ΠΛΗΡΟΦΟΡΙΚΗ Ι

Σημειώσεις

Αριθμητική Κινητής Υποδιαστολής

Συστήματα αριθμών και στοιχεία αριθμητικής υπολογιστών

Συστήματα αριθμών

Κάθε αριθμός

$$z_{(\beta)} = \pm\, d_{n-1}\, d_{n-2} \ldots d_1\, d_0 \,,\, d_{-1}\, d_{-2} \ldots d_{-m}$$

ενός συστήματος αρίθμησης μπορεί να τεθεί υπό την ακόλουθη γενική μορφή:

$$z_{(b)} = \pm\, d_{n-1}\, b^{n-1} + \cdots + d_1\, b^1 + d_0\, b^0 + d_{-1}\, b^{-1} + \cdots + d_{-m}\, b^{-m}$$

$$= \pm \sum_{i=-m}^{n-1} d_i\, b^i \qquad (\ .1)$$

όπου:

$z_{(b)}$: = αριθμός ενός συστήματος αρίθμησης
b : = βάση του συστήματος αρίθμησης ($b \geqslant 2$, με b αριθμό του δεκαδικού συστήματος)
d_i : = ψηφία του αριθμού: $0 \leqslant d_i \leqslant b_{-1}$
n : = αριθμός ψηφίων προ της υποδιαστολής
m : = αριθμός ψηφίων μετά την υποδιαστολή

Σημείωση: Συνήθη συστήματα αριθμών είναι τα ακόλουθα:

α) Δεκαδικό με ψηφία τα {0, 1, 2, 3, 4, 5, 6, 7, 8, 9}
β) Δυαδικό με ψηφία τα {0, 1}
γ) Οκταδικό με ψηφία τα {0, 1, 2, 3, 4, 5, 6, 7}
δ) Δεκαεξαδικό με ψηφία τα {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F}. Επειδή στο δεκαεξαδικό σύστημα τα δέκα ψηφία του δεκαδικού δεν αρκούν, συμπληρώνουμε τα υπόλοιπα ψηφία (με τιμές 10, 11, 12, 13, 14, 15) με τα έξι πρώτα κεφαλαία γράμματα του λατινικού αλφάβητου.

Παράδειγμα

α) Έστω $z_{(b)} = 125.35_{(10)}$

$$z_{(b)} = 1\cdot10^2 + 2\cdot10^1 + 5\cdot10^0 + 3\cdot10^{-1} + 5\cdot10^{-2}$$

β) Έστω $z_{(b)} = 10011_{(2)}$

$$z_{(b)} = 1\cdot2^4 + 0\cdot2^3 + 0\cdot2^2 + 1\cdot2^1 + 1\cdot2^0$$

γ) Έστω $z_{(b)} = E02.4_{(16)}$

$$z_{(b)} = E\cdot16^2 + 0\cdot16^1 + 2\cdot16^0 + 4\cdot16^{-1}$$

Μετατροπή αριθμού $z_{(b)}$ σε $z_{(10)}$

Η μετατροπή γίνεται με βάση την παράσταση (1) αρκεί να θεωρηθούν τα ψηφία d_i, του συστήματος με βάση b, ως αριθμοί του δεκαδικού συστήματος.

Παράδειγμα

α) Έστω $z_{(b)} = E02.4_{(16)}$

$$\begin{aligned} z_{(10)} &= E\cdot16^2 + 0\cdot16^1 + 2\cdot16^0 + 4\cdot16^{-1} \\ &= 14\cdot16^2 + 0\cdot16^1 + 2\cdot16^0 + 4\cdot16^{-1} \\ &= 14\cdot256 + 0 + 2 + 0.25 = 3586.25_{(10)} \end{aligned}$$

β) Έστω $z_{(b)} = 100101_{(2)}$

$$\begin{aligned} z_{(10)} &= 1\cdot2^5 + 0\cdot2^4 + 0\cdot2^3 + 1\cdot2^2 + 0\cdot2^1 + 1\cdot2^0 \\ &= 1\cdot32 + 0 + 0 + 4 + 0 + 1 = 37_{(10)} \end{aligned}$$

Μετατροπή ακέραιου αριθμού $z_{(10)}$ σε $z_{(b)}$

Η μετατροπή των ακεραίων δεκαδικών αριθμών σε αριθμούς άλλων συστημάτων, βασίζεται πάλι στον πολυωνυμικό τρόπο γραφής (1) ενός αριθμού. Έστω ο αριθμός

$$z_{(10)} = d_{n-1}b^{n-1} + \cdots + d_1b^1 + d_0 \qquad (.2)$$

Σ' αυτόν τον αριθμό δίνονται τα $z_{(10)}$, b και ζητούνται τα d_i, $i = 0(1)n-1$.

Με τη βοήθεια του σχήματος Horner η παράσταση (.2) γράφεται:

$$\begin{aligned} z_{(10)} &= d_{n-1}b^{n-1} + \cdots + d_1b^1 + d_0 \\ &= d_0 + b(d_1 + b(d_2 + b(d_3 + \cdots + bd_{n-1}))\cdots) \end{aligned} \qquad (.3)$$

Ο προσδιορισμός των d_i γίνεται με βάση την (2.3) ως εξής:

1) Διαιρούμε τον $z_{(10)}$ δια $b \Rightarrow d_0$ (υπόλοιπο διαίρεσης)

2) Το πηλίκο $d_1 + b(d_2 + b(d_3 + \cdots + bd_{n-1}))\cdots)$ το διαιρούμε πάλι με $b \Rightarrow d_1$ (υπόλοιπο διαίρεσης)

.

