{F}^n)$ have a least upper bound in $\mathcal{D}$
- $\langle \mathcal{D}, \sqsubseteq \rangle \xrightleftharpoons[\alpha]{\gamma} \langle \mathbb{P}^{¤}, \sqsubseteq^{¤} \rangle$
- $\alpha \circ \mathscr{F} \circ \gamma \mathrel{\dot{\sqsubseteq}}^{¤} \mathscr{F}^{¤}$

then

$$\mathsf{lfp}^{\sqsubseteq} \mathscr{F} \sqsubseteq \gamma(\mathsf{lfp}^{\sqsubseteq^{¤}})\mathscr{F}^{¤}$$

Reachability

Reachability abstraction (exact)

- Abstract a set of traces into a map from initial states to reachable states at each program point

$$\langle \wp(\mathbb{T}^+ \times \mathbb{T}^+),\ \subseteq\rangle \xleftarrow[\alpha^{\vec{r}}]{\gamma^{\vec{r}}} \langle \wp(\mathbb{E}\mathbb{v}) \to \mathbb{L} \mapsto \wp(\mathbb{E}\mathbb{v}),\ \dot{\subseteq}\rangle$$

- $\alpha^{\vec{r}}(\mathcal{S})\,\mathcal{R}_0\,\ell \quad \triangleq \quad \{\varrho(\pi_0{}^{\ell_0}\pi^{\ell'}) \mid \langle \pi_0{}^{\ell_0},\ {}^{\ell_0}\pi^{\ell'}\pi'\rangle \in \mathcal{S} \wedge \varrho(\pi_0{}^{\ell_0}) \in \mathcal{R}_0 \wedge \ell' = \ell\}$

Reachability for assignment

Reachability of an assignment statement $\mathtt{S ::= x = E ;}$

$$\widehat{\mathcal{S}}^{\vec{r}}[\![\mathtt{S}]\!]\,\mathcal{R}_0\,\ell \;=\; (\!|\; \ell = \mathsf{at}[\![\mathtt{S}]\!] \;?\; \mathcal{R}_0 \\ \|\; \ell = \mathsf{after}[\![\mathtt{S}]\!] \;?\; \mathsf{assign}^{\vec{r}}[\![\mathtt{x}, \mathtt{A}]\!]\,\mathcal{R}_0 \\ :\; \varnothing \;|\!) \tag{17.12}$$

$$\mathsf{assign}^{\vec{r}}[\![\mathtt{x}, \mathtt{A}]\!]\,\mathcal{R}_0 \;\triangleq\; \{\rho[\mathtt{x} \leftarrow \mathcal{A}[\![\mathtt{A}]\!]\rho] \mid \rho \in \mathcal{R}_0\}$$

Reachability for iteration

Reachability of an iteration statement $\texttt{S} ::= \texttt{while}\ \ell\ \texttt{(B)}\ \texttt{S}_b$

$$\widehat{\mathcal{S}}^{\vec{r}}[\![\texttt{S}]\!]\ \mathcal{R}_0\ \ell' \quad = \quad (\mathsf{lfp}^{\dot{\subseteq}}\ \mathcal{F}^{\vec{r}}[\![\texttt{while}\ \ell\ \texttt{(B)}\ \texttt{S}_b]\!]\ \mathcal{R}_0)\ \ell' \tag{17.16}$$

$$\mathcal{F}^{\vec{r}}[\![\texttt{while}\ \ell\ \texttt{(B)}\ \texttt{S}_b]\!]\ \mathcal{R}_0\ X\ \ell' \quad =$$

$$\begin{aligned}
&(\!|\ \ell' = \ell\ ?\ \mathcal{R}_0 \cup \widehat{\mathcal{S}}^{\vec{r}}[\![\texttt{S}_b]\!]\ (\mathsf{test}^{\vec{r}}[\![\texttt{B}]\!]X(\ell))\ \ell && \text{(a)}\\
&\|\ \ell' \in \mathsf{in}[\![\texttt{S}_b]\!] \setminus \{\ell\}\ ?\ \widehat{\mathcal{S}}^{\vec{r}}[\![\texttt{S}_b]\!]\ (\mathsf{test}^{\vec{r}}[\![\texttt{B}]\!]X(\ell))\ \ell' && \text{(b)}\\
&\|\ \ell' = \mathsf{after}[\![\texttt{S}]\!]\ ?\ \overline{\mathsf{test}}^{\vec{r}}[\![\texttt{B}]\!](X(\ell)) \cup \bigcup_{\ell'' \in \mathsf{breaks\text{-}of}[\![\texttt{S}_b]\!]} \widehat{\mathcal{S}}^{\vec{r}}[\![\texttt{S}_b]\!]\ (\mathsf{test}^{\vec{r}}[\![\texttt{B}]\!]X(\ell))\ \ell'' && \text{(c)}\\
&\ :\ \varnothing\ |\!)
\end{aligned}$$

