A Generalization of P-boxes to Affine Arithmetic, and Applications to Static Analysis of Programs

O. Bouissou, E. Goubault, J. Goubault-Larrecq and S. Putot

CEA-LIST, MEASI (ModElisation and Analysis of Systems in Interaction) and ENS Cachan/INRIA Saclay/CNRS, LSV (Laboratory for Specification and Verification)

SCAN 2010, Lyon

Agence Nationale de la Recherche GIP

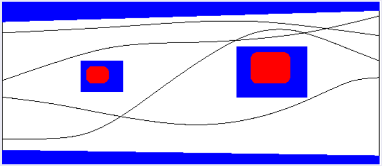
Confidence Proof Probabilities

Contents

- Static analysis of programs
  - Why do we want to mix non-determinism and probabilities?
- A quick recap on affine forms, and P-boxes/Dempster-Shafer structures
- Combining the two approaches
- Examples and first experiments

Static analysis of programs

- Find outer-approximation of sets of reachable values of variables at some program points
- To ensure absence of runtime errors typically



Example

```
float x;
x=[0,1];           [1]
while (x<=1) {     [2]
  x = x-0.5*x;     [3]
}                  [4]
```

$$\begin{aligned} x_1 &= [0,1] \\ x_2 &=]-\infty,1] \cap (x_1 \cup x_3) \\ x_3 &= x_2 - 0.5x_2 \\ x_4 &=]1,\infty[\cap x_2 \end{aligned}$$

(final smallest invariant: $x_2 \in [0,1]$, $x_4 = \emptyset$)

Primary application/motivation

- Critical embedded software such as
  - flight control computers
  - control of industrial installations (e.g. nuclear plants)
  - etc.
- Automatic proof of safety properties of such systems
  - Bounds for program variables (viz expected range)
  - Proof of absence of runtime errors (no overflow, no division by zero, no array out of bound dereference etc.)
  - More refined properties (numerical stability etc.)

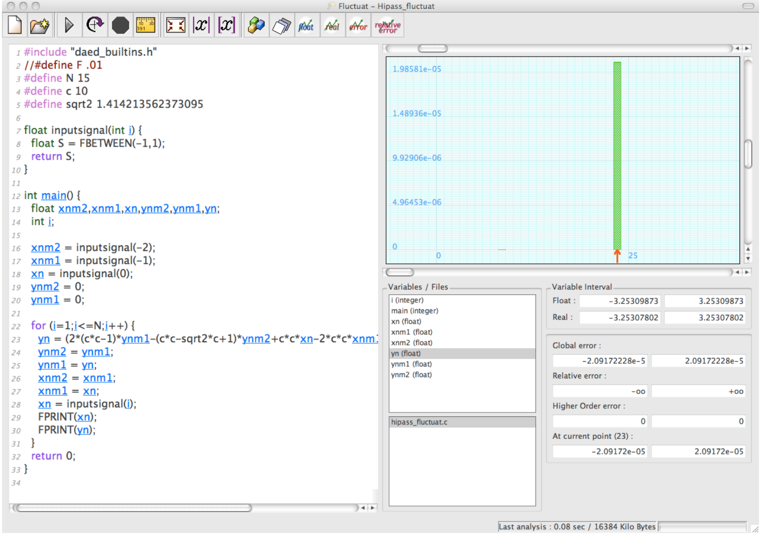
- Some inputs being known set theoretically (non-deterministic inputs) or in probability (probabilistic inputs)
  - e.g. temperature distribution maybe known but we might only know a range for pressure, in some software-driven apparatus
- In fact, more generally, inputs may be thought of as given by imprecise probabilities

Running example (high-pass, Butterworth order 2)

```
float filter() {
 float xnm2,xnm1,xn,ynm2,ynm1,yn;
 ynm2 = 0; ynm1 = 0; xnm2 = inputsignal(-2);
 xnm1 = inputsignal(-1); xn = inputsignal(0);
 for (int i=1;i<=N;i++) {
  yn = (2*(c*c-1)*ynm1-(c*c-sqrt2*c+1)*ynm2
+c*c*xn-2*c*c*xnm1+c*c*xnm2)/(c*c+sqrt2*c+1);
  ynm2 = ynm1; ynm1 = yn;
  xnm2 = xnm1; xnm1 = xn;
  xn = inputsignal(i);
 }
return yn; }
```

