Static analysis of finite precision computations

Eric Goubault and Sylvie Putot

Laboratory for the Modelling and Analysis of Interacting Systems, CEA LIST

January 25, VMCAI 2011, Austin, Texas

Motivations

- Validate numerical programs
  - Check/infer invariant properties both on the floating-point number and on the real number semantics
  - Find out the difference between the two semantics and its origin in the program
- In particular:
  - range of program variables: the difference between the semantics characterizes that the program computes something close to what is expected
    - accuracy of results
    - behaviour of the program (control flow)
  - check functional properties
    - method error (real-number semantics)
    - implementation error (difference between semantics)

Example: Householder scheme for square root approx

Householder

Outline of the talk

- Modelling finite precision computations
- Affine sets for zonotopic abstraction of real computations
- Zonotopic abstraction of finite precision computations
- Example with the Fluctuat static analyzer

Concrete semantics

- Aim: compute rounding errors and their propagation
  - we need the floating-point values
  - relational (thus accurate) analysis more natural on real values
  - for each variable, we compute (f^x, r^x, e^x)

```
float x,y,z;
x = 0.1; // [1]
y = 0.5; // [2]
z = x+y; // [3]
t = x*z; // [4]
```

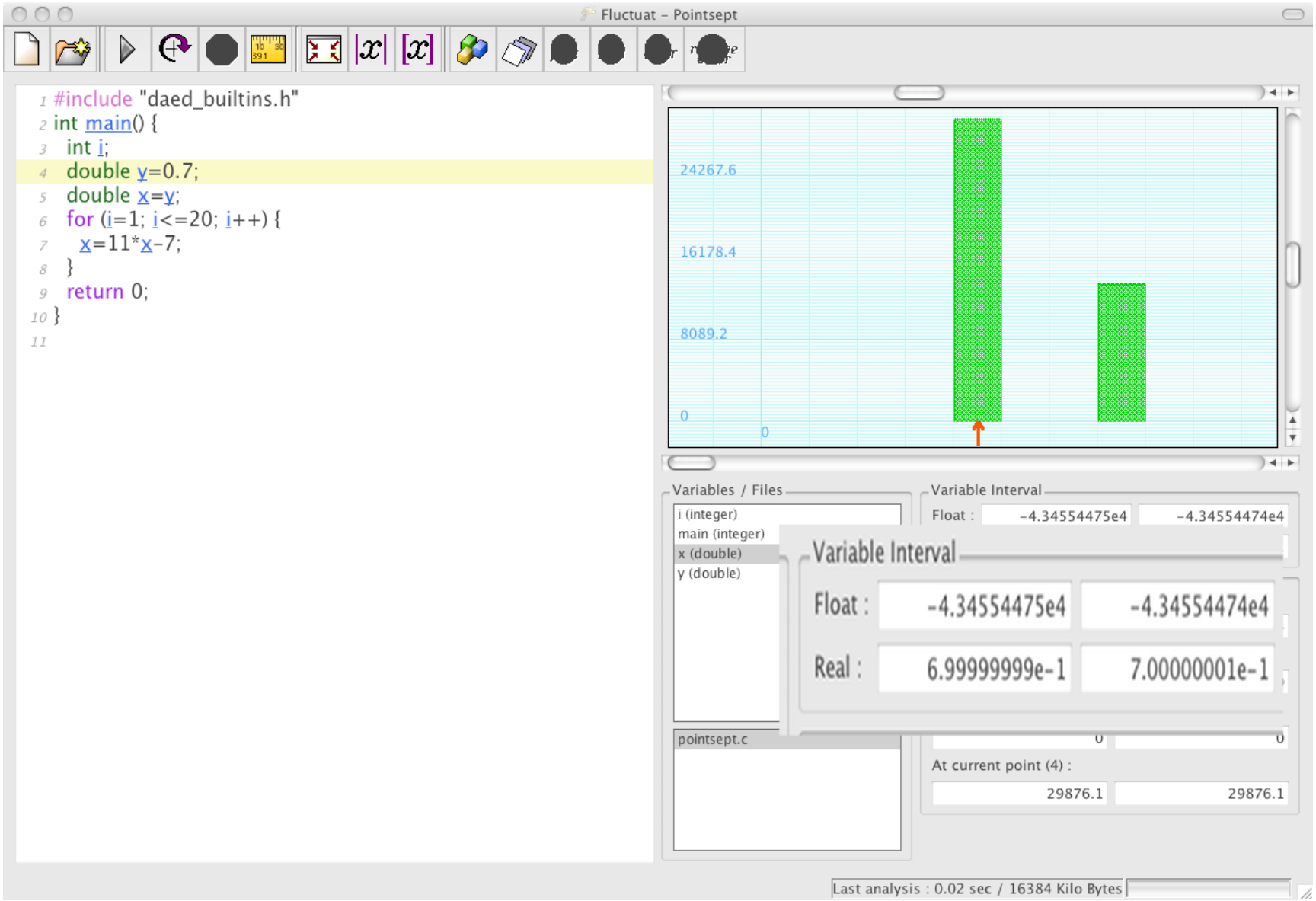
$$f^x = 0.1 + 1.49e^{-9}\ [1]$$

$$f^y = 0.5$$

$$f^z = 0.6 + 1.49e^{-9}\ [1] + 2.23e^{-8}\ [3]$$

$$f^t = 0.06 + 1.04e^{-9}\ [1] + 2.23e^{-9}\ [3] - 8.94e^{-10}\ [4] - 3.55e^{-17}\ [ho]$$

Example (Fluctuat)



Towards an abstract model

- IEEE norm allows to prove properties on programs using f.p. numbers:
  - elementary rounding error when rounding r^x to $\uparrow_\circ r^x$: there exist $\delta_r > 0$ and $\delta_a > 0$

$$|r^x - \uparrow_\circ r^x| \leq max(\delta_r | \uparrow_\circ r^x|, \delta_a)$$

