

Principles of Abstract Interpretation

MIT press

Ch. 3, Syntax, semantics, properties, and static analysis of expressions

Patrick Cousot

pcousot.github.io

PrAbsInt@gmail.com github.com/PrAbsInt/

These slides are available at
<http://github.com/PrAbsInt/slides/slides/slides-03--expressions-PrAbsInt.pdf>

Chapter 3

Ch. 3, Syntax, semantics, properties, and static analysis of expressions

The objective of this [Chapter 3 \(Syntax, semantics, properties, and static analysis of expressions\)](#) is to [introduce abstract interpretation](#) using an extremely simple example: the [rule of signs](#)

Product of two integers

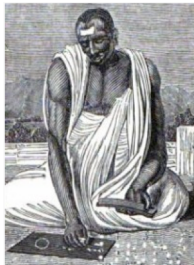
·	−	+
−	+	−
+	−	+

In words, we have:

- Minus times Minus gives Plus
- Minus times Plus gives Minus
- Plus times Minus gives Minus
- Plus times Plus gives Plus

[en.wikipedia.org/wiki/Product_\(mathematics\)](https://en.wikipedia.org/wiki/Product_(mathematics))

Brahmagupta



- Brahmagupta (born c. 598 CE¹, died after 665 CE) was an Indian mathematician and astronomer;
- Invented the rule of signs (including to compute with zero);
- Probably the very first recorded historical example of abstract interpretation :)

en.wikipedia.org/wiki/Brahmagupta

¹Common Era

Syntax of expressions

Syntax of expressions

$x, y, \dots \in \mathcal{V}$	variables (\mathcal{V} not empty)
$A \in \mathcal{A} ::= 1 \mid x \mid A_1 - A_2$	arithmetic expressions
$B \in \mathcal{B} ::= A_1 < A_2 \mid B_1 \text{ nand } B_2$	boolean expressions
$E \in \mathcal{E} ::= A \mid B$	expressions

This is an example of *context-free grammar*.

Binary operators are left associative and arithmetic operators have priority over boolean operators (so $1 - 1 < 1 - 1 - 1$ is $((1 - 1) < ((1 - 1) - 1))$ *i.e.* false ff).

[en.wikipedia.org/wiki/Syntax_\(programming_languages\)](https://en.wikipedia.org/wiki/Syntax_(programming_languages))
en.wikipedia.org/wiki/Context-free_grammar

Semantics of expressions

Environment

- The value of an expression depends on the value of the free variables e.g.
 $x - 1$ is 2 when $x = 3$, $x - 1$ is 42 when $x = 43$, etc.;
- We cannot enumerate the infinitely many cases;
- The computer uses values of variables stored in **memory**;
- The evaluation of expressions by the computer can be explained independently of the memory content;
- We formalize the memory by **environments** assigning values to variables (**assignments** in logic);
- An environment

$$\rho \in \mathcal{V} \rightarrow \mathbb{Z}$$

is a total function ρ mapping a variable $x \in \mathcal{V}$ to its integer value $\rho(x) \in \mathbb{Z}$;

en.wikipedia.org/wiki/Typing_environment

[en.wikipedia.org/wiki/Valuation_\(logic\)](https://en.wikipedia.org/wiki/Valuation_(logic))

Semantics of expressions

$$\begin{aligned}
 \mathcal{A} \llbracket 1 \rrbracket \rho &\triangleq 1 \\
 \mathcal{A} \llbracket x \rrbracket \rho &\triangleq \rho(x) \\
 \mathcal{A} \llbracket A_1 - A_2 \rrbracket \rho &\triangleq \mathcal{A} \llbracket A_1 \rrbracket \rho - \mathcal{A} \llbracket A_2 \rrbracket \rho \\
 \mathcal{B} \llbracket A_1 < A_2 \rrbracket \rho &\triangleq \mathcal{A} \llbracket A_1 \rrbracket \rho < \mathcal{A} \llbracket A_2 \rrbracket \rho \\
 \mathcal{B} \llbracket B_1 \text{ nand } B_2 \rrbracket \rho &\triangleq \mathcal{B} \llbracket B_1 \rrbracket \rho \uparrow \mathcal{B} \llbracket B_2 \rrbracket \rho \\
 \mathcal{S} \llbracket E \rrbracket &\triangleq \mathcal{A} \llbracket E \rrbracket && \text{when } E \in \mathcal{A} \\
 \mathcal{S} \llbracket E \rrbracket &\triangleq \mathcal{B} \llbracket E \rrbracket && \text{when } E \in \mathcal{B}
 \end{aligned}
 \tag{3.4}$$

