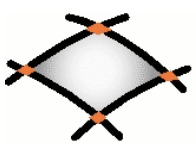
Interprétation abstraite pour la précision numérique : de la détection à la correction d'erreurs

Matthieu Martel

DALI - Université de Perpignan Via Domitia
LIRMM - CNRS: UMR 5506 - Université Montpellier 2, France

matthieu.martel@univ-perp.fr

DÉCEMBRE 2012



UPVD
Université de Perpignan Via Domitia

Laboratoire d'Informatique de Robotique et de Microélectronique de Montpellier
LIRMM

Introduction

Flottants utilisés à la place des réels pour les calculs sur ordinateurs

Arithmétiques très différentes :

Opérations non associatives, non distributives, non inversibles

- $\sqrt{2.0} = 1.414\ldots = x \quad x^2 = 1.9999\ldots$
- dans les doubles :

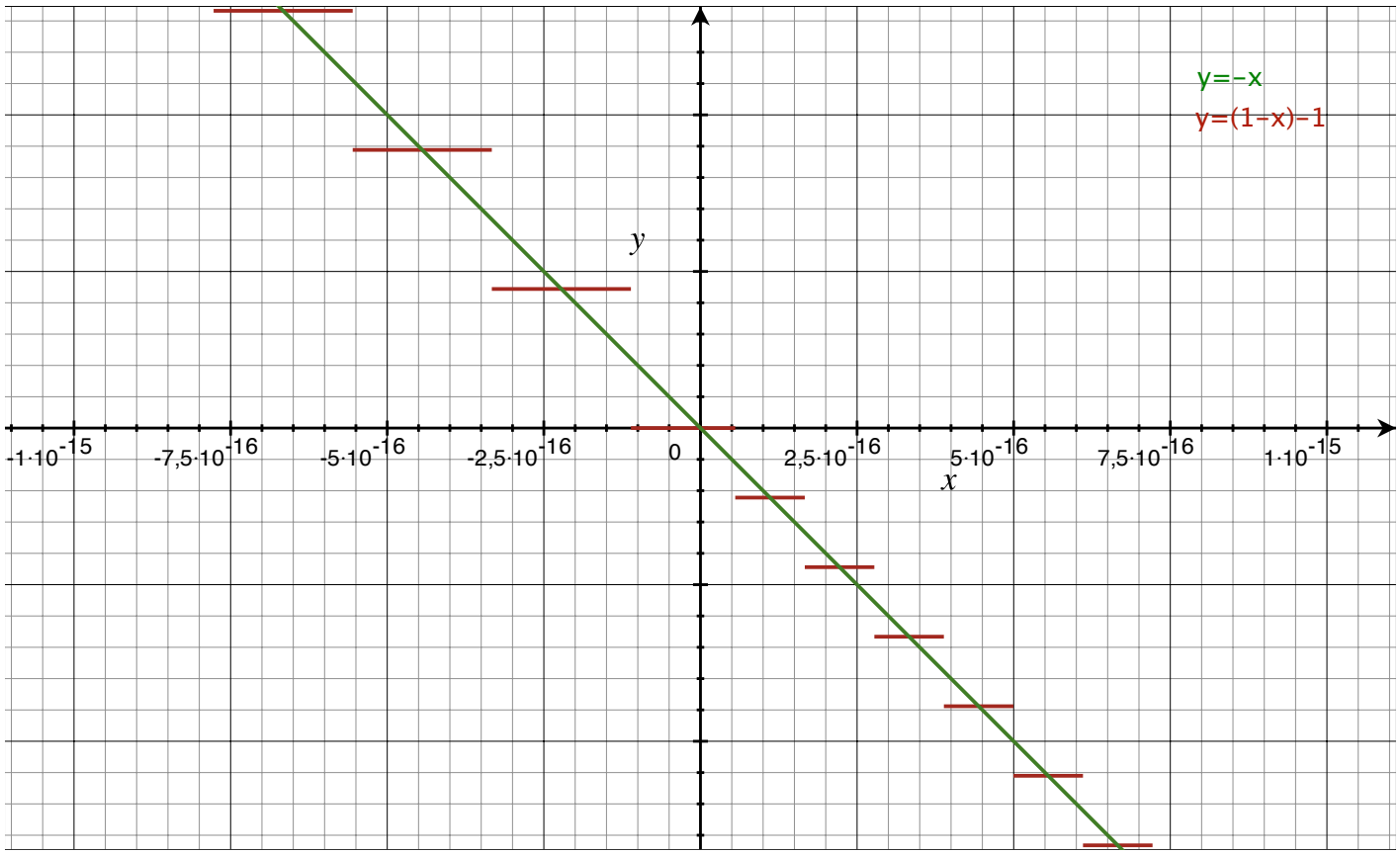
$x + 16.0 = 16.0$ ssi $x \in [-8.881784197001252 32e^{-16}, 1.7763568394002504 6e^{-15}]$

Nombre fini de décimales

- Exceptions : overflow, underflow, NaN.
  `if (x<>0) y = 1/(x*x)`
- Pertes de précision : erreurs “subjectives” ne déclenchent pas d’exception/interruption

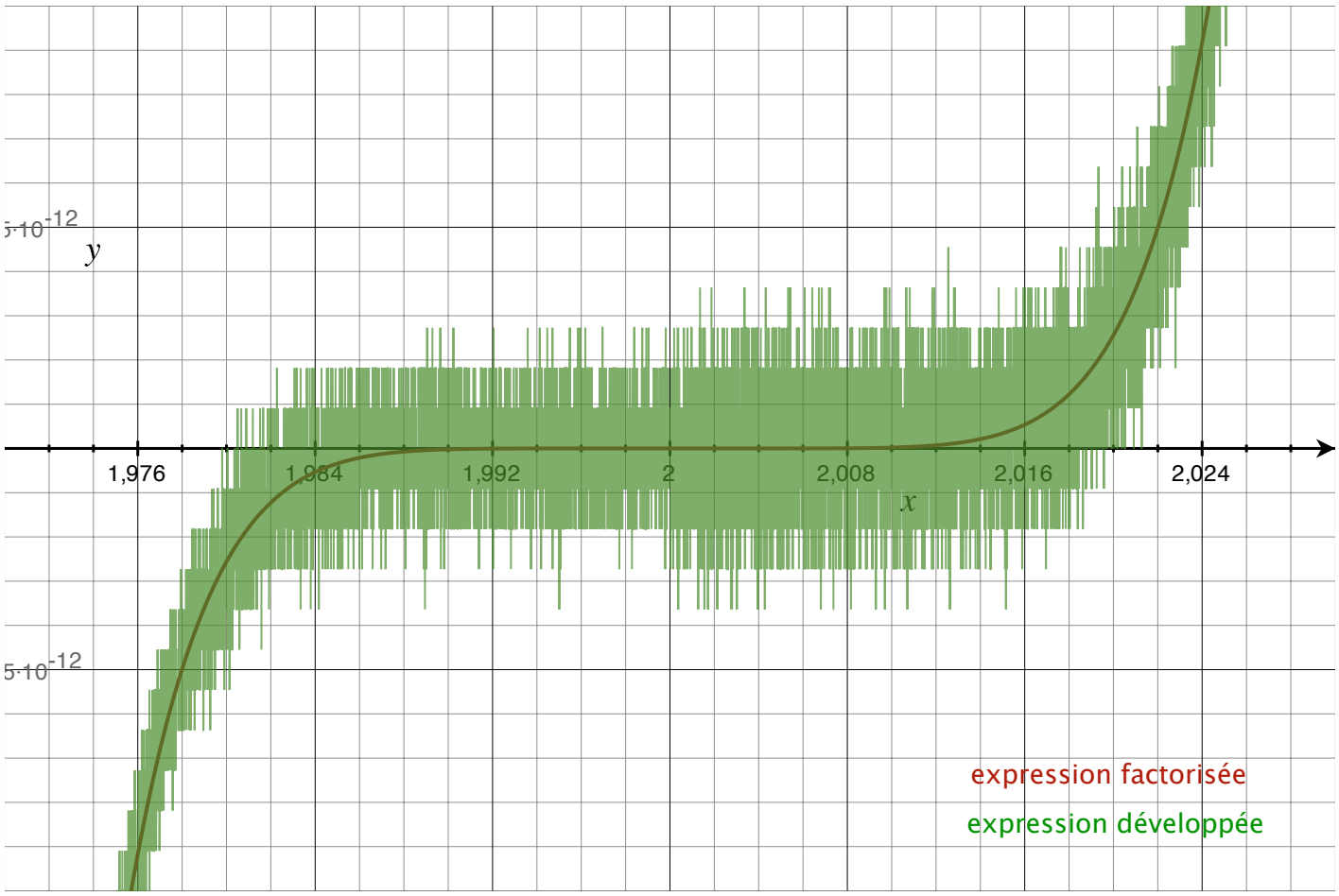
Exemple 1

$$f : \left| \begin{array}{ccc} \mathbb{F} & \to & \mathbb{F} \\ x & \mapsto & (1-x)-1 \end{array} \right.$$



Exemple 2

$$f : \left| \begin{array}{ccc} \mathbb{F} & \to & \mathbb{F} \\ x & \mapsto & (x-2)^7 \end{array} \right.$$



Plan

Généralités sur les flottants

Norme IEEE 754

Mesure des erreurs

La norme IEEE 754

$$\begin{aligned} f &= \pm d_0.d_1 d_2 \ldots d_{p-1} \beta^e \\ &= \pm \left(d_0 + d_1 \beta^{-1} + \ldots d_{p-1} \beta^{-(p-1)} \right) \beta^e \end{aligned} \tag{1}$$

β : base, e : exposant ($e_{min} \leq e \leq e_{max}$), p : précision, $d_0.d_1 \ldots d_{p-1}$ mantisse avec $0 \leq d_i < \beta$

Nombres *flottants* : nombres représentables par (1)

Nombres *normalisés* : nombres flottants t.q. $d_0 \neq 0 \quad \Rightarrow$ unicité

Nombres non-représentables :

- Hors des bornes : $x < \beta^{e_{min}}$ ou $x > \beta^{e_{max}}$
- Précision : π, 0.1 en base 2

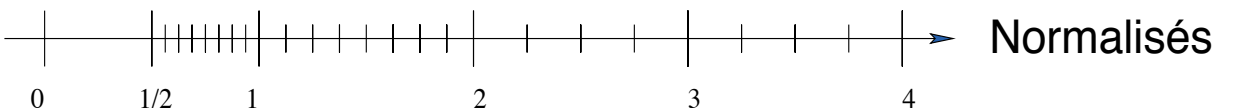
Norme IEEE 754 : formats

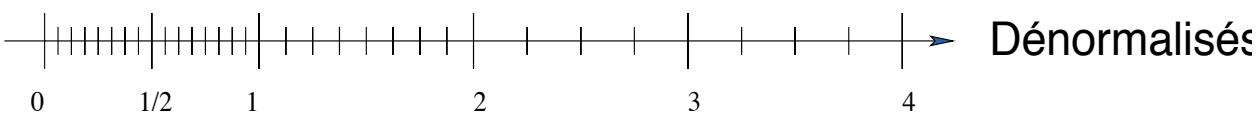
Type	Mantisse (bits)	e_{max}	e_{min}	Exposant (bits)
Single	23+1	+127	-126	8
Single Extended	$\geq$ 32	$\geq$+1023	$\leq$-1022	$\geq$11
Double	52+1	+1023	-1022	11
Double Extended	$\geq$64	$>$16383	$\leq$-16382	$\geq$15

- Rayon d'un proton : $1.2 \times 10^{-15} m$
- Masse d'un electron : $9.1 \times 10^{-31} kg$
- Masse de la voie lactée : $2.2 \times 10^{41} kg$
- Age de l'univers : $4.17 \times 10^{17} s$

Valeurs particulières

Valeur	Signification	Exposant	Mantisse
$\pm\infty$	Dépacement	$e_{max}+1$	0
NaN	Not a Number	e_{max}	$\neq 0$
± 0	Zéro signé	$-e_{max}$	0
$0.dd\ldots$	Dénormalisés	$-e_{max}$	$\neq 0$





$\beta = 2,\ p = 3,\ -1 \leq e \leq 1$

Opérations élémentaires sur les flottants

- Entre valeurs particulières :

$$-\infty \times -\infty = +\infty, \quad \pm 0 \div \pm\infty = \pm 0, \quad \pm\infty \div \pm\infty = \mathit{NaN}, \ \mathit{etc.}$$

- Pour les valeurs “classiques” $\rightarrow$ garantie de l’arrondi
- 4 modes d’arrondi : vers 0, vers $+\infty$, au plus près, vers $-\infty$

$$\uparrow_\circ : \mathbb{R} \rightarrow \mathbb{F}$$

$$\text{Pour } \lozenge \in \{+, -, \times, \div, \sqrt{}\}, \ x \lozenge_{\mathbb{F}} y = \uparrow_\circ (x \lozenge_{\mathbb{R}} y)$$

- Obtenu en utilisant des bits de garde (bits supplémentaires)
- Donne une sémantique précise aux opérations sur les flottants

Mesure des erreurs

Erreurs déclenchant une exception : Overflow Underflow NaN

Pertes de précision (ne déclenchent pas d'exception) :

- Absorption $x + y = x$ si $x \gg y$ (cf. $x + 16 = 16$)
- Branchement / Test instable
- Elimination catastrophique (cancellation), $x - y$ avec $x \approx y$

$$A = \sqrt{s(s-a)(s-b)(s-c)} \quad \text{avec } s = \frac{a+b+c}{2}$$

Si le triangle est plat, $a \approx b + c$ et $s \approx a$

Si $a = 9.00$, $b = c = 4.53$:

$s_{\mathbb{R}}$	$A_{\mathbb{R}}$	$s_{\mathbb{F}}$	$A_{\mathbb{F}}$	Erreur
9.03	2.34	9.04	2.71	15%

Ulp, erreurs relatives

ulp : **U**nit in the **L**ast **P**lace, ordre de grandeur du plus petit chiffre significatif d'un nombre

ex : si $p = 3$, $\beta = 10$, $x = 1.23 \cdot 10^4$, $ulp(x) = 1 \cdot 10^2$

Erreur relative : $e_r = \left| \frac{r_{\text{exact}} - r_{\text{approché}}}{r_{\text{exact}}} \right|$

Elimination catastrophique $\Rightarrow$ erreur relative importante

aire du triangle : dans $\mathbb{R}$, $9.03 - 9.0 = 0.03$; dans $\mathbb{F}$, $9.04 - 9.0 = 0.04$;
$e_r = \left| \frac{0.03 - 0.04}{0.03} \right| = 33\%$

ϵ-machine, Erreurs d'ordre supérieur

- ϵ-machine : maximum de l'erreur relative due à un arrondi au voisinage d'un point x.

$$\epsilon = ulp(1) \qquad x = 1.23 \cdot 10^4,\ ulp(x) = 1 \cdot 10^2 = 10^4\epsilon$$

- Erreurs d'ordre supérieur :

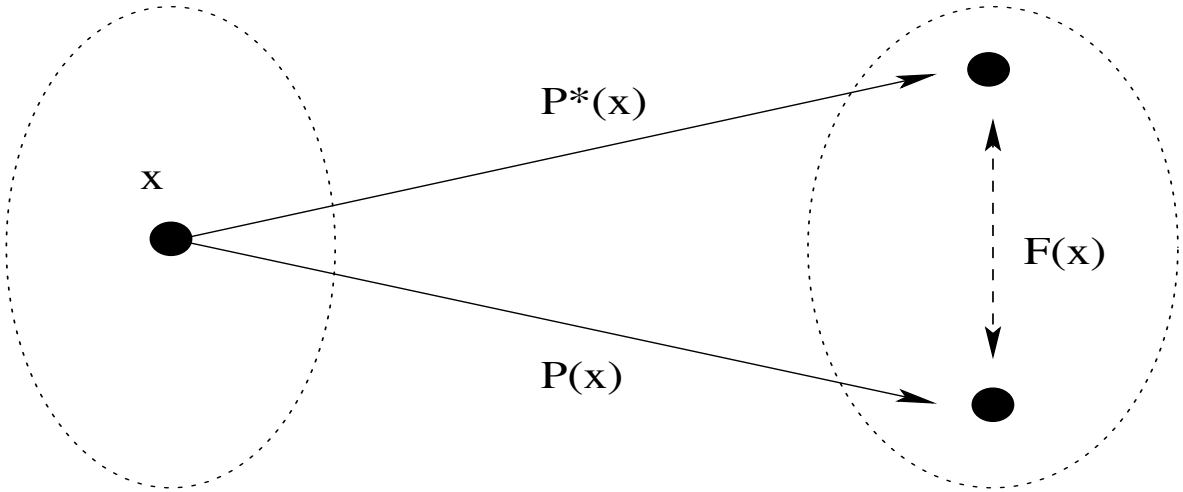
$$x_1 = f_1 + \epsilon_1,\ x_2 = f_2 + \epsilon_2, \quad x_1 \times x_2 = \underbrace{f_1 f_2}_{resultat} + \underbrace{f_1\epsilon_2 + f_2\epsilon_1}_{1er\ ordre} + \underbrace{\epsilon_1\epsilon_2}_{2d\ ordre}$$

- plus généralement, des erreurs d'ordre n peuvent apparaître au cours d'un calcul.
- les erreurs d'ordre supérieur sont généralement négligeables par rapport à celles d'ordre 1 mais $\exists$ des exceptions

Erreur en avant

$F(x) = \text{dist}(p(x), p^*(x))$

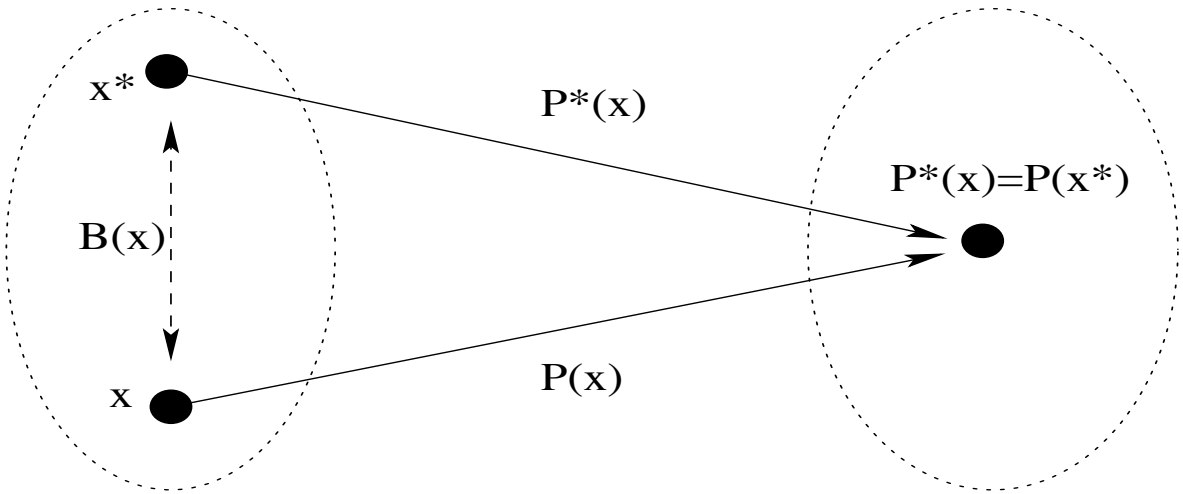
$\begin{cases} p & : \text{ calcul exact} \\ p^* & : \text{ calcul approché} \end{cases}$



Estime la précision de la solution en fonction de la précision d'une entrée particulière

Erreur en arrière

$$B(x) = \text{Inf } \left\{ \text{dist}(x, x^*) \ : \ x^* = p^{-1}(p^*(x)) \right\}$$



Détermine si une solution approchée est égale à la solution exacte d'un problème voisin

Plan

Généralités sur les nombres flottants

Détection : sémantique des séries d'erreurs

Correction : transformation sémantique d'expressions

Séries d'erreurs

Introduction

Sémantique générale

Restriction aux erreurs d'ordre n

Grain d'erreur

Utilisation de la Norme IEEE 754

Arrondi des nombres réels (vers 0, $-\infty$, $+\infty$ et au plus près) :

$$\uparrow_\circ : \mathbb{R} \to \mathbb{F}$$

Résultats des opérations : pour $\Diamond \in \{+, -, \times, \div, \sqrt{}\}$ on a :

$$x \Diamond_{\mathbb{F}} y = \uparrow_\circ (x \Diamond_{\mathbb{R}} y)$$

Erreur d'arrondi :