$$\mathsf{test}^{\vec{r}}[\![\texttt{B}]\!]\mathcal{R}_0 \triangleq \{\rho \in \mathcal{R}_0 \mid \mathcal{B}[\![\texttt{B}]\!]\rho = \mathsf{tt}\}$$

$$\overline{\mathsf{test}}^{\vec{r}}[\![\texttt{B}]\!]\mathcal{R}_0 \triangleq \{\rho \in \mathcal{R}_0 \mid \mathcal{B}[\![\texttt{B}]\!]\rho = \mathsf{ff}\}$$

Interval analysis

Interval abstraction (approximate)

- Abstract the set of possible values of a variable by the interval of its minimum and maximum value (or ∞)

$$\langle\wp(\mathbb{V}),\ \subseteq\rangle \xleftarrow[\alpha^i]{\gamma^i} \langle\mathbb{P}^i,\ \sqsubseteq^i\rangle \qquad \mathbb{P}^i \triangleq \{[l,h] \mid l \leqslant h\} \cup \{\varnothing\}$$

$$= \qquad \alpha^i(\varnothing) \triangleq \varnothing \quad \alpha^i(V) \triangleq [\min V, \max V]$$

$$\langle\wp(\mathbb{E}\mathbb{v}),\ \subseteq\rangle \xleftarrow[\dot{\alpha}^i]{\dot{\gamma}^i} \langle\mathbb{V} \to \mathbb{P}^i,\ \dot{\sqsubseteq}^i\rangle$$

$$\dot{\alpha}^i(E) \triangleq \boldsymbol{\lambda}\,\mathsf{x} \cdot \dot{\alpha}^i(\{\rho(\mathsf{x}) \mid \rho \in E\})$$

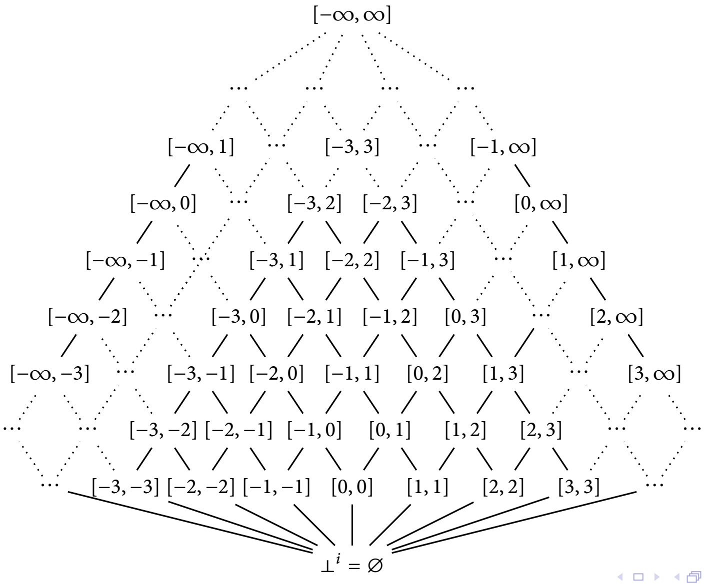
$$\langle\mathbb{L} \to \wp(\mathbb{E}\mathbb{v}),\ \dot{\subseteq}\rangle \xleftarrow[\ddot{\alpha}^i]{\ddot{\gamma}^i} \langle\mathbb{L} \to \mathbb{V} \to \mathbb{P}^i,\ \ddot{\sqsubseteq}^i\rangle$$

$$\ddot{\alpha}^i(I) \triangleq \boldsymbol{\lambda}\,\ell \cdot \dot{\alpha}^i(I(\ell))$$