a	tt	tt	ff	ff
b	tt	ff	tt	ff
$a \uparrow b$	ff	tt	tt	tt

- This is an example of well-defined structural definition.
- $\mathcal{A} \llbracket A \rrbracket$ and $\mathcal{B} \llbracket B \rrbracket$ are total functions (in \mathbb{Z}), proof by structural induction.

[en.wikipedia.org/wiki/Semantics_\(computer_science\)](https://en.wikipedia.org/wiki/Semantics_(computer_science))

Semantic properties of expressions

Properties

- We represent a property by the set of elements that have this property.
- For example
 - “ x is an even natural” is “ $x \in \{0, 2, 4, \dots\}$ ”.
 - “ x is constant equal to 1” is “ $x \in \{1\}$ ”.

So a property of a natural is an element of $\wp(\mathbb{N})$.

For example

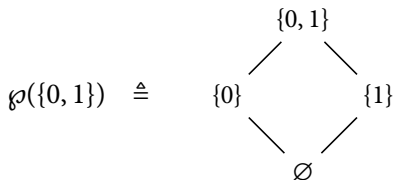
- The property $\{0, 2, 4, \dots\}$ is “to be even”.
- The property $\{1\}$ is “to be one”.

Powerset

- If S is a set then $\wp(S)$ is the *powerset* of S ,

$$\wp(S) \triangleq \{X \mid X \subseteq S\}$$

- Example: $\wp(\{0, 1\}) \triangleq \{\emptyset, \{0\}, \{1\}, \{0, 1\}\}$
- Hasse diagram:



en.wikipedia.org/wiki/Power_set

en.wikipedia.org/wiki/Hasse_diagram

Implication, weaker and stronger properties

- When considering properties as sets, **logical implication** is subset inclusion \subseteq .
- For example “to be greater than 42 implies to be positive” is $\{x \in \mathbb{Z} \mid x > 42\} \subseteq \{x \in \mathbb{Z} \mid x \geq 0\}$.
- If $P \subseteq Q$ then P is said to be **stronger/more precise** than Q and Q is said to be **weaker/less precise** than P .
- Stronger/more precise properties are satisfied by less elements while weaker/less precise properties are satisfied by more elements.
- False **ff** i.e. \emptyset is the strongest property while true **tt** i.e. \mathbb{Z} is the weakest property of integers.
- conjunction \wedge is intersection \cap and disjunction \vee is union \cup .

en.wikipedia.org/wiki/Logical_consequence

en.wikipedia.org/wiki/Subset

Semantics properties of expressions

- By property of an expression, we mean a semantic property, that is a property of its semantics;
- The **semantic** belongs to $(\mathcal{V} \rightarrow \mathbb{Z}) \rightarrow \mathbb{Z}$;
- So a **semantic property** is an element of $\wp((\mathcal{V} \rightarrow \mathbb{Z}) \rightarrow \mathbb{Z})$;
- Arithmetic expression **A** is said to have semantic property $P \in \wp((\mathcal{V} \rightarrow \mathbb{Z}) \rightarrow \mathbb{Z})$ if and only if $\mathcal{A} \llbracket A \rrbracket \in P$;
- Semantic properties **P** of expressions are just a particular case of property of expressions *i.e.* the property $\{A \in \mathcal{E} \mid \mathcal{A} \llbracket A \rrbracket \in P\}$ ².

²This will be discussed in greater details in Chapter 9 (Undecidability and Rice theorem)

Collecting semantics of expressions

Collecting semantics of expressions

- The **collecting semantics** of expressions is the strongest property of an expression.

$$\mathcal{S}^c[A] \triangleq \{\mathcal{A}[A]\} \in \wp((V \rightarrow \mathbb{Z}) \rightarrow \mathbb{Z}) \quad (3.13)$$

- Arithmetic expression A is said to have semantic property $P \in \wp((V \rightarrow \mathbb{Z}) \rightarrow \mathbb{Z})$ if and only if $\mathcal{A}[A] \in P$
- Equivalently $\mathcal{S}^c[A] \subseteq P$ (so we don't need to use \in)
- $\mathcal{S}^c[A]$ is the **strongest property** of A .