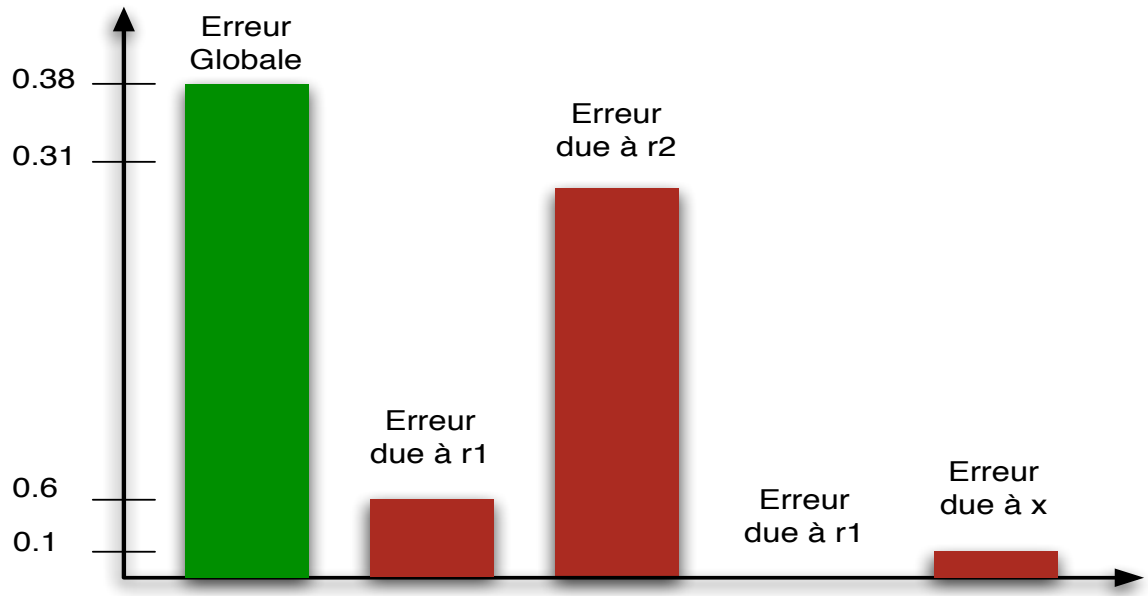
$$\downarrow_\circ : \mathbb{R} \to \mathbb{R} \qquad \downarrow_\circ (r) = r - \uparrow_\circ (r)$$

Exemple introductif

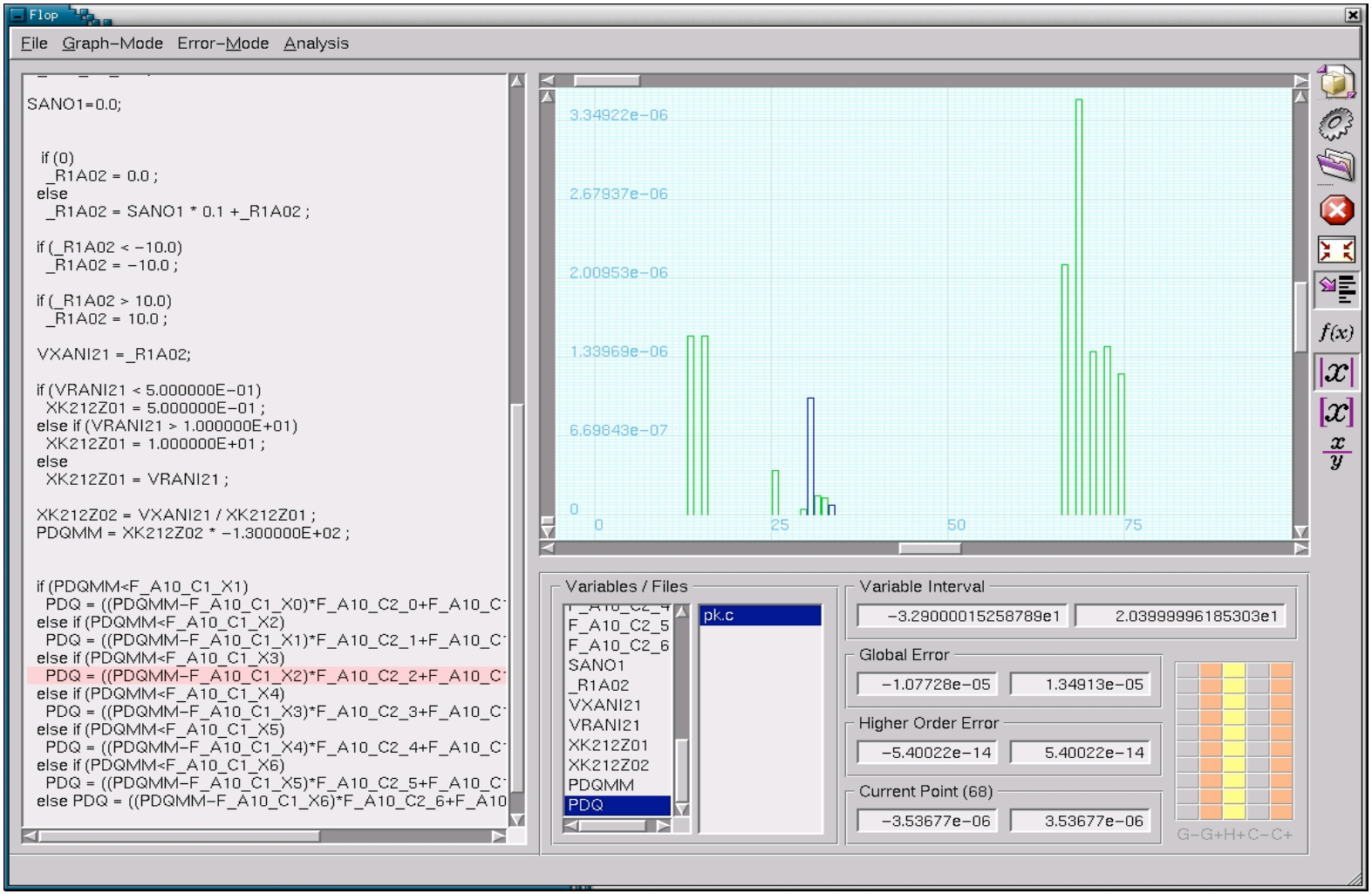
$$
\begin{array}{rrcrl}
 & 621.3\vec{\varepsilon} & + & 0.05\vec{\varepsilon}_{\ell_1} & r_1^{\ell_1} \\
\times^{\ell_3} & 1.287\vec{\varepsilon} & + & 0.0005\vec{\varepsilon}_{\ell_2} & r_2^{\ell_2} \\
\hline
= & 799.6131\vec{\varepsilon}\cdot\vec{\varepsilon} & & & \text{Résultat} \\
 & & + & 0.06435\vec{\varepsilon}\cdot\vec{\varepsilon}_{\ell_1} & \text{Erreur due à } r_1^{\ell_1} \\
 & & + & 0.31065\vec{\varepsilon}\cdot\vec{\varepsilon}_{\ell_2} & \text{Erreur due à } r_2^{\ell_2} \\
 & & + & 0.000025\vec{\varepsilon}_{\ell_1}\cdot\vec{\varepsilon}_{\ell_2} & \text{Terme du second ordre} \\
\hline
= & 799.6\vec{\varepsilon} & & & \text{Résultat machine } =\uparrow_\circ (r_1 \times r_2) \\
 & & + & 0.06435\vec{\varepsilon}_{\ell_1} & \text{Erreur due à } r_1^{\ell_1} \\
 & & + & 0.31065\vec{\varepsilon}_{\ell_2} & \text{Erreur due à } r_2^{\ell_2} \\
 & & + & 0.000025\vec{\varepsilon}_{\ell_1\ell_2} & \text{Terme du second ordre} \\
 & & + & 0.0131\vec{\varepsilon}_{\ell_3} & \text{Erreur due à } \times^{\ell_3} =\downarrow_\circ (r_1 \times r_2)
\end{array}
$$

Exemple Introductif (suite)

$$799.988125 = 799.6\vec{\varepsilon} + 0.06435\vec{\varepsilon}_{\ell_1} + 0.31065\vec{\varepsilon}_{\ell_2} + 0.000025\vec{\varepsilon}_{\ell_1\ell_2} + 0.0131\vec{\varepsilon}_{\ell_3}$$

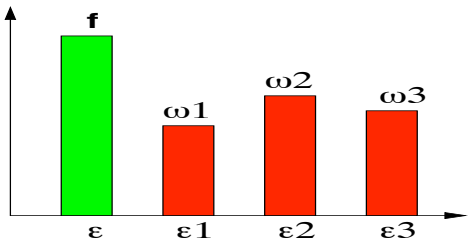


Fluctuat


Flop
File Graph-Mode Error-Mode Analysis
SANO1=0.0;
if (0)
_R1A02 = 0.0 ;
else
_R1A02 = SANO1 * 0.1 +_R1A02 ;
if (_R1A02 < -10.0)
_R1A02 = -10.0 ;
if (_R1A02 > 10.0)
_R1A02 = 10.0 ;
VXANI21 =_R1A02;
if (VRANI21 < 5.000000E-01)
XK212Z01 = 5.000000E-01 ;
else if (VRANI21 > 1.000000E+01)
XK212Z01 = 1.000000E+01 ;
else
XK212Z01 = VRANI21 ;
XK212Z02 = VXANI21 / XK212Z01 ;
PDQMM = XK212Z02 * -1.300000E+02 ;
if (PDQMM<F_A10_C1_X1)
PDQ = ((PDQMM-F_A10_C1_X0)*F_A10_C2_0+F_A10_C
else if (PDQMM<F_A10_C1_X2)
PDQ = ((PDQMM-F_A10_C1_X1)*F_A10_C2_1+F_A10_C
else if (PDQMM<F_A10_C1_X3)
PDQ = ((PDQMM-F_A10_C1_X2)*F_A10_C2_2+F_A10_C
else if (PDQMM<F_A10_C1_X4)
PDQ = ((PDQMM-F_A10_C1_X3)*F_A10_C2_3+F_A10_C
else if (PDQMM<F_A10_C1_X5)
PDQ = ((PDQMM-F_A10_C1_X4)*F_A10_C2_4+F_A10_C
else if (PDQMM<F_A10_C1_X6)
PDQ = ((PDQMM-F_A10_C1_X5)*F_A10_C2_5+F_A10_C
else PDQ = ((PDQMM-F_A10_C1_X6)*F_A10_C2_6+F_A10
3.34922e-06
2.67937e-06
2.00953e-06
1.33969e-06
6.69843e-07
0
0
25
50
75
f(x)
Variables / Files
F_A10_C2_5
F_A10_C2_6
SANO1
_R1A02
VXANI21
VRANI21
XK212Z01
XK212Z02
PDQMM
PDQ
pk.c
Variable Interval
-3.29000015258789e1
2.03999996185303e1
Global Error
-1.07728e-05
1.34913e-05
Higher Order Error
-5.40022e-14
5.40022e-14
Current Point (68)
-3.53677e-06
3.53677e-06
G-G+H+C-C+

Sémantique générale $[\![.]\!]^{\mathcal{L}^*}$

1^er ordre : $r^{\mathcal{L}^*} = f\vec{\varepsilon} + \sum_{\ell \in \mathcal{L}} \omega^\ell \vec{\varepsilon}_\ell$



- $f \in \mathbb{F}$ est le flottant utilisé par la machine au lieu de la valeur exacte r
- $\vec{\varepsilon}$ est une variable formelle toujours attachée au flottant f
- $\mathcal{L}$ ensemble des labels du programme
- $\vec{\varepsilon}_\ell$ variable formelle correspondant à l'erreur due au point $\ell \in \mathcal{L}$
- $\omega^\ell \in \mathbb{R}$ poids de l'erreur introduite au point $\ell \in \mathcal{L}$
- La valeur exacte dans $\mathbb{R}$ est : $r = f + \sum_{\ell \in \mathcal{L}} \omega^\ell$

Représentation des Erreurs d'Ordre Supérieur

- Erreur du 1^{er} ordre due au point ℓ attachée à la variable $\vec{\varepsilon}_\ell$
- Erreur du second ordre = produit de deux termes d'erreur du 1^{er} ordre, attaché à la variable formelle $\vec{\varepsilon}_{\ell_1 \ell_2}$

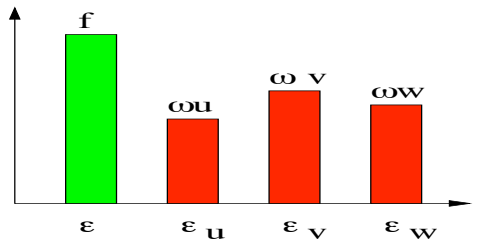
Exemple :

$$(f_1 \vec{\varepsilon} + \omega^{\ell_1} \vec{\varepsilon}_{\ell_1}) \times (f_2 \vec{\varepsilon} + \omega^{\ell_2} \vec{\varepsilon}_{\ell_2}) = f_1 f_2 \vec{\varepsilon} + f_2 \omega^{\ell_1} \vec{\varepsilon}_{\ell_1} + f_1 \omega^{\ell_2} \vec{\varepsilon}_{\ell_2} + \omega^{\ell_1} \omega^{\ell_2} \vec{\varepsilon}_{\ell_1 \ell_2}$$

- Plus généralement $\vec{\varepsilon}_{\ell_1 \ldots \ell_n}$ est la variable correspondant à l'erreur d'ordre n due aux points $\ell_1, \ldots, \ell_n$

Représentation des Erreurs d'Ordre Supérieur (2)

$$r^{\mathcal{L}^*} = f\vec{\varepsilon} + \sum_{u \in \overline{\mathcal{L}^+}} \omega^u \vec{\varepsilon}_u$$



- $f \in \mathbb{F}$ est le nombre flottant utilisé par la machine au lieu de la valeur exacte $r \in \mathbb{R}$
- $\overline{\mathcal{L}^+} \subseteq \mathcal{L}^*$ est un ensemble de mots sur l'alphabet $\mathcal{L}$ =Lab(prg)
- Pour tout mot $u = \ell_1 \ldots \ell_n \in \overline{\mathcal{L}^+}$, $\vec{\varepsilon}_u$ est une variable formelle correspondant à l'erreur d'ordre n due aux points $\ell_1, \ldots, \ell_n$
- $\omega^u \in \mathbb{R}$ est le coefficient de $\vec{\varepsilon}_u$
- La valeur exacte dans $\mathbb{R}$ est : $r = f + \sum_{u \in \overline{\mathcal{L}^+}} \omega^\ell$

Représentation des Erreurs d'Ordre Supérieur (3)

$$r^{\mathcal{L}^*} = f\vec{\varepsilon} + \sum_{u \in \overline{\mathcal{L}^+}} \omega^u \vec{\varepsilon}_u$$

- $\mathcal{L}$ ensemble des étiquettes, utilisé comme alphabet, $\mathcal{L}^*$ mots de $\mathcal{L}$
- ϵ mot vide, $\vec{\varepsilon} = \vec{\varepsilon}_\epsilon =$ variable attachée au flottant $f = \omega^\epsilon$
- $\mathcal{L}^+ = \mathcal{L} \setminus \{\epsilon\}$
- $\overline{\mathcal{L}^+}$ mots de $\mathcal{L}^+$ composés des mêmes lettres ($\vec{\varepsilon}_{\ell_1\ell_2} = \vec{\varepsilon}_{\ell_2\ell_1}$)

Opérations Elémentaires (Addition)

$$r_1 = f_1\vec{\varepsilon} + \sum_{u\in\overline{\mathcal{L}^+}} \omega_1^u \vec{\varepsilon}_u \qquad r_2 = f_2\vec{\varepsilon} + \sum_{u\in\overline{\mathcal{L}^+}} \omega_2^u \vec{\varepsilon}_u$$

$$r_1 +^{\ell_i} r_2 \overset{\text{def}}{=} \uparrow_\circ (f_1 + f_2)\vec{\varepsilon} + \sum_{u\in\overline{\mathcal{L}^+}} (\omega_1^u + \omega_2^u)\vec{\varepsilon}_u + \downarrow_\circ (f_1 + f_2)\vec{\varepsilon}_{\ell_i}$$

$$\begin{array}{rrcr} & 621.3\vec{\varepsilon} & + & 0.05\vec{\varepsilon}_{\ell_1} \\ +^{\ell_3} & 1.287\vec{\varepsilon} & + & 0.0005\vec{\varepsilon}_{\ell_2} \\ \hline = & 622.5\vec{\varepsilon} & & \\ & & + & 0.05\vec{\varepsilon}_{\ell_1} \\ & & + & 0.0005\vec{\varepsilon}_{\ell_2} \\ & & + & 0.087\vec{\varepsilon}_{\ell_3} \end{array}$$

$r_1^{\ell_1}$

$r_2^{\ell_2}$

Résultat machine $=\uparrow_\circ (r_1 + r_2)$

Erreur due à $r_1^{\ell_1}$

Erreur due à $r_2^{\ell_2}$

Erreur due à $+^{\ell_3} = \downarrow_\circ (r_1 + r_2)$

Opérations Elémentaires (Multiplication)

$$r_1 \times^{\ell_i} r_2 \overset{\text{def}}{=} \uparrow_\circ (f_1 f_2)\vec{\varepsilon} + \sum_{\substack{u \in \overline{\mathcal{L}^*},\ v \in \overline{\mathcal{L}^*} \\ |u.v| > 0}} \omega_1^u \omega_2^v \vec{\varepsilon}_{u.v} + \downarrow_\circ (f_1 f_2)\vec{\varepsilon}_{\ell_i}$$

	$621.3\vec{\varepsilon}$	$+$	$0.05\vec{\varepsilon}_{\ell_1}$	$r_1^{\ell_1}$
$\times^{\ell_3}$	$1.287\vec{\varepsilon}$	$+$	$0.0005\vec{\varepsilon}_{\ell_2}$	$r_2^{\ell_2}$
$=$	$799.6\vec{\varepsilon}$			Résultat machine $=\uparrow_\circ (r_1 \times r_2)$
		$+$	$0.06435\vec{\varepsilon}_{\ell_1}$	Erreur due à $r_1^{\ell_1}$
		$+$	$0.31065\vec{\varepsilon}_{\ell_2}$	Erreur due à $r_2^{\ell_2}$
		$+$	$0.000025\vec{\varepsilon}_{\ell_1 \ell_2}$	Terme du second ordre
		$+$	$0.0131\vec{\varepsilon}_{\ell_3}$	Erreur due à $\times^{\ell_3} = \downarrow_\circ (r_1 \times r_2)$

Cas de la division

$$\frac{1}{1+x} = \sum_{n\geq 0}(-1)^n x^n \text{ pour tout } x \text{ t.q. } -1 < x < 1$$

Nous avons :

$$\frac{1}{f+e} = \frac{1}{f} \times \frac{1}{1+\frac{e}{f}} = \frac{1}{f} \times \sum_{n\geq 0}(-1)^n \frac{e^n}{f^n}$$

et :

$$[\![\frac{1}{f\vec{\varepsilon}+e\vec{\varepsilon}_\ell}]\!]^{\mathcal{L}^*} = \uparrow_\circ \left(\frac{1}{f}\right)\vec{\varepsilon}_f + \left[\downarrow_\circ \left(\frac{1}{f}\right) + \sum_{n\geq 1}(-1)^n \frac{e^n}{f^{n+1}}\right]\vec{\varepsilon}_\ell$$

Valable pour $-1 < \frac{e}{f} < 1$ ou pour $|e| \leq |f|$, c.à.d. tant que l'erreur est inférieure au flottant.

