Agrégation externe de mathématiques, session 2010
Épreuve de modélisation, option informatique

(*public 2010*)

Résumé : on définit deux méthodes de représentation des nombres entiers relatifs et on étudie la complexité de l'implantation des opérations usuelles entre nombres entiers ainsi représentés.

Mots clefs : circuits élémentaires.

► *Il est rappelé que le jury n'exige pas une compréhension exhaustive du texte. Vous êtes laissé(e) libre d'organiser votre discussion comme vous l'entendez. Des suggestions de développement, largement indépendantes les unes des autres, vous sont proposées en fin de texte. Vous n'êtes pas tenu(e) de les suivre. Il vous est conseillé de mettre en lumière vos connaissances à partir du fil conducteur constitué par le texte. Le jury appréciera que la discussion soit accompagnée d'exemples traités sur ordinateur.*

1. Représentation des nombres entiers

Il existe plusieurs conventions permettant de représenter dans une machine un nombre entier a par une suite finie de symboles appartenant à un alphabet fini donné. On s'intéressera ici uniquement aux représentations par une liste de chiffres dans une base B donnée, et plus précisément à deux conventions de permettant de représenter des entiers positifs ou négatifs et permettant de passer facilement de la représentation d'un nombre à celle de son opposé : soient $a \in \mathbf{Z}$, $B \in \mathbf{N}$ avec $B \geq 2$, $\beta \in \mathbf{N}$ avec $\beta < B \leq 2\beta + 1$ et $n \in \mathbf{N}^*$.

Une écriture de a en base B sur n chiffres en complément à la base est une décomposition :

$$a = \sum_{k=0}^{n-1} a_k B^k - \varepsilon B^n \text{ avec } \begin{cases} a_k \in [\![0, B-1]\!] & \text{pour tout } k \text{ ;} \\ \varepsilon = 0 & \text{si } 2a_{n-1} < B \text{ ;} \\ \varepsilon = 1 & \text{sinon.} \end{cases} \tag{1}$$

Une écriture de a en base B sur n chiffres signés entre $-\beta$ et β est une décomposition :

$$a = \sum_{k=0}^{n-1} a_k B^k \text{ avec } a_k \in [\![-\beta, \beta]\!] \text{ pour tout } k. \tag{2}$$

Proposition 1.

(1) *a admet une écriture de type* (1) *si et seulement si* $-B^{n-1}\lfloor B/2 \rfloor \leq a < B^{n-1}\lceil B/2 \rceil$. *Dans ce cas cette écriture est unique.*

(2) *a admet une écriture de type* (2) *si et seulement si* $|a| \leq \beta(B^n-1)/(B-1)$.
Si $B = 2\beta+1$ *alors il y a unicité.*
Si $B < 2\beta+1$ *et* $n \geq 2$*, il n'y a en général pas unicité.*

La représentation en complément à la base est celle qui est généralement implantée dans les ordinateurs. La propriété d'unicité de l'écriture d'un entier représentable permet de tester facilement l'égalité deux entiers. De plus, le signe d'un entier est en évidence : positif ou nul si $2a_{n-1} < B$, négatif sinon. Enfin on peut classer deux entiers en comparant lexicographiquement leurs chiffres.

La représentation en chiffres signés a été introduite par AVIZIENIS pour pouvoir réaliser des additionneurs fonctionnant en temps constant en exploitant la *redondance* de cette représentation, c'est-à dire la possibilité d'écrire un même nombre de plusieurs manières (cf. proposition 2, page suivante). Voici, à titre d'exemple, diverses représentations en chiffres signés, de longueurs 4 ou 5, pour le nombre 2006 avec $B = 10$ et $\beta = 8$ (le chiffre de poids faible est celui le plus à droite) :

$$2006 = (2,0,0,6)_{10,8} = (2,0,1,-4)_{10,8} = (1,-8,0,0,6)_{10,8} = (1,-8,0,1,-4)_{10,8}.$$

Malgré cette propriété de non unicité, la représentation en chiffres signés permet de comparer facilement un entier $a = \sum_{k=0}^{n-1} a_k B^k$ à zéro : on a $a = 0$ si et seulement si tous les chiffres a_k sont nuls, et dans le cas contraire le signe de a est celui de son dernier chiffre non nul.

2. Modélisation d'un circuit de calcul

On étudie dans les sections suivantes l'implantation matérielle ou logicielle des opérations usuelles sur les nombres entiers relatifs (addition, multiplication et division). Les opérandes et le résultat à calculer sont définis par des écritures dans une base B, c'est-à-dire par des suites finies de chiffres. On suppose savoir effectuer les opérations de base entre chiffres, et il s'agit de décrire comment agencer ces opérations de base de façon à obtenir des opérations entre suites de chiffres de longueurs arbitraires.

Formellement, on appelle *circuit arithmétique* un « dispositif » prenant en entrée un ou plusieurs chiffres ainsi que des informations auxiliaires telles que des retenues ou des variables booléennes, et fournissant en sortie un résultat, fonction des entrées, sous forme d'un ou plusieurs chiffres et d'informations auxiliaires. On dit que le circuit *réalise le calcul* de la fonction ou *implante* la fonction. Le circuit est dit *élémentaire* si les nombres d'entrées et de sorties sont bornés et s'il implante une fonction appartenant à une liste prédéfinie ; il est dit *composé* s'il est construit par assemblage de circuits élémentaires.

Dans l'optique d'une implantation matérielle, les circuits élémentaires sont les « composants de base », par exemple les portes logiques `et`, `ou` et `non` et les circuits composés sont les machines construites par interconnexion de composants de base. Dans l'optique d'une implantation logicielle, les circuits élémentaires sont les instructions ou les opérations disponibles dans un

langage de programmation et les circuits composés sont les programmes écrits avec ces instructions.

Deux paramètres mesurent la complexité d'un circuit composé :

– la *taille* du circuit : c'est le nombre de circuits élémentaires utilisés ;
– la *profondeur* du circuit : c'est le nombre maximal de circuits élémentaires intercalés entre une entrée et une sortie.

Si l'on suppose que tous les circuits élémentaires fonctionnent en parallèle et ont un temps de calcul identique, alors la taille et la profondeur donnent des indications approximatives sur le coût de réalisation du circuit et sur le temps nécessaire pour calculer le résultat. Si l'on suppose au contraire que les circuits élémentaires fonctionnent l'un après l'autre, par exemple si ce sont les instructions d'un programme exécuté sur une machine séquentielle, alors la taille du circuit composé donne une indication sur la complexité temporelle du programme. Dans ce cadre, la profondeur n'a pas d'interprétation simple.

3. Implantation de l'addition

Soient $a, b \in \mathbf{Z}$, donnés par des écritures dans une base B et $c \in \{-1, 0, 1\}$; on veut déterminer une écriture du nombre $a+b+c$ dans cette base. On suppose que les écritures de a et b obéissent à la même convention (en complément à la base ou en chiffres signés entre $-\beta$ et β), que les écritures de a et b sont de même longueur n, et que le résultat $a+b+c$ doit être donné sous la forme $a+b+c = s+dB^n$ où s est un nombre à n chiffres suivant la même convention de représentation que a et b, et $d \in \{-1, 0, 1\}$. On vérifie aisément qu'une telle décomposition de $a+b+c$ est toujours possible, tant pour l'écriture en complément à la base que pour l'écriture en chiffres signés. Les nombres c et d sont appelés respectivement *retenue entrante* et *retenue sortante.*

Proposition 2.

(1) *L'addition de deux nombres de n chiffres en complément à la base est réalisable avec un circuit composé de taille* $\Theta(n)$ *et de profondeur* $\Theta(\log n)$.

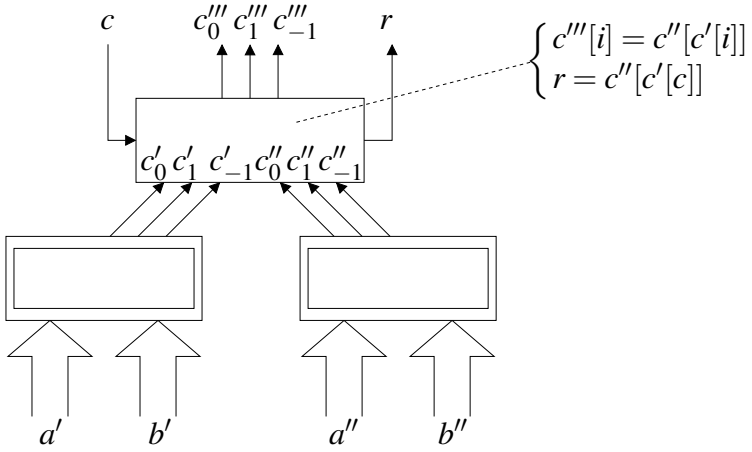
(2) *Si* $\beta < B < 2\beta$ *alors l'addition de deux nombres de n chiffres signés entre* $-\beta$ *et* β *est réalisable avec un circuit composé de taille* $\Theta(n)$ *et de profondeur* $\Theta(1)$.

Démonstration succincte

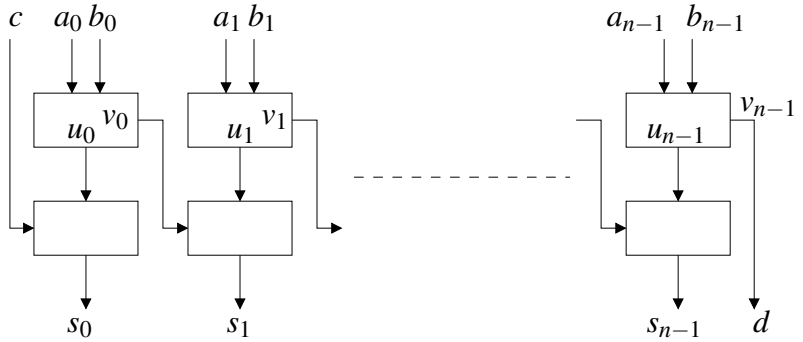
(1) Dans le cas de l'écriture en complément à la base, le point délicat est de déterminer les retenues intermédiaires lorsque l'on additionne les opérandes chiffre à chiffre. En supposant que n est pair, on « découpe » les listes des chiffres de a et b en deux listes de $n/2$ chiffres. Soient : $a' = \sum_{i=0}^{n/2-1} a_i B^i$, $a'' = \sum_{i=0}^{n/2-1} a_{i+n/2} B^i$, $a''' = \sum_{i=0}^{n-1} a_i B^i = a' + B^{n/2} a''$, et b', b'', b''' définis de même à partir de b. On calcule alors les six quantités suivantes :

$$\begin{aligned} c'[0] &= \lfloor (a'+b')/B^{n/2} \rfloor ; & c''[0] &= \lfloor (a''+b'')/B^{n/2} \rfloor ; \\ c'[1] &= \lfloor (a'+b'+1)/B^{n/2} \rfloor ; & c''[1] &= \lfloor (a''+b''+1)/B^{n/2} \rfloor ; \\ c'[-1] &= \lfloor (a'+b'-1)/B^{n/2} \rfloor ; & c''[-1] &= \lfloor (a''+b''-1)/B^{n/2} \rfloor . \end{aligned}$$

$c'[0], \ldots, c''[-1]$ sont appelées *retenues conditionnelles pour $a'+b'$ et $a''+b''$*. Ces informations suffisent à calculer à l'aide de circuits élémentaires appropriés les retenues conditionnelles pour $a'''+b''' : c'''[i] = \lfloor (a'''+b'''+i)/B^n \rfloor$ pour $i \in \{0,1,-1\}$, ainsi que la *retenue effective* $r = \lfloor (a'''+b'''+c)/B^n \rfloor$ en fonction de la retenue entrante c. Par application récursive de cette méthode au calcul de $c'[0], \ldots, c''[-1]$, on obtient un circuit composé calculant toutes les retenues intermédiaires, de taille $\Theta(n)$ et de profondeur $\Theta(\log n)$.



(2) Dans le cas de l'écriture en chiffres signés, on calcule pour $0 \leq i < n$ la somme $a_i + b_i$ et on la décompose sous la forme $a_i + b_i = u_i + Bv_i$ avec $u_i \in [\![-\beta+1, \beta-1]\!]$ et $v_i \in \{-1,0,1\}$. La somme $a+b+c = s+dB^n$ est alors obtenue par : $s_i = u_i + v_{i-1}$ (avec $v_{-1} = c$) et $d = v_{n-1}$.



■

Dans le cas de l'écriture en complément à la base, le résultat précédent est asymptotiquement optimal :

Proposition 3. (Winograd, 1965)
Tout circuit composé calculant la retenue sortante pour l'addition d'une retenue entrante et de deux nombres écrits en complément à la base sur n chiffres a une profondeur $\Omega(\log n)$.

Dans le cas de l'écriture en chiffres signés, la condition $\beta < B < 2\beta$ est non satisfiable pour $B = 2$. On peut toutefois réaliser un circuit additionneur de profondeur bornée avec $B = 2$ et $\beta = 1$ en regroupant les chiffres deux par deux.

4. Implantation de la multiplication

Ici les données sont deux entiers a et b définis par des écritures en base B sur n et p chiffres avec $n \leq p$ et on veut déterminer le produit ab par une écriture dans cette base, sur $n+p$ chiffres. Une telle écriture de ab existe si a, b et ab sont tous les trois représentés avec la même convention (complément à la base ou chiffres signés).

On peut procéder par découpage de l'opérande le plus court. Par exemple en supposant n pair : $a = a' + a''B^{n/2}$, d'où $ab = a'b + (a''b)B^{n/2}$. En procédant de la même manière pour calculer les produits $a'b$ et $a''b$ on obtient un circuit composé calculant ab, dont la taille est $\Theta(np)$ et la profondeur est :

$\Theta(\log n)$	si a, b, ab et les résultats intermédiaires sont écrits en chiffres signés ;
$\Theta(\log(n)\log(p))$	s'ils sont écrits en complément à la base ;
$\Theta(\log p)$	si a, b, et ab sont écrits en complément à la base et si les additions intermédiaires sont effectuées en chiffres signés.

On peut diminuer la complexité en taille en découpant les deux opérandes à la fois. Supposons pour simplifier que $n = p$, que n est pair, et décomposons : $a = a' + a''B^{n/2}$, $b = b' + b''B^{n/2}$. On a alors $ab = a'b' + a''b''B^n + ((a' + a'')(b' + b'') - a'b' - a''b'')B^{n/2}$. Cette relation, due à KARATSUBA, permet de réaliser un circuit multiplieur pour deux opérandes de n chiffres à l'aide de trois circuits multiplieurs pour des opérandes de $(n/2+1)$ chiffres et de circuits additionneurs. On en déduit récursivement un circuit multiplieur de taille $\Theta(n^\alpha)$ et de profondeur $\Theta(\log n)$ avec $\alpha = \log(3)/\log(2)$. Dans le cas où le circuit décrit un programme à exécuter sur une machine séquentielle, on obtient un algorithme calculant le produit de deux nombres de n chiffres avec une complexité temporelle $\Theta(t^\alpha)$ où t désigne la taille des données, et on peut faire en sorte que la complexité spatiale soit $\Theta(t)$.

5. Implantation de la division

Ici les données sont deux entiers a et b définis par des écritures en base B et on veut déterminer $c = \lfloor a/b \rfloor$. On suppose que l'on a $0 \leq a < B^{2n}$ et $B^{n-1} \leq b < B^n$ pour un certain entier $n \in \mathbf{N}^*$. L'algorithme suivant calcule c dans ce cas particulier :

(1) $u \leftarrow \lceil B^{2n}/b \rceil$;

(2) $v \leftarrow \lfloor ua/B^{2n} \rfloor$;

(3) `si` $a - bv \geq 0$ `alors` $c \leftarrow v$ `sinon` $c \leftarrow v - 1$.

Validité

Écrivons $B^{2n} = ub - r$ avec $0 \leq r < b$ et $ua = B^{2n}v + s$ avec $0 \leq s < B^{2n}$. Alors :

$$B^{2n}(a - bv) = (ub - r)a - B^{2n}bv = b(ua - B^{2n}v) - ra = bs - ra.$$

D'où $-b < a - bv < b$, ce qui prouve que la valeur attribuée à c en ligne 3 est correcte. ∎

Il reste à expliciter une méthode de calcul de u. En remarquant que $u \approx B^{2n}/b$, on utilise une version adaptée aux nombres entiers de l'algorithme de NEWTON pour résoudre l'équation $B^{2n}/x - b = 0$:

(1) $x \leftarrow B^n$;

(2) `répéter` $y \leftarrow x$; $x \leftarrow 2y - \lfloor by^2/B^{2n} \rfloor$ `jusqu'à avoir` $x \leq y$;

(3) $u \leftarrow y$.

Proposition 4.
L'algorithme précédent calcule $u = \lceil B^{2n}/b \rceil$, et la ligne 2 est répétée $\mathcal{O}(\log n)$ fois.

On en déduit que le calcul du quotient c peut être implanté par un circuit composé de taille $\mathcal{O}(M(n)\log n)$ où $M(n)$ désigne la taille d'un circuit multiplieur sur $\mathcal{O}(n)$ chiffres. Toutefois cette complexité en taille n'est pas optimale : il existe des algorithmes plus sophistiqués pour la division d'un nombre de $2n$ chiffres par un nombre de n chiffres ayant une complexité $\mathcal{O}(M(n))$.

Exercice de programmation :

► *Il vous est demandé de rédiger un programme conforme aux spécifications ci-dessous dans l'un des langages C, Caml ou Java à votre choix. Ce programme devra être accompagné d'un exemple d'exécution permettant d'en vérifier le bon fonctionnement. La clarté et la concision du programme seront des éléments importants d'appréciation pour le jury.*

Écrire un programme prenant en entrée deux nombres a, b, représentés par des écritures en base B avec des chiffres signés entre $-\beta$ et β, et retournant un résultat ternaire indiquant si $a < b$, si $a = b$ ou si $a > b$. On supposera que $\beta < B < 2\beta$, et que les écritures de a et b ont même longueur.

Suggestions pour le développement

► *Soulignons qu'il s'agit d'un menu à la carte et que vous pouvez choisir d'étudier certains points, pas tous, pas nécessairement dans l'ordre, et de façon plus ou moins fouillée. Vous pouvez aussi vous poser d'autres questions que celles indiquées plus bas. Il est très vivement souhaité que vos investigations comportent une partie traitée sur ordinateur et, si possible, des représentations graphiques de vos résultats.*

- Compléter la démonstration de la proposition 2 et démontrer les propositions 3 et 4.
- Justifier les complexités données à la section 4.
- Présenter et analyser d'autres implantations de la multiplication.
- Présenter une autre convention d'écriture des entiers relatifs et décrire les implantations des opérations usuelles avec cette convention.
- Étudier l'implantation d'autres opérations entre nombres entiers (racine carrée, puissance, logarithme, pgcd, …)