- The collecting semantics of boolean expressions is

$$\mathcal{S}^c[B] \triangleq \{\mathcal{B}[B]\} \in \wp((V \rightarrow \mathbb{Z}) \rightarrow \mathbb{B})$$

Structural collecting semantics

$$\mathcal{S}^c[\mathbf{1}] = \{\rho \in (\mathcal{V} \rightarrow \mathbb{Z}) \mapsto 1\}$$

$$\mathcal{S}^c[\mathbf{x}] = \{\rho \in (\mathcal{V} \rightarrow \mathbb{Z}) \mapsto \rho(\mathbf{x})\}$$

$$\mathcal{S}^c[\mathbf{A}_1 - \mathbf{A}_2] = \{\rho \in (\mathcal{V} \rightarrow \mathbb{Z}) \mapsto f_1(\rho) - f_2(\rho) \mid f_1 \in \mathcal{S}^c[\mathbf{A}_1] \wedge f_2 \in \mathcal{S}^c[\mathbf{A}_2]\}$$

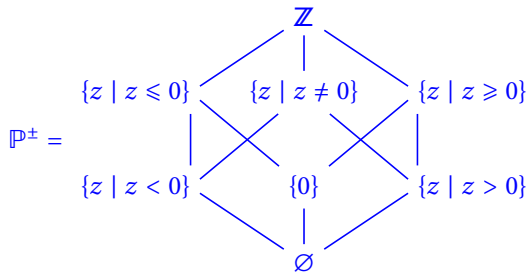
$$\mathcal{S}^c[\mathbf{A}_1 < \mathbf{A}_2] = \{\rho \in (\mathcal{V} \rightarrow \mathbb{Z}) \mapsto f_1(\rho) < f_2(\rho) \mid f_1 \in \mathcal{S}^c[\mathbf{A}_1] \wedge f_2 \in \mathcal{S}^c[\mathbf{A}_2]\}$$

$$\mathcal{S}^c[\mathbf{B}_1 \text{ nand } \mathbf{B}_2] = \{\rho \in (\mathcal{V} \rightarrow \mathbb{Z}) \mapsto f_1(\rho) \uparrow f_2(\rho) \mid f_1 \in \mathcal{S}^c[\mathbf{B}_1] \wedge f_2 \in \mathcal{S}^c[\mathbf{B}_2]\}$$

$x \mapsto t$ is the function f such that for parameter x , the value $f(x)$ of f at x is equal to the value of the term t (depending upon x). $x \in X \mapsto t$ states that f is undefined when $x \notin X$.

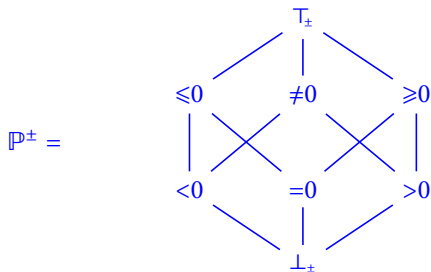
Sign abstraction

Sign property (of an individual variable)



The **Hasse diagram** for partial order \subseteq , \cup is the join, \cap is the meet, *etc.*

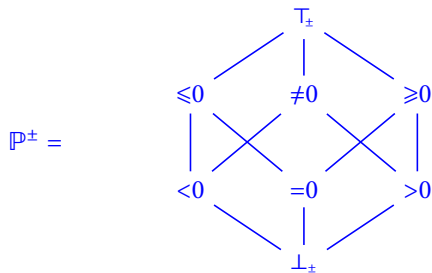
Encoding of sign properties (of an individual variable)



Concretization function:

$\gamma_\pm(\perp_\pm)$	\triangleq	\emptyset	$\gamma_\pm(\leq 0)$	\triangleq	$\{z \mid z \leq 0\}$
$\gamma_\pm(< 0)$	\triangleq	$\{z \mid z < 0\}$	$\gamma_\pm(\neq 0)$	\triangleq	$\{z \mid z \neq 0\}$
$\gamma_\pm(= 0)$	\triangleq	$\{0\}$	$\gamma_\pm(\geq 0)$	\triangleq	$\{z \mid z \geq 0\}$
$\gamma_\pm(> 0)$	\triangleq	$\{z \mid z > 0\}$	$\gamma_\pm(\top_\pm)$	\triangleq	\mathbb{Z}

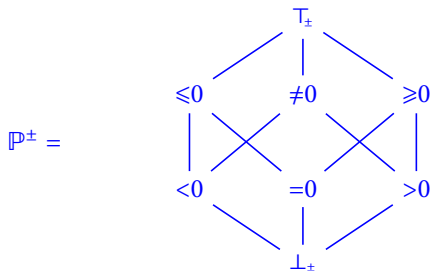
The lattice of abstract properties



The **Hasse diagram** for partial order \sqsubseteq , \sqcup is the join, \sqcap is the meet, *etc.*

e.g. $\sqcap\{\leq 0, \neq 0\} = < 0$, $\sqcap \emptyset = \top_\pm$

Encoding of sign properties (of an individual variable)



Abstraction function: $\alpha_\pm(P) \triangleq \left(\begin{array}{l} P \subseteq \emptyset \text{ ? } \perp_\pm \\ P \subseteq \{z \mid z < 0\} \text{ ? } <0 \\ P \subseteq \{0\} \text{ ? } =0 \\ P \subseteq \{z \mid z > 0\} \text{ ? } >0 \\ P \subseteq \{z \mid z \leq 0\} \text{ ? } \leq 0 \\ P \subseteq \{z \mid z \neq 0\} \text{ ? } \neq 0 \\ P \subseteq \{z \mid z \geq 0\} \text{ ? } \geq 0 \\ \text{? } \top_\pm \end{array} \right) \quad (3.30)$

Galois connection

- The pair $\langle \alpha_+, \gamma_+ \rangle$ of functions satisfies $\alpha_+(P) \sqsubseteq Q \Leftrightarrow P \subseteq \gamma_+(Q)$
- For example,

$$(\alpha_+({-2, -1})) \triangleq <0 \sqsubseteq \neq 0 \Leftrightarrow ({-2, -1} \subseteq \{z \mid z \neq 0\} \triangleq \gamma_+(\neq 0))$$

- Let us prove that we have a Galois connection between concrete and abstract properties

Galois connection

- The pair $\langle \alpha_{\pm}, \gamma_{\pm} \rangle$ of functions satisfies $\alpha_{\pm}(P) \sqsubseteq Q \Leftrightarrow P \subseteq \gamma_{\pm}(Q)$

$$\alpha_{\pm}(P) \sqsubseteq Q$$

$$\Leftrightarrow \alpha_{\pm}(P) \sqsubseteq \neq 0$$

{in case $Q = \neq 0$, other cases are similar}

$$\Leftrightarrow \alpha_{\pm}(P) \in \{\perp_{\pm}, <0, \neq 0, >0\}$$

{def. \sqsubseteq }

$$\Leftrightarrow P \subseteq \emptyset \vee P \subseteq \{z \mid z < 0\} \vee P \subseteq \{z \mid z > 0\} \vee P \subseteq \{z \mid z \neq 0\}$$

{def. α_{\pm} }

$$\Leftrightarrow P \subseteq \{z \mid z \neq 0\}$$

{def. \subseteq }

$$\Leftrightarrow P \subseteq \gamma_{\pm}(\neq 0)$$

{def. γ_{\pm} }

$$\Leftrightarrow P \subseteq \gamma_{\pm}(Q)$$

{case $Q = \neq 0$ }

- This is the definition of a **Galois connection**

- We write $\langle \wp(\mathbb{Z}), \subseteq \rangle \xrightleftharpoons[\alpha_{\pm}]{\gamma_{\pm}} \langle \mathbb{P}^{\pm}, \sqsubseteq \rangle$

- This will be further generalized.