Opérations Elémentaires (Division)

- Inverse : obtenu par un développement en séries (problème : convergence)

$$(r_1)^{-1^{\ell_i}} \overset{\text{def}}{=} \uparrow_\circ (f_1^{-1})\vec{\varepsilon} + \frac{1}{f_1}\sum_{n\geq 1}(-1)^n \left(\sum_{u\in\mathcal{W}^+} \frac{\omega_1^u}{f_1}\vec{\varepsilon}_u\right)^n + \downarrow_\circ (f_1^{-1})\vec{\varepsilon}_{\ell_i}$$

- Division : combine produit et inverse

$$r_1 \div^{\ell_i} r_2 \overset{\text{def}}{=} \uparrow_\circ \left(\frac{f_1}{f_2}\right)\vec{\varepsilon} + \sum_{u\in\mathcal{W}} \frac{\omega_1^u}{f_2}\sum_{n\geq 0}(-1)^n \left(\sum_{v\in\mathcal{W}^+} \frac{\omega_2^v}{f_2}\vec{\varepsilon}_v\right)^n \vec{\varepsilon}_u + \downarrow_\circ \left(\frac{f_1}{f_2}\right)\vec{\varepsilon}_{\ell_i}$$

- remarque : $r_1 \div^{\ell_i} r_2 \neq r_1 \times^{\ell_i} r_2^{-1^{\ell_i}}$

Convergence

- Rayon de convergence de la série

$$\frac{1}{1+x} = \sum_{n \geq 0} (-1)^n x^n \text{ pour tout } x \text{ t.q. } -1 < x < 1$$

- Pour $x = \sum_{u \in \mathcal{W}^+} \frac{\omega_1^u}{f_1}$ il faut

$$-1 < \sum_{u \in \mathcal{W}^+} \frac{\omega_1^u}{f_1} < 1$$

- c.à.d.

$$-f_1 < \sum_{u \in \mathcal{W}^+} \omega_1^u < f_1$$

- L'erreur globale doit être inférieure au flottant

Correction (1)

$$r_1 = \sum_{u \in \mathcal{W}} \omega_1^u \vec{\varepsilon}_u \qquad r_2 = \sum_{u \in \mathcal{W}} \omega_2^u \vec{\varepsilon}_u \qquad r = r_1 \Diamond^{\ell_i} r_2 = \sum_{u \in \mathcal{W}} \omega^u \vec{\varepsilon}_u$$

- Principe : montrer que $\mathbb{R}(r_1) \Diamond \mathbb{R}(r_2) = \mathbb{R}(r)$

$$\sum_{u \in \mathcal{W}} \omega^u = \left(\sum_{u \in \mathcal{W}} \omega_1^u \right) \Diamond \left(\sum_{u \in \mathcal{W}} \omega_2^u \right)$$

- **Trop faible, par exemple:**

$$(f_1 \vec{\varepsilon} + \omega^{\ell_1} \vec{\varepsilon}_{\ell_1}) +^{\ell_{\ell_i}} (f_2 \vec{\varepsilon} + \omega^{\ell_2} \vec{\varepsilon}_{\ell_2}) \overset{\text{faux!}}{=} \uparrow_\circ (f_1 + f_2) \vec{\varepsilon} + \omega^{\ell_1} \vec{\varepsilon}_{\ell_2} + \omega^{\ell_2} \vec{\varepsilon}_{\ell_1} + \downarrow_\circ (f_1 + f_2) \vec{\varepsilon}_{\ell_i}$$

Correction (2)

$$r_1 = \sum_{u \in \mathcal{W}} \omega_1^u \vec{\varepsilon}_u \qquad r_2 = \sum_{u \in \mathcal{W}} \omega_2^u \vec{\varepsilon}_u \qquad r = r_1 \Diamond^{\ell_i} r_2 = \sum_{u \in \mathcal{W}} \omega^u \vec{\varepsilon}_u$$

- Principe : étudier les variations des coefficients ω_1^u et ω_2^u
- Les variations de $\omega_1^{u_1} \dots \omega_1^{u_k} \dots \omega_2^{u_{k+1}} \dots \omega_2^{u_n}$ sont données par $\frac{\partial^n}{\partial \omega_{k_1}^{u_1} \dots \omega_{k_n}^{u_n}}$

Propriété

$$\frac{\partial (r_1 \Diamond r_2)}{\partial \omega_1^{u_0}} = \frac{\partial r}{\partial \omega_1^{u_0}} \quad \textit{et} \quad \frac{\partial (r_1 \Diamond r_2)}{\partial \omega_2^{u_0}} = \frac{\partial r}{\partial \omega_2^{u_0}}$$

Propriété

$$\frac{\partial^n (r_1 \Diamond r_2)}{\partial \omega_{k_1}^{u_1} \dots \omega_{k_n}^{u_n}} = \frac{\partial^n r}{\partial \omega_{k_1}^{u_1} \dots \omega_{k_n}^{u_n}}$$

Preuve (multiplication)

D'un côté nous avons : $\frac{\partial (r_1 \times^{\ell i} r_2)}{\partial \omega_1^{u_0}} = \frac{\partial}{\partial \omega_1^{u_0}} \left(\sum_{u,v \in \mathcal{W}} \omega_1^u \omega_2^v \vec{\varepsilon}_{uv} \right)$ En utilisant l'égalité

$$\sum_{u,v \in \mathcal{W}} \omega_1^u \omega_2^v \vec{\varepsilon}_{uv} = \sum_{v \in \mathcal{W}} \omega_1^{u_0} \omega_2^v \vec{\varepsilon}_{u_0 v} + \sum_{u \in \mathcal{W} \setminus \{u_0\}, v \in \mathcal{W}} \omega_1^u \omega_2^v \vec{\varepsilon}_{uv}$$

on obtient : $\frac{\partial (r_1 \times^{\ell i} r_2)}{\partial \omega_1^{u_0}} = \sum_{v \in \mathcal{W}} \omega_2^v \vec{\varepsilon}_{u_0 v}$ Par ailleurs,

$$\frac{\partial}{\partial \omega_1^{u_0}} \left(\sum_{u \in \mathcal{W}} \omega_1^u \vec{\varepsilon}_u \times \sum_{v \in \mathcal{W}} \omega_2^u \vec{\varepsilon}_u \right) =$$

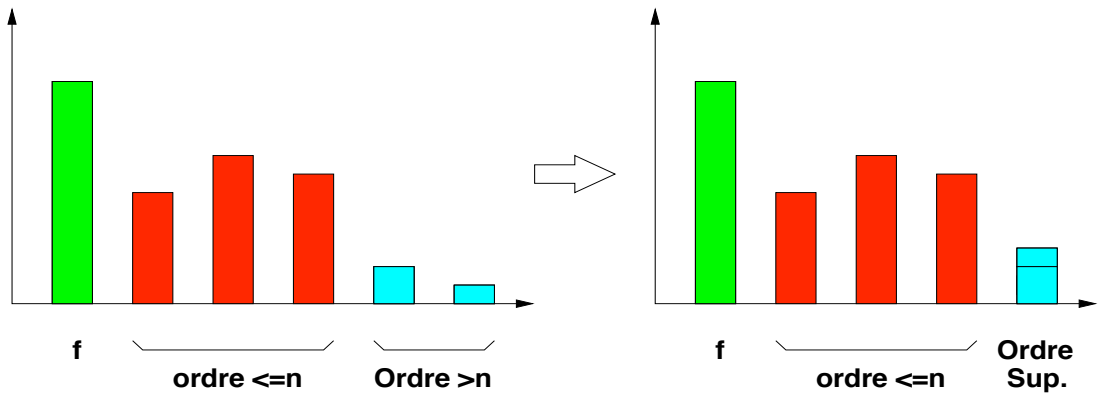
$$\sum_{u \in \mathcal{W}} \omega_1^u \vec{\varepsilon}_u \times \frac{\partial}{\partial \omega_1^{u_0}} \left(\sum_{v \in \mathcal{W}} \omega_2^v \vec{\varepsilon}_v \right) + \sum_{v \in \mathcal{W}} \omega_2^v \vec{\varepsilon}_v \times \frac{\partial}{\partial \omega_1^{u_0}} \left(\sum_{u \in \mathcal{W}} \omega_1^u \vec{\varepsilon}_u \right) =$$

$$0 + \left(\sum_{v \in \mathcal{W}} \omega_2^v \vec{\varepsilon}_v \right) \times \vec{\varepsilon}_{u_0} = \sum_{v \in \mathcal{W}} \omega_2^v \vec{\varepsilon}_{u_0 v}$$

Restriction à l'Ordre n : $[\![.]\!]^{\mathcal{L}^n}$

- n opérations introduisent éventuellement des erreurs d'ordre n
  - Comment réduire la taille des valeurs?
- En pratique :
  - Seules les erreurs du premier ordre sont significatives
  - Très rarement, celles d'ordre 2 ne sont pas négligeables
- Principe :
  - Détailler la provenance des erreurs des n premiers ordres uniquement
  - Vérifier que les erreurs d'ordre supérieur sont globalement négligeables

Erreurs d'Ordre n : Principe



- Détaille la contribution des erreurs d'ordre $\leq n$
- Indique globalement le poids des erreurs $> n$
  - On a $\omega_{hi} =$ somme des erreurs d'ordre $> n$

Opérations Elémentaires ($\mathcal{W} = \overline{\mathcal{L}^n}$)

- $\overline{\mathcal{L}^n} = \{u \in \overline{\mathcal{L}^*} \ : \ |u| \leq n\} \cup \{hi\}$
- Concaténation : $u \cdot_n v = \begin{cases} u.v \text{ si } |u.v| \leq n \\ hi \text{ sinon} \end{cases}$

$$r_1 +^{\ell_i} r_2 \stackrel{\text{def}}{=} \uparrow_\circ (f_1 + f_2)\vec{\varepsilon} + \sum_{u \in \mathcal{W}^+} (\omega_1^u + \omega_2^u)\vec{\varepsilon}_u + \downarrow_\circ (f_1 + f_2)\vec{\varepsilon}_{\ell_i}$$

$$r_1 \times^{\ell_i} r_2 \stackrel{\text{def}}{=} \uparrow_\circ (f_1 f_2)\vec{\varepsilon} + \sum_{\substack{u \in \mathcal{W},\ v \in \mathcal{W} \\ |u.v| > 0}} \omega_1^u \omega_2^v \vec{\varepsilon}_{u.v} + \downarrow_\circ (f_1 f_2)\vec{\varepsilon}_{\ell_i}$$

Relation entre $[\![.]\!]^{\mathcal{L}^n}$ et $[\![.]\!]^{\mathcal{L}^m}$

- $[\![.]\!]^{\mathcal{L}^n}$ détaille la provenance des erreurs jusqu'à l'ordre n et calcule la somme des erreurs d'ordre $> n$
- Pour $m < n$ la sémantique à l'ordre m est une approximation de celle à l'ordre n :

$$\langle \wp(\mathcal{F}(\mathbb{R}, \overline{\mathcal{L}^n})), \subseteq \rangle \quad \underset{\alpha^{n,m}}{\overset{\gamma^{m,n}}{\leftrightarrows}} \quad \langle \wp(\mathcal{F}(\mathbb{R}, \overline{\mathcal{L}^m})), \subseteq \rangle$$

- On obtient une chaine de sémantiques de plus en plus précises :

$$[\![.]\!]^{\mathcal{L}^*}(a_0^{\ell_0}) \leftrightarrows \ldots [\![.]\!]^{\mathcal{L}^n}(a_0^{\ell_0}) \leftrightarrows [\![.]\!]^{\mathcal{L}^{n-1}}(a_0^{\ell_0}) \ldots \leftrightarrows [\![.]\!]^{\mathcal{L}^0}(a_0^{\ell_0})$$

Relation entre $[\![.]\!]^{\mathcal{L}^n}$ et $[\![.]\!]^{\mathcal{L}^m}$

$$\alpha^{n,m}\left(\sum_{u\in\overline{\mathcal{L}^n}}\omega^u\vec{\varepsilon}_u\right)\stackrel{\text{def}}{=}\sum_{u\in\overline{\mathcal{L}^m}\setminus\{hi\}}\omega^u\vec{\varepsilon}_u+\left(\sum_{u\in(\overline{\mathcal{L}^n}\setminus\overline{\mathcal{L}^m})\cup\{hi\}}\omega^u\right)\vec{\varepsilon}_{hi}$$

$$\gamma^{m,n}\left(\sum_{u\in\overline{\mathcal{L}^m}}\nu^u\vec{\varepsilon}_u\right)\stackrel{\text{def}}{=}\left\{\sum_{u\in\overline{\mathcal{L}^n}}\omega^u\vec{\varepsilon}_u\ :\ \left|\begin{array}{l}\omega^u=\nu^u\text{ if }u\in\overline{\mathcal{L}^m}\setminus\{hi\}\\ \sum_{u\in(\overline{\mathcal{L}^n}\setminus\overline{\mathcal{L}^m})\cup\{hi\}}\omega^u=\nu^{hi}\end{array}\right.\right\}$$

$$\alpha^{n,m}(X)=\{\alpha^{n,m}(x)\ :\ x\in X\}\quad\gamma^{m,n}(X)=\cup_{x\in X}\gamma^{m,n}(x)$$

Correction

Propriété

Soit ℓ_i un point de contrôle, soit $r^{\mathcal{L}^n}, s^{\mathcal{L}^n} \in \mathcal{F}(\mathbb{R}, \overline{\mathcal{L}^n})$, des séries d'erreurs telles que $r^{\mathcal{L}^m} = \alpha^{n,m}(r^{\mathcal{L}^n})$, $s^{\mathcal{L}^m} = \alpha^{n,m}(s^{\mathcal{L}^n})$, $1 \leq m \leq n$. Pour toute opération $\Diamond \in \{+, -, \times, \div\}$ nous avons

$$r^{\mathcal{L}^n} \Diamond^{\ell_i} s^{\mathcal{L}^n} \in \gamma^{m,n}(r^{\mathcal{L}^m} \Diamond^{\ell_i} s^{\mathcal{L}^m})$$

Preuve (multiplication)

Soit

$$r^{\mathcal{L}^n} = \sum_{u \in \overline{\mathcal{L}^n}} \omega_r^u \vec{\varepsilon}_u$$

et

$$s^{\mathcal{L}^n} = \sum_{u \in \overline{\mathcal{L}^n}} \omega_s^u \vec{\varepsilon}_u$$

On a :

$$\begin{array}{rcl} t^{\mathcal{L}^n} & = & r^{\mathcal{L}^n} \times^{\ell_i} s^{\mathcal{L}^n} \\ & = & \uparrow_\circ (\omega_r^\epsilon \omega_s^\epsilon) \vec{\varepsilon}_\epsilon + \sum_{\substack{u,v \in \overline{\mathcal{L}^n} \\ |u.v|>0}} \omega_r^u \omega_s^v \vec{\varepsilon}_{u.v} + \downarrow_\circ (\omega_r^\epsilon \omega_s^\epsilon) \vec{\varepsilon}_{\ell_i} \end{array} \quad (2)$$

De même, si $r^{\mathcal{L}^m} = \sum_{u \in \overline{\mathcal{L}^m}} \nu_r^u \vec{\varepsilon}_u$ et $s^{\mathcal{L}^m} = \sum_{u \in \overline{\mathcal{L}^m}} \nu_s^u \vec{\varepsilon}_u$ alors

$$\begin{array}{rcl} t^{\mathcal{L}^m} & = & r^{\mathcal{L}^m} \times^{\ell_i} s^{\mathcal{L}^m} \\ & = & \uparrow_\circ (\nu_r^\epsilon \nu_s^\epsilon) \vec{\varepsilon}_\epsilon + \sum_{\substack{u,v \in \overline{\mathcal{L}^m} \\ |u.v|>0}} \nu_r^u \nu_s^v \vec{\varepsilon}_{u.v} + \downarrow_\circ (\nu_r^\epsilon \nu_s^\epsilon) \vec{\varepsilon}_{\ell_i} \end{array} \quad (3)$$

Preuve (multiplication)

Par definition de $\gamma^{m,n}$,

$$\gamma^{m,n}(t^{\mathcal{L}^m}) = \left\{ \sum_{u \in \overline{\mathcal{L}^n}} \omega^u \vec{\varepsilon}_u \ : \ \left| \begin{array}{l} \omega^u = \nu_t^u \text{ if } u \in \overline{\mathcal{L}^m} \setminus \{hi\} \\ \sum_{u \in (\overline{\mathcal{L}^n} \setminus \overline{\mathcal{L}^m}) \cup \{hi\}} \omega^u = \nu_t^{hi} \end{array} \right. \right\} \tag{4}$$

où, dans (4),

$$\nu_t^u = \sum_{u_1 u_2 = u} \nu_r^{u_1} \nu_s^{u_2}$$

et

$$\nu_t^{hi} = \sum_{u_1 u_2 = u \in (\overline{\mathcal{L}^n} \setminus \overline{\mathcal{L}^m}) \cup \{hi\}} \nu_r^{u_1} \nu_s^{u_2}$$

Nous devons montrer que $\sum_{u \in (\overline{\mathcal{L}^n} \setminus \overline{\mathcal{L}^m}) \cup \{hi\}} \omega_t^u = \nu_t^{hi}$. Notation : $M = \overline{\mathcal{L}^m} \setminus \{hi\}$ et $N = (\overline{\mathcal{L}^n} \setminus \overline{\mathcal{L}^m}) \cup \{hi\}$. On a :

$$\begin{aligned} \sum_{u \in N} \omega_t^u &= \sum_{\substack{u \in \overline{\mathcal{L}^n},\, v \in \overline{\mathcal{L}^n} \\ u.v \in N}} \omega_r^u \omega_s^v \\ &= \sum_{\substack{u,\, v \in M \\ u.v \in N}} \omega_r^u \omega_s^v + \sum_{u,\, v \in N} \omega_r^u \omega_s^v \\ &+ \sum_{\substack{u \in M,\, v \in N \\ u.v \in N}} \omega_r^u \omega_s^v + \sum_{\substack{u \in N,\, v \in M \\ u.v \in N}} \omega_r^u \omega_s^v \end{aligned}$$

Preuve (multiplication)

Puisque $r^{\mathcal{L}^m} = \alpha^{n,m}(r^{\mathcal{L}^n})$ et $s^{\mathcal{L}^m} = \alpha^{n,m}(s^{\mathcal{L}^n})$, $\sum_{u \in N} \omega_r^u = \nu_r^{hi}$ et $\sum_{u \in N} \omega_s^u = \nu_s^{hi}$. Donc,

$$\sum_{u \in N} \omega_t^u = \sum_{\substack{u,\, v \in M \\ u.v \in N}} \omega_r^u \omega_s^v + \nu_r^{hi} \nu_s^{hi} + \sum_{u \in M} \omega_r^u \nu_s^{hi} + \sum_{v \in M} \nu_r^{hi} \omega_s^v$$

Puisque $r^{\mathcal{L}^m} = \alpha^{n,m}(r^{\mathcal{L}^n})$ et $s^{\mathcal{L}^m} = \alpha^{n,m}(s^{\mathcal{L}^n})$, pour tout mot u tel que $|u| \leq m$, nous avons $\omega_r^u = \nu_r^u$ et $\omega_s^u = \nu_s^u$, donc

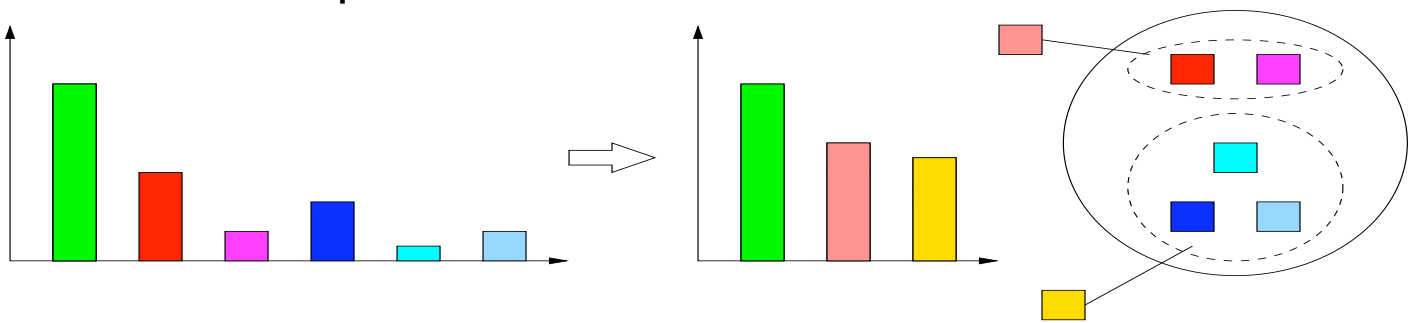
$$\begin{aligned} \sum_{u \in N} \omega_t^u &= \sum_{\substack{u,\, v \in M \\ u.v \in N}} \nu_r^u \nu_s^v + \nu_r^{hi} \nu_s^{hi} + \sum_{u \in M} \nu_r^u \nu_s^{hi} + \sum_{v \in M} \nu_r^{hi} \nu_s^v \\ &= \sum_{\substack{u,\, v \in \overline{\mathcal{L}^m} \\ u.v \in N}} \nu_r^u \nu_s^v = \nu_t^{hi} \end{aligned}$$