$$\langle\wp(\mathbb{E}\mathbb{v}) \to (\mathbb{L} \to \wp(\mathbb{E}\mathbb{v})),\ \ddot{\subseteq}\rangle \xleftarrow[\dddot{\alpha}^i]{\dddot{\gamma}^i} \langle(\mathbb{V} \to \mathbb{P}^i) \to (\mathbb{L} \to \mathbb{V} \to \mathbb{P}^i),\ \dddot{\sqsubseteq}^i\rangle$$

$$\dddot{\alpha}^i(T) \triangleq \ddot{\alpha}^i \circ T \circ \dot{\gamma}^i$$

$\langle \mathbb{P}^i, \sqsubseteq^i \rangle$ is an infinite complete lattice



Analysis of an iteration

Consider the simple diverging program $\mathtt{P}_1 \quad = \quad \mathtt{while}\ \ell_1\ (\mathtt{tt})$
$\qquad \ell_2\ \mathtt{x = x + 1\ ;}$
ℓ_3

The interval static analysis from an initial assignment $\overline{\rho}_0$ of intervals to variables is

$$\widehat{\mathcal{S}}^{\,i}[\![\mathtt{P}_1]\!]\,\overline{\rho}_0 \quad = \quad \mathsf{lfp}^{\sqsubseteq^i}\,(\mathcal{F}^{\,i}[\![\mathtt{while}\ \ell_1\ (\mathtt{tt})\ \ell_2\ \mathtt{x = x + 1\ ;}]\!]\,\overline{\rho}_0)$$

where

$$\begin{aligned}
\mathcal{F}^{\,i}[\![\mathtt{while}\ \ell_1\ (\mathtt{tt})\ \ell_2\ \mathtt{x = x + 1\ ;}]\!]\,\overline{\rho}_0\, X\, \ell' \quad &= \quad (\!|\ \ell' = \ell_1\ \text{?}\ \overline{\rho}_0 \sqcup^i X(\ell_1)[x \leftarrow X(\ell_1)(\mathtt{x}) \oplus^i [1,1]])\\
&\qquad [\!]\ \ell' = \ell_2\ \text{?}\ X(\ell_1)\\
&\qquad \text{:}\ /\star\ \ \ell' = \ell_3\ \ \star/\ \ \mathtt{x} \in \mathbb{V} \mapsto \bot^i\,|\!)\\
[\ell_1, h_1] \oplus^i [\ell_2, h_2] \quad &= \quad [\ell_1 + \ell_2, h_1 + h_2]
\end{aligned}$$

- Assume that initially $\overline{\rho}_0(\mathsf{x}) = [0,0]$ and let $x = X(\ell_1)(\mathsf{x})$. The fixpoint computation amounts to solving the fixpoint equation

$$x \quad = \quad \mathscr{F}^{i}(x) \quad \text{where} \quad \mathscr{F}^{i}(x) \quad = \quad [0,0] \sqcup^{i} (x \oplus^{i} [1,1])$$

Let us solve iteratively.

$$
\begin{array}{rcllllll}
x^0 & = & \bot^{i} \\
x^1 & = & \mathscr{F}^{i}(x^0) & = & [0,0] \sqcup^{i} (x^0 \oplus^{i} [1,1]) & = & [0,0] \\
x^2 & = & \mathscr{F}^{i}(x^1) & = & [0,0] \sqcup^{i} (x^1 \oplus^{i} [1,1]) & = & [0,0] \sqcup^{i} [1,1] & = \quad [0,1] \\
\ldots \\
x^n & = & [0, n-1] & & & & & \text{induction hypothesis} \\
x^{n+1} & = & \mathscr{F}^{i}(x^n) & = & [0,0] \sqcup^{i} (x^n \oplus^{i} [1,1]) \\
 & & & = & [0,0] \sqcup^{i} [1,n] \quad = \quad [0,(n+1)-1] \\
\ldots \\
x^{\omega} & = & \bigsqcup^{i}_{n \in \mathbb{N}} [0, n-1] \quad = \quad [0, \infty] & & & & & \text{limit} \\
x^{\omega+1} & = & \mathscr{F}^{i}(x^{\omega}) & = & [0,0] \sqcup^{i} (x^{\omega} \oplus^{i} [1,1]) & = & [0,0] \sqcup^{i} [1, \infty + 1] & = \quad [0, \infty] \\
 & = & \mathsf{lfp}^{\sqsubseteq^{i}}\, x \mapsto [0,0] \sqcup^{i} (x \oplus^{i} [1,1])
\end{array}
$$