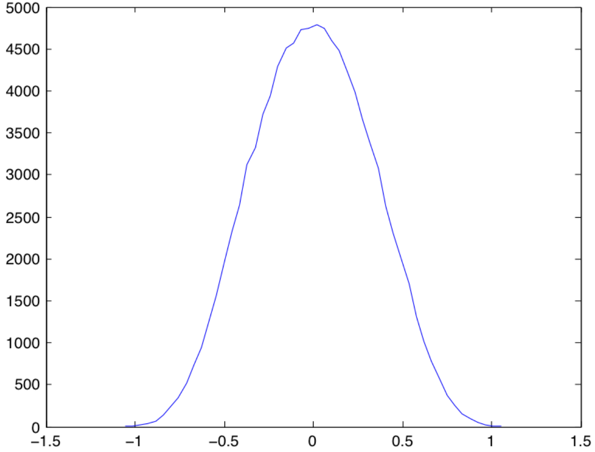
`inputsignal` comes from sensor: deterministic/probabilistic info

Inputs known set-theoretically ($\in [-1, 1]$) - by FLUCTUAT



y_{15} in $[-3.25, 3.25]$; what if better knowledge of the inputs?

Inputs known in probability (uniform) - Matlab simulation



(density function) But not guaranteed, and in general, imprecise probability for the input only...

A known method: P-boxes/Demster-Shafer

- Model "imprecise probabilities"
- Generalize both probabilities and interval computations
  - model for non-deterministic and probabilistic events

- Can be thought of as representations of sets of probability distributions

P-boxes

- Given by upper and lower "probabilities" (CDF form, not necessarily normalized) on $\mathbb{R}$: $\underline{f}$ and $\overline{f}$ from $\mathbb{R}$ to $\mathbb{R}^+$
- for all $x \in \mathbb{R}$, $\underline{f}(x) \leq \overline{f}(x)$

Confidence Proof Probabilities

Dempster-Shafer structures

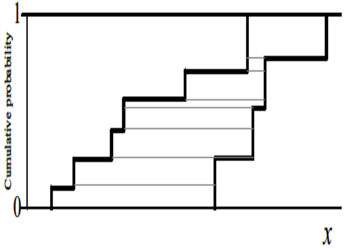
- Based on a notion of focal elements ($\in F$ - here F is a set of subsets of $\mathbb{R}$):
  - sets of non-deterministic events/values
- Weights (positive reals) associated to focal elements ($w : F \rightarrow \mathbb{R}^+$)
  - probabilistic information only available on the belonging to the focal elements, not to precise events

- Determine a belief function Bel and a plausibility function Pl from $\wp(E)$ to $\mathbb{R}$:
  - $Pl(S) = \sum_{T, T \cap S \neq \emptyset} w(T)$
  - $Bel(S) = \sum_{T, T \subseteq S} w(T)$
- $Bel(]-\infty, x]) \leq Pl(]-\infty, x])$ generate a P-box

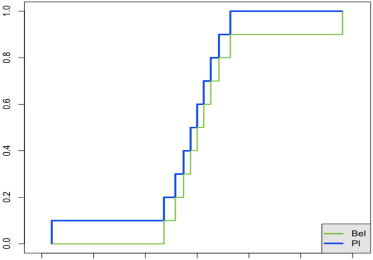
Confidence Proof Probabilities

Given a P-box $(\underline{f}, \overline{f})$

- Subdivide $supp(\underline{f}) \cup supp(\overline{f})$ and take outer approximation by stair functions on this subdivision
- Focal elements and weights can be deduced easily



(taken from SANDIA 2002-4015)



10 subd. on Gaussian(mean=0,std.dev.=0.1)

P-box→DS→P-box gives a "bigger" P-box $(\overline{f}' \geq \overline{f}, \underline{f}' \leq \underline{f})$

Some computation rules: $z = x \square y$ ($\square$=+,-,$\times$,/ etc.)

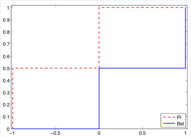
Independent variables x, y (i.e. `xn`, `xnm1` etc.)