en.wikipedia.org/wiki/Galois_connection
en.wikipedia.org/wiki/Évariste_Galois

Sign abstract semantics

$$\begin{aligned}
 \mathcal{S}[[A]] &\in (V \rightarrow \mathbb{P}^\pm) \rightarrow \mathbb{P}^\pm & (3.21) \\
 \mathcal{S}[[1]]P &\triangleq >0 \\
 \mathcal{S}[[x]]P &\triangleq P(x) \\
 \mathcal{S}[[A_1 - A_2]]P &\triangleq \mathcal{S}[[A_1]]P \neg_\pm \mathcal{S}[[A_2]]P
 \end{aligned}$$

$x \neg_{\pm} y$		y								
		\perp_{\pm}	<0	$=0$	>0	≤ 0	$\neq 0$	≥ 0	\top_{\pm}	
x	\perp_{\pm}	\perp_{\pm}	\perp_{\pm}	\perp_{\pm}	\perp_{\pm}	\perp_{\pm}	\perp_{\pm}	\perp_{\pm}	\perp_{\pm}	
	<0	\perp_{\pm}	\top_{\pm}	<0	<0	\top_{\pm}	\top_{\pm}	<0	\top_{\pm}	
	$=0$	\perp_{\pm}	>0	$=0$	<0	≥ 0	$\neq 0$	≤ 0	\top_{\pm}	
	>0	\perp_{\pm}	>0	>0	\top_{\pm}	>0	\top_{\pm}	\top_{\pm}	\top_{\pm}	
	≤ 0	\perp_{\pm}	>0	≤ 0	\top_{\pm}	\top_{\pm}	\top_{\pm}	≤ 0	\top_{\pm}	
	$\neq 0$	\perp_{\pm}	\top_{\pm}	$\neq 0$	\top_{\pm}	\top_{\pm}	\top_{\pm}	\top_{\pm}	\top_{\pm}	
	≥ 0	\perp_{\pm}	>0	≥ 0	\top_{\pm}	≥ 0	\top_{\pm}	\top_{\pm}	\top_{\pm}	
	\top_{\pm}	\perp_{\pm}	\top_{\pm}	\top_{\pm}	\top_{\pm}	\top_{\pm}	\top_{\pm}	\top_{\pm}	\top_{\pm}	

This is a specification of an abstract interpreter I

```
type aexpr = One | Var of string | Minus of aexpr * aexpr;;
```

```
let bot = 0 and neg = 1 and is0 = 2 and pos = 3 and  
    neg0 = 4 and not0 = 5 and pos0 = 6 and top = 7;;
```

```
let print s = match s with  
    0 -> "bot" | 1 -> "neg" | 2 -> "is0" | 3 -> "pos" |  
    4 -> "neg0" | 5 -> "not0" | 6 -> "pos0" | 7 -> "top" |  
    _ -> failwith "incorrect sign";;
```

```
let minus= [| [|bot; bot; bot; bot; bot; bot; bot; bot|];  
    [|bot; top; neg; neg; top; top; neg; top|];  
    [|bot; pos; is0; neg; pos0; not0; neg0; top|];  
    [|bot; pos; pos; top; pos; top; top; top|];  
    [|bot; pos; neg0; top; top; top; neg0; top|];  
    [|bot; top; not0; top; top; top; top; top|];  
    [|bot; pos; pos0; top; pos0; top; top; top|];  
    [|bot; top; top; top; top; top; top; top|];  
|];;
```

This is a specification of an abstract interpreter II

```
type environment = (string * int) list;;

let rec sign a r = match a with
| One -> pos
| Var x -> List.assoc x r
| Minus (a1, a2) -> minus.(sign a1 r).(sign a2 r);;

- : aexpr -> environment -> int = <fun>

let r = [("x",pos); ("y",neg)];;

print (sign (Minus ((Var "x"),(Var "y"))) r);;

- : string = "pos"
```

Computational design of the rule of signs

$$\begin{aligned} & >0 \text{ }_{\pm} \leq 0 \\ \triangleq & \alpha_{\pm}(\{x - y \mid x \in \gamma_{\pm}(>0) \wedge y \in \gamma_{\pm}(\leq 0)\}) \\ = & \alpha_{\pm}(\{x - y \mid x > 0 \wedge y \leq 0\}) \\ = & \alpha_{\pm}(\{z \mid z > 0\}) \\ & \left\{ \begin{array}{l} \text{for } \subseteq, x > 0 \wedge y \leq 0 \Rightarrow x - y > 0; \\ \text{for } \supseteq \text{ if } z > 0 \text{ then take } x = z \text{ and } y = 0 \text{ so } z \in \{x - y \mid x > 0 \wedge -y \geq 0\} \end{array} \right\} \\ = & >0 \end{aligned}$$

Same calculus for all other cases (can be automated with a theorem prover).