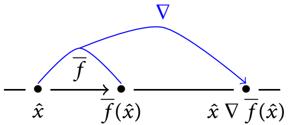
Non convergence

- Unfortunalely computerized methods to infer induction hypotheses, to simplify the iteration terms, and to pass to the limit are not effective.
- We soundly automatize the induction and passage to the limit at the price of a loss of precision to enforce rapid convergence. This is the purpose of widenings and narrowings.

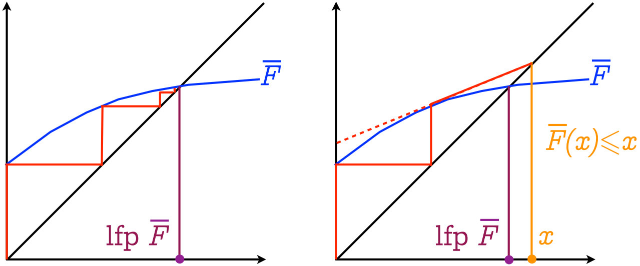
Fixpoint extrapolation (widening) and interpolation (narrowing)

Widening

- The idea of the widening is to extrapolate from an iterate x^n and the next one x^{n+1} to an upper bound $x^n \nabla x^{n+1}$ so as accelerate or enforce the convergence of the iterates in finitely many steps.
- This is an extrapolation



- The price to be paid is a loss of precision



Interval widening

Let us consider for example the interval widening

$$
\begin{aligned}
\bot^i \,\nabla^i\, x \triangleq x \,\nabla^i\, \bot^i \quad &\triangleq \quad x \\
[\ell_1, h_1] \,\nabla^i\, [\ell_2, h_2] &\triangleq [\llparenthesis\, \ell_2 < \ell_1 \mathrel{?} -\infty \mathrel{\raisebox{0.2ex}{$\circ$}} \ell_1 \,\rrparenthesis, \llparenthesis\, h_2 > h_1 \mathrel{?} \infty \mathrel{\raisebox{0.2ex}{$\circ$}} h_1 \,\rrparenthesis]
\end{aligned}
\tag{31.4}
$$

that essentially pushes unstable bounds to infinity.

Example of loss of precision by widening

$$\texttt{P}^{1001} \quad = \quad \begin{array}{l}\textbf{while}\ \ell_1\ \texttt{(x<1001)}\\ \quad \ell_2\ \texttt{x = x+1 ;}\\ \ell_3\end{array}$$

Assuming initially $\overline{\rho}_0(\mathtt{x}) = [0,0]$, $x = X(\ell_1)(\mathtt{x})$, and $y = X(\ell_3)(\mathtt{x})$, the fixpoint computation amounts to solving the fixpoint system of equations

$$\begin{cases} x &= \mathscr{F}^i(x) \\ y &= x \sqcap^i [1001, \infty] \end{cases} \quad \text{where} \quad \mathscr{F}^i(x) = [0,0] \sqcup^i ((x \sqcap^i [-\infty, 1000]) \oplus^i [1,1])$$

Example of loss of precision by widening (cont'd)