Grain d'Erreur : $[\![.]\!]^{\mathcal{J}^n}$

- But : limiter le nombre de termes dans les séries d'erreurs
- Principe :
  - Ne plus considérer les erreurs introduites par des opérations élémentaires
  - Calculer les erreurs due à des morceaux de code partitionnant le programme
- Exemples :
  - Erreur due à une formule intermédiaire
  - Erreur introduite par un bloc de programme C
  - $[\![.]\!]^{\mathcal{L}^*}$, $[\![.]\!]^{\mathcal{L}^n}$ correspondent à une partition particulière, celle des singletons de Lab(prg)

Grain d'Erreur : Principe

- $\mathcal{J} = \{J_1, \ldots J_m\}$ partition de l'ensemble $\mathcal{L}$ des étiquettes
- Pour les termes du premier ordre:

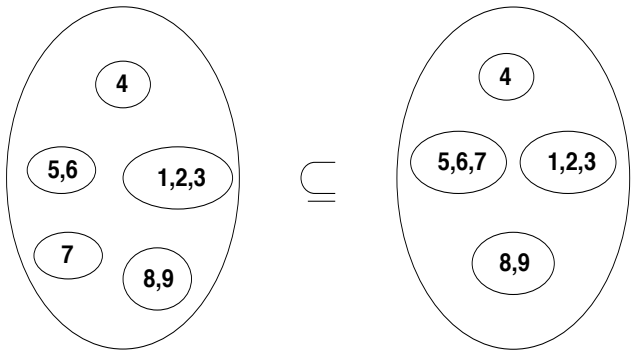


- Pour les termes d'ordre supérieur : $\omega^{J_1 J_2} = \sum_{\ell_1 \in J_1, \ell_2 \in J_2} \omega^{\ell_1} \omega^{\ell_2}$
- Sémantique des opérations : $\mathcal{W} = \overline{\mathcal{T}^n}$

$$r_1 +^{\ell_i} r_2 \stackrel{\text{def}}{=} \uparrow_\circ (f_1 + f_2)\vec{\varepsilon} + \sum_{u \in \mathcal{W}^+} (\omega_1^u + \omega_2^u)\vec{\varepsilon}_u + \downarrow_\circ (f_1 + f_2)\vec{\varepsilon}_{\ell_i}$$

Correction de $[\![.]\!]^{\mathcal{I}^n}$

- Ordre partiel $\dot{\subseteq}$: $\mathcal{I}_1$ est plus précis que $\mathcal{I}_2$ si $\mathcal{I}_2$ regroupe des éléments de $\mathcal{I}_1$



- La sémantique $[\![.]\!]^{\mathcal{I}_2^n}$ utilisant la partition $\mathcal{I}_2$ t.q. $\mathcal{I}_1 \dot{\subseteq} \mathcal{I}_2$ est une approximation de la sémantique $[\![.]\!]^{\mathcal{I}_1^n}$

$$\langle \wp(\mathcal{F}(\mathbb{R}, \overline{\mathcal{I}_1^n})), \subseteq \rangle \quad \underset{\alpha^{\mathcal{I}_1^n, \mathcal{I}_2^n}}{\overset{\gamma^{\mathcal{I}_2^n, \mathcal{I}_1^n}}{\leftrightarrows}} \quad \langle \wp(\mathcal{F}(\mathbb{R}, \overline{\mathcal{I}_2^n})), \subseteq \rangle$$

Correction de $[\![.]\!]^{\mathcal{J}^n}$

$$\langle \wp(\mathcal{F}(\mathbb{R}, \overline{\mathcal{J}_1^n})), \subseteq \rangle \quad \underset{\alpha^{\mathcal{J}_1^n, \mathcal{J}_2^n}}{\overset{\gamma^{\mathcal{J}_2^n, \mathcal{J}_1^n}}{\leftrightarrows}} \quad \langle \wp(\mathcal{F}(\mathbb{R}, \overline{\mathcal{J}_2^n})), \subseteq \rangle$$

$$\tau_{\mathcal{J}_1^n, \mathcal{J}_2^n}(J_1.u) = J_2.\tau_{\mathcal{J}_1^n, \mathcal{J}_2^n}(u) \text{ avec } J_1 \in \mathcal{J}_1,\ J_1 \subseteq J_2,\ J_2 \in \mathcal{J}_2$$

$$\alpha^{\mathcal{J}_1^n, \mathcal{J}_2^n}\left(\sum_{u \in \overline{\mathcal{J}_1^n}} \omega_i^u \vec{\varepsilon}_u\right) \overset{\text{def}}{=} \sum_{u \in \overline{\mathcal{J}_1^n}} \omega^u \vec{\varepsilon}_{\tau_{\mathcal{J}_1^n, \mathcal{J}_2^n}(u)}$$

$$\gamma^{\mathcal{J}_2^n, \mathcal{J}_1^n}\left(\sum_{v \in \overline{\mathcal{J}_2^n}} \nu^u \vec{\varepsilon}_v\right) \overset{\text{def}}{=} \left\{ \sum_{u \in \overline{\mathcal{J}_1^n}} \omega^u \vec{\varepsilon}_u \ : \sum_{\tau_{\mathcal{J}_1^n, \mathcal{J}_2^n}(u) = v} \omega^u = \nu^v \right\}$$

Correction

Propriété

Soit ℓ_i un point de contrôle, soit $\mathcal{J}_1$ et $\mathcal{J}_2$ des partitions de $\mathcal{L}$ telles que $\mathcal{J}_1 \dot{\subseteq} \mathcal{J}_2$ et soient $r^{\mathcal{J}_1^n}, s^{\mathcal{J}_1^n} \in \mathcal{F}(\mathbb{R}, \overline{\mathcal{J}_1^n})$. Si $r^{\mathcal{J}_2^n} = \alpha^{\mathcal{J}_1^n, \mathcal{J}_2^n}(r^{\mathcal{J}_1^n})$, $s^{\mathcal{J}_2^n} = \alpha^{\mathcal{J}_1^n, \mathcal{J}_2^n}(s^{\mathcal{J}_1^n})$ alors pour tout opérateur $\Diamond \in \{+, -, \times, \div\}$ nous avons :

$$r^{\mathcal{J}_1^n} \Diamond^{\ell_i} s^{\mathcal{J}_1^n} \in \gamma^{\mathcal{J}_2^n, \mathcal{J}_1^n}(r^{\mathcal{J}_2^n} \Diamond^{\ell_i} s^{\mathcal{J}_2^n})$$

Preuve (multiplication)

On utilise les notations :

$$r^{\mathcal{J}_1^n} = \sum_{u \in \overline{\mathcal{J}_1^n}} \omega_r^u \vec{\varepsilon}_u, \qquad s^{\mathcal{J}_1^n} = \sum_{u \in \overline{\mathcal{J}_1^n}} \omega_s^u \vec{\varepsilon}_u,$$

$$r^{\mathcal{J}_2^n} = \sum_{u \in \overline{\mathcal{J}_2^n}} \nu_r^u \vec{\varepsilon}_u, \qquad s^{\mathcal{J}_2^n} = \sum_{u \in \overline{\mathcal{J}_2^n}} \nu_s^u \vec{\varepsilon}_u.$$

Soit $\tau(u) = \tau_{\mathcal{J}_1^n, \mathcal{J}_2^n}(u)$. Nous avons :

$$t^{\mathcal{J}_1^n} = r^{\mathcal{J}_1^n} \times^{\ell_i} s^{\mathcal{J}_1^n} = \uparrow_\circ (\omega_r^\epsilon \omega_s^\epsilon) \vec{\varepsilon}_\epsilon + \sum_{\substack{u, v \in \overline{\mathcal{J}_1^n} \\ |u.v| > 0}} \omega_r^u \omega_s^v \vec{\varepsilon}_{u.v} + \downarrow_\circ (\omega_r^\epsilon \omega_s^\epsilon) \vec{\varepsilon}_{\ell_i}$$

$$t^{\mathcal{J}_2^n} = r^{\mathcal{J}_2^n} \times^{\ell_i} s^{\mathcal{J}_2^n} = \uparrow_\circ (\nu_r^\epsilon \nu_s^\epsilon) \vec{\varepsilon}_\epsilon + \sum_{\substack{u, v \in \overline{\mathcal{J}_2^n} \\ |u.v| > 0}} \nu_r^u \nu_s^v \vec{\varepsilon}_{u.v} + \downarrow_\circ (\nu_r^\epsilon \nu_s^\epsilon) \vec{\varepsilon}_{\ell_i}$$

Le point principal de la preuve consiste à montrer que pour tout $u \in \overline{\mathcal{J}_2^n}$, $\sum_{\tau(v)=u} \omega_t^v = \nu_t^u$.

Preuve (multiplication)

Le point principal de la preuve consiste à montrer que pour tout $u \in \overline{\mathcal{T}_2^n}$, $\sum_{\tau(v)=u} \omega_t^v = \nu_t^u$.

$$\begin{aligned}
\sum_{\tau(v)=u} \omega_t^v &= \sum_{\tau(v_1 \cdot v_2)=u} \omega_r^{v_1} \omega_s^{v_2} = \sum_{\substack{\tau(v_1).\tau(v_2) = u_1.u_2 \\ u_1.u_2 = u}} \omega_r^{v_1} \omega_s^{v_2} \\
&= \sum_{u_1 \cdot u_2 = u} \left(\sum_{\substack{\tau(v_1) = u_1 \\ \tau(v_2) = u_2}} \omega_r^{v_1} \omega_s^{v_2} \right) \\
&= \sum_{u_1 \cdot u_2 = u} \left(\sum_{\tau(v_1)=u_1} \omega_r^{v_1} \times \sum_{\tau(v_2)=u_2} \omega_s^{v_2} \right) = \sum_{u_1 \cdot u_2 = u} \nu_r^{u_1} \nu_s^{u_2} = \nu_t^u
\end{aligned}$$

Relation Entre les Différentes Sémantiques

$$
\begin{array}{ccccccc}
[.]^{\mathcal{J}_0^*}(a_0^{\ell 0}) & \leftrightarrows & \cdots & \underset{\alpha^{n+1,n}}{\overset{\gamma^{n,n+1}}{\longleftrightarrows}} & [.]^{\mathcal{J}_0^n}(a_0^{\ell 0}) & \underset{\alpha^{n,n-1}}{\overset{\gamma^{n-1,n}}{\longleftrightarrows}} \cdots & \leftrightarrows & [.]^{\mathcal{J}_0^0}(a_0^{\ell 0}) \\
\uparrow\downarrow & & \uparrow\downarrow & & {}_{\alpha}\mathcal{J}_1^n,\mathcal{J}_0^n \uparrow\downarrow {}_{\gamma}\mathcal{J}_0^n,\mathcal{J}_1^n & & \uparrow\downarrow & & \uparrow\downarrow \\
[.]^{\mathcal{J}_1^*}(a_0^{\ell 0}) & \leftrightarrows & \cdots & \leftrightarrows & [.]^{\mathcal{J}_1^n}(a_0^{\ell 0}) & \leftrightarrows \cdots & \leftrightarrows & [.]^{\mathcal{J}_1^0}(a_0^{\ell 0}) \\
\uparrow\downarrow & & \uparrow\downarrow & & \uparrow\downarrow & & \uparrow\downarrow & & \uparrow\downarrow \\
\cdots & \leftrightarrows & \cdots & \leftrightarrows & \cdots & \leftrightarrows & \cdots & \leftrightarrows & \cdots \\
\uparrow\downarrow & & \uparrow\downarrow & & \uparrow\downarrow & & \uparrow\downarrow & & \uparrow\downarrow \\
[.]^{\mathcal{L}^*}(a_0^{\ell 0}) & \leftrightarrows & \cdots & \leftrightarrows & [.]^{\mathcal{L}^n}(a_0^{\ell 0}) & \leftrightarrows \cdots & \leftrightarrows & [.]^{\mathcal{L}^0}(a_0^{\ell 0})
\end{array}
$$

Partitionnement des points de contrôle

Analyse statique fondée sur $[\![.]\!]^{\mathcal{T}^n}$:

intervalles de flottants pour le terme flottant

intervalles multi-précision pour les termes d'erreurs

choix d'une partition

Le choix d'une partition a une grande importance sur la précision de l'analyse

Abstraction par des intervalles

$$\langle \wp(\mathcal{F}(\mathbb{R}, \overline{\mathcal{T}^n})), \subseteq \rangle \quad \underset{\alpha^{\mathcal{I}}}{\overset{\gamma^{\mathcal{I}}}{\leftrightarrows}} \quad \langle \mathcal{F}(\mathcal{I}_{\mathbb{F}}, \overline{\mathcal{T}^n}), \sqsubseteq \rangle$$

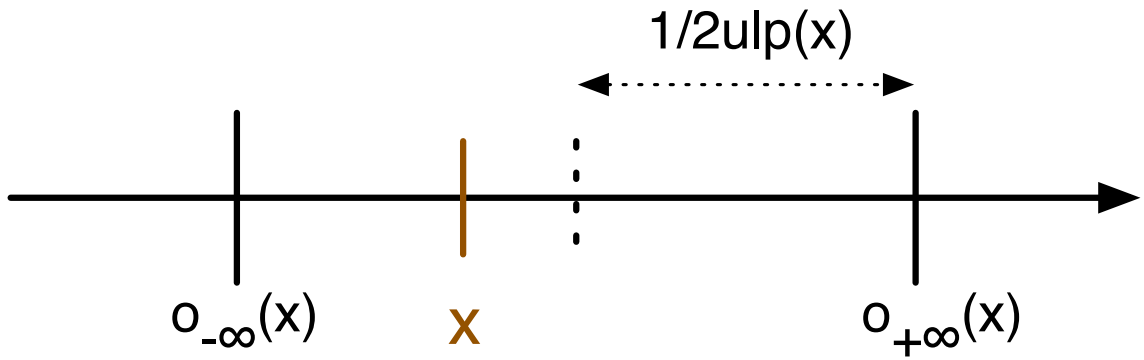
$$\sum_{u \in \overline{\mathcal{T}^n}} \omega_1^u \vec{\varepsilon}_u \sqsubseteq \sum_{u \in \overline{\mathcal{T}^n}} \omega_2^u \vec{\varepsilon}_u \; ; \iff ; \; \forall u \in \overline{\mathcal{T}^n}, \; \omega_1^u \subseteq \omega_2^u$$

$$\alpha^{\mathcal{I}} \left(\left\{ \sum_{u \in \overline{\mathcal{T}^n}} \omega_i^u \vec{\varepsilon}_u \; : \; i \in I \right\} \right) \overset{\text{def}}{=} \sum_{u \in \overline{\mathcal{T}^n}} \Phi \left(\{ \omega_i^u \; : \; i \in I \} \right) \vec{\varepsilon}_u$$

$$\gamma^{\mathcal{I}} \left(\sum_{u \in \overline{\mathcal{T}^n}} \nu^u \vec{\varepsilon}_u \right) \overset{\text{def}}{=} \left\{ \sum_{u \in \overline{\mathcal{T}^n}} \omega^u \vec{\varepsilon}_u \; : \; \forall u \in \overline{\mathcal{T}^n}, \; \omega^u \in \nu^u \right\}$$

Encadrement des erreurs

Erreurs d'arrondi :



Dans l'arithmétique des intervalles :

x borné par $[\circ_{-\infty}(x), \circ_{+\infty}(x)]$ with error $[-\frac{1}{2}ulp(x), +\frac{1}{2}ulp(x)]$

```
 -] 0.1
ans =
     float64:  [9.999999999999999E-2,1.000000000000001E-1]
       error:  [-6.938893903907228E-18,6.938893903907229E-18]

-] [2.0,3.0,0.0,0.05]
ans =
     float64:  [2.000000000000000E0,3.000000000000000E0]
       error:  [0.000000000000000E-1,5.000000000000001E-2]
```

Propagation des erreurs

Opérations élémentaires :

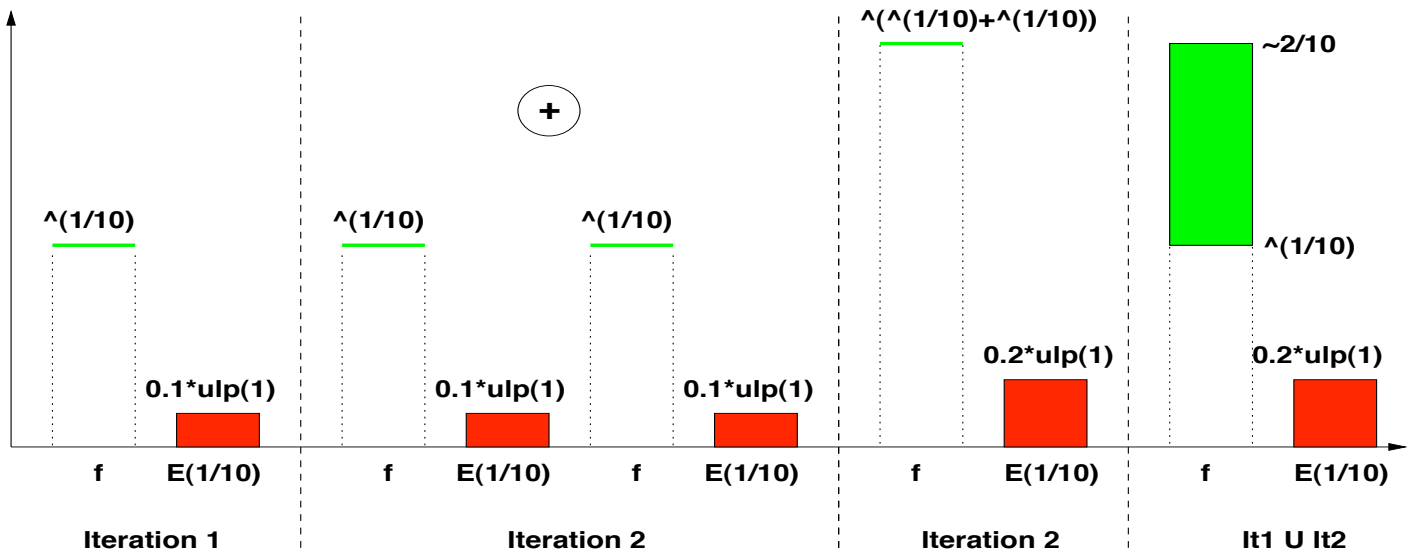
$$
\begin{array}{rcc}
 & [x_1] & , \quad [\varepsilon_1] \\
+ & [x_2] & , \quad [\varepsilon_2] \\
\hline
= & [x_1] \oplus_{\sim} [x_2] & , \quad \begin{array}{c} [\varepsilon_1] \oplus_{\leftrightarrow} [\varepsilon_2] \\ \oplus_{\leftrightarrow} \\ \left[\pm \frac{1}{2} ulp([x_1] +_{\sim} [x_2]) \right] \end{array}
\end{array}
$$

$$
\begin{array}{rcc}
 & [x_1] & , \quad [\varepsilon_1] \\
\times & [x_2] & , \quad [\varepsilon_2] \\
\hline
= & [x_1] \otimes_{\sim} [x_2] & , \quad \begin{array}{c} [\varepsilon_1] \otimes_{\leftrightarrow} [x_2] \\ \oplus_{\leftrightarrow} \\ [\varepsilon_2] \otimes_{\leftrightarrow} [x_1] \\ \oplus_{\leftrightarrow} \\ [\varepsilon_1] \otimes_{\leftrightarrow} [\varepsilon_2] \\ \oplus_{\leftrightarrow} \\ \left[\pm \frac{1}{2} ulp([x_1] \otimes_{\sim} [x_2]) \right] \end{array}
\end{array}
$$

```
 -] 0.1
ans =
     float64:  [9.999999999999999E-2,1.000000000000001E-1]
       error:  [-6.938893903907228E-18,6.938893903907229E-18]
-] 0.2
ans =
     float64:  [1.999999999999999E-1,2.000000000000001E-1]
       error:  [-1.387778780781445E-17,1.387778780781446E-17]
-] 0.1+0.2
ans =
     float64:  [2.999999999999999E-1, 3.000000000000001E-1]
       error:  [-4.857225732735059E-17, 4.857225732735060E-17]
```