Soundness

Sign concretization

- Sign

$$\begin{array}{ll} \gamma_{\pm}(\perp_{\pm}) & \triangleq \emptyset \\ \gamma_{\pm}(<0) & \triangleq \{z \in \mathbb{Z} \mid z < 0\} \\ \gamma_{\pm}(=0) & \triangleq \{0\} \\ \gamma_{\pm}(> 0) & \triangleq \{z \in \mathbb{Z} \mid z > 0\} \end{array} \quad \begin{array}{ll} \gamma_{\pm}(\leq 0) & \triangleq \{z \in \mathbb{Z} \mid z \leq 0\} \\ \gamma_{\pm}(\neq 0) & \triangleq \{z \in \mathbb{Z} \mid z \neq 0\} \\ \gamma_{\pm}(\geq 0) & \triangleq \{z \in \mathbb{Z} \mid z \geq 0\} \\ \gamma_{\pm}(\top_{\pm}) & \triangleq \mathbb{Z} \end{array} \quad (3.23)$$

- Sign environment

$$\dot{\gamma}_{\pm}(\dot{\rho}) \triangleq \{\rho \in \mathcal{V} \rightarrow \mathbb{Z} \mid \forall x \in \mathcal{V} . \rho(x) \in \gamma_{\pm}(\dot{\rho}(x))\} \quad (3.24)$$

- Sign abstract property

$$\ddot{\gamma}_{\pm}(\overline{P}) \triangleq \{\mathcal{S} \in (\mathcal{V} \rightarrow \mathbb{Z}) \rightarrow \mathbb{Z} \mid \forall \dot{\rho} \in \mathcal{V} \rightarrow \mathbb{P}^{\pm} . \forall \rho \in \dot{\gamma}_{\pm}(\dot{\rho}) . \mathcal{S}(\rho) \in \gamma_{\pm}(\overline{P}(\dot{\rho}))\} \quad (3.25)$$

Sign abstraction

- Value property

$$\alpha_{\pm}(P) \triangleq \begin{aligned} & \llbracket P \subseteq \emptyset \text{ ? } \perp_{\pm} \\ & \parallel P \subseteq \{z \mid z < 0\} \text{ ? } < 0 \\ & \parallel P \subseteq \{0\} \text{ ? } = 0 \\ & \parallel P \subseteq \{z \mid z > 0\} \text{ ? } > 0 \\ & \parallel P \subseteq \{z \mid z \leq 0\} \text{ ? } \leq 0 \\ & \parallel P \subseteq \{z \mid z \neq 0\} \text{ ? } \neq 0 \\ & \parallel P \subseteq \{z \mid z \geq 0\} \text{ ? } \geq 0 \\ & \circ \top_{\pm} \rrbracket \end{aligned} \quad (3.30)$$

- Environment property

$$\dot{\alpha}_{\pm}(P) \triangleq x \in \mathcal{V} \mapsto \alpha_{\pm}(\{\rho(x) \mid \rho \in P\}) \quad (3.33)$$

- Semantics property

$$\ddot{\alpha}_{\pm}(P) \triangleq \overset{\pm}{\rho} \in \mathcal{V} \rightarrow \mathbb{P}^{\pm} \mapsto \alpha_{\pm}(\{\mathcal{S}(\rho) \mid \mathcal{S} \in P \wedge \rho \in \dot{\gamma}_{\pm}(\overset{\pm}{\rho})\}) \quad (3.34)$$