The upward iterates with widening are now

$$
\begin{array}{rcl}
\hat{x}^0 & = & \bot^i, \quad \hat{y}^0 \;=\; \bot^i \\
\hat{x}^1 & = & \hat{x}^0 \,\nabla^i\, \mathscr{F}^i(\hat{x}^0) \;=\; \hat{x}^0 \,\nabla^i\, ([0,0] \sqcup^i ((\hat{x}^0 \sqcap^i [-\infty,1000]) \oplus^i [1,1])) \\
 & = & \bot^i \,\nabla^i\, [0,0] \;=\; [0,0] \qquad \text{since } \mathscr{F}^i(\hat{x}^0) \;=\; [0,0] \not\sqsubseteq^i \bot^i \;=\; \hat{x}^0 \\
\hat{x}^2 & = & \hat{x}^1 \,\nabla^i\, \mathscr{F}^i(\hat{x}^1) \;=\; \hat{x}^1 \,\nabla^i\, ([0,0] \sqcup^i ((\hat{x}^1 \sqcap^i [-\infty,1000]) \oplus^i [1,1])) \\
 & = & [0,0] \,\nabla^i\, ([0,0] \sqcup^i [1,1]) \;=\; [0,0] \,\nabla^i\, [0,1] \\
 & = & [0,\infty] \qquad \text{since } \mathscr{F}^i(\hat{x}^1) \;=\; [0,1] \not\sqsubseteq^i \hat{x}^1 \;=\; [0,0] \\
\hat{x}^n & = & \hat{x}^2, \qquad n \geqslant 2 \\
 & & \text{since } \mathscr{F}^i(\hat{x}^2) \;=\; ([0,0] \sqcup^i ((\hat{x}^2 \sqcap^i [-\infty,1000]) \oplus^i [1,1])) \\
 & & \qquad = ([0,0] \sqcup^i (([0,\infty] \sqcap^i [-\infty,1000]) \oplus^i [1,1])) \\
 & & \qquad = ([0,0] \sqcup^i [1,1001]) \;=\; [0,1001] \;\sqsubseteq^i\; \hat{x}^2 \;=\; [0,\infty] \\
\hat{y} & = & \hat{x}^2 \sqcap^i [1001,\infty] \;=\; [0,\infty] \sqcap^i [1001,\infty] \;=\; [1001,\infty]
\end{array}
$$

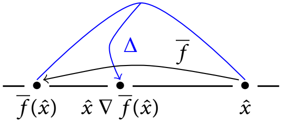
Improving the solution

- The solution found is therefore $\hat{x} = [0, \infty]$ and $= \hat{y} = [1001, \infty]$
- This is frustrating since $\mathscr{F}^i(\hat{x}) = [0, 1001]$ provides a better solution.
- We can improve the solution by a decreasing iteration
- This iteration may be infinite or very long for intervals, we stop it by a narrowing

Interval narrowing

$$\bot^i \,\Delta^i\, x \;\triangleq\; x \,\Delta^i\, \bot^i \;\;\triangleq\;\; \bot^i \qquad (31.6)$$
$$[\ell_1, h_1] \,\Delta^i\, [\ell_2, h_2] \;\triangleq\; [\llparenthesis\, \ell_1 = -\infty \;?\; \ell_2 \;:\; \ell_1 \,\rrparenthesis, \llparenthesis\, h_1 = \infty \;?\; h_2 \;:\; h_1 \,\rrparenthesis]$$

which attempts to improve infinite bounds only. This is an interpolation

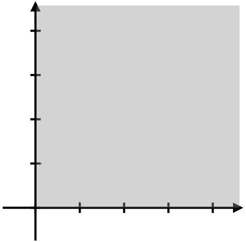


Downward iterates with narrowing

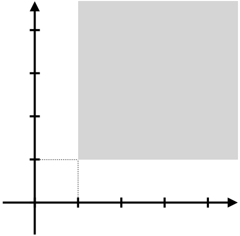
$$
\begin{array}{rcl}
\check{x}^0 & = & \hat{x} \;=\; [0,\infty], \quad \check{y} \;=\; \hat{y} \;=\; [1001,\infty] \\
\check{x}^1 & = & \check{x}^0 \,\Delta^i\, \mathscr{F}^i(\check{x}^0) \;=\; \check{x}^0 \,\Delta^i\, ([0,0] \sqcup^i ((\check{x}^0 \sqcap^i [-\infty,1000]) \oplus^i [1,1])) \\
 & = & [0,\infty] \,\Delta^i\, [0,1001] \;=\; [0,1001] \\
 & & \qquad \text{since } \mathscr{F}^i(\check{x}^0) \;=\; [0,1001] \;\neq\; [0,\infty] \;=\; \check{x}^0 \\
\check{x}^n & = & \check{x}^1, \qquad n \geqslant 1 \\
 & & \text{since } \mathscr{F}^i(\check{x}^1) \;=\; ([0,0] \sqcup^i ((\check{x}^2 \sqcap^i [-\infty,1000]) \oplus^i [1,1])) \\
 & & \qquad = \; ([0,0] \sqcup^i (([0,1001] \sqcap^i [-\infty,1000]) \oplus^i [1,1])) \\
 & & \qquad = \; ([0,0] \sqcup^i [1,1001]) \;=\; [0,1001] \;=\; \check{x}^1 \\
\check{y} & = & \check{x}^1 \sqcap^i [1001,\infty] \;=\; [0,1001] \sqcap^i [1001,\infty] \;=\; [1001,1001].
\end{array}
$$