Exemple

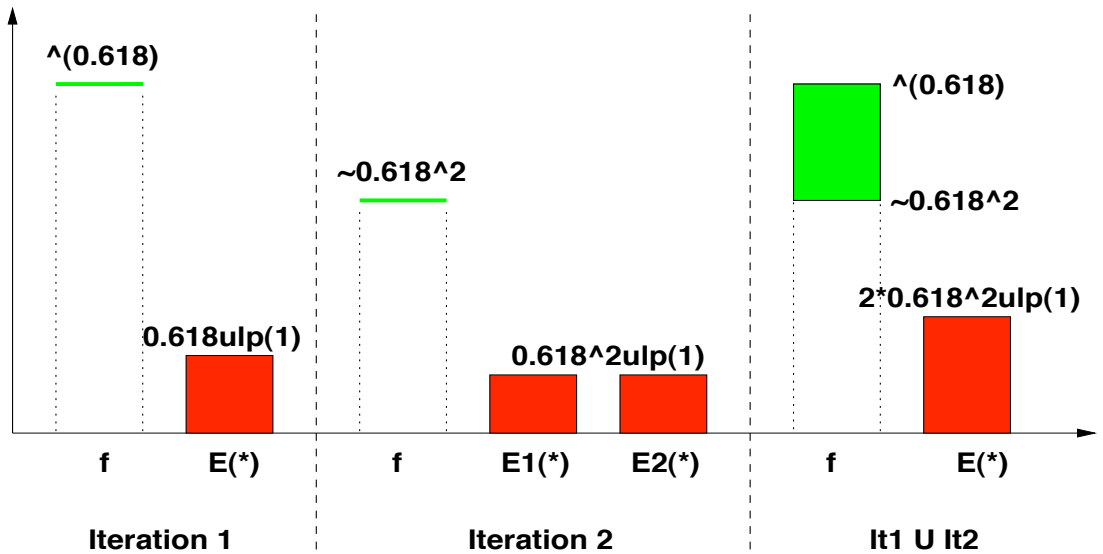
```
t = 0.0;
for (;;)
  t = t+1/10;
```



$t_1 < t_2$, par widening on obtient : $t = [\uparrow_\circ (1/10), +\infty]\vec{\varepsilon} + [-\infty, +\infty]\vec{\varepsilon}_{1/10}$

Autre Exemple

```
t = 1.0;
for (i=1; i<=20; i++)
  t = t*0.618;
```



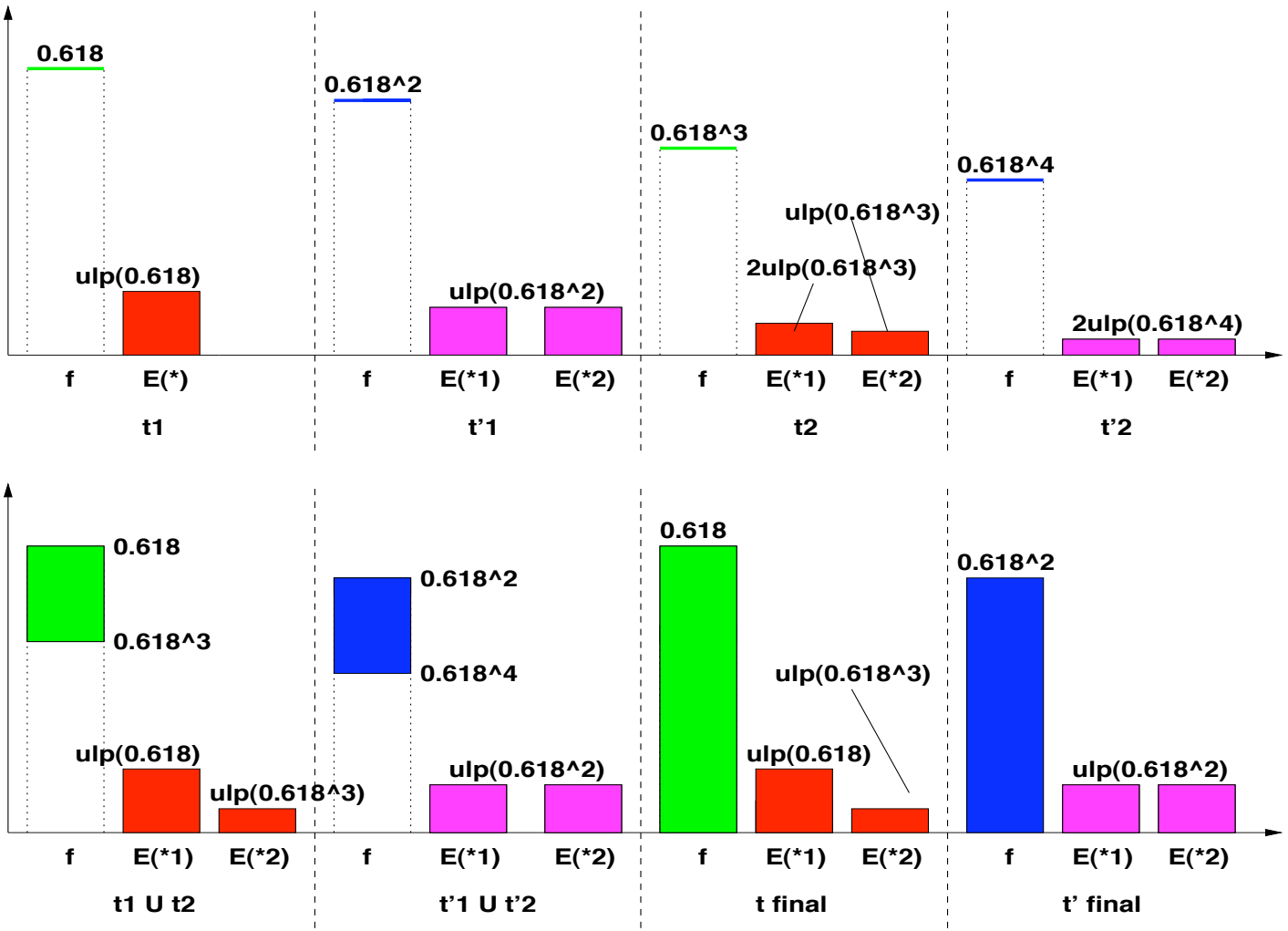
$0.618 < 2 \times 0.618^2$, l'erreur calculée augmente alors que l'erreur réelle diminue

Dépliage des boucles

Réécriture du programme :

```
t = 1;
for (i=1 ; i<=20 ; i++)
{
  t = t*0.618; (multiplication 1)
  if (i>20) break;
  i++;
  t = t*0.618; (multiplication 2)
}
```

Dépliage des boucles



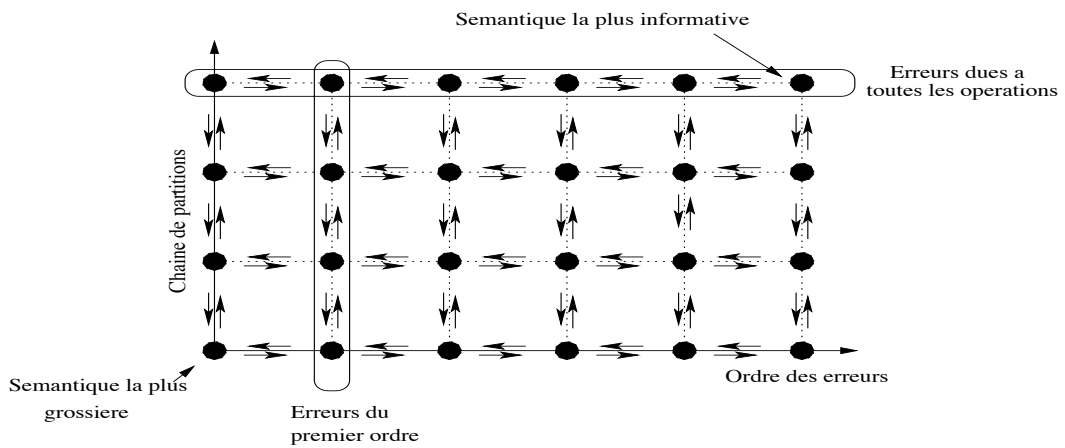
Partitionnement Dynamique

Principe :

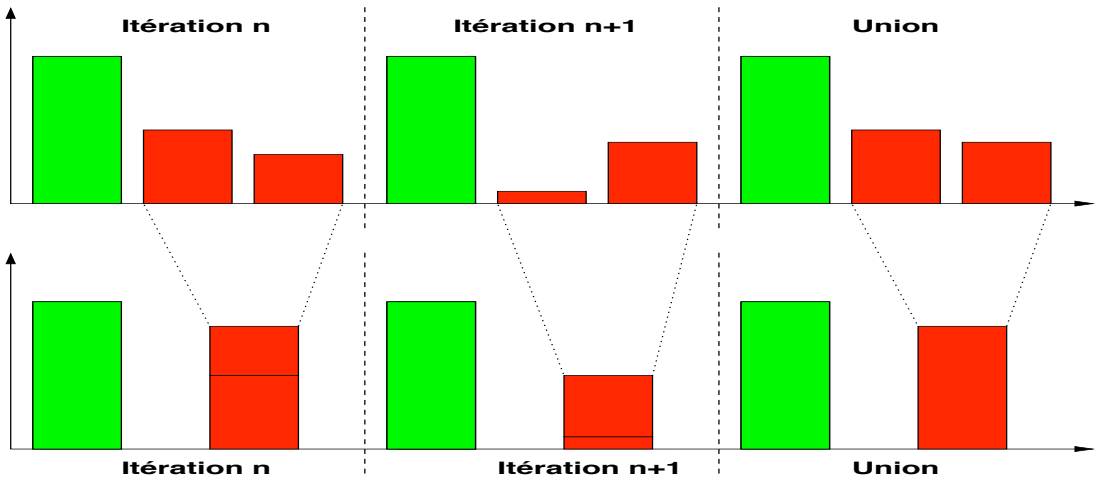
Regrouper les erreurs dues à certains points de contrôle (pour limiter l'occupation mémoire)

En isoler certain autres (pour éviter les pertes de précision)

Faire ce choix dynamiquement, c.à.d. en cours d'analyse



Partitionnement Dynamique (2)



Difficulté :

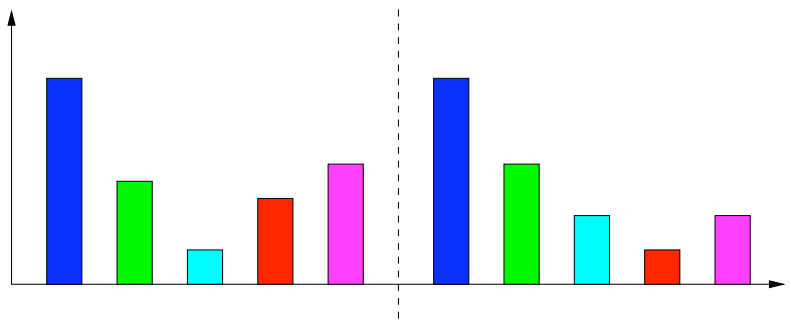
Des points regroupés ne peuvent ensuite être dissociés (correction de l'analyse)

Trouver la plus grande partition pour laquelle on peut affirmer que le calcul est stable

Partitionnement Dynamique : Complexité

Problème DP : Etant donnés deux tuples d'entiers $W = \langle \omega_1, \ldots, \omega_n \rangle$ et $W' = \langle \omega'_1, \ldots, \omega'_n \rangle$, existe-t-il une partition $\mathcal{P}$ de $\{1, \ldots, n\}$ de taille t telle que

$$\forall X \in \mathcal{P}, \sum_{i \in X} \omega_i \leq \sum_{i \in X} \omega'_i$$



Problème NP-Complet (preuve à partir de 2-Partition)

2-Partition : rappel

Etant donné un ensemble d'entiers positifs $A = \{a_1, \ldots, a_n\}$, existe-t-il un sous-ensemble I de A tel que :

$$\sum_{a_i \in I} a_i = \sum_{a_i \in A \setminus I} a_i$$

Preuve (NP-Complétude de DP)

- DP appartient à NP car vérification en temps polynomial.
- Soit $A = \{a_1, \ldots, a_n\}$ une instance I_1 de Partition. Construction à partir de I_1, de I_2 instance de DP

$$
\begin{aligned}
t &= 2 \\
\omega_i &= \left(\sum_{a_j \in A} a_j\right) + a_i,\ 1 \leq i \leq n \\
\omega'_i &= \left(\sum_{a_j \in A} a_j\right) - a_i,\ 1 \leq i \leq n \\
\omega_{n+1} = \omega_{n+2} &= 0 \\
\omega'_{n+1} = \omega'_{n+2} &= \sum_{a_j \in A} a_j
\end{aligned}
$$

Preuve (NP-Complétude de DP)

- On suppose connaître un algorithme A polynomial pour DP
- A trouve une solution à I_2 qui satisfait :

$$\forall X \in \mathcal{P}, \sum_{i \in X} \omega_i \leq \sum_{i \in X} \omega'_i \tag{5}$$

- Remarque 1 : puisque $t = 2$, $\mathcal{P} = \{X, \overline{X}\}$, où $\overline{X} = \{1, 2, \ldots, n+2\} \setminus X$.
- Remarque 2 : $n+1$ et $n+2$ ne sont pas dans la même classe car $\omega'_i < \omega_i$, $1 \leq i \leq n$. Si $n+1$ et $n+2$ sont dans X (resp. $\overline{X}$), alors (5) n'est pas respectée par $\overline{X}$ (resp. X).

Preuve (NP-Complétude de DP)

Nous avons pour X :

$$\begin{aligned}
\sum_X \omega_i &\leq \sum_X \omega'_i \\
|X| \sum_A a_i + \sum_X a_i &\leq |X| \sum_A a_i - \sum_X a_i + \omega'_{n+1} \\
\sum_X a_i &\leq \omega'_{n+1} - \sum_X a_i \\
\sum_X a_i &\leq \sum_A a_i - \sum_X a_i = \sum_{\overline{X}} a_i
\end{aligned}$$

Pour $\overline{X}$, on obtient par la même preuve :

$$\sum_{\overline{X}} a_i \leq \sum_X a_i$$

On en déduit l'égalité

Plan

Généralités sur les nombres flottants

Détection : sémantique des séries d'erreurs

Correction : transformation sémantique d'expressions

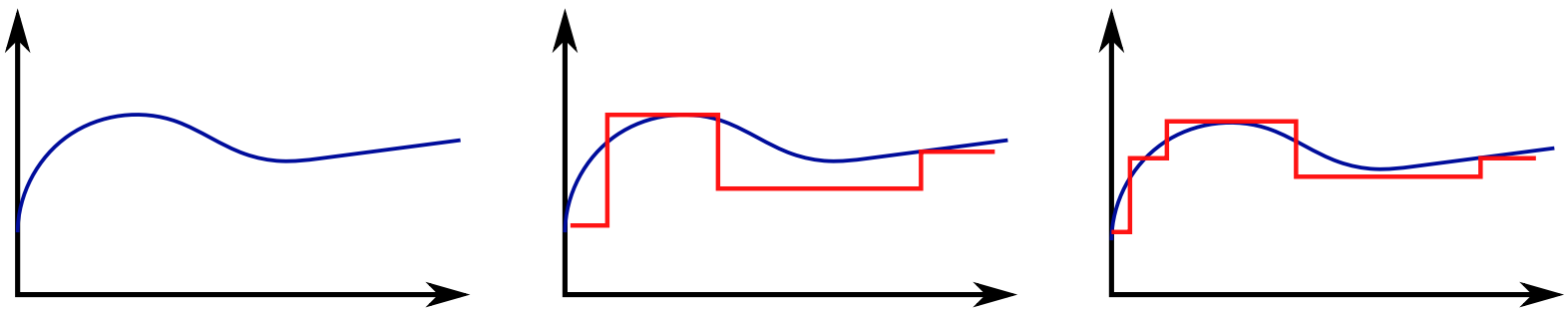
Objectif

Améliorer la précision des calculs à la compilation

Spécification	Implémentation
$x^2 + 2x + 1 = (x+1)^2 = \ldots$	`(x*x+2*x)+1` $\neq$ `(x+1)*(x+1)`

Première sémantique : corps des réels
précision infinie, lois de composition (associativité, distributivité, etc.)

Seconde sémantique : ADO / flottants IEEE754
erreurs d'arrondi, pas de lois de composition



spécification | implémentation | implém. optimisée

Correction : transformation sémantique d'expressions

Introduction

Abstract Program Equivalence Graphs

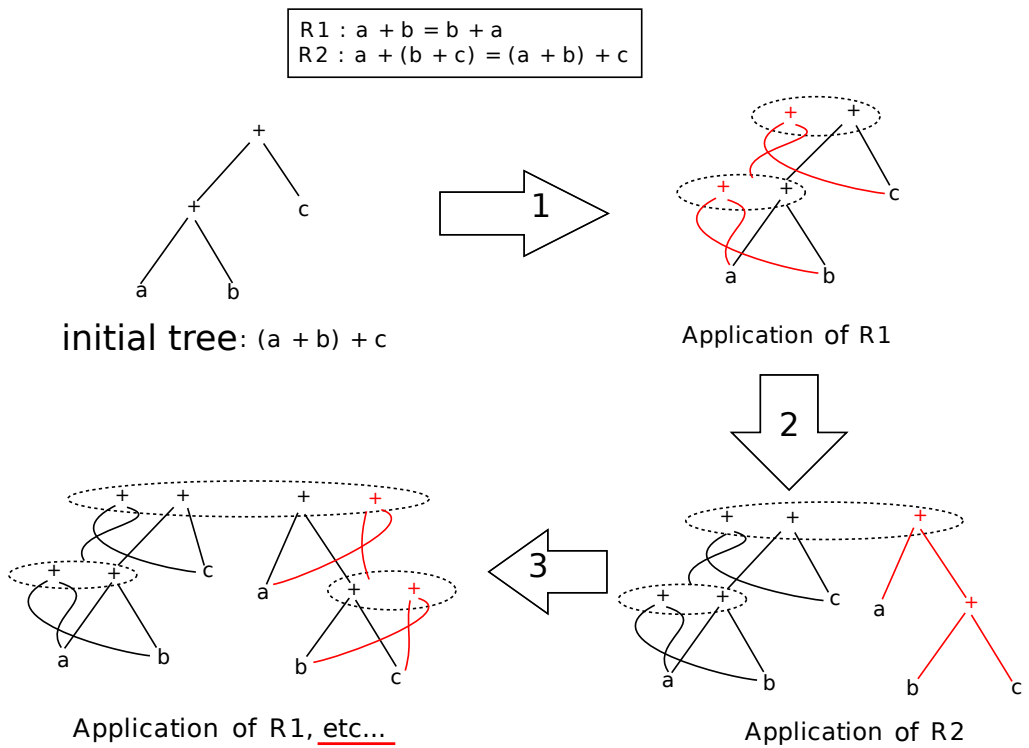
Analyse de profitabilité

Benchmarks

Equivalent Program Expansion Graphs

EPEGs [R.Tate, M.Stepp, Z.Tatlock, S.Lerner, POPL'09]

Développés pour le *phase ordering problem*. Utilisent un ensemble de ré-écritures optimisantes Notion de classe d'équivalence



Approche retenue

Représenter le plus grand nombre de versions d'un programme

Rester polynomial en taille

Extraire un programme dont la précision numérique est meilleure

$(2n-1)!!$[1] façons de sommer n termes ($\frac{2n!}{n! \times (n+1)!}$ parenthésages)

Les EPEGs sont exponentiels en taille

Les EPEGs peuvent aussi être infinis (ex : $a = 1 \times a$)

[1] $(2n-1)!! = 1 \times 3 \times 5 \times \cdots \times n$

Abstraction

APEGs = Abstract Program Equivalence Graphs

Polynomiaux en taille

Représentent un grand nombre de programmes

Ne représentent QUE des programmes équivalents

Contiennent des boîtes d'abstraction

Une boîte d'abstraction : $\boxed{*, (p_1, p_2, \ldots, p_n)}$

Représente tous les parenthésages des expressions composées avec l'opérateur $*$ et les noeuds $p_1, \ldots, p_n$. ($*$ doit être symétrique)

Les p_i peuvent être des constantes, des variables, des expressions ou d'autres boîtes d'abstraction

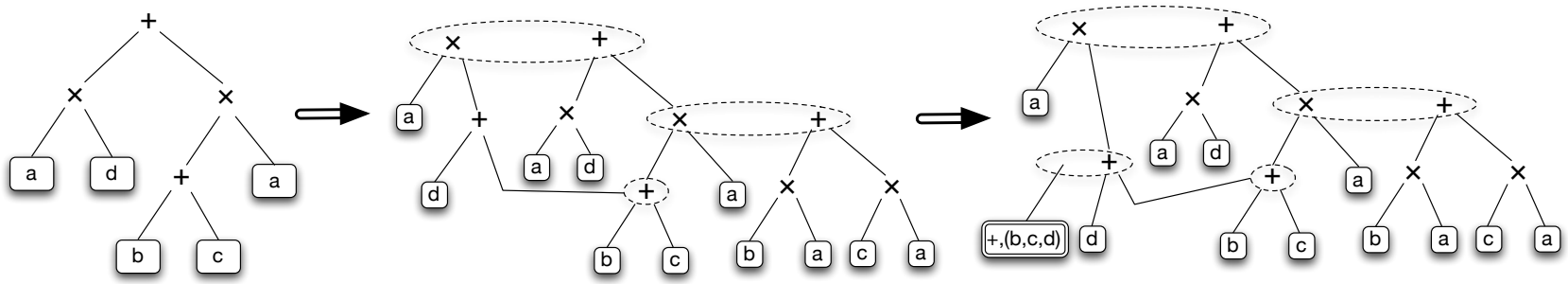
Construction d'un APEG

Etapes :