- Easy using DS: x (resp. y) given by focal elements F^x (resp. F^y) and weights w^x (resp. w^y)
- Define DS for z: $F^z = \{f^x \square f^y \mid f^x \in F^X, f^y \in F^y\}$ and $w^z(f^x \square f^y) = w^x(f^x)w^y(f^y)$ (and renormalize)

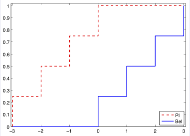
Example

- x with $F^x = \{[-1,0],[0,1]\}$, $w^x([-1,0]) = w^x([0,1]) = \frac{1}{2}$ (approximation of uniform distribution on [-1,1])
- y with $F^y = \{[-2,0],[0,2]\}$, $w^y([-2,0]) = w^y([0,2]) = \frac{1}{2}$

$x; y$	[-2,0], $\frac{1}{2}$	[0,2], $\frac{1}{2}$
[-1,0], $\frac{1}{2}$	[-3,0], $\frac{1}{4}$	[-1,2], $\frac{1}{4}$
[0,1], $\frac{1}{2}$	[-2,1], $\frac{1}{4}$	[0,3], $\frac{1}{4}$



CDF of x



CDF of $x + y$

Some computation rules (here $\square = +$)

Dependent variables x, y with unknown dependencies (i.e. `yn`, `ynm1` etc.)

- Easy using P-boxes (consequence of Fréchet bounds): x (resp. y) given by upper and lower probabilities $(\overline{f}^x, \underline{f}^x)$ (resp. $(\overline{f}^y, \underline{f}^y)$)
- Define P-box for z:

$$\overline{f}^z(x) = \inf_{u+v=x} \min\left(\overline{f}^x(u) + \overline{f}^y(v), 1\right)$$

$$\underline{f}^z(x) = \sup_{u+v=x} \max\left(\underline{f}^x(u) + \underline{f}^y(v) - 1, 0\right)$$

- Use transfo DS$\leftrightarrow$P-box to find the right formulas on DS directly - expanded in full paper; or use Williamson and Downs/Ferson et al. (LP)/Berleant et al.

Sum $z = x + y$ of DS x, y with unknown dependence

Suppose:

- DS for x (similarly for y) given on $F^x = \{[a_i^x, b_i^x] \mid i = 1, \ldots, n\}$ by $w^x([a_i^x, b_i^x]) = w_i^x$

Compute the stair functions given by values at $a_k^x + a_l^y$, $b_k^x + b_l^y$:

$$\overline{f}_z(a_k^x + a_l^y) = \min\left(\inf_{a_i^x + a_j^y = a_k^x + a_l^y} \sum_{i' \leq i} w_{i'}^x + \sum_{j' \leq j} w_{j'}^y, 1\right)$$

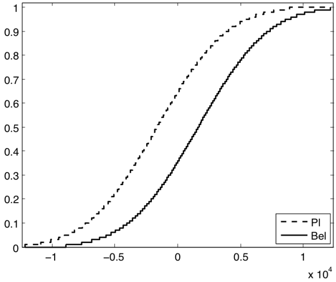
$$\underline{f}_z(b_k^x + b_l^y) = \max\left(\sup_{b_i^x + b_j^y = b_k^x + b_l^y} \sum_{i' \leq i} w_{i'}^x + \sum_{j' \leq j} w_{j'}^y - 1, 0\right)$$

Deduce F^z and w^z

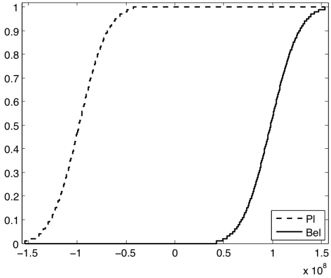
- See "from P-box to DS and vice-versa" (or Mobius inversion formula)

This is all nice but...

Suffer from the same disease as interval computations (wrapping effect), and costly computations (here using Matlab/IPPtoolbox by P. Limburg - 100 subd. for each uniform distrib.):



15 iterations (4.65 seconds)



30 iterations (9.30 seconds)

Our approach

- Encode as much deterministic dependencies as possible

```
yn = ...*ynm1-...*ynm2 // not unknown dep.
+...*xn-...*xnm1+...*xnm2); // indep.
```

  - use affine arithmetic based abstraction
  - linearization of dependencies
  - representation on a basis of independent noise symbols

- Use P-boxes for probabilistic values, independent as much as possible...
  - associate a P-box to each noise symbol
  - technicality: some noise symbols (coming from non-linear terms in particular) have unknown dependencies...