  - the f.p. result of arithmetic elementary operations $+, -, \times, /, \sqrt{}$ is the rounded value of the real result

- For each variable x, a triplet (f^x, r^x, e^x):
  - r^x is an abstraction of the real (ideal) value of x
  - e^x is an abstraction of the difference, i.e. of initial or rounding errors and their propagation in computations
  - f^x is an abstraction of the machine (finite precision) value of x

Outline of the talk

- Modelling finite precision computations
- Affine sets for zonotopic abstraction of real computations
- Zonotopic abstraction of finite precision computations
- Example with the Fluctuat static analyzer

Zonotopic abstraction based on Affine Arithmetic (Stolfi 93)

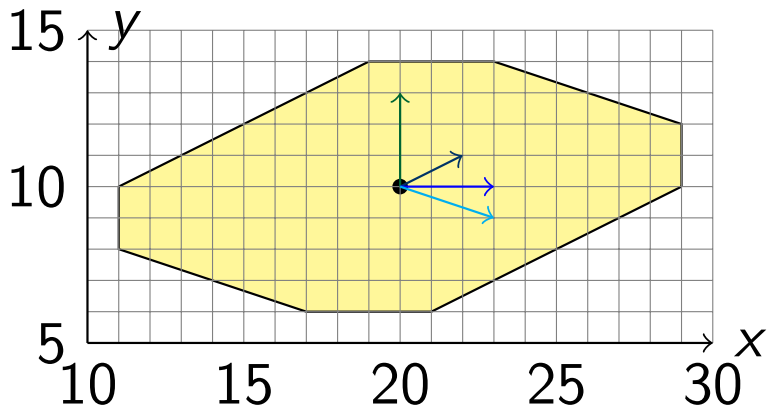
- The *real value* of variable x is represented by an affine form $\hat{r}^x$:

$$\hat{r}^x = r_0^x + r_1^x \varepsilon_1 + \ldots + r_n^x \varepsilon_n,$$

where $r_i^x \in \mathbb{R}$ and the ε_i are independent symbolic variables with unknown value in $[-1, 1]$.

- Sharing ε_i between variables expresses *implicit dependency*: concretization as a zonotope

$$\begin{aligned} \hat{x} &= 20 - 4\varepsilon_1 + 2\varepsilon_3 + 3\varepsilon_4 \\ \hat{y} &= 10 - 2\varepsilon_1 + \varepsilon_2 - \varepsilon_4 \end{aligned}$$



Abstract domain based on affine arithmetic

- *Assignment* of a variable x whose real value is given in a range $[a, b]$ introduces a noise symbol ε_i :

$$\hat{r}^x = \frac{(a+b)}{2} + \frac{(b-a)}{2}\varepsilon_i.$$

  - functional abstraction: link to the inputs via the noise symbols, allowing sensitivity analysis and worst case generation

- *Addition* is computed componentwise (no new noise symbol):

$$\hat{r}^x + \hat{r}^y = (r_0^x + r_0^y) + (r_1^x + r_1^y)\varepsilon_1 + \ldots + (r_n^x + r_n^y)\varepsilon_n$$

- *Multiplication* : we select an approximate linear form, the approximation error creates a new noise term :