Example of environment property abstraction

- The property of environments such that x is equal to 1:

$$\{\rho \in \mathcal{V} \rightarrow \mathbb{Z} \mid \rho(x) = 1\}$$

- Sign abstraction:

$$\alpha_{\pm}(\{\rho \in \mathcal{V} \rightarrow \mathbb{Z} \mid \rho(x) = 1\})$$

$$\triangleq y \in \mathcal{V} \mapsto \alpha_{\pm}(\{\rho(y) \mid \rho \in \{\rho \in \mathcal{V} \rightarrow \mathbb{Z} \mid \rho(x) = 1\}\})$$

$$= y \in \mathcal{V} \mapsto ([y = x \text{ ? } \alpha_{\pm}(\{1\}) : \alpha_{\pm}(\mathbb{Z})])$$

$$= y \in \mathcal{V} \mapsto ([y = x \text{ ? } >0 : \top_{\pm}])$$

- Sign concretization:

$$\gamma_{\pm}(y \in \mathcal{V} \mapsto ([y = x \text{ ? } >0 : \top_{\pm}]))$$

$$\triangleq \{\rho \in \mathcal{V} \rightarrow \mathbb{Z} \mid \forall z \in \mathcal{V} . \rho(z) \in \gamma_{\pm}(y \in \mathcal{V} \mapsto ([y = x \text{ ? } >0 : \top_{\pm}]))(z)\}$$

$$= \{\rho \in \mathcal{V} \rightarrow \mathbb{Z} \mid \rho(x) > 0\}$$

Galois connections

- Value to sign

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

- Value environment to sign environment

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

- Semantic to sign abstract semantic property

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

Soundness of the abstract sign semantics

- The abstract sign semantics is an abstraction of the collecting property

$$\begin{aligned} \mathcal{S}^c[A] &\subseteq \gamma_{\pm}(\mathcal{S}^{\pm}[A]) \\ \Leftrightarrow \alpha_{\pm}(\mathcal{S}^c[A]) &\sqsubseteq \mathcal{S}^{\pm}[A] \end{aligned}$$

- Precision loss: if the sign of x is ≤ 0 then the sign of $x - x$ is \top_{\pm} not $=0$
- The absolute value is abstracted away
- No precision loss for multiplication \times

en.wikipedia.org/wiki/Soundness

Next objective ...

Now that we have defined the **collecting semantics** $\mathcal{S}^c \llbracket A \rrbracket \in \wp((V \rightarrow \mathbb{Z}) \rightarrow \mathbb{Z})$

$$\mathcal{S}^c \llbracket 1 \rrbracket = \{\rho \in (V \rightarrow \mathbb{Z}) \mapsto 1\}$$

$$\mathcal{S}^c \llbracket x \rrbracket = \{\rho \in (V \rightarrow \mathbb{Z}) \mapsto \rho(x)\}$$

$$\mathcal{S}^c \llbracket A_1 - A_2 \rrbracket = \{\rho \in (V \rightarrow \mathbb{Z}) \mapsto f_1(\rho) - f_2(\rho) \mid f_1 \in \mathcal{S}^c \llbracket A_1 \rrbracket \wedge f_2 \in \mathcal{S}^c \llbracket A_2 \rrbracket\}$$

and the **sign abstraction**

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

value properties

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

environment properties

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

semantic properties

we are ready to calculate the **sign abstract semantics** $\mathcal{S}^{\pm} \llbracket A \rrbracket \in (V \rightarrow \mathbb{P}^{\pm}) \rightarrow \mathbb{P}^{\pm}$ by over approximation of the collecting semantics

$$\ddot{\alpha}_{\pm}(\mathcal{S}^c \llbracket A \rrbracket) \sqsubseteq \mathcal{S}^{\pm} \llbracket A \rrbracket$$

This sign abstract semantics is a specification of the **sign static analyzer**.