Affine Arithmetic for real numbers - quick recap

Originally: Comba, de Figueiredo and Stolfi 1993

- A variable x is represented by an affine form $\hat{x}$:

$$\hat{x} = x_0 + x_1\varepsilon_1 + \ldots + x_n\varepsilon_n,$$

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

  - $x_0 \in \mathbb{R}$ is the *central value* of the affine form
  - the coefficients $x_i \in \mathbb{R}$ are the *partial deviations*
  - the ε_i are the *noise symbols*

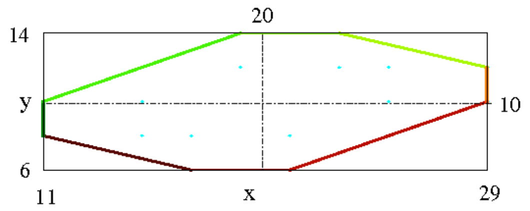
- The sharing of noise symbols between variables expresses *implicit dependency*

(FLUCTUAT is based on ideas coming from affine arithmetic)

They form sub-polyhedric relations

Concretization is a center-symmetric convex polytope

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



Affine Arithmetic for over-approximation (some functions)

Assignment

of a variable x whose value is given in a range $[a, b]$ at label i

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

Addition

$$\hat{x} + \hat{y} = (\alpha_0^x + \alpha_0^y) + (\alpha_1^x + \alpha_1^y)\varepsilon_1 + \ldots + (\alpha_n^x + \alpha_n^y)\varepsilon_n$$

For example, with real (exact) coefficients , $f - f = 0$.

Multiplication

creates a new noise term (can do better):

$$\hat{x} \times \hat{y} = \alpha_0^x\alpha_0^y + \sum_{i=1}^{n}(\alpha_i^x\alpha_0^y + \alpha_i^y\alpha_0^x)\varepsilon_i + \left(\sum_{i=1}^{n}|\alpha_i^x|.|\sum_{i=1}^{n}|\alpha_i^y|\right)\eta_1.$$

Careful: η symbols have unknown dependencies with the ε_i!

Affine P-boxes

P-forms

- Affine forms based on two sets of noise symbols:
  - ε_i independent with each other, created by inputs
  - η_j unknown dependencies with each other and with the ε_i, created by non-linear computation (including branching)
- Together with (imprecise) probabilistic information:
  - DS associated to ε_i: (F^i, w^i); DS associated to η_j: (G^j, v^j)

Remarks

- Notation: for each program variable x associate

$$\hat{x} = c_0^x + \sum_{i=1}^{n} c_i^x \varepsilon_i + \sum_{j=1}^{m} p_j^x \eta_j$$

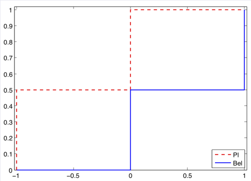
- Experiments in what follows using a toy implementation based on IPPtoolbox under Matlab

Operator γ

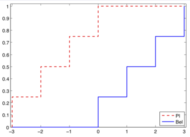
- Easy: use computation rules for addition on independent ε_i and for η_j with unknown dependency (Fréchet bounds)

Example

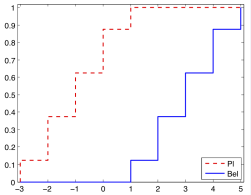
Consider $\hat{x} = 1 + \varepsilon_1 - 2\varepsilon_2 + \eta_1$, all noise symbols being uniform distributions on $[-1, 1]$, approximated by $F = \{[-1, 0], [0, 1]\}$, $w([-1, 0]) = w([0, 1]) = \frac{1}{2})$



CDF of ε_1



CDF of $\varepsilon_1 - 2\varepsilon_2$



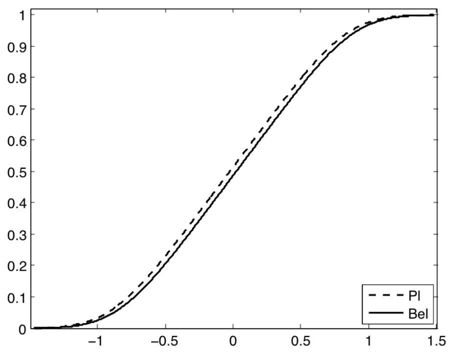
CDF of $\hat{x}$

Computation rules on P-forms (I)

Linear transformations

- Easy: no new noise symbol, exact linear transformation on the affine (deterministic) part

Example: high-pass filter, 30 iterations - 100 subd. for ε_i



Some preliminary experiments (on this Mac+ Matlab+IPP)

Using 100 subdivisions for the initial uniform P-boxes

#iter	t P-Box	t P-form	supp P-box	supp P-form
15	4.65s	0.53s	$\sim 10^4$	~ 1
30	9.30s	1.11s	$\sim 10^8$	~ 1
45	13.79s	1.72s	$\sim 10^{12}$	~ 1

With 1000 subdivisions

- 30 iterations: using P-boxes 279.94s, using P-forms 38.23s

Confidence Proof Probabilities

Computation rules on P-forms (II)