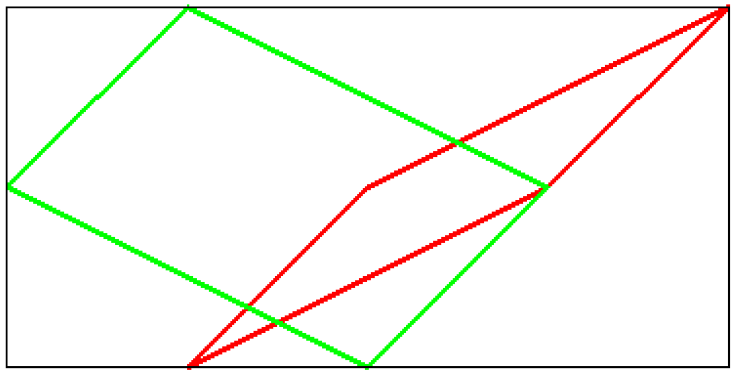
$$\hat{r}^x \times \hat{r}^y = r_0^x r_0^y + \sum_{i=1}^{n}(r_i^x r_0^y + r_i^y r_0^x)\varepsilon_i + (\sum_{i=1}^{n}|r_i^x|.|\sum_{i=1}^{n}|r_i^y|)\varepsilon_{n+1}.$$

- *Non linear operations* : approximate linear form (Taylor expansion), new noise term for the approximation error

Example computation of a (m)ub

$$\begin{aligned} \hat{r}^x &= 3 + \varepsilon_1 + 2\varepsilon_2 \\ \hat{r}^y &= 1 - 2\varepsilon_1 + \varepsilon_2 \\ \hat{r}^u &= \varepsilon_1 + \varepsilon_2 \end{aligned}$$

$$\hat{r}^z = \hat{r}^x \cup \hat{r}^y = 2 + 0\varepsilon_1 + 1\varepsilon_2 + 3\varepsilon_U$$

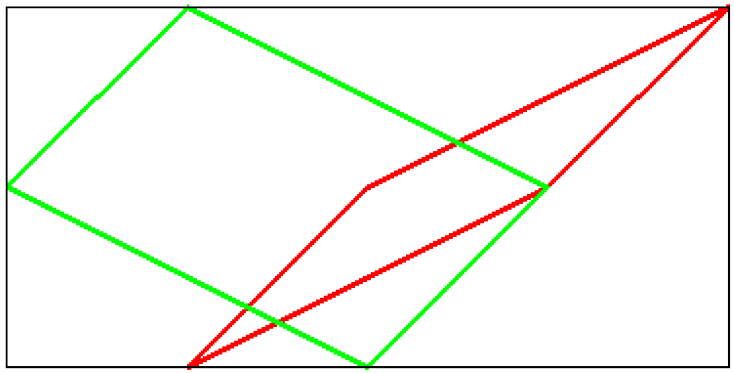


$\hat{r}^x$, $\hat{r}^y$ functions of $\hat{r}^u$

Example computation of a (m)ub

$$\begin{aligned} \hat{r}^x &= 3 + \varepsilon_1 + 2\varepsilon_2 \\ \hat{r}^y &= 1 - 2\varepsilon_1 + \varepsilon_2 \\ \hat{r}^u &= \varepsilon_1 + \varepsilon_2 \end{aligned}$$

$$\hat{r}^z = \hat{r}^x \cup \hat{r}^y = 2 + 0\varepsilon_1 + 1\varepsilon_2 + 3\varepsilon_U$$



$\hat{r}^x$, $\hat{r}^y$ functions of $\hat{r}^u$

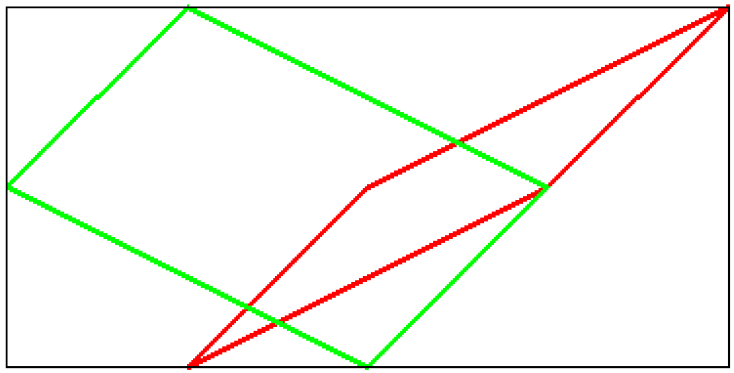
Keep "minimal common dependencies":

$$r_i^z = \underset{r_i^x \wedge r_i^y \leq r \leq r_i^x \vee r_i^y}{\operatorname{argmin}} |r|, \ \forall i \geq 1$$