A partir de l'arbre syntaxique

Introduction de nouvelles expressions

Ajout de boîtes d'abstraction



Introduction de nouvelles expressions

Tout en restant polynomial en taille

Distribuer les multiplications (facteurs maximaux)

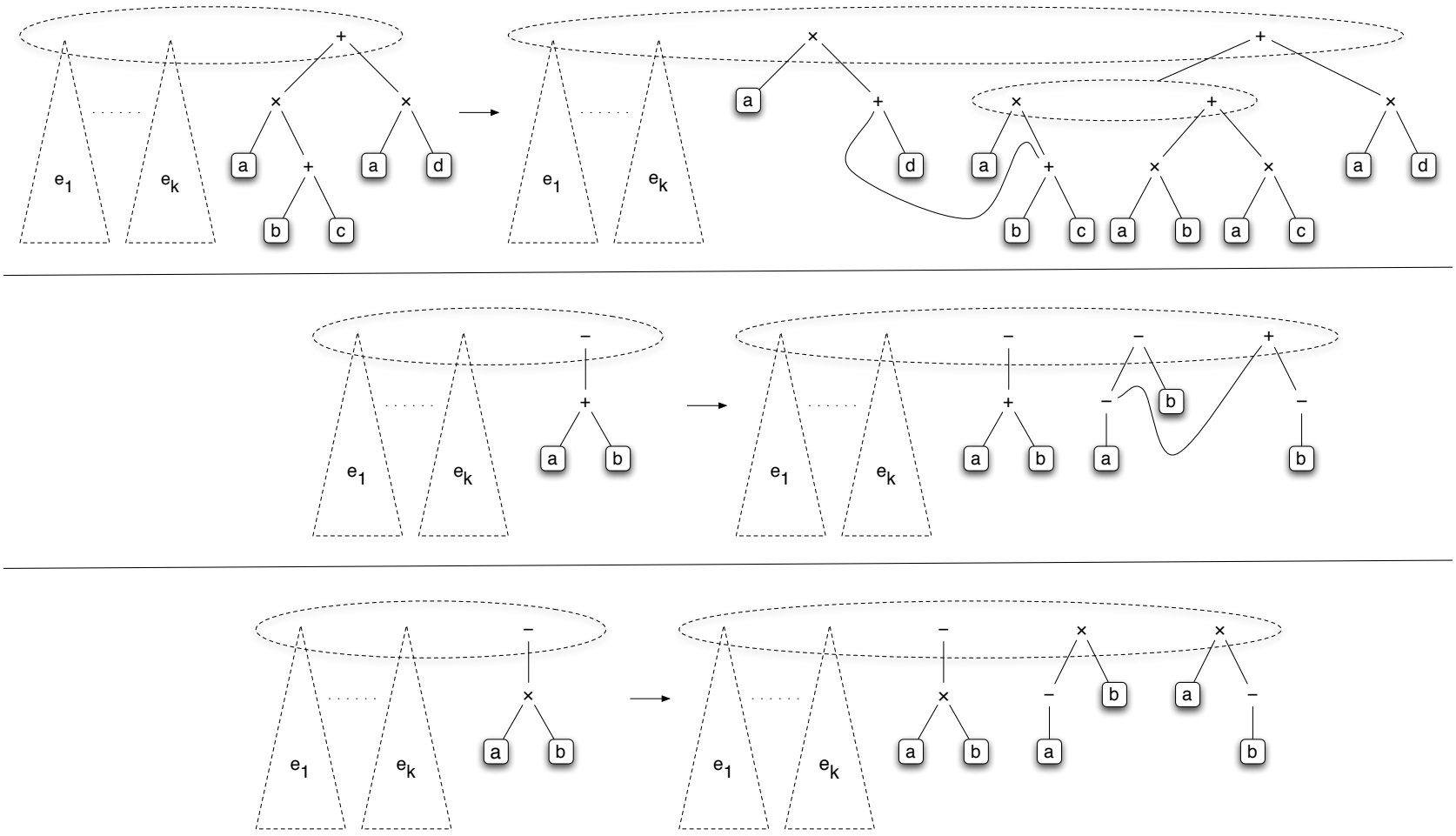
Factoriser au maximum chaque produit

Propager les soustractions

Objectif :

Permettre la créations de grandes boîtes d'abstraction

Règles de transformation des APEGs



Création de boîtes

Repose sur la notion de zone homogène : partie de l'APEG où un opérateur symétrique est répété

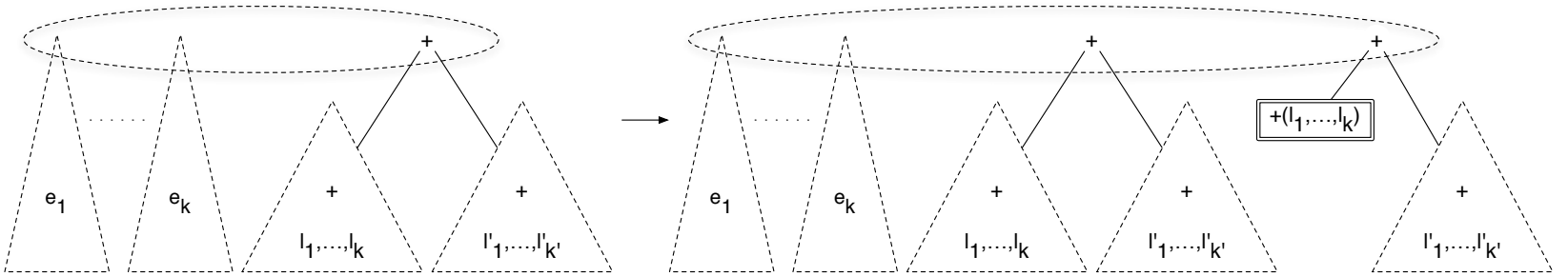
3 algorithmes sont actuellement utilisés :

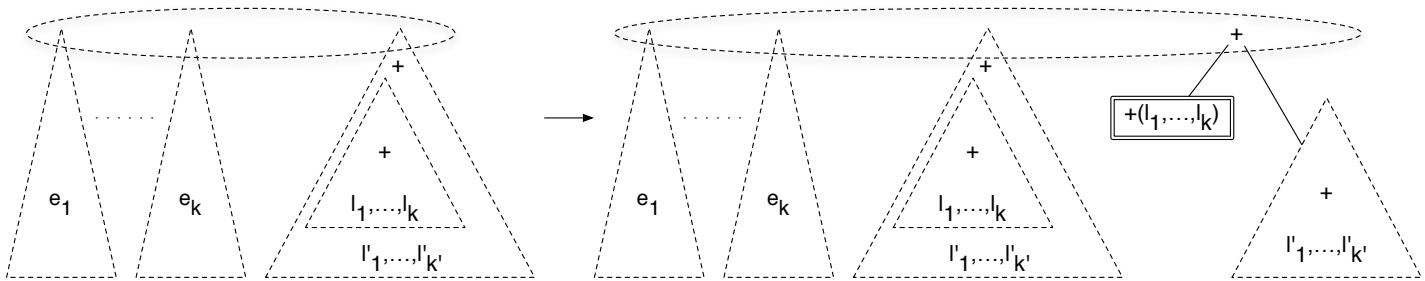
Horizontal : abstrait récursivement la partie gauche puis la partie droite d'une zone homogène

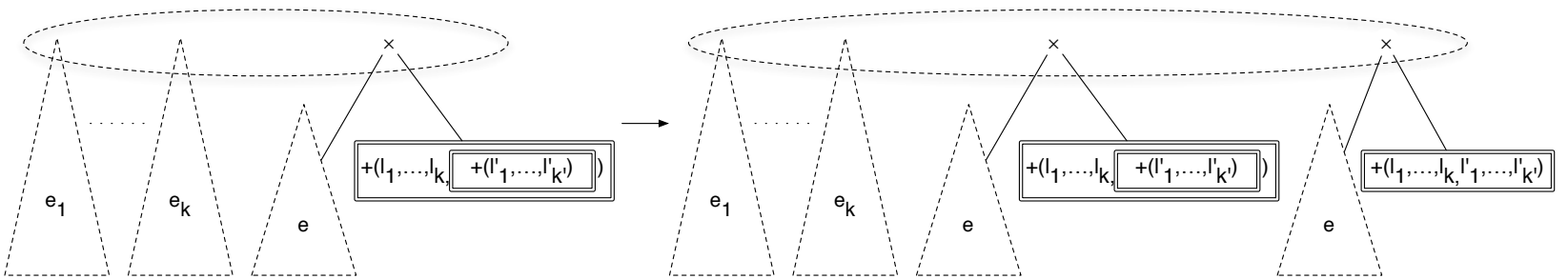
Vertical : abstrait progressivement la partie supérieure d'une zone homogène

Expansion de boîte : les boîtes imbriquées avec le même opérateur sont fusionnées

Les algorithmes de création de boîtes







Exemples

$$(a + (b + c)) + (d + e) \rightarrow \boxed{+\ \mathrm{a}\ \mathrm{b}\ \mathrm{c}} + (d + e)$$

$$a + (b + (c + (d + e))) \rightarrow \boxed{+\ \mathrm{a}\ \mathrm{b}\ \mathrm{c}} + (d + e)$$

$$\boxed{+\ \ \mathrm{a}\ \mathrm{b}\ \boxed{+\ \mathrm{c}\ \mathrm{d}\ \mathrm{e}}} \rightarrow \boxed{+\ \mathrm{a}\ \mathrm{b}\ \mathrm{c}\ \mathrm{d}\ \mathrm{e}}$$

Correction : transformation sémantique d'expressions

Introduction

Abstract Program Equivalence Graphs

Analyse de profitabilité

Benchmarks

Analyse de Profitabilité : boîtes d'abstraction

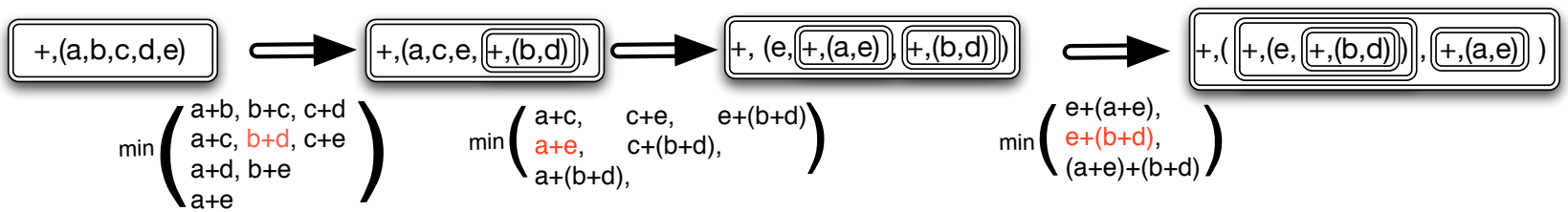
Une boîte d'abstraction = $(2n-1)!!$ expressions

On en cherche une avec une bonne précision

Heuristique gloutonne

A chaque étape, chercher les 2 expressions qui minimisent l'erreur

Approche locale, exécution en $O(n^2)$



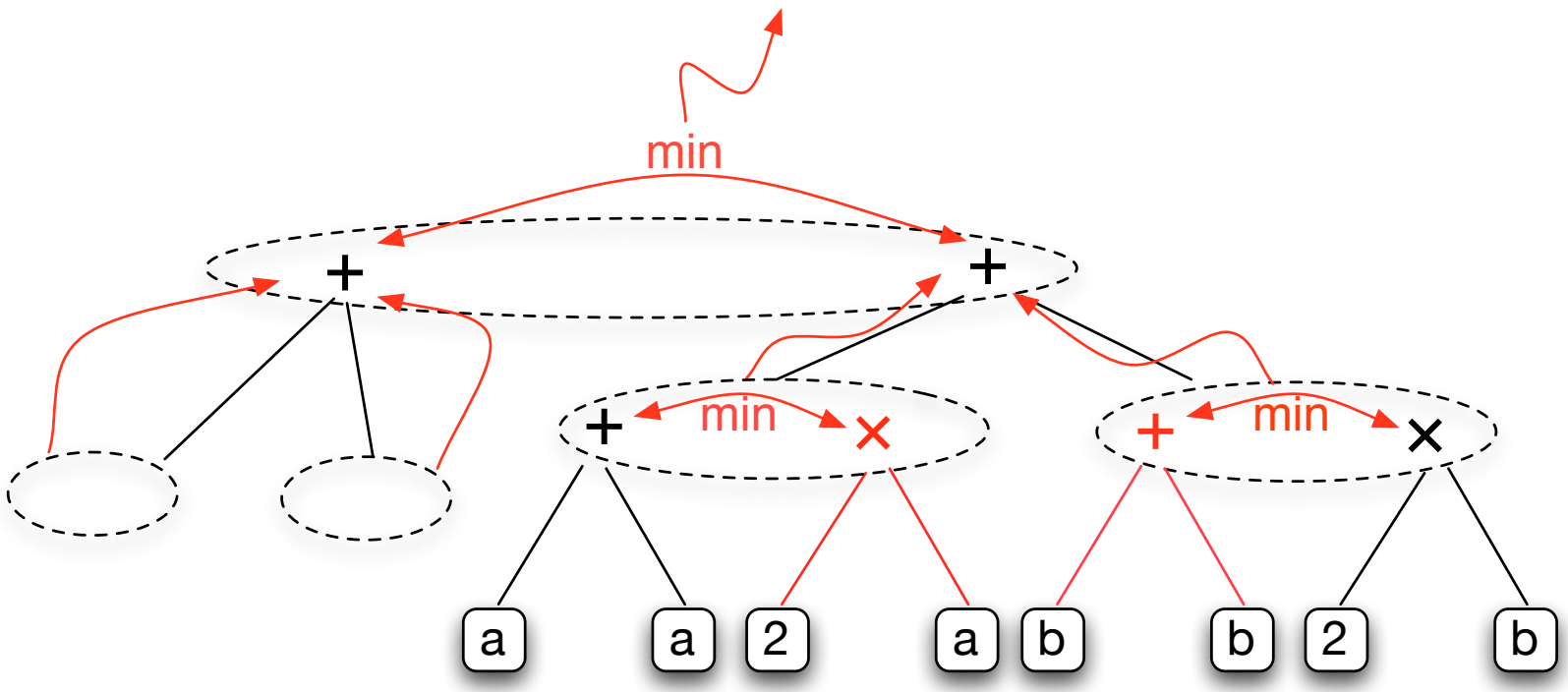
On génère l'expression : $(a+e)+(e+(b+d))$

Analyse de Profitabilité : APEGs

Profitabilité naïve

Pour chaque opérateur, on sélectionne les deux sous-expressions qui minimisent l'erreur (sans remanier les sous-expressions)

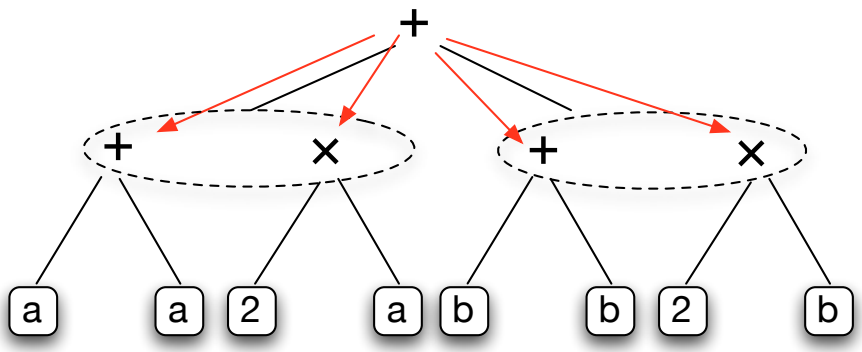
S'exécute en $O(n)$



Amélioration

Ne plus considérer seulement le minimum de chaque classe

Rechercher le minimum en fonction des classes en dessous



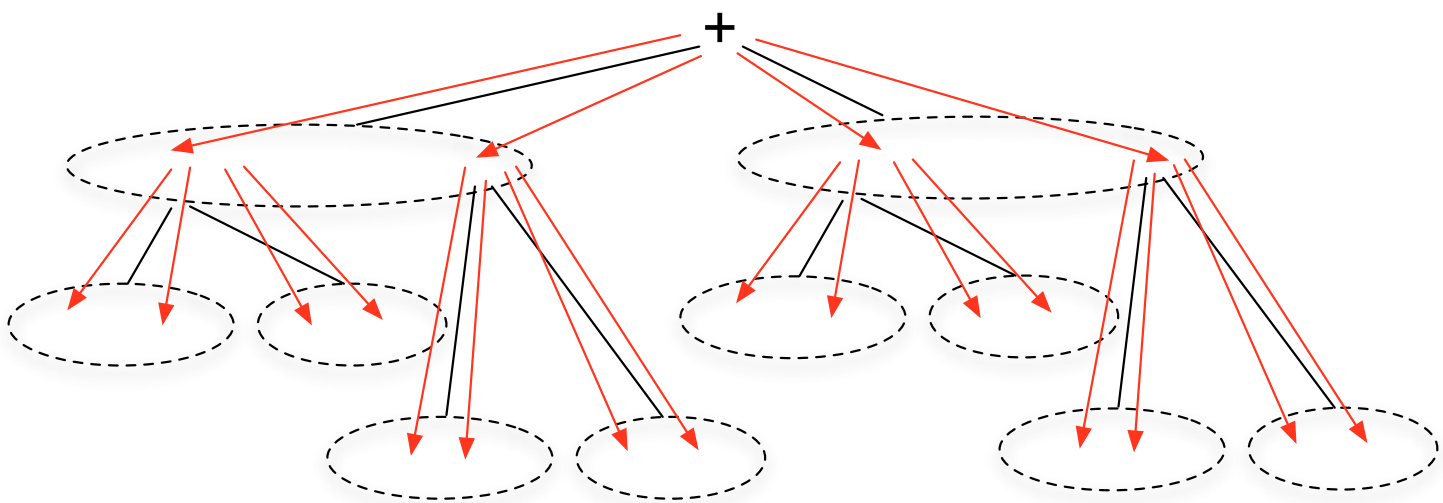
On teste les erreurs sur les expressions :

- $(a + a) + (b + b)$
- $(a + a) + (2 \times b)$
- $(2 \times a) + (b + b)$
- $(2 \times a) + (2 \times b)$

Généralisation

Considérer toutes les classes d'équivalence jusqu'à une profondeur k

Complexité en $O(n^{2\times k})$



Correction : transformation sémantique d'expressions

Introduction

Abstract Program Equivalence Graphs

Analyse de profitabilité

Benchmarks

Benchmarks

Divers cas étudiés (double précision)

Sommations

Polynômes univariés écrits sous forme développée

Développements limités de fonctions usuelles

Méthode :

Générer le plus grand nombre d'expressions équivalentes (toutes?)

Choisir des jeux de données pour le problème considéré

Utiliser Sardana pour optimiser chaque expression initiale

Pour chaque erreur trouvée, compter combien d'expressions l'ont générée

Sommations

4 scenarii :

> 0, 20% grosses valeurs $\approx 10^{16}$ parmi petites $\approx 10^{-16}$

> 0, 20% grosses valeurs parmi petites et moyennes ≈ 1

20% de grandes valeurs des 2 signes, parmi des petites et moyennes

> 0 et < 0, peu de petites, autant de moyennes que de grandes

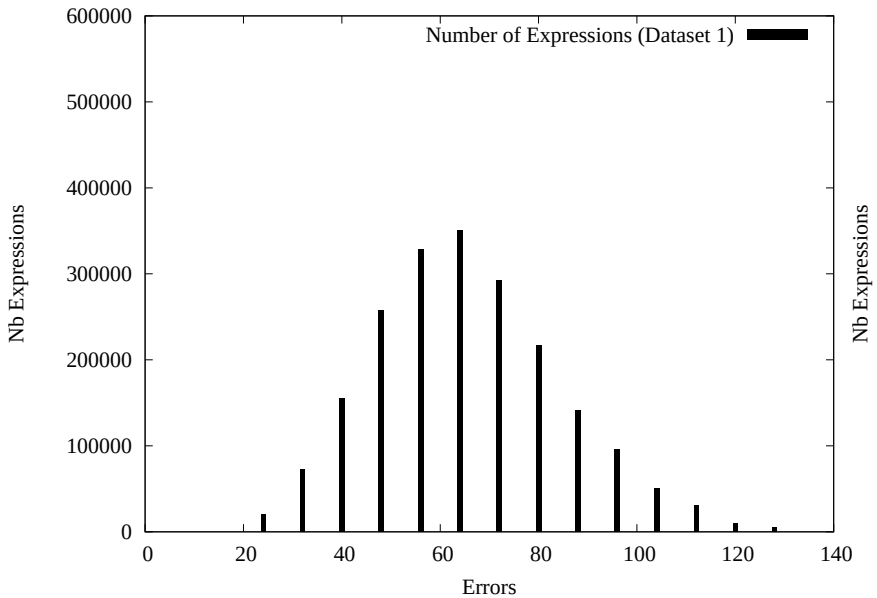
2 tailles d'intervalles :

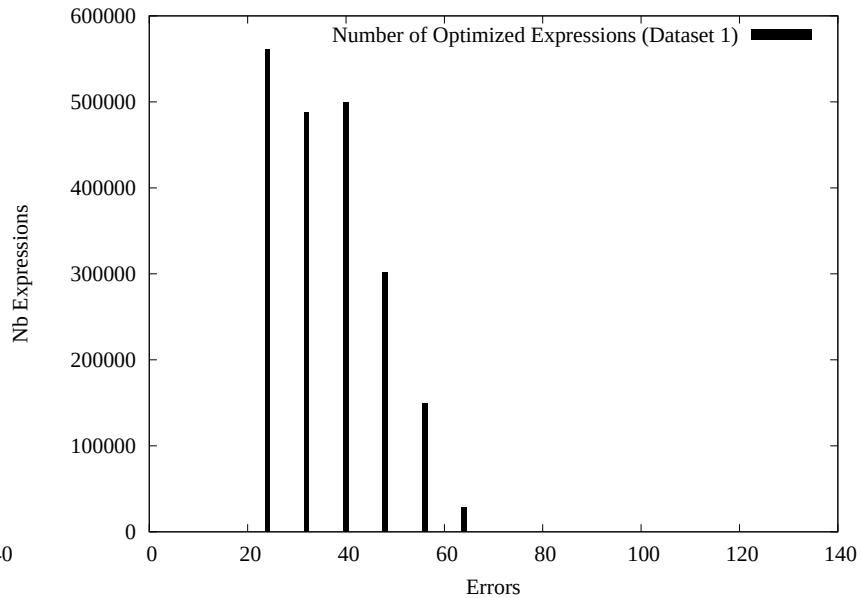
Largeur $=$ 10% de la valeur centrale de l'intervalle

Largeur = 10^{-12} fois plus petite que la valeur centrale de l'intervalle

Somme de 9 termes : 2 millions de cas (config. 1)

nombres positifs, 20% de grosses valeurs au milieu de petites

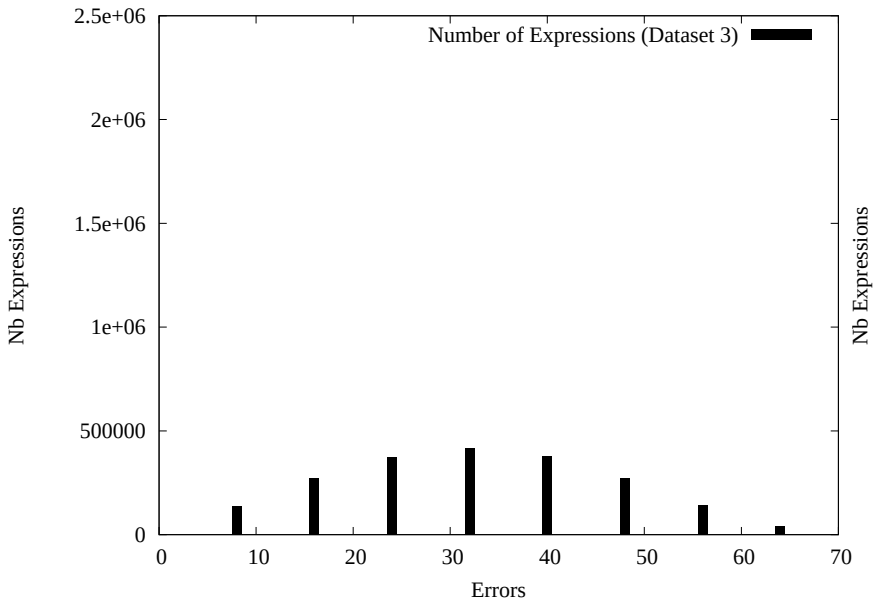


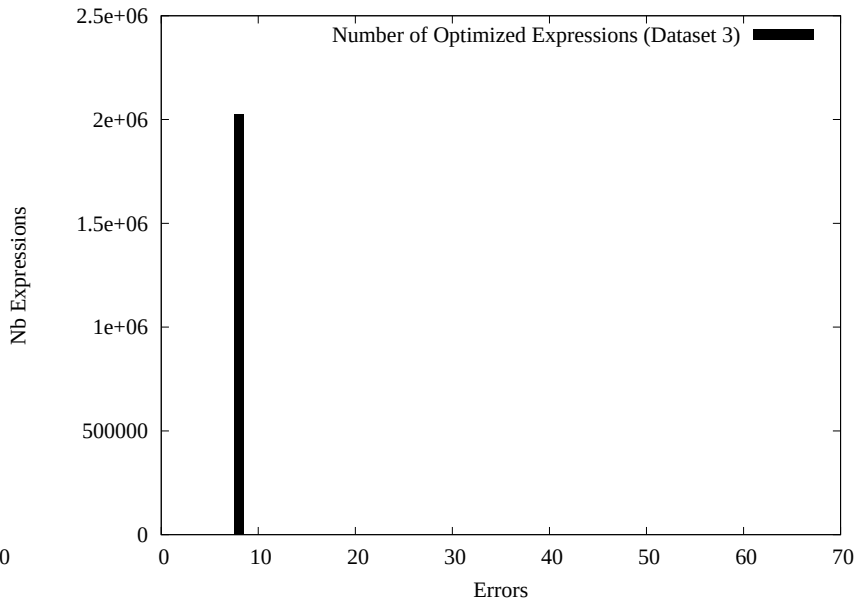


Intervalles larges. (résultats identiques avec intervalles petits)

Somme de 9 termes : 2 millions de cas (config. 2)

nombres positifs, 20% de grosses valeurs au milieu de petites et de moyennes

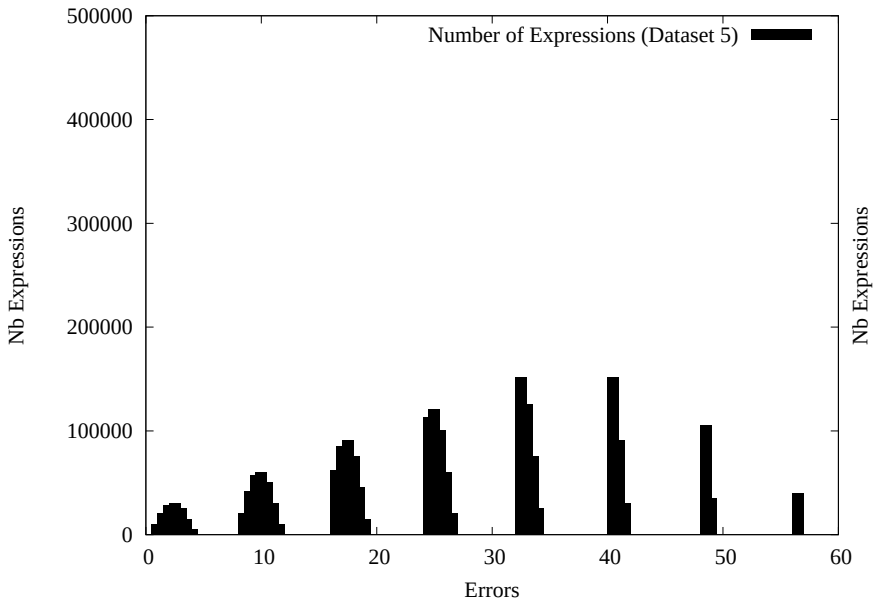


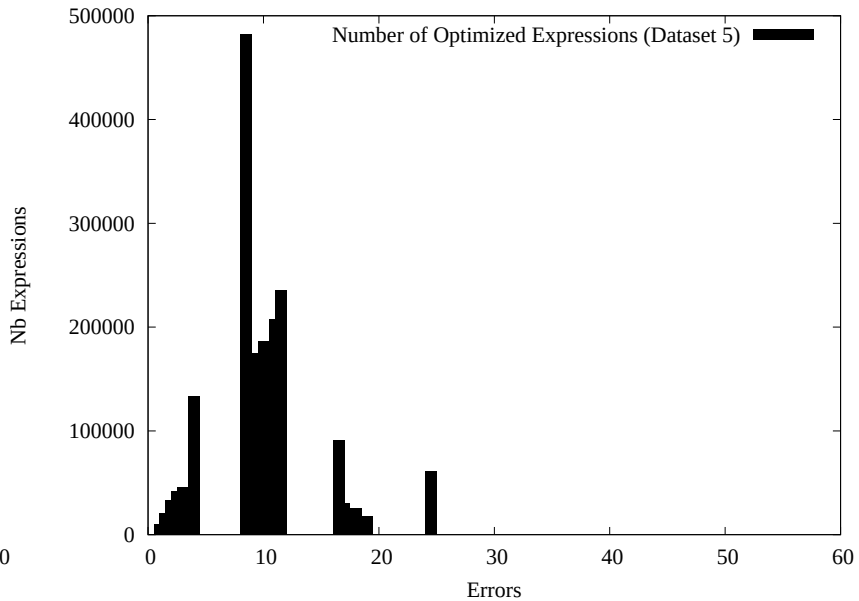


Intervalles larges. (résultats identiques avec intervalles petits)

Somme de 9 termes : 2 millions de cas (config. 3)

20% de grandes valeurs des 2 signes, au milieu de petites et moyennes

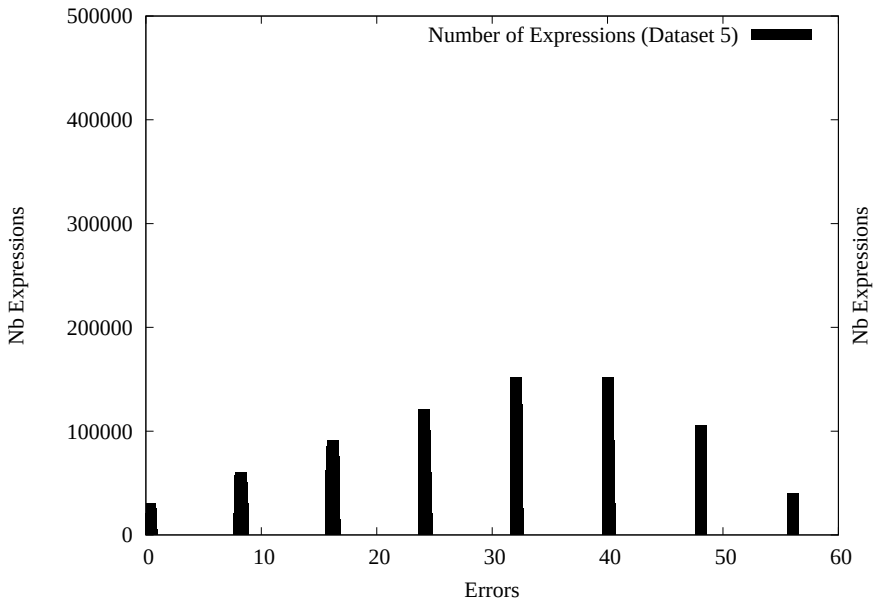


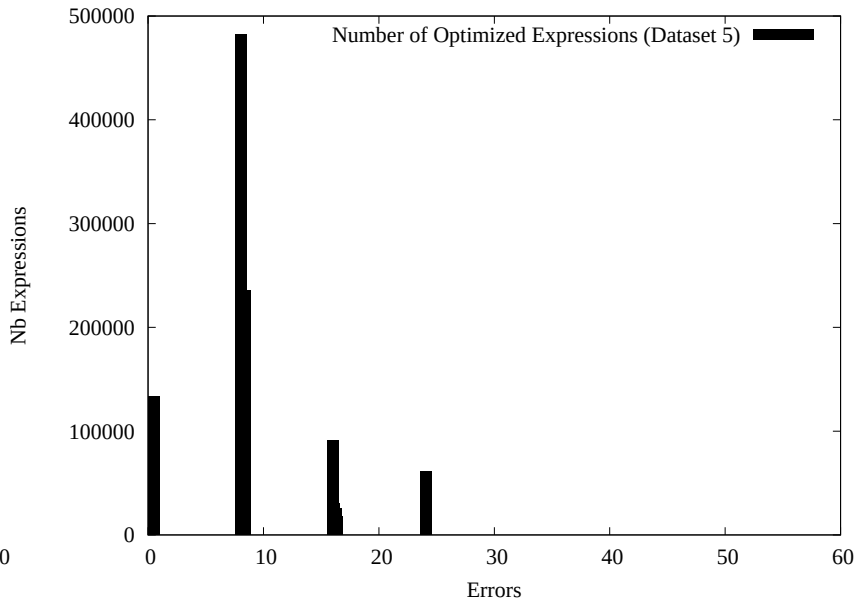


Intervalles larges

Somme de 9 termes : 2 millions de cas (config. 3)

20% de grandes valeurs des 2 signes, au milieu de petites et moyennes

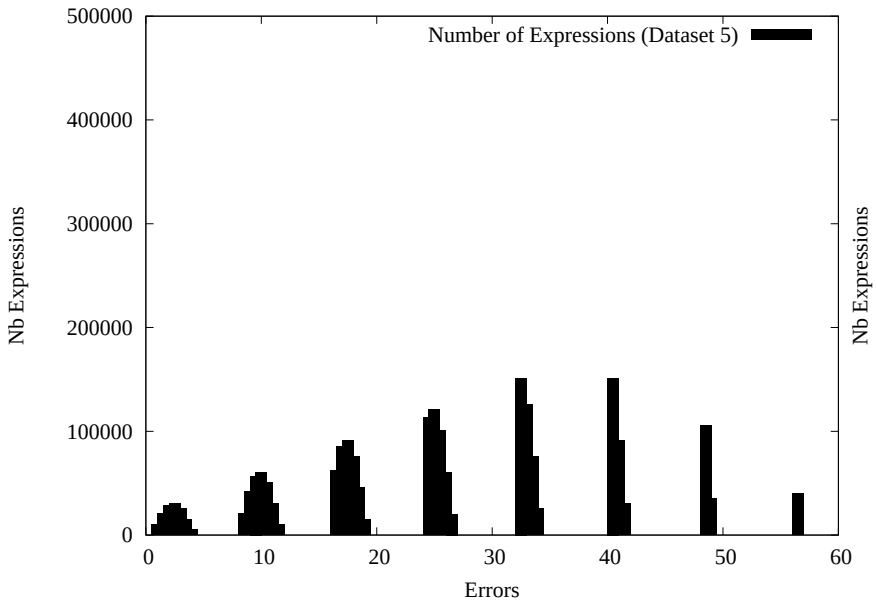


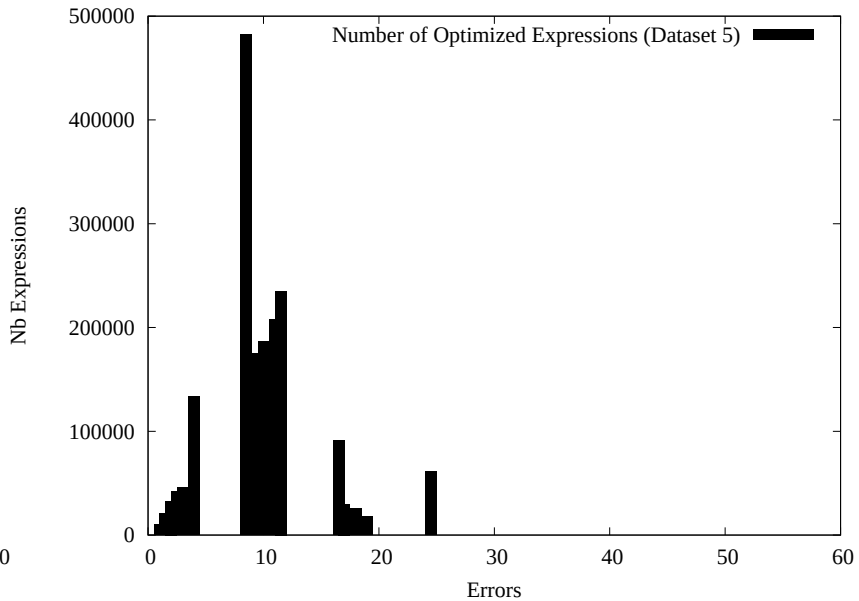


Intervalles petits

Somme de 9 termes : 2 millions de cas (config. 4)

valeurs positives et négatives, peu de petites valeurs et autant de moyennes et de grandes

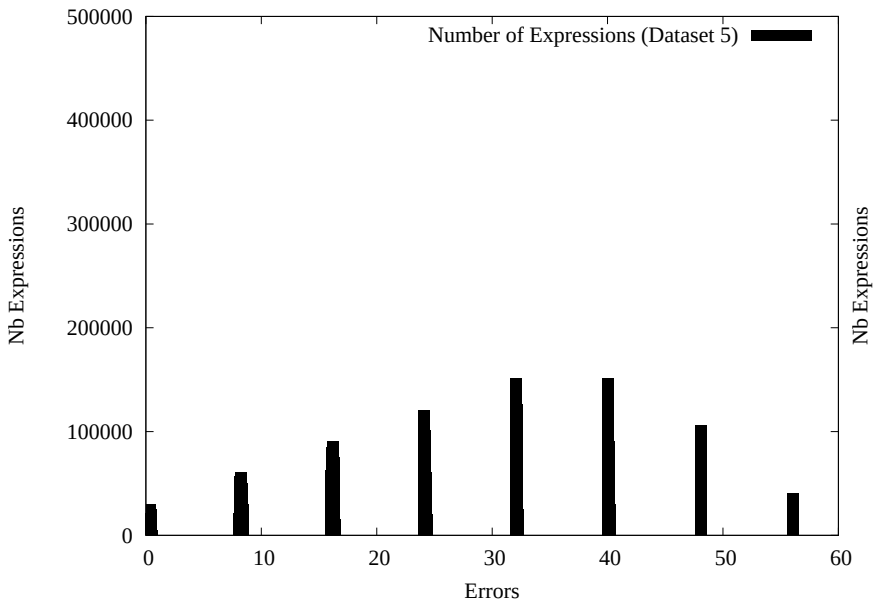


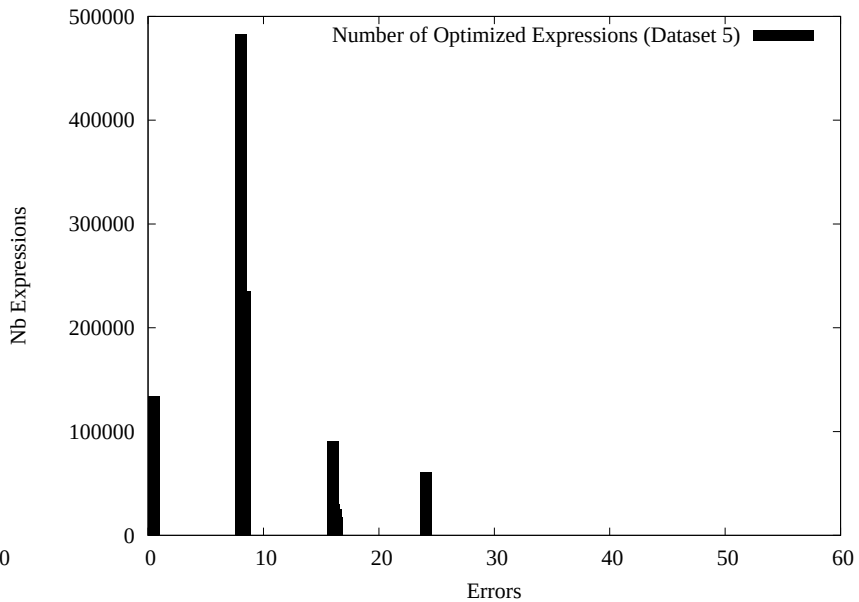


Intervalles larges

Somme de 9 termes : 2 millions de cas (config. 4)

valeurs positives et négatives, peu de petites, autant de moyennes et de grandes





Intervalles petits

Polynomes développés

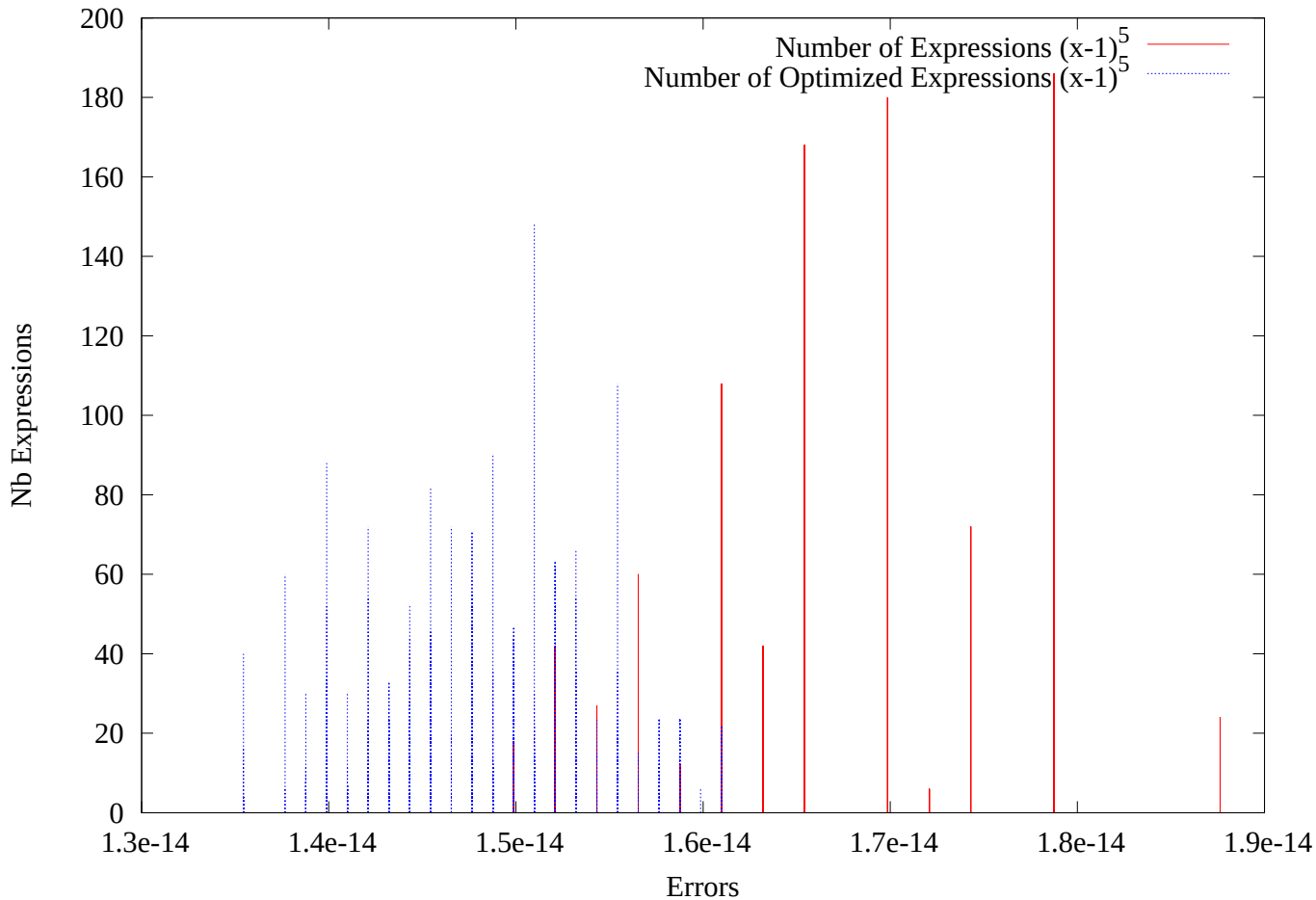
Polynomes utilisés : $(x-1)^n = \sum_{k=0}^{n}(-1)^k \times \binom{n}{k} \times x^k, n \in [2,6]$

Quand n augmente l'erreur augmente autour de la racine multiple

On s'autorise des factorisations dans notre analyseur, donc le résultat n'est pas forcément sous forme développée

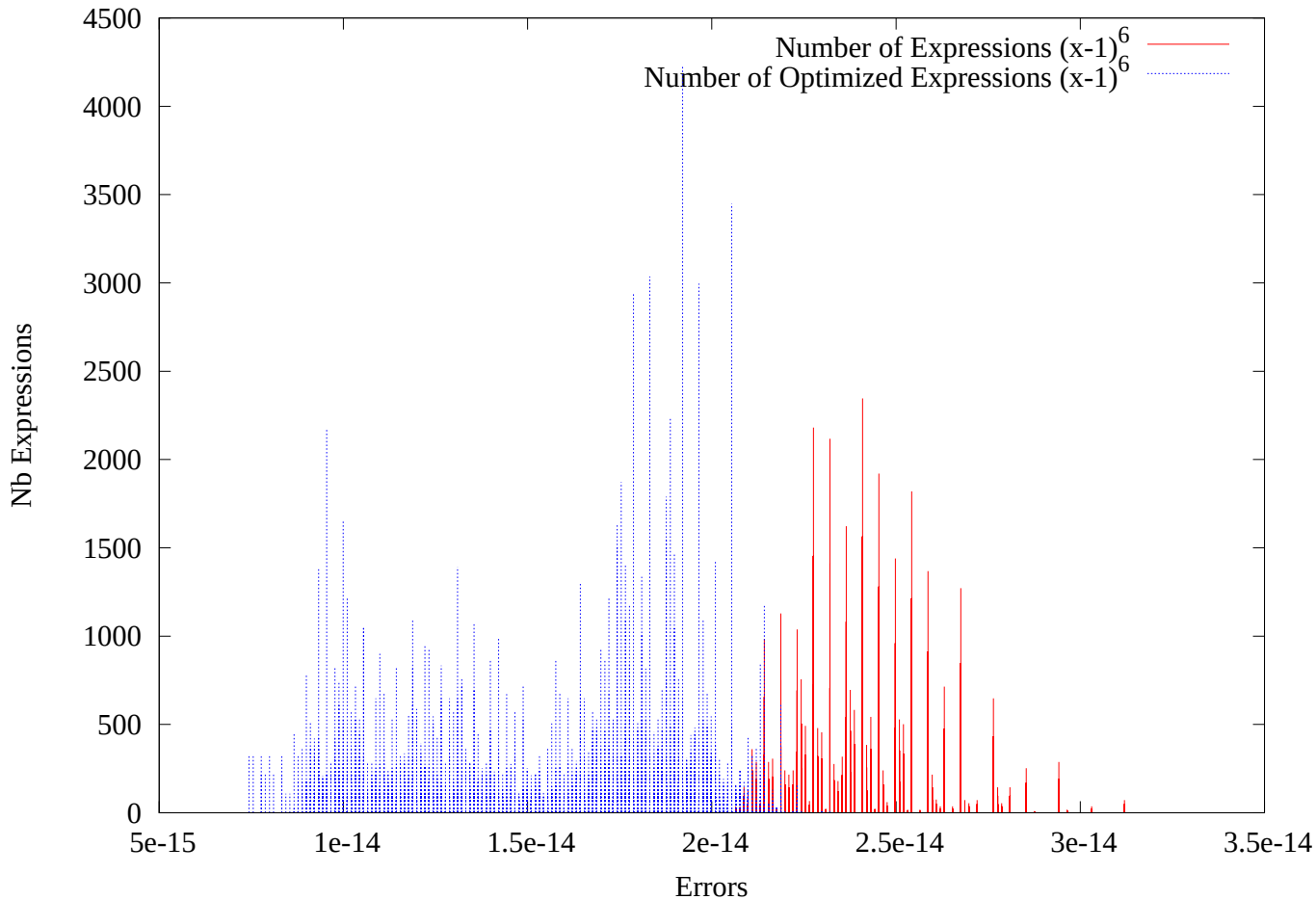
On génère toute les sommes possibles de monômes $\times$ tous les manières d'écrire les monômes x^i

Polynômes développés avec $n = 5$, 5.670 cas



En rouge les bornes d'erreurs initiales, en bleue les optimisées

Polynomes développés avec $n = 6$, 374.220 cas



En rouge les bornes d'erreurs initiales, en bleue les optimisées

Développements de Taylor

Différentes fonctions considérés :

$\cos x = \sum_{n=0}^{+\infty} (-1)^n \frac{x^{2n}}{(2n)!}$

$\sin x = \sum_{n=0}^{+\infty} (-1)^n \frac{x^{2n+1}}{(2n+1)!}$

$ln(2+x) = \sum_{n=1}^{+\infty} (-1)^{n-1} \frac{x^n}{n \times 2^n}$

Pour différents ordres :

pour cos: $n \in \{4, 6, 8\}$

pour sin: $n \in \{5, 7, 9\}$

pour ln(2+x): $n \in \{4, 5\}$

Intervalles centrés sur racines, largeur = 10% de valeur centrale

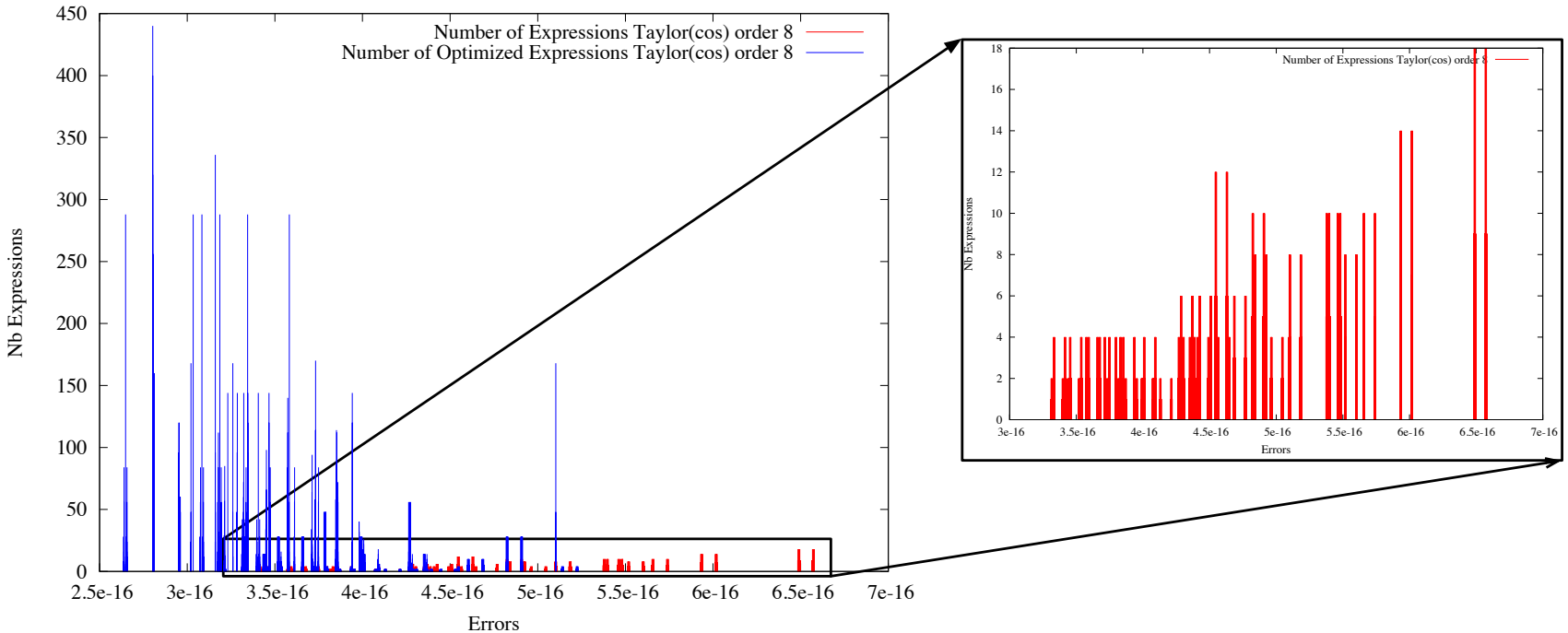
Développements de Taylor

Méthodologie, forme générale d'un développement : $\sum c_i x^i$

On génère toutes les ré-écritures de la somme de termes $\times$ toutes les ré-écriture des produits internes de x^i

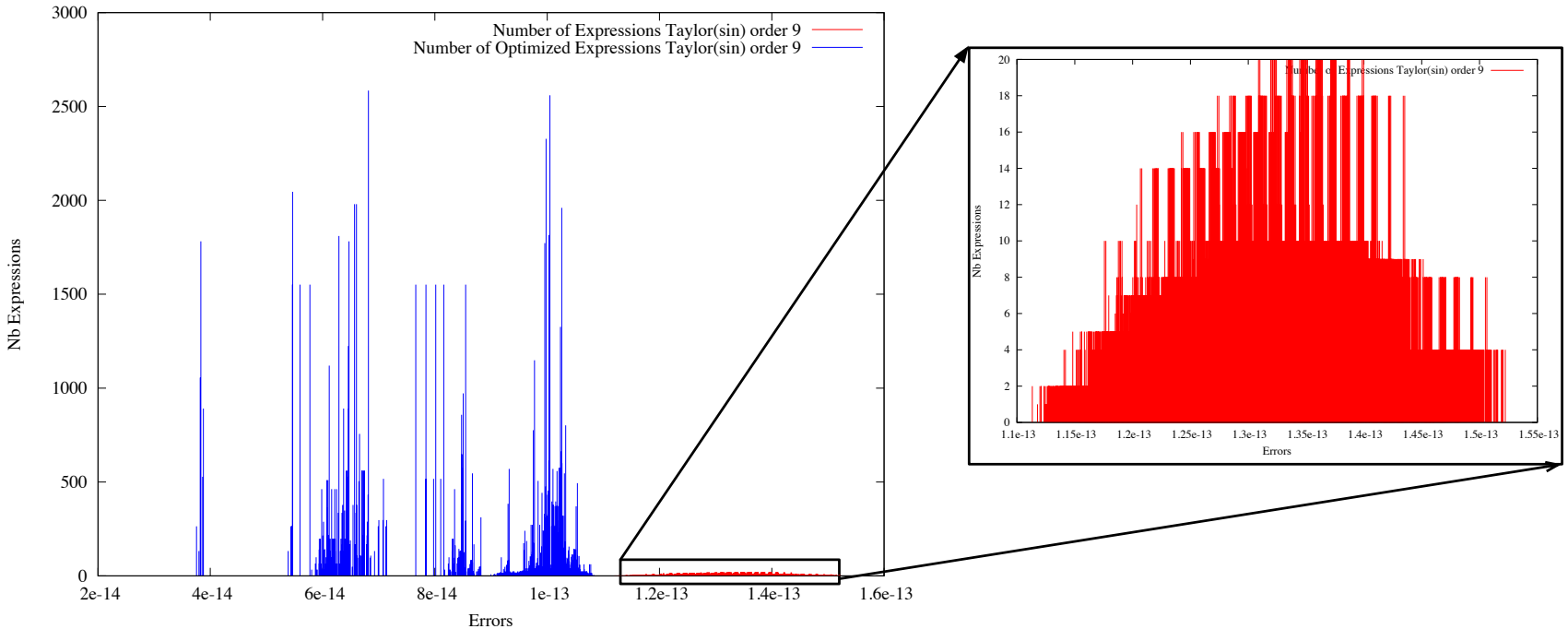
Quand n augmente, l'erreur près d'une racine de la fonction cible augmente, car le développement s'annule aussi sur une valeur de plus en plus proche

Résultats sur cos avec $n = 8$, 30.240 cas



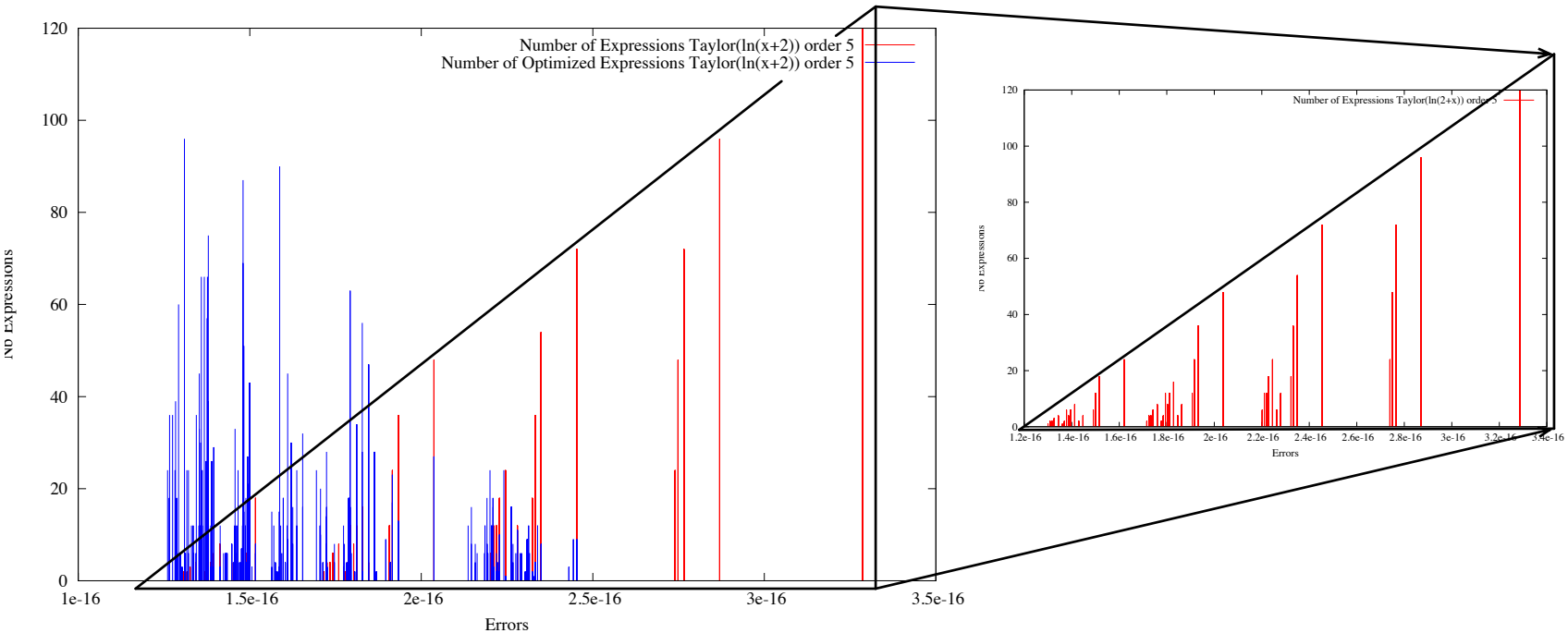
En bleu les résultats des expressions optimisées, en rouge les initiales, à droite une vue zoomée des résultats des expressions initiales

Résultats sur sin avec $n = 9$, 162.855 cas



En bleu les résultats des expressions optimisées, en rouge les initiales, à droite une vue zoomée des résultats des expressions initiales

Analyse sur $ln(2 + x)$ avec $n = 5$, 5.670 cas



En bleu les résultats des expressions optimisées, en rouge les initiales, à droite une vue zoomée des résultats des expressions initiales

Références

Jean-Michel Muller et al, Handbook of Floating-Point Arithmetic, Birkhauser, décembre 2009

David Monniaux. The pitfalls of verifying floating-point computations. TOPLAS, 30(3):12, 2008

Matthieu Martel, Enhancing the Implementation of Mathematical Formulas for Fixed-Point and Floating-Point Arithmetics, Journal of Formal Methods in System Design, volume 35, pages 265-278, 2009

Matthieu Martel, Semantics of roundoff error propagation in finite precision computations, Journal of Higher Order and Symbolic Computation, 19:7-30, 2006

David Delmas, Eric Goubault, Sylvie Putot, Jean Souyris, Karim Tekkal et Franck VédrineTowards an Industrial use of FLUCTUAT on Safety-Critical Avionics Software, FMICS 2009