Computational design of the sign semantics

Case of a variable x

$$\begin{aligned}
 & \ddot{\alpha}_{\pm}(\mathcal{S}^c \llbracket x \rrbracket) \\
 = & \alpha_{\pm}(\{\mathcal{S}(\rho) \mid \mathcal{S} \in \mathcal{S}^c \llbracket x \rrbracket \wedge \rho \in \dot{\gamma}_{\pm}(\dot{\rho})\}) && \{ \text{def. (3.34) of } \ddot{\alpha}_{\pm} \} \\
 = & \alpha_{\pm}(\{\mathcal{A} \llbracket x \rrbracket(\rho) \mid \rho \in \dot{\gamma}_{\pm}(\dot{\rho})\}) && \{ \text{def. (3.13) of } \mathcal{S}^c \llbracket x \rrbracket \} \\
 = & \alpha_{\pm}(\{\rho(x) \mid \rho \in \dot{\gamma}_{\pm}(\dot{\rho})\}) && \{ \text{def. (3.4) of } \mathcal{A} \llbracket x \rrbracket \} \\
 = & \alpha_{\pm}(\{\rho(x) \mid \forall y \in V . \rho(y) \in \gamma_{\pm}(\dot{\rho}(y))\}) && \{ \text{def. (3.24) of } \dot{\gamma}_{\pm} \} \\
 \sqsubseteq & \alpha_{\pm}(\{\rho(x) \mid \rho(x) \in \gamma_{\pm}(\dot{\rho}(x))\}) \\
 & \{ \text{if } y = x, \text{ the condition } \rho(x) \in \gamma_{\pm}(\dot{\rho}(x)) \text{ is the same;} \\
 & \text{if } y \neq x \text{ the condition } \rho(y) \in \gamma_{\pm}(\dot{\rho}(y)) \text{ is disregarded;} \\
 & \text{So the set } \{\rho(x) \mid \rho(x) \in \gamma_{\pm}(\dot{\rho}(x))\} \text{ is larger and } \alpha_{\pm} \text{ is increasing} \} \\
 = & \alpha_{\pm}(\{x \mid x \in \gamma_{\pm}(\dot{\rho}(x))\}) && \{ \text{letting } x = \rho(x) \} \\
 = & \alpha_{\pm}(\gamma_{\pm}(\dot{\rho}(x))) && \{ \text{since } S = \{x \mid z \in S\} \text{ for any set } S \} \\
 = & \dot{\rho}(x) && \{ \text{since } \alpha_{\pm} \circ \gamma_{\pm} \text{ is the identity} \} \\
 \triangleq & \mathcal{S}^{\pm} \llbracket x \rrbracket^{\dot{\rho}} && \{ \text{in accordance with (3.21)} \}
 \end{aligned}$$

Other cases

- similar for $\ddot{\alpha}_{\pm}(\mathcal{S}^c[[1]])^{\pm}$
- by structural induction for $\ddot{\alpha}_{\pm}(\mathcal{S}^c[[A_1 - A_2]])$
- See the book [Cousot, 2021] for more details.

Extension to programs

Automatic static sign program analysis

```
#include <stdio.h>
int main () {
    int x;
    scanf("%d",&x);
1:
    while 2: (x>0) {
3:
        x = x-1;
4:
    }
5:    printf("%d\n",x);
    return x;
}
```

What is the sign of x when printing?

Conclusion

Conclusion I

- We have **formally defined** the semantics of expressions, their properties, their collecting semantics, the sign abstraction, and designed, by calculus, a sign analysis that we have implemented.
- Of course the rule of signs looks trivial, but one can get it wrong! [Sintzoff, 1972]
- The sign analysis is not very precise, but Section **34.11** shows that it is always possible to use **infinite abstractions** to guarantee more precise results³.
- For another **informal introduction** to abstract interpretation, you can read [P. Cousot and R. Cousot, 2010]

³e.g. Chapter 33 (Static interval analysis) for signs.

Bibliography

Cousot, Patrick (2021). *Principles of Abstract Interpretation*. 1st ed. MIT Press.

P. Cousot and R. Cousot (2010). “A gentle introduction to formal verification of computer systems by abstract interpretation”. In: *Logics and Languages for Reliability and Security*. Ed. by J. Esparza, O. Grumberg, and M. Broy. NATO Science Series III: Computer and Systems Sciences. IOS Press, pp. 1–29.

Sintzoff, Michel (1972). “Calculating Properties of Programs by Valuations on Specific Models”. In: *Proceedings of ACM Conference on Proving Assertions About Programs*. ACM, pp. 203–207.

Home work

Read Ch. 3 “Syntax, semantics, properties, and static analysis of expressions” of

Principles of Abstract Interpretation

Patrick Cousot

MIT Press

The End, Thank you