Example computation of a (m)ub

$$\begin{aligned} \hat{r}^x &= 3 + \varepsilon_1 + 2\varepsilon_2 \\ \hat{r}^y &= 1 - 2\varepsilon_1 + \varepsilon_2 \\ \hat{r}^u &= \varepsilon_1 + \varepsilon_2 \end{aligned}$$

$$\hat{r}^z = \hat{r}^x \cup \hat{r}^y = 2 + 0\varepsilon_1 + 1\varepsilon_2 + 3\varepsilon_U$$



$\hat{r}^x$, $\hat{r}^y$ functions of $\hat{r}^u$

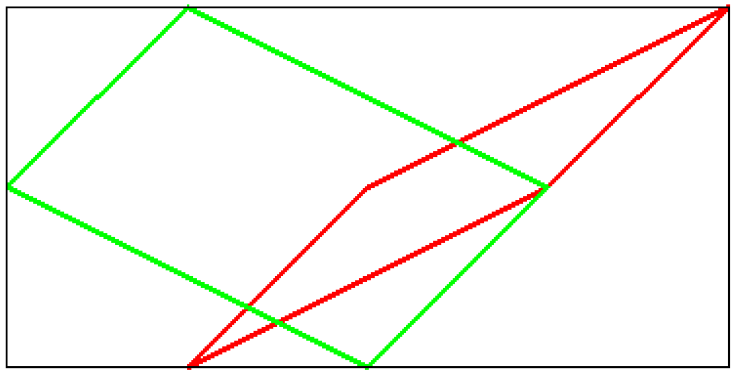
Keep "minimal common dependencies":

$$r_i^z = \underset{r_i^x \wedge r_i^y \leq r \leq r_i^x \vee r_i^y}{\operatorname{argmin}} |r|, \ \forall i \geq 1$$

Example computation of a (m)ub

$$\begin{aligned} \hat{r}^x &= 3 + \varepsilon_1 + 2\varepsilon_2 \\ \hat{r}^y &= 1 - 2\varepsilon_1 + \varepsilon_2 \\ \hat{r}^u &= \varepsilon_1 + \varepsilon_2 \end{aligned}$$

$$\hat{r}^z = \hat{r}^x \cup \hat{r}^y = 2 + 0\varepsilon_1 + 1\varepsilon_2 + 3\varepsilon_U$$



$\hat{r}^x$, $\hat{r}^y$ functions of $\hat{r}^u$

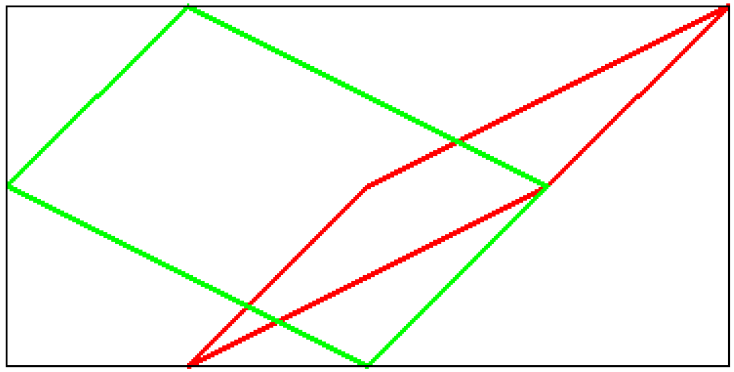
Keep "minimal common dependencies":

$$r_i^z = \underset{r_i^x \wedge r_i^y \leq r \leq r_i^x \vee r_i^y}{\operatorname{argmin}} |r|, \ \forall i \geq 1$$

Example computation of a (m)ub

$$\begin{aligned} \hat{r}^x &= 3 + \varepsilon_1 + 2\varepsilon_2 \\ \hat{r}^y &= 1 - 2\varepsilon_1 + \varepsilon_2 \\ \hat{r}^u &= \varepsilon_1 + \varepsilon_2 \end{aligned}$$

$\hat{r}^z = \hat{r}^x \cup \hat{r}^y = 2 + 0\varepsilon_1 + 1\varepsilon_2 + 3\varepsilon_U$



