32

Algorithmes pour la conception de circuits arithmétiques rapides

Jean-Michel Muller
ENS, Lyon

[résumé par Valérie Ménissier-Morain & François Morain]

1 Introduction

Deux thèmes sont abordés: l'arithmétique "en ligne" (c'est un mode de calcul dans lequel les opérandes circulent en série, poids forts en tête, et sont représentés dans un système redondant d'écriture des nombres), et les algorithmes "de type Cordic" de calcul des fonctions élémentaires.

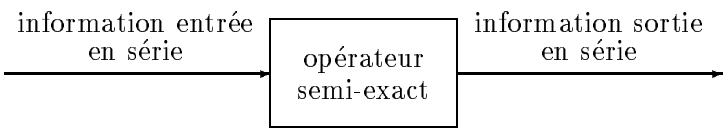
2 L'arithmétique "en ligne"

L'arithmétique "en ligne" est un moyen de calcul qui permet d'atteindre de hautes performances en permettant un "pipe-line" au niveau du chiffre.
Le sens de circulation "poids forts en tête" des chiffres est plus naturel que le sens "poids faibles en tête", car il correspond à la notion de continuité (on va du grossier au précis) et fournit un contrôle d'erreur naturel. De surcroît, le sens "poids faibles en tête" ne permettrait pas d'effectuer des divisions ou de calculer des fonctions élémentaires.
Deux questions se posent: quel système d'écriture des nombres utiliser et quels algorithmes, adaptés à quelles architectures, choisir pour les opérations arithmétiques ($+$, $-$, $\times$, $/$) et les fonctions transcendantes?
Il existe divers types d'arithmétique:

exacte: entiers, rationnels, manipulation symbolique d'objets non finis,

approchée: c'est l'arithmétique flottante usuelle,

semi-exacte: cette arithmétique fonctionne suivant le schéma suivant:



L'idée du calcul en série est de ne consommer qu'un grain d'information de chaque opérande pour produire un grain d'information en sortie.
Il y a deux avantages à cela: d'une part le *layout* se trouve simplifié et d'autre part cela permet un "pipe-line" au niveau du chiffre et, par conséquent, un enchaînement rapide des calculs.
Mais ce système ne marche que si les chiffres circulent *toujours* dans le même sens.
Hélas, les algorithmes classiques d'addition et de multiplication produisent les chiffres de droite à gauche à cause de la propagation des retenues, alors que celui de la division produit les chiffres de gauche à droite.

2.1 Représentation d'Avizienis et algorithmes d'addition

On utilise alors une représentation redondante des nombres due à Avizienis ([1] et [4] pp. 34-40): on travaille dans une base B et les chiffres sont des entiers entre $-a$ et $+a$, où a est un chiffre en base B ($1 \leq a \leq B-1$).
Avizienis a établi deux résultats:

- *on peut écrire tous les nombres entiers dans cette représentation si et seulement si* $2a+1 \geq B$,
- *si* $2a \geq B+1$, *alors il existe un algorithme pour effectuer des additions de manière* complètement parallèle, *sans propagation de retenue.*

Voici une description de cet algorithme d'addition:

$$\begin{aligned} t_{i+1} &= \begin{cases} -1 & \text{si} \quad x_i + y_i \leq -a \\ +1 & \text{si} \quad x_i + y_i \geq a \\ 0 & \text{si} \quad -a+1 \leq x_i + y_i \leq a-1 \end{cases} \\ w_i &= x_i + y_i - B t_{i+1} \\ s_i &= w_i + t_i \end{aligned}$$

avec $w_n = t_0 = 0$.
On montre que cet algorithme ne produit jamais de retenue.
Les deux conditions $a \leq B-1$ et $2a \geq B+1$ ne peuvent être satisfaites que si B est supérieur ou égal à 3. La méthode d'Avizienis ne peut donc être utilisée que dans ce cas-ci.
Il est possible de faire également des additions de manière totalement parallèle (c'est-à-dire en un temps totalement indépendant de la taille des opérandes) en base 2, mais la méthode utilisée est alors plus complexe que celle d'Avizienis [2]. Cet algorithme a pour délai 2 (c'est-à-dire le nombre de grains d'information sur l'opérande à fournir avant de produire un grain d'information pour le résultat), qui est le délai minimum pour l'addition en base 2.

2.2 Algorithmes de multiplication et de division "en ligne"

Ercegovac et Trivedi ([3] et [4] pp. 134-135 pour la multiplication, pp. 161-164 pour la division) ont proposé en 1977 des algorithmes de multiplication et division "en ligne" de délai 5.

2.3 Que peut-on calculer en ligne?

Remarquons tout d'abord que la calculabilité nécessite la continuité de l'opérateur, ceci vient du fait que l'on calcule avec des systèmes redondants d'écriture des nombres.

- On sait calculer avec un opérateur indépendant de la taille des nombres manipulés (*un automate fini*): $+$, $-$, max, min, et toute application affine avec des constantes rationnelles,
- On sait calculer avec un opérateur "linéaire" constitué d'un même élément répliqué un nombre de fois égal à la taille des nombres manipulés: $\times$, $/$, $\sqrt{}$, et tout polynôme à coefficients rationnels ou son inverse,
- On sait calculer avec des opérateurs constitués d'une partie "linéaire" et d'une table contenant un nombre de constantes proportionnel à la taille des nombres manipulés: sin, cos, arctan, exp et log.