Multiplication

- Linear term (on ε_i) is easy, same as for affine forms
- Associate a new noise symbol η_{m+1} to the non-linear terms:

$$a = \sum_{1 \le r,l \le n} c_r^x c_l^y \varepsilon_r \varepsilon_l + \sum_{1 \le r,l \le m} p_r^x p_l^y \eta_r \eta_l + \sum_{1 \le r,l \le n,m} (c_r^x p_l^y + c_r^y p_l^x) \epsilon_r \eta_l \ .$$

- Associate to η_{m+1} the correct DS, based on the following facts:
  (1) $\epsilon_r \epsilon_l$ $(r \neq l)$ is a product of two independent DS
  (2) $\eta_r \eta_l$ $(l \neq r)$ and $\epsilon_r \eta_l$ are products of two DS whose dependence is unknown.
  (3) $\epsilon_r \epsilon_l$ with $r = l$ and $\eta_r \eta_l$ with $r = l$ are products of two P-boxes whose dependence is perfectly known (squares).

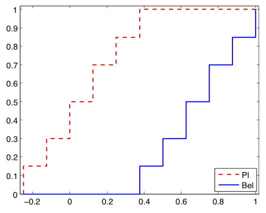
Multiplication example

Iterated power on uniform laws

```
float x in [0,1]; float y in [0,1];
for (int i=1;i<10;i++) x = x*y;
```

Initially - small dependence between x and y:

- $\hat{x} = \frac{1}{2} + \frac{1}{2}\varepsilon_1$,
- $\hat{y} = \frac{1}{2} + \frac{1}{4}\varepsilon_1 + \frac{1}{4}\varepsilon_2$, ε_1 has $F^1 = \{[-1,0],[0,1]\}$, $w^1([-1,0]) = w^1([0,1] = \frac{1}{2}$ (same for ε_2)



Concentrates around 0
Outer approximation but keeps dependencies!

Computation rules on P-forms (III)

Union

- Creates an extra η_k noise symbol per variable

$$\hat{z} = c_0^z + \sum_{i=1}^{n} c_i^z \varepsilon_i + \sum_{j=1}^{m+1} p_j^z \eta_j$$

- Based on the "argmin" formula (Goubault & Putot) for c_i^z
- P-box (hence DS) associated to η_{m+1} estimated as interval union of $\gamma\left(\hat{x} - c_0^z - \sum_{i=1}^{n} c_i^z \varepsilon_i\right)$ with $\gamma\left(\hat{y} - c_0^z - \sum_{i=1}^{n} c_i^z \varepsilon_i\right)$

Intersection

- Interpret `if` $(expr == 0)$ `then` ...
- From $expr == 0$, contraints on DS associated with ε_i and η_j
- Link with Ghorbal & Goubault & Putot constraint affine forms' domain

Conclusion and future work

Theory

- Compare with Williamson & Downs, with Statool (Berleant et al.), RiskCalc (Ferson et al.)
- Complete semantic framework (order etc.)
- Probabilistic computations (not only inputs)
- Link with under-approximations (different notions of dependencies) - link with copula based (and other) approaches

Static analysis

- Comparison with D. Monniaux probabilistic static analyzes

Other application

- Probabilistic computations...
- Floating-point numbers (deterministic model for reals, probabilistic model for errors)

Related work

- Dependence in Dempster-Shafer Theory and Probability Bounds Analysis, Ferson, Hajagos, Berleant, Zhang, Tucker, Ginzburg, Oberkampf, SANDIA Report 2004
- Clouds, Fuzzy Sets and Probability Intervals, Neumaier, Reliable Computing, 2004
- Probabilistic Arithmetic I: Numerical Methods for Calculating Convolutions and Dependency Bounds, Williamson and Downs, Journal of Approximate Reasoning, 1990
- Abstraction of expectation functions using gaussian distributions, Monniaux, VMCAI, 2003
- Prevision Domains and Convex Powercones, Goubault-Larrecq, FoSSaCS, 2008
- A zonotopic framework for functional abstractions, Goubault, Putot, arxiv:0910.1763, 2009
- Absolute Bounds on the Mean of Sum, Product, Max and Min: A Probabilistic Extension to Interval Arithmetic, Ferson, Ginzburg, Kreinovich, Lopez
- Bounding the Results of Arithmetic Operations on Random Variables of Unknown Dependency using Intervals, Berleant, Goodman-Strauss, Reliable Computing 1998