$\hat{r}^x$, $\hat{r}^y$ functions of $\hat{r}^u$

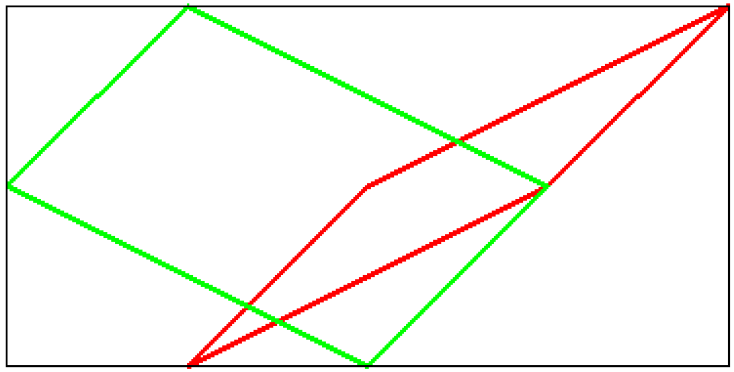
For each var, the concretization is the interval union of the concretizations ($\gamma(\hat{z}) = [-2, 6]$) :

$$\begin{aligned} &i) \quad r_0^z = mid(\gamma(\hat{r}^x) \cup \gamma(\hat{r}^y)) \\ &ii) \quad \beta^z = \sup \gamma(\hat{r}^x) \cup \gamma(\hat{r}^y) - r_0^z - \|\hat{r}^z\|_1 \end{aligned}$$

Example computation of a (m)ub

$$\begin{aligned} \hat{r}^x &= 3 + \varepsilon_1 + 2\varepsilon_2 \\ \hat{r}^y &= 1 - 2\varepsilon_1 + \varepsilon_2 \\ \hat{r}^u &= \varepsilon_1 + \varepsilon_2 \end{aligned}$$

$$\hat{r}^z = \hat{r}^x \cup \hat{r}^y = 2 + 0\varepsilon_1 + 1\varepsilon_2 + 3\varepsilon_U$$



$\hat{r}^x$, $\hat{r}^y$ functions of $\hat{r}^u$

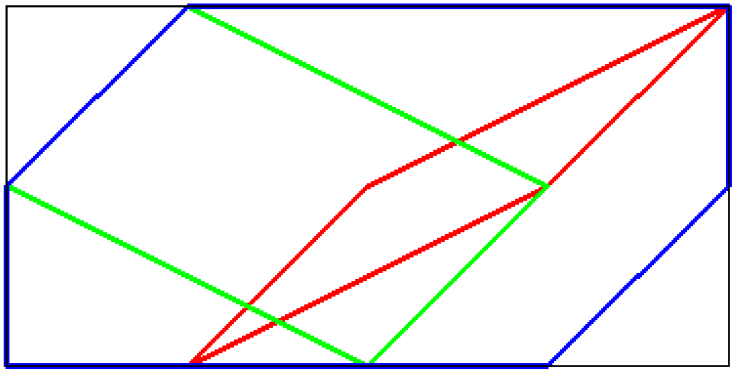
For each var, the concretization is the interval union of the concretizations ($\gamma(\hat{z}) = [-2, 6]$) :

$$\begin{aligned} &i) \quad r_0^z = mid(\gamma(\hat{r}^x) \cup \gamma(\hat{r}^y)) \\ &ii) \quad \beta^z = \sup \gamma(\hat{r}^x) \cup \gamma(\hat{r}^y) - r_0^z - \|\hat{r}^z\|_1 \end{aligned}$$

Example computation of a (m)ub

$$\begin{aligned} \hat{r}^x &= 3 + \varepsilon_1 + 2\varepsilon_2 \\ \hat{r}^y &= 1 - 2\varepsilon_1 + \varepsilon_2 \\ \hat{r}^u &= \varepsilon_1 + \varepsilon_2 \end{aligned}$$

$$\hat{r}^z = \hat{r}^x \cup \hat{r}^y = 2 + 0\varepsilon_1 + 1\varepsilon_2 + 3\varepsilon_U$$



$\hat{r}^x$, $\hat{r}^y$ and $\hat{r}^z$ functions of $\hat{r}^u$

For each var, the concretization is the interval union of the concretizations ($\gamma(\hat{z}) = [-2, 6]$) :

$$\begin{aligned} &i) \quad r_0^z = mid(\gamma(\hat{r}^x) \cup \gamma(\hat{r}^y)) \\ &ii) \quad \beta^z = \sup \gamma(\hat{r}^x) \cup \gamma(\hat{r}^y) - r_0^z - \|\hat{r}^z\|_1 \end{aligned}$$

Outline of the talk

- Modelling finite precision computations
- Affine sets for zonotopic abstraction of real computations
- Zonotopic abstraction of finite precision computations
  - real/double to float conversions
  - arithmetic operations on float/double
  - operations on integers also handled (not detailed here)
  - join and meet
- Example with the Fluctuat static analyzer

Full abstraction

Abstract value: an interval and two affine forms $x = (\mathbf{f}^x, \hat{r}^x, \hat{e}^x)$:

$$\begin{aligned} \hat{r}^x &= r_0^x + \sum_i r_i^x \, \varepsilon_i^r \\ \hat{e}^x &= e_0^x + \sum_i e_i^x \, \varepsilon_i^r + \sum_l e_l^x \, \varepsilon_l^e \end{aligned}$$