3 Les algorithmes "de type Cordic"

Les algorithmes "de type Cordic" [5], bien qu'assez anciens (le premier algorithme de cette classe a trois siècles et a été inventé par Briggs, un contemporain de Neper, pour construire la première table de logarithmes), sont un des moyens de calcul des fonctions élémentaires (sinus, exponentielle, logarithme, ...) les plus employés (on trouve des algorithmes de ce type dans la plupart des calculatrices, et dans des coprocesseurs comme l'Intel 8087 ou le Motorola 68881).
A titre d'exemple, considérons le problème du calcul des fonctions circulaires élémentaires cosinus et sinus. Soit θ un réel, $0 \leq \theta \leq \pi/2$. Pour calculer $(\cos\theta, \sin\theta)$, il suffit de faire une rotation d'angle θ à partir du point $(1,0)$ et de lire les coordonnées du point d'arrivée. L'idée est alors de décomposer cette rotation en petites rotations d'angle $d_n e_n$ avec $d_i = \pm 1$, de sorte que les rotations d'angle $\pm e_i$ soient faciles à effectuer et que bien sûr $\theta = \sum_{i=0}^{\infty} d_i e_i$. On pose $\theta_n = \sum_{i=0}^{n} d_i e_i$. A la n-ième étape, on tournera d'un angle $+e_n$ si $\theta_n \leq \theta$, et de $-e_n$ sinon. On montre alors le résultat suivant :

Si (e_n) est une suite décroissante et sommable de réels positifs, vérifiant

$$\forall n, \quad e_n \leq \sum_{i=n+1}^{\infty} e_i$$

alors, pour tout $\theta \in [-\sum_{i=0}^{\infty} e_i, +\sum_{i=0}^{\infty} e_i]$, la suite (θ_n) définie par

$$\begin{array}{lcl} \theta_0 & = & 0 \\ d_n & = & 1 \text{ si } \theta_n \leq \theta, -1 \text{ sinon} \\ \theta_{n+1} & = & \theta_n + d_n e_n \end{array}$$

a pour limite θ quand n tend vers l'infini.

L'algorithme de calcul de $(\cos\theta, \sin\theta)$ se fait alors en prenant pour suite (e_n) la suite de terme général $e_n = \arctan(2^{-n})$. Écrivons :

$$V_n = \begin{pmatrix} x_n \\ y_n \end{pmatrix} = \begin{pmatrix} \cos\theta_n \\ \sin\theta_n \end{pmatrix}.$$

On passe de V_n à V_{n+1} par une rotation d'angle $d_n e_n$. Autrement dit :

$$\begin{pmatrix} x_{n+1} \\ y_{n+1} \end{pmatrix} = \begin{pmatrix} \cos(d_n e_n) & -\sin(d_n e_n) \\ \sin(d_n e_n) & \cos(d_n e_n) \end{pmatrix} \begin{pmatrix} x_n \\ y_n \end{pmatrix}$$

ce qui conduit, avec le choix particulier de $e_n = \arctan(2^{-n})$ à

$$V_{n+1} = \frac{1}{\sqrt{1+2^{-2n}}} \begin{pmatrix} 1 & -d_n 2^{-n} \\ d_n 2^{-n} & 1 \end{pmatrix} V_n.$$

Quand n tend vers l'infini, on obtient

$$V_\infty = \begin{pmatrix} \cos\theta \\ \sin\theta \end{pmatrix} = \prod_{n=0}^{\infty} \frac{\begin{pmatrix} 1 & -d_n 2^{-n} \\ d_n 2^{-n} & 1 \end{pmatrix}}{\sqrt{1+2^{-2n}}} \begin{pmatrix} x_0 \\ y_0 \end{pmatrix} = K \prod_{n=0}^{\infty} \begin{pmatrix} 1 & -d_n 2^{-n} \\ d_n 2^{-n} & 1 \end{pmatrix} V_0.$$

L'algorithme de Cordic consiste à calculer la suite V'_n définie par

$$V'_0 = \begin{pmatrix} 1/K \\ 0 \end{pmatrix},$$

$$V'_{n+1} = \begin{pmatrix} x'_{n+1} \\ y'_{n+1} \end{pmatrix} = \begin{pmatrix} 1 & -d_n 2^{-n} \\ d_n 2^{-n} & 1 \end{pmatrix} V'_n$$

qui converge par construction vers

$$V'_\infty = V_\infty = \begin{pmatrix} \cos\theta \\ \sin\theta \end{pmatrix}.$$

On montre que de petites modifications de ces algorithmes permettent l'utilisation de systèmes redondants d'écriture des nombres, et autorisent ainsi d'excellentes performances de calcul. On peut notamment utiliser ces algorithmes modifiés pour calculer "en ligne" (méthodes de Takagi ; améliorations de l'auteur).

Références

[1] A. Avizienis. Signed-digit number representations for fast parallel arithmetic. *IRE Transactions on electronic computers*, 10:pp. 389–400, 1961.

[2] C. Y. Chow and J. E. Robertson. Logical design of a redundant binary adder. In *Proceedings of the* 4^{th} *symposium on computer arithmetic*, pages pp. 109–115, October 1978.

[3] M. D. Ercegovac and K. S. Trivedi. On-line algoritms for division and multiplication. *IEEE Transactions on Computers*, C-26:pp. 681–687, July 1977.

[4] J. M. Muller. *Arithmétique des ordinateurs.* études et recherches en informatique. Masson, 1989.

[5] J. Volder. The CORDIC computing technique. *IRE Transactions on Computers*, September 1959.