.

.

n) $\Rightarrow d_{n-1}$

Το τελευταίο βήμα εμφανίζεται όταν ο διαιρετέος είναι μικρότερος του b και επομένως έχουμε πηλίκο 0 και υπόλοιπο d_{n-1}.

Στη συνέχεια δίνουμε ως εφαρμογή τη μετατροπή ενός αριθμού $z_{(10)}$ σε έναν π.χ. τετραψήφιο ακέραιο στο σύστημα με βάση b.

Εφαρμογή

$$z_{(10)} = d_0 + \underbrace{b(d_1 + b(d_2 + bd_3))}_{\alpha_0}$$

$$\alpha_1 = d_1 + b(d_2 + bd_3)$$

$$\alpha_2 = d_2 + bd_3$$

$$\alpha_3 = d_3$$

$$z_{(10)} = d_3 d_2 d_1 d_{0_{(b)}}$$

Σχηματική πορεία αλγόριθμου

$\alpha_0 : b = \alpha_1$ υπολ. d_0

$\alpha_1 : b = \alpha_2$ υπολ. d_1

$\alpha_2 : b = \alpha_3$ υπολ. d_2

$\alpha_3 : b = 0$ υπολ. d_3

Παράδειγμα

α) Να μετατραπεί ο αριθμός $42_{(10)}$ σε δυαδικό

$$\left.\begin{array}{rlll} 42:2 = 21 & \text{υπολ.} & 0 & (d_0) \\ 21:2 = 10 & » & 1 & (d_1) \\ 10:2 = 5 & » & 0 & (d_2) \\ 5:2 = 2 & » & 1 & (d_3) \\ 2:2 = 1 & » & 0 & (d_4) \\ 1:2 = 0 & » & 1 & (d_5) \end{array}\right\} \quad 42_{(10)} = 101010_{(2)}$$

β) Να μετατραπεί ο αριθμός $42_{(10)}$ σε δεκαεξαδικό

$$\left.\begin{array}{rlll} 42:16 = 2 & \text{υπολ.} & 10 & (d_0) \\ 2:16 = 0 & » & 2 & (d_1) \end{array}\right\} \quad 42_{(10)} = 2A_{(16)}$$

Μετατροπή δεκαδικού κλάσματος $.z_{(10)}$ σε $.z_{(b)}$

Θεωρούμε το δεκαδικό κλάσμα

$$z_{(10)} - [z_{(10)}] = .z_{(10)}$$

Αυτό τίθεται στη μορφή

$$.z_{(10)} = d_{-1} b^{-1} + d_{-2} b^{-2} + \cdots + d_{-m} b^{-m} \quad ή$$

$$.z_{(10)} = b^{-1} (d_{-1} + b^{-1} (d_{-2} + \cdots + b^{-1} d_{-m})) \cdots) \qquad (2.4)$$

Στον αριθμό της παράστασης (2.4) δίνονται τα $.z_{(10)}$ και b και ζητούνται τα d_i, $i = -1(-1)-m$. Ο προσδιορισμός αυτών γίνεται από την (2.4) ως εξής:

— Πολλαπλασιάζουμε και τα δύο μέλη της (2.4) επί b $\Rightarrow d_{-1}$

— Διαγράφουμε το d_{-1} και το υπόλοιπο $b^{-1}(d_{-2} + \cdots + b^{-1} d_{-m}) \cdots)$ το πολλαπλασιάζουμε πάλι επί b $\Rightarrow d_{-2}$ κ.ο.κ.

Ο αλγόριθμος διακόπτεται είτε αν έχουμε υπολογίσει έναν ικανοποιητικό αριθμό ψηφίων, είτε αν μετά τον πολλαπλασιασμό με το b δεν προκύπτει υπόλοιπο.

Εφαρμογή. Μετατροπή ενός αριθμού $.z_{(10)}$ σε έναν αριθμό, με τρία ψηφία μετά την υποδιαστολή, στο σύστημα με βάση b.

Σχηματική πορεία αλγόριθμου

$$.z_{(10)} = \underbrace{b^{-1} (d_{-1} + b^{-1} (d_{-2} + b^{-1} d_{-3}))}_{\alpha_{-1}} \qquad \alpha_{-1} \cdot b = d_{-1} + \alpha_{-2}$$

$$\alpha_{-2} = b^{-1} (d_{-2} + b^{-1} d_{-3}) \qquad \alpha_{-2} \cdot b = d_{-2} + \alpha_{-3}$$

$$\alpha_{-3} = b^{-1} d_{-3} \qquad \alpha_{-3} \cdot b = d_{-3} + 0$$

Άρα $.z_{(10)} = .d_{-1} d_{-2} d_{-3_{(b)}}$

Παράδειγμα

α) Να μετατραπεί ο αριθμός $0.825_{(10)}$ σε δεκαεξαδικό

$$0.825 \cdot 16 = 13 + 0.2$$

$$0.2 \cdot 16 = 3 + 0.2$$

$$0.2 \cdot 16 = 3 + 0.2$$

Άρα $0.825_{(10)} = 0.D33 \ldots_{(16)}$

β) Μετατροπή του $0.03125_{(10)}$ σε δεκαεξαδικό

$0.03125 \cdot 16 = 0 + 0.5$

$0.5 \cdot 16 = 8 + 0$

Άρα $0.03125_{(10)} = 0.08_{(16)}$

γ) Μετατροπή του $0.2_{(10)}$ σε δυαδικό

$0.2 \cdot 2 = 0 + 0.4$

$0.4 \cdot 2 = 0 + 0.8$

$0.8 \cdot 2 = 1 + 0.6$

$0.6 \cdot 2 = 1 + 0.2$

$0.2 \cdot 2 = 0 + 0.4$

κ.λπ. επαναλαμβάνονται περιοδικά τα 4 προηγούμενα ψηφία.