- $\mathbf{f}^x = [\underline{f^x}, \overline{f^x}]$ bounds the finite prec value, $(\underline{f^x}, \overline{f^x}) \in \mathbb{F} \times \mathbb{F}$,
- $e_l^x \, \varepsilon_l^e$: uncertainty on the rounding error committed at point l of the program (its center being in e_0^x), and its propagation through further computations,
- $e_i^x \, \varepsilon_i^r$: propagation of the uncertainty on value at point i, on the error term,
- dependency between errors and values partially modelled

Real/double to float conversion $y = (float)^n x$

Central operation because involved in all arithmetic operations

$$y = (float)^n x \quad = \quad ((float)(\mathbf{f}^x),\ \hat{r}^x,\ \hat{e}^x + new_{\varepsilon_n^e}(\mathbf{e}(\mathbf{f}^x))),$$

- Rounding error of a real/double value given in $\mathbf{f}^x$ to its finite precision representation bounded by the interval

$$\mathbf{e}(\mathbf{f}^x) = [-u^x, u^x] \cap ([\underline{f^x}, \overline{f^x}] - [fl(\underline{f^x}), fl(\overline{f^x})]),$$

  where $u^x = \max(\delta_r \max(|fl(\underline{f^x})|, |fl(\overline{f^x})|), \delta_a)$.
  - computed as the intersection of the bound given by the norm, and the interval difference between the real and the finite precision values.
- Creation of a new error noise symbol ε_n^e associated to control point n: for an interval I, $new_{\varepsilon_n^e}(I) = mid(I) + dev(I)\varepsilon_n^e$

Multiplication

$$z = x \times^n y = ((\mathbf{f}^x \times_{\mathbb{F}} \mathbf{f}^y) \cap \gamma(\hat{r}^z - \hat{e}^z), \hat{r}^x \hat{r}^y, \hat{e}^z),$$

where

$$\hat{e}^z = \hat{r}^y \hat{e}^x + \hat{r}^x \hat{e}^y - \hat{e}^x \hat{e}^y + new_{\varepsilon_n^e}(\mathbf{e}(\gamma(\hat{r}^z - (\hat{r}^y \hat{e}^x + \hat{r}^x \hat{e}^y - \hat{e}^x \hat{e}^y))))$$

Example:

```
float x = [0,1];      [1]
float y = 0.1;        [2]
float z = x*y;        [3]
```

$$x = ([0,1], 0.5 + 0.5\epsilon_1^r, 0)$$

Multiplication

$$z = x \times^n y = ((\mathbf{f}^x \times_{\mathbb{F}} \mathbf{f}^y) \cap \gamma(\hat{r}^z - \hat{e}^z), \hat{r}^x \hat{r}^y, \hat{e}^z),$$

where

$$\hat{e}^z = \hat{r}^y \hat{e}^x + \hat{r}^x \hat{e}^y - \hat{e}^x \hat{e}^y + new_{\varepsilon_n^e}(\mathbf{e}(\gamma(\hat{r}^z - (\hat{r}^y \hat{e}^x + \hat{r}^x \hat{e}^y - \hat{e}^x \hat{e}^y))))$$

Example:

```
float x = [0,1];      [1]
float y = 0.1;        [2]
float z = x*y;        [3]
```

$$\begin{aligned} x &= ([0,1], 0.5 + 0.5\epsilon_1^r, 0) \\ y &= (fl(0.1), 0.1, -1.59e^{-9}) \end{aligned}$$

Multiplication

$$z = x \times^n y = ((\mathbf{f}^x \times_{\mathbb{F}} \mathbf{f}^y) \cap \gamma(\hat{r}^z - \hat{e}^z), \hat{r}^x \hat{r}^y, \hat{e}^z),$$

where

$$\hat{e}^z = \hat{r}^y \hat{e}^x + \hat{r}^x \hat{e}^y - \hat{e}^x \hat{e}^y + new_{\varepsilon_n^e}(\mathbf{e}(\gamma(\hat{r}^z - (\hat{r}^y \hat{e}^x + \hat{r}^x \hat{e}^y - \hat{e}^x \hat{e}^y))))$$

Example:

```
float x = [0,1];      [1]
float y = 0.1;        [2]
float z = x*y;        [3]
```

$$\begin{aligned} x &= ([0,1], 0.5 + 0.5\epsilon_1^r, 0) \\ y &= (fl(0.1), 0.1, -1.59e^{-9}) \\ z &= ([0, fl(0.1)], 0.05(1+\varepsilon_1^r), -7.45e^{-10}(1+\varepsilon_1^r) + 3.72e^{-9}\varepsilon_3^e) \end{aligned}$$