Examples of static analyzes

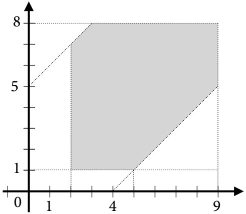
Examples of abstract domains



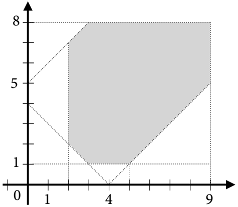
signs



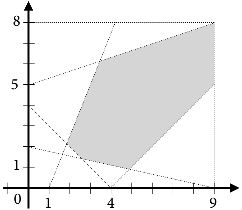
intervals



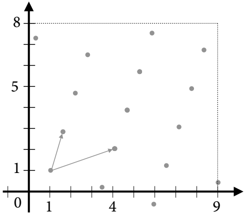
zones



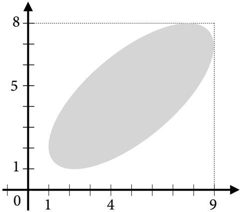
octagons



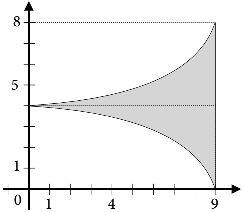
polyhedra



congruences



ellipses



exponentials

Example of octagon analysis

```
l1:  {T} i = 0;
while l2:  (i < n) {i>=0}
  l3:  {i>=0, i<=n-1} i = (i + 1);
l4:  {i>=0, i>=n}
```

Conclusion

Conclusion

- Static analysis is undecidable

  i.e. no terminating algorithm can always automatically analyze correctly any program with best possible precision

- Abstract interpretation theory can be used to build static analyzers that are
  - fully automatic (no human intervention needed)
  - always terminating
  - always sound/correct

  but

  - may sometimes be imprecise

- example: `Astrée` (`https://www.absint.com/astree/index.htm`)

Conclusion

- This light introduction to abstract interpretation should be sufficient to follow the invited talk "Calculational design of a regular model checker by abstract interpretation" on November 2, 2019, 9:00–10:30
- Reading these slides by yourself can be helpful
- These slides are available at
  https://cs.nyu.edu/~pcousot/summerschools/ICTAC-2029/Cousot-tutorial.pdf
- I will attend the tutorials and conference, so I am available at any time for questions, don't hesitate!

Other online resources

- MIT course web.mit.edu/16.399/
- NYU course https://cs.nyu.edu/~pcousot/courses/spring19/CSCI-GA.3140-001 (send me an email at `pcousot@cs.nyu.edu` to get access)

Bibliography

Basic references I

Bertrane, Julien, Patrick Cousot, Radhia Cousot, Jérôme Feret, Laurent Mauborgne, Antoine Miné, and Xavier Rival (2015). "Static Analysis and Verification of Aerospace Software by Abstract Interpretation". *Foundations and Trends in Programming Languages* 2.2-3, pp. 71–190.

Cousot, Patrick (1999). "The Calculational Design of a Generic Abstract Interpreter". In: M. Broy and R. Steinbrüggen, eds. *Calculational System Design*. NATO ASI Series F. IOS Press, Amsterdam.

– (2015). "Abstracting Induction by Extrapolation and Interpolation". In: *VMCAI*. Vol. 8931. Lecture Notes in Computer Science. Springer, pp. 19–42.

Cousot, Patrick and Radhia Cousot (1977). "Abstract Interpretation: A Unified Lattice Model for Static Analysis of Programs by Construction or Approximation of Fixpoints". In: *POPL*. ACM, pp. 238–252.

– (1979). "Systematic Design of Program Analysis Frameworks". In: *POPL*. ACM Press, pp. 269–282.

The End, Thank you