Άρα $0.2_{(10)} = 0.0011\,0011\ldots_{(2)}$

Παράδειγμα Μετατροπή του $213.04_{(10)}$ σε δεκαεξαδικό αριθμό.

Μετατροπή <u>ακεραίου μέρους</u>:

$213 : 16 = 13$ υπολ. 5

$13 : 16 = 0$ υπολ. 13

Μετατροπή του <u>δεκαδικού κλάσματος</u>:

$0.04 \cdot 16 = 0 + 0.64$

$0.64 \cdot 16 = 10 + 0.24$

$0.24 \cdot 16 = 3 + 0.84$

$0.84 \cdot 16 = 13 + 0.44$

.

.

.

Άρα $213.04_{(10)} = D5.0A3D\ldots_{(16)}$

1.1.2 Υπολογισμοί Κινητής Υποδιαστολής

Το πρόβλημα αυτού του περιορισμού μπορεί να ξεπερασθεί εάν για παράδειγμα τον ακέραιο 976000000000000 τον παραστήσουμε ως $\boxed{9.76 * 10^{14}}$ και τον ακέραιο 0.00000000000000976 τον παραστήσουμε ως $9.76 * 10^{-14}$. Μετακινώντας έτσι δυναμικά το δεκαδικό σημείο σε κατάλληλη θέση και χρησιμοποιώντας τον εκθέτη του 10 ώστε να αποθηκεύεται η μετακίνηση αυτή, αυξάνουμε το εύρος παράστασης των

αριθμών και πετυχαίνουμε τη δυνατότητα παράστασης πολύ μεγάλων και πολύ μικρών αριθμών με μόνο λίγα ψηφία. Εισάγουμε έτσι την έννοια των υπολογισμών κινητής υποδιαστολής, που μπορούν να ορισθούν ως εξής:

Ενας διάφορος του μηδενός δεκαδικός αριθμός παριστάνεται ως **κανονικοποιημένος floating point** αριθμός σαν

$$x = \sigma \cdot (.a_1 a_2 \ldots a_t)_\beta \cdot \beta^e$$

όπου σ το πρόσημο του, e ο εκθέτης και a_i τα ψηφία της χαρακτηριστικής (mantissa), έτσι ώστε $0 \leq a_i \leq \beta - 1, a_1 \neq 0$. Επίσης $m \leq e \leq M$, και γενικά $m = -M$ ή $m = -M + 1$.

Ορισμός: Το σύνολο $\mathcal{M} \subset \mathbb{R}$ όλων των υπό χανονιχοποιημένη παράσταση αριθμών $x = \sigma \cdot \bar{x} \cdot \beta^e$,

$$\bar{x} = (0.a_1 a_2 \ldots a_t)_\beta = \sum_{k=1}^{t} a_k \cdot \beta^{-k}$$

όπου σ υποδηλώνει το πρόσημο, $t \in \mathbb{N}$, $1 \leq a_1 \leq \beta - 1$, $0 \leq a_k \leq \beta - 1$, $k = 2, 3, \ldots, t$, $m \leq e \leq M$, $m \leq 0$, $M \geq 1$, και $e, m, M \in \mathbb{Z}$ καλείται **σύστημα κινητής υποδιαστολής** και συμβολίζεται με $\mathcal{M}(\beta, t, m, M)$ ή $\mathcal{M}$.
Το $0 \in \mathcal{M}$ (από τον ορισμό) και είναι ο αριθμός $0 := +0.00 \ldots 0\beta^m$
Κάθε $x \in \mathcal{M}$ ονομάζεται αριθμός μηχανής

Παρατηρήσεις:

- $\mathcal{M} \subset \mathbb{R}$
- κάθε $x \in \mathcal{M}$ παριστάνει έναν **πραγματικό αριθμό**
- $0, 1 \in \mathcal{M}$ β^0
- $\forall x \in \mathcal{M} \Rightarrow -x \in \mathcal{M}$
- Το σύνολο των κανονικοποιημένων floating point αριθμών που ανήκουν στο $\mathcal{M}$ ισούται με $2(\beta - 1)\beta^{t-1}(M - m + 1) + 1$.

και οι αρνητικοί — ο αριθμός 0 — διαστημάτων

ΑΣΚΗΣΗ :

Να προσδιοριστούν όλοι οι αριθμοί μηχανής που ανήκουν στο $\mathbb{M}(2, 3, -1, 1)$
Να παρασταθεί η ευθεία των αριθμών μηχανής

Κατ' αρχήν στο $\mathbb{M}(\underset{\beta}{2}, \underset{t}{3}, \underset{m}{-1}, \underset{M}{1})$ ανήκουν

$$2(\beta-1)\beta^{t-1}(1-(-1)+1) = 2\cdot\underbrace{1\cdot 2^2}_{4 \text{ χαρακτηρ.}}\cdot 3+1 = 25$$

Κάθε $x \in \mathbb{M}$ είναι της μορφής

$$x = \sigma\cdot(.a_1a_2a_3)_2\cdot 2^e$$

Δυνατές χαρακτηριστικές

$$(0.100)_{(2)} = 1*2^{-1} = 1/2_{(10)}$$

$$(0.101)_{(2)} = 1*2^{-1}+1*2^{-3}_{(2)} = \frac{5}{8}_{(10)}$$

$$(0.110)_{(2)} = 1*2^{-1}+1*2^{-2}_{(2)} = \frac{3}{4}_{(10)}$$

$$(0.111)_{(2)} = 1*2^{-1}+1*2^{-2}+1*2^{-3} = \frac{7}{8}_{(10)}$$

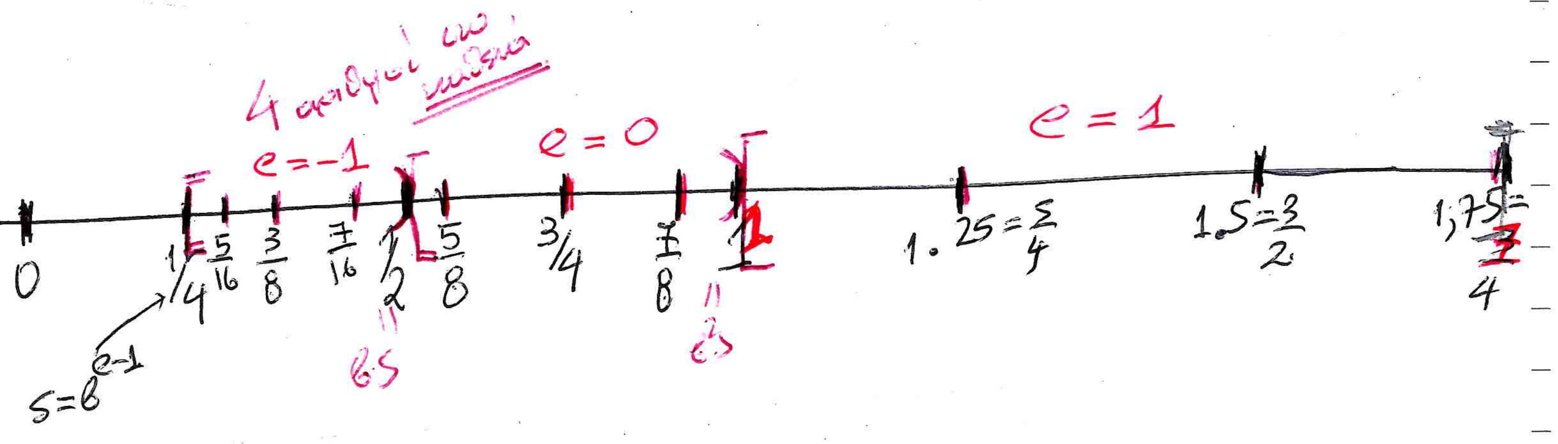
Για ευκολία θα αναπαραστήσουμε τους αριθμούς στο δεκαδικό σύστημα (δεκαδική αναπαράσταση).

Ο παρακάτω πίνακας καταγράφει τους 12 υπάρχοντες θετικούς αριθμούς μηχανής από τον τύπο $\boxed{x = m \cdot 2^{e}}$

εκθ e \ mantissa m	$\frac{1}{2}$	$\frac{5}{8}$	$\frac{3}{4}$	$\frac{7}{8}$
-1	$\frac{1}{2} * 2^{-1} = \frac{1}{4}$ μικρότερος	$\frac{5}{8} * 2^{-1}$ $\frac{5}{16}$	$\frac{3}{4} * 2^{-1}$ $\frac{3}{8}$	$\frac{7}{8} * 2^{-1}$ $\frac{7}{16}$
0	$\frac{1}{2}$	$\frac{5}{8}$	$\frac{3}{4}$	$\frac{7}{8}$
1	$\frac{1}{2} * 2$ 1	$\frac{5}{8} * 2$ $\frac{5}{4}$	$\frac{3}{4} * 2$ $\frac{3}{2}$	$\frac{3}{4} * 2$ $\frac{7}{4}$ μεγαλύτερος

Μεγαλύτερο t → πιο πυκνοί αριθμοί

Μεγαλύτερο e → Μεγαλύτερο εύρος αναπαράστασης



Παρατηρούμε ότι οι αριθμοί είναι πυκνότεροι κοντά στο 0 και αραιότεροι όσο απομακρυνόμαστε από αυτό

Παραδείγματα Overflow - Underflow

Πιο σημαντικό πρόβλημα εμφανίζεται στην υπερχείλιση (overflow) όπου δίνεται σαν τελικό αποτέλεσμα η τιμή $\pm\infty$.
Σαν αποτέλεσμα της υποχείλισης δίνεται η τιμή 0 ή $\pm\beta^m$ ή κάποιος μη κανονικοποιημένος αριθμός.

Υπερχείλιση

$\beta = 10,\ t = 3,\ m = -4,\ M = 4$
$a = 0.111 \cdot 10^4$
$b = 0.120 \cdot 10^4$
$c = a \cdot b = 0.133 \cdot 10^7$

Υποχείλιση

$\beta = 10,\ t = 3,\ m = -3,\ M = 4$
$a = 0.1 \cdot 10^{-3}$
$b = 0.2 \cdot 10^{-3}$
$c = a \cdot b = 0.2 \cdot 10^{-7}$

Απλές μαθηματικές πράξεις μπορεί να οδηγήσουν σε υπερχείλιση. Με κατάλληλη οργάνωση μπορεί αυτό να αποφευχθεί.

Παράδειγμα 1.1.6 *Υπολογίστε τη $\|\underline{x}\|_2^2$, $\underline{x} \in \mathbb{R}^n$ με τον καλύτερο δυνατό τρόπο.*

Απόδειξη: Αυτός ο υπολογισμός μπορεί να εκτελεσθεί με έναν εκ των δύο αλγορίθμων.

Αλγόριθμος 1

$\|\underline{x}\|_2 = \sqrt{x_1^2 + x_2^2 + \ldots + x_n^2}$

Αλγόριθμος 2

$m = \max(|x_1|, \ldots, |x_n|)$
$y_i = \frac{x_i}{m}, i = 1, \ldots, n$
$\|\underline{x}\|_2 = m \cdot \sqrt{y_1^2 + y_2^2 + \ldots + y_n^2}$

Ενώ ο αλγόριθμος 1 μπορεί να οδηγήσει σε υπερχείλιση π.χ. εάν ο μεγαλύτερος παραστήσιμος floating point αριθμός είναι ο 10^{50} και ορισμένες συντεταγμένες του $\underline{x}$ είναι της τάξης του 10^{30} εάν εφαρμοσθεί η τεχνική του αλγορίθμου 2 τα αποτελέσματα ελέγχονται. Εν γένει, εάν δουλεύουμε με αριθμούς φραγμένους (ιδανική περίπτωση εάν όλοι είναι μικρότεροι της μονάδας) ελέγχεται το εύρος των αναμενόμενων τιμών. □

Example: Αποφυγή Overflow

$$x = (10^{30}, 10^{20}, 10^{10})^t$$

εάν μεγ. παραστ αριθ $= 10^{50}$, ελάχιστος 10^{-50}

Αλγ 1: $\|x\|_2 = \sqrt{\underbrace{(10^{30})^2}_{\text{overflow}} + (10^{20})^2 + (10^{10})^2}$

$$y = \left(1, \frac{1}{10^{10}}, \frac{1}{10^{20}}\right)^t$$

Αλγ 2: $\|x\|_2 = 10^{30} \cdot \sqrt{1 + \left(\frac{1}{10^{10}}\right)^2 + \left(\frac{1}{10^{20}}\right)^2}$

$$10^{30} \cdot \sqrt{1 + 10^{-20} + 10^{-40}}$$

Στρογγύλευση και αποκοπή

Εάν ο πραγματικός αριθμός δεν έχει mantissa $\bar{x}$ που να ταιριάζει στη διαθέσιμη ακρίβεια των t ψηφίων θα πρέπει να περικοπεί. Ετσι εισάγονται τα λεγόμενα σφάλματα στρογγύλευσης (rounding errors). Υπάρχουν οι τεχνικές της αποκοπής (όσα ψηφία είναι παραπάνω από t αποκόπτονται) και της στρογγύλευσης.

Εστω $\hat{x}$ προσέγγιση της τιμής x. Το σφάλμα μπορεί να εκτιμηθεί με τις ποσότητες:

Απόλυτο σφάλμα $= |\hat{x} - x|$

Σχετικό σφάλμα $= \frac{|\hat{x}-x|}{|x|}, x \neq 0$

Το σχετικό σφάλμα δίνει πάντα καλλίτερη εκτίμηση απ' ότι το απόλυτο και είναι ανεξάρτητο από το scaling. Συγκεκριμένα, το σχετικό σφάλμα των ποσοτήτων x και $\hat{x}$ είναι ίδιο με το σχετικό σφάλμα των ποσοτήτων ax και $a\hat{x}$.

Παράδειγμα 1.1.7 *Εστω* $x_1 = \mathbf{1.31}$, $\hat{x}_1 = \mathbf{1.30}$ *και* $x_2 = \mathbf{0.12}$, $\hat{x}_2 = \mathbf{0.11}$. *Ποια εκ των δύο προσεγγίσεων είναι καλλίτερη;*

Απόδειξη:

$$|\hat{x_1} - x_1| = |\hat{x_2} - x_2| = 0.01$$

$$\left|\frac{\hat{x_1} - x_1}{x_1}\right| = 0.0076$$

$$\left|\frac{\hat{x_2} - x_2}{x_2}\right| = 0.0833$$

Άρα το $\hat{x_1}$ είναι κοντύτερα στο x_1 απ' ότι το $\hat{x_2}$ στο x_2.

Αρα το $\hat{x}_1$ είναι κοντύτερα στο x_1 απ'οτι το $\hat{x}_2$ στο ...

ΕΚΤΕΛΕΣΗ ΠΡΑΞΕΩΝ

Έστω $\beta = 10, t = 3, u = 0.005$
$x = 0.101 * 10^2, y = -0.994 * 10^1$ (**Θεωρητικά:** $x + y = 0.16$)
Να υπολογιστεί το $fl(x+y)$ χρησιμοποιώντας επεξεργαστή διπλής ακριβείας (double precision accumulator).

Απόδειξη:

Βήμα 1:

$$\text{Ευθυγράμμιση στους εκθέτες} \begin{cases} x = 0.101\overbrace{000} * 10^2 \\ y = -0.099\underbrace{400} * 10^2 \end{cases}$$

Ευθυγράμμιση μικρότερου αριθμού

Βήμα 2: Εκτελούμε την πρόσθεση (εσωτερικά κρατάμε ψηφία 6 θέσεων)

$$x + y = 0.001600 * 10^2$$

Βήμα 3: Κανονικοποιούμε

$$fl(x+y) = 0.160 * 10^0$$

Τελικά $fl(x+y) = (x+y)(1+\delta)$, $\delta = 0$

Εάν δε χρησιμοποιηθεί επεξεργαστής διπλής ακριβείας η πράξη θα εκτελεσθεί ως εξής:

Βήμα 1: Υποθέτουμε ότι τα επιπλέον ψηφία που δε χωράνε στη mantissa, απλώς διαγράφονται.

$$\begin{cases} x = 0.101 * 10^2 \\ y = -0.099 * 10^2 \end{cases}$$

Βήμα 2:

$$fl(x+y) = 0.002 * 10^2$$

Βήμα 3: Κανονικοποίηση $fl(x+y) = 0.2$
Τελικά

$$fl(x+y) = (x+y)(1+\sigma) \;\; \text{όπου } \sigma = 0.25 = \mathbf{50u}$$

$0.2 = 0.16 * 1.25$

Συμπέρασμα: Πάντα χρησιμοποιείται επεξεργαστής διπλής ακριβείας. □

Άσκηση

Σε αριθμητική κινητής υποδιαστολής $M(10,4,4,4)$, να προστεθούν οι ακόλουθοι αριθμοί αρχικά εξ αριστερών προς τα δεξιά και στη συνέχεια εκ δεξιών προς τα αριστερά

$$x_1 = 0.1025 * 10^4$$
$$x_2 = -0.9123 * 10^3$$
$$x_3 = -0.9663 * 10^2$$
$$x_4 = -0.9315 * 10$$

Θεωρητική τιμή

$$x_1+x_2+x_3+x_4 = 6.755$$

1) $\overrightarrow{fl}(x_1+x_2+x_3+x_4) =$

$$\underbrace{fl\Big(\big(\underbrace{fl(x_1+x_2)}_{s_2}+x_3\big)}_{s_3}+x_4\Big)\ \to s_4$$

$$s_2 = fl(x_1+x_2)$$

$$= \begin{array}{r} 0.1025 * 10^4 \\ -0.9123 * 10^3 \end{array} \xrightarrow[\text{εκθέτη}]{\text{ευθυγρ}} \begin{array}{r} 0.10250 * 10^4 \\ -0.09123 * 10^4 \\ \hline 0.01127 * 10^4 \end{array}$$

$\downarrow$ κανονικοποίηση

$$0.1127 * 10^3$$

no rounding

$$s_3 = fl(s_2+x_3)$$

$$= \begin{array}{r} 0.1127 * 10^3 \\ -0.9663 * 10^2 \end{array} \xrightarrow[\text{εκθ}]{\text{ευθ}} \begin{array}{r} 0.11270 * 10^3 \\ -0.09663 * 10^3 \\ \hline 0.01607 * 10^3 \end{array}$$

$\downarrow$ κανονικοποίηση

$$0.1607 * 10^2$$

no rounding

$s_4 = fl(s_3 + x_4) =$

$$\begin{array}{r} 0.1607 * 10^2 \\ -0.9315 * 10 \end{array} \xrightarrow[\text{εκθετών}]{\text{εξισ.}} \begin{array}{r} 0.16070 * 10^2 \\ -0.09315 * 10^2 \\ \hline 0.06755 * 10^2 \end{array}$$

$$\Big\downarrow \text{κανονικοποίηση}$$

$$\boxed{0.6755 * 10} = 6.755$$

2) $fl(\underbrace{\underbrace{\underbrace{x_4 + x_3}_{s_3} + x_2}_{s_2} + x_1}_{s_1})$

$s_3 = fl(x_4 + x_3)$

$$= \begin{array}{r} -0.9663 * 10^2 \\ -0.9315 * 10 \end{array} \xrightarrow{\text{εξισ.}} \begin{array}{r} -0.96630 * 10^2 \\ -0.09315 * 10^2 \\ \hline -1.05945 * 10^2 \end{array}$$

$$\Big\downarrow \text{κανονικ.}$$

$$-0.105945 * 10^3$$

$$\Big\downarrow \text{στρογγυλ. σε 4 ψηφία}$$

$$-0.1059 * 10^3$$

$s_2 = fl(s_3 + x_2) =$

$$\begin{array}{r} -0.1059 * 10^3 \\ -0.9123 * 10^3 \\ \hline -1.0182 * 10^3 \end{array}$$

$\Big\downarrow$ κανονικ.

$-0.10182 * 10$

$\Big\downarrow$ στρογγ. σε 4 ψηφία

$-0.1018 * 10^4$

$s_1 = fl(s_2 + x_1)$

$$= \begin{array}{r} 0.1025 * 10^4 \\ -0.1018 * 10^4 \\ \hline 0.0007 * 10^4 \end{array}$$

$\Big\downarrow$ κανονκ.

$0.7 * 10$ υπολογισμ. τιμή

Σχετικό Σφάλμα (Relative error) $:= \dfrac{|\text{θεωρ. τιμή} - \text{υπολογισμ}|}{|\text{θεωρ}|} = \dfrac{|6.755 - 7|}{|6.755|} = 0.0362694$

$(O(10^{-2}))$

Facts

Basic (Rules) in Floating Point Arithmetic.

Characteristics

In a system $M(b, t, \varepsilon)$ it holds

1) $fl(1 + b^{-t}) = 1$, αγνοείται το b^{-t}
2) $fl(1 + b^{t}) = b^{t}$, αγνοείται η μονάδα

3) $fl(1 - b^{(t+1)}) = -b^{(t+1)}$, αγνοείται η μονάδα
4) $fl(1 - b^{-(t+1)}) = 1$, αγνοείται το $b^{-(t+1)}$

Παράδειγμα:

Έστω το σύστημα κινητής υποδιαστολής

$M(10, 3, e)$

στο οποίο οι πράξεις εκτελούνται με επεξεργαστή διπλής ακρίβειας.

1) $fl(1 + 10^{-3}) = 1$

$1 \longrightarrow 0.100|000 * 10$

$10^{-3} \longrightarrow 0.000|100 * 10$

$\| $

0.001

$+$ ________________

$0.100|100 * 10$

είναι normalized

↓ στρογγυλοποίηση σε 3 ψηφία

$0.1 * 10 = 1$

Δηλαδή

$fe(1 + 10^{-3}) = 1 \neq$ θεωρητική τιμή $= \underline{1.001}$

Αγνοείται ο όρος 10^{-3}

---

2) $fe(1 + 10^{3})$

$10^{3} \longrightarrow 0.100 \vdots 000 \times 10^{4}$

$1 \longrightarrow 0.000 \vdots 100 \times 10^{4}$

$+ \quad 0.100 \vdots 100 \times 10^{4}$

είναι normalized

↓ στρογγυλέυση σε 3 ψηφία

$0.100 \times 10^{4} = 10^{3}$

Δηλαδή

$fe(1+10^{3}) = 10^{3} \neq$ θεωρητική τιμή 1001

Αγνοείται η μονάδα

3) $fl(1 - 10^4)$

$-10^4 \longrightarrow -0.100\,000 * 10^5$

$1 \longrightarrow 0.000\,01 * 10^5$

$-0.09999 * 10^5$

↓ κανονικοποίηση

$-0.9999 * 10^4$

↓ στρογγυλοποίηση σε 3 ψηφία

$+0.001$

$-1.000 * 10^4$

Τελικά $fl(1-10^4) = -10^4 \neq$ θεωρητική τιμή $= -9999$

Αγνοείται η μονάδα

4) $fl(1-10^{-4})$

$1 \longrightarrow 0.100\,00 * 10^1$

$-10^{-4} \longrightarrow -0.000\,01 * 10^1$

$0.09999 * 10$

↓ κανονικοποίηση

0.9999

$\downarrow$ στρογγυλέυση σε 3 ψηφία

$$\begin{array}{r} 0.999{:}9 \\ +\ 0.001 \\ \hline 1.000 \end{array}$$

Τελικά $fl(1-10^{-4}) = 1 \neq$ θεωρητική τιμή $= 0.9999$

Αγνοείται ο όρος 10^{-4}

<u>Παρατήρηση</u>

Σε ένα σύστημα κινητής υποδιαστολής $M(b,t,e)$ το b^{-t} είναι ο μεγαλύτερος δεκτικός αριθμός x για τον οποίο ισχύει $1+x=1$

Εάν χρησιμοποιήσουμε το IEEE format 754 τότε $fl(1+b^{-(t+1)})=1$

Πχ $fl(1+10^{-4})=1$ για $M(10,\underset{3}{t},e)$ σε

$$\begin{array}{lr} 1 \longrightarrow & 1.000\,000 \\ 10^{-4} = 0.0001 \longrightarrow & .000\,100 \ + \\ \hline & 1.000{:}100 \end{array}$$

είναι normalized

$\downarrow$ στρογγυλέυση σε 3 θέσεις

1.000

Παράδειγμα 1.1.3 *Έστω ότι σε ένα σύστημα κινητής υποδιαστολής έχουμε* $\beta = 10, t = 3, m = -1, M = 2$. *Αναπαραστήστε στο σύστημα αυτό τους αριθμούς* $a = 11.2,\ b = 1.13,\ c = a \cdot b$.

Απόδειξη: Έχουμε, $a = 11.2 = 0.112 \cdot 10^2$,
$b = 1.13 = 0.113 \cdot 10^1$
$c = a \cdot b = 12.656 = 0.12656 \cdot 10^2$
Άρα ο αριθμός c δεν ανήκει στο $\mathcal{M}(10, 3, -1, 2)$. □

Η mantissa περιέχει t ψηφία της εκάστοτε βάσης β και όλοι οι αριθμοί που είναι μεγαλύτεροι πρέπει να αποκοπούν (shortened) σ' αυτό το δεδομένο μήκος με άμεση συνέπεια να περιορίζεται η ακρίβεια κάθε υπολογισμού. Επομένως έχουμε **t-ψηφίων αριθμητική κινητής υποδιαστολής**. Ένα ερώτημα που γεννιέται είναι ο τρόπος αποθήκευσης αυτών των αριθμών στη μνήμη του υπολογιστή.

Αποθήκευση floating point αριθμού

Στην αποθήκευση των floating point αριθμών τα bits της λέξης του υπολογιστή κατανέμονται για την αποθήκευση των προσήμων εκθέτη και χαρακτηριστικής, του εκθέτη και της mantissa. Εάν για την αποθήκευση των αριθμών χρησιμοποιηθούν ψηφία από δύο χαρακτηριστικές (δηλαδή δεσμευθούν δύο λέξεις) τότε μιλάμε για υπολογισμούς διπλής ακριβείας (**double precision**) σε αντίθεση με τους υπολογισμούς απλής ακριβείας (**single precision**) για τους οποίους χρησιμοποιείται μία μόνο λέξη υπολογιστή. Οι υπολογισμοί διπλής ακριβείας, όπως φαίνεται από την ονομασία τους, δίνουν διπλή ακρίβεια αλλά είναι κάπως αργότεροι στην εκτέλεση.

Παράσταση λέξης υπολογιστή σε IEEE αριθμητική

Οι αριθμοί κινητής υποδιαστολής αποτελούν τον πυρήνα κάθε αριθμητικού υπολογισμού, είναι λοιπόν απαραίτητο να δούμε πως τους αποθηκεύει και τους επεξεργάζεται ο υπολογιστής. Επειδή η αποθήκευση είναι στενά συνδεδεμένη με το hardware του υπολογιστή, το Institute of Electrical and Electronic Engineers (IEEE) καθιέρωσε το 1985 τις βασικές αρχές παράστασης και επεξεργασίας των αριθμών κινητής υποδιαστολής. Η IEEE δυαδική αριθμητική standard binary arithmetic, χρησιμοποιείται σήμερα από όλους τους κατασκευαστές υπολογιστών στο σχεδιασμό των μονάδων επεξεργασίας της αριθμητικής κινητής υποδιαστολής, εξασφαλίζοντας έτσι τη συμβατότητα μεταξύ όλων των προγραμμάτων, ανεξαρτήτως της μηχανής που χρησιμοποιείται για την εκτέλεση.

Η IEEE περιλαμβάνει δύο κατηγορίες αριθμών κινητής υποδιαστολής· απλής ακρίβειασ(single precision) με λέξεις των 32 bits και διπλής ακρίβειασ(double precision) αριθμούς, λέξεις 64 bits. Συνήθως $\beta = 2$.

IEEE Floating Point Standard 754-1985

Format απλής ακρίβειας

- πρόσημο mantissa 1 bit
- εκθέτης 8 bits
- mantissa (επονομάζεται significand) 23 bits

1 bit	8 bits	23 bits
πρόσημο	biased exponent[1]	mantissa

single format

Κατά συνέπεια ο υπολογιστής αυτός έχει το ακόλουθο εύρος παράστασης αριθμών:

$$x = \sigma * (.a_1 a_2 \ldots a_{23})_2 * 2^e$$

$$|e| \leq (11111111)_2 = 2^8 - 1 = 255$$

Το εύρος παράστασης είναι από 0 έως 255. Επειδή ο εκθέτης θα παίρνει και θετικές και αρνητικές τιμές, αντί να χρησιμοποιείται ένα ξεχωριστό bit για το πρόσημο του εκθέτη, το IEEE Floating Point Standard 754-1985 χρησιμοποιεί τη biased representation με τιμή για το bias το 127. Από το εύρος λοιπόν του εκθέτη αφαιρούμε το 127 και το τελικό εύρος είναι από -127 έως 128. Οι εκθέτες -127 (όλα 0) και +128 (όλα 1) δεσμεύονται για την αναπαράσταση ειδικών αριθμών.
Για να αυξηθεί περισσότερο η ακρίβεια στη mantissa, το IEEE 754 standard χρησιμοποιεί κανονικοποιημένη mantissa. Η πρώτη μονάδα που πάντα θα εμφανίζεται λόγω κανονικοποίησης, δε θα αποθηκεύεται και έτσι έχουμε ότι το πεδίο των 23-bits χρησιμοποιείται για την αποθήκευση σε mantissa 24-bit με τιμή ανάμεσα στο 0.5 και 1. Θεωρούμε δε ότι αυτή η μονάδα θα εμφανίζεται αριστερά της νοητής υποδιαστολής. Ετσι ένας κανονικοποιημένος floating point αριθμός στο IEEE Floating Point Standard 754-1985 θα παριστάνεται ως εξής:

$$x = (-1)^s * (1.a_1 a_2 \ldots a_{23})_2 * 2^{(exponent-127)}$$

όπου s είναι το sign bit με τιμή 0 για θετικό αριθμό και 1 για αρνητικό.

Έτσι λοιπόν ο μικρότερος θετικός παραστήσιμος αριθμός είναι: $(1.00...0)_2 * 2^{-126} = 1 * 2^{-126} \approx 1.1755 * 10^{-38}$
και ο μέγιστος θετικός παραστήσιμος αριθμός είναι: $(1.11...1)_2 * 2^{127} = 1 + (1 - 2^{-23}) * 2^{127} \approx 3.403 * 10^{38}$

ενώ για fixed point η ίδια μηχανή έχει μέγιστο παραστήσιμο: $2^{31} - 1 = 2147483647$.

[1] biased representation είναι μια σταθερή τιμή που ονομάζεται the bias και αφαιρείται από το πεδίο για να μας δώσει την αληθή τιμή τού εκθέτη

Παράδειγμα 1.1.4 *Να αποθηκευθεί ο αριθμός* 52.21875 *σε IEEE 754-32 bit floating point format.*

Απόδειξη: Από τα όσα έχουμε αναφέρει μέχρι τώρα μπορούμε εύκολα να δούμε ότι η αποθήκευση θα γίνει ως εξής:

Κατάρχήν ο πραγματικός αριθμός θα μετατραπεί στον ισοδύναμό του στο δυαδικό σύστημα.

$52.21875 = 110100.00111 \qquad = 1.1010000111 * 2^5$

Κανονικοποιημένη mantissa=.1010000111

Για το εκθέτη έχουμε $(exponent - 127) = 5$ και άρα ο εκθέτης θα είναι 132. Εχουμε ότι

$132 = 10000100$

Τελικά ακολουθώντας το IEEE format η αναπαράσταση των bits θα είναι ως εξής:

1 bit	8 bits	