Multiplication

$$z = x \times^n y = ((\mathbf{f}^x \times_{\mathbb{F}} \mathbf{f}^y) \cap \gamma(\hat{r}^z - \hat{e}^z), \hat{r}^x \hat{r}^y, \hat{e}^z),$$

where

$$\hat{e}^z = \hat{r}^y \hat{e}^x + \hat{r}^x \hat{e}^y - \hat{e}^x \hat{e}^y + new_{\varepsilon_n^e}(\mathbf{e}(\gamma(\hat{r}^z - (\hat{r}^y \hat{e}^x + \hat{r}^x \hat{e}^y - \hat{e}^x \hat{e}^y))))$$

Example:

```
float x = [0,1];      [1]
float y = 0.1;        [2]
float z = x*y;        [3]
```

$$\begin{aligned} x &= ([0,1], 0.5 + 0.5\epsilon_1^r, 0) \\ y &= (fl(0.1), 0.1, -1.59e^{-9}) \\ z &= ([0, fl(0.1)], 0.05(1+\varepsilon_1^r), -7.45e^{-10}(1+\varepsilon_1^r) + 3.72e^{-9}\varepsilon_3^e) \end{aligned}$$

Multiplication

$$z = x \times^n y = ((\mathbf{f}^x \times_{\mathbb{F}} \mathbf{f}^y) \cap \gamma(\hat{r}^z - \hat{e}^z), \hat{r}^x \hat{r}^y, \hat{e}^z),$$

where

$$\hat{e}^z = \hat{r}^y \hat{e}^x + \hat{r}^x \hat{e}^y - \hat{e}^x \hat{e}^y + new_{\varepsilon_n^e}(\mathbf{e}(\gamma(\hat{r}^z - (\hat{r}^y \hat{e}^x + \hat{r}^x \hat{e}^y - \hat{e}^x \hat{e}^y))))$$

Example:

```
float x = [0,1];      [1]
float y = 0.1;        [2]
float z = x*y;        [3]
```

$$\begin{aligned} x &= ([0,1], 0.5 + 0.5\epsilon_1^r, 0) \\ y &= (fl(0.1), 0.1, -1.59e^{-9}) \\ z &= ([0, fl(0.1)], 0.05(1+\varepsilon_1^r), -7.45e^{-10}(1+\varepsilon_1^r) + 3.72e^{-9}\varepsilon_3^e) \end{aligned}$$

Multiplication

$$z = x \times^n y = \left((\mathbf{f}^x \times_{\mathbb{F}} \mathbf{f}^y) \cap \gamma(\hat{r}^z - \hat{e}^z), \hat{r}^x \hat{r}^y, \hat{e}^z \right),$$

where

$$\hat{e}^z = \hat{r}^y \hat{e}^x + \hat{r}^x \hat{e}^y - \hat{e}^x \hat{e}^y + new_{\varepsilon_n^e} \left(\mathbf{e} \left(\gamma \left(\hat{r}^z - \left(\hat{r}^y \hat{e}^x + \hat{r}^x \hat{e}^y - \hat{e}^x \hat{e}^y \right) \right) \right) \right)$$

Example:

```
float x = [0,1];      [1]
float y = 0.1;        [2]
float z = x*y;        [3]
```

$$\begin{aligned} x &= ([0,1], 0.5 + 0.5\epsilon_1^r, 0) \\ y &= (fl(0.1), 0.1, -1.59e^{-9}) \\ z &= ([0, fl(0.1)], 0.05(1+\varepsilon_1^r), -7.45e^{-10}(1+\varepsilon_1^r) + 3.72e^{-9}\varepsilon_3^e) \end{aligned}$$

Join operation

Component-wise, using the join over intervals and affine forms
$x \cup y = (\mathbf{f}^x \cup \mathbf{f}^y, \hat{r}^x \cup \hat{r}^y, \hat{e}^x \cup \hat{e}^y)$.

- the join over affine forms keeps only common relation
- sources of errors that are not common to all execution paths of the programs will be lost and assigned to the label of the join.

Example:

```
double x,y,z;
x = [-1,1];
y = [0,1]*2;                    [2]
{ if (x < 0) z = 2*y + 1;
else z = y + 2;}                [4]   // after join
```

$$y = ([0,2],\ 1+\varepsilon_2^r,\ 2.22e^{-16}\varepsilon_2^e)$$
$$z = ([1,5],\ 3+\varepsilon_2^r+\varepsilon_4^r,\ 2.22e^{-16}\varepsilon_2^e+1.11e^{-15}\varepsilon_4^e)$$

Test interpretation/meet

Stable test assumption

- assumption that the control flow of the program is the same for the finite precision and real values of the program
- if this is found not to be the case when evaluating a boolean condition in real and in floats: unstable test warning
- the finite precision control flow is followed: in case of unstable test, possibly unsound error bounds

Tests thus interpreted over both real and float values

- affine forms unchanged; extra constraints on noise symbols,
- constraints generated both by $\hat{r}^x \cap \hat{r}^y$ and $(\hat{r}^x - \hat{e}^x) \cap (\hat{r}^y - \hat{e}^y)$
- the error terms can be reduced using these constraints

Test interpretation: example

```
float x,y,z=0;
x = [0,1];    [1]
y = 2*x;      [2]
if (y <= x)
  z = x-y;    [3]   // if branch
```

$$\begin{aligned} x &= ([0,1],\ 0.5 + 0.5\varepsilon_1^r,\ 0) \\ y &= ([0,2],\ 1 + \varepsilon_1^r,\ 1.19e^{-7}\varepsilon_2^e) \end{aligned}$$

Test $y \leq x$ yields in the `if` branch:

$$\text{real:}\quad 1 + \varepsilon_1^r \leq 0.5 + 0.5\varepsilon_1^r \quad \Rightarrow \quad \varepsilon_1^r = -1$$

Thus, in the `if` branch:

$$z = ([-1.19e^{-7}, 1.19e^{-7}], 0,\ -1.19e^{-7}\varepsilon_2^e)$$

Test interpretation: example

```
float x,y,z=0;
x = [0,1];    [1]
y = 2*x;      [2]
if (y <= x)
  z = x-y;    [3]   // if branch
```

$$\begin{aligned} x &= ([0,1],\ 0.5 + 0.5\varepsilon_1^r,\ 0) \\ y &= ([0,2],\ 1 + \varepsilon_1^r,\ 1.19e^{-7}\varepsilon_2^e) \end{aligned}$$

Test $y \leq x$ yields in the `if` branch:

$$\begin{array}{lrcl} \text{real:} & 1 + \varepsilon_1^r \leq 0.5 + 0.5\varepsilon_1^r & \Rightarrow & \varepsilon_1^r = -1 \\ \text{but also } (\mathbf{f}^y \geq 0) & 1 + \varepsilon_1^r - 1.19e^{-7}\varepsilon_2^e \geq 0 & \Rightarrow & \varepsilon_2^e \leq 0. \end{array}$$

Thus, in the `if` branch:

$$z = ([0, 1.19e^{-7}], 0,\ -1.19e^{-7}\varepsilon_2^e,\ \varepsilon_2^e \leq 0)$$

Test interpretation: example

```
float x,y,z=0;
x = [0,1];    [1]
y = 2*x;      [2]
if (y <= x)
  z = x-y;    [3]   // if branch
```

$$\begin{aligned} x &= ([0,1],\ 0.5 + 0.5\varepsilon_1^r,\ 0) \\ y &= ([0,2],\ 1 + \varepsilon_1^r,\ 1.19e^{-7}\varepsilon_2^e) \end{aligned}$$

Test $y \leq x$ yields in the `if` branch:

real:	$1 + \varepsilon_1^r \leq 0.5 + 0.5\varepsilon_1^r$	$\Rightarrow$	$\varepsilon_1^r = -1$
float:	$1 + \varepsilon_1^r - 1.19e^{-7}\varepsilon_2^e \leq 0.5 + 0.5\varepsilon_1^r$	$\Rightarrow$	$\varepsilon_2^e \geq 0.$
but also ($\mathbf{f}^y \geq 0$)	$1 + \varepsilon_1^r - 1.19e^{-7}\varepsilon_2^e \geq 0$	$\Rightarrow$	$\varepsilon_2^e \leq 0.$

Thus, in the `if` branch:

$$z = ([0,0], 0,\ 0)$$

Outline of the talk

- Modelling finite precision computations
- Affine sets for zonotopic abstraction of real computations
- Zonotopic abstraction of finite precision computations
- Example with the Fluctuat static analyzer

Back to the Householder scheme

Householder

Conclusion

- Relational semantics both in real-numbers and floating point numbers, with relations between the two semantics, also enabling
  - test case generation for extremal values and errors
  - sensitivity analysis etc.
- Successfully used on non-trivial programs (see paper):
  - subsets of navigation and physical process monitoring codes of about 10KLOCs
  - parameterized second-order filters, conjugate gradient etc.
- Future work includes:
  - Improvement of the resolution of fixpoint equations through policy iteration (building on our CAV 2005 paper)
  - Combination of deterministic and probabilistic information for value and error abstractions (SCAN 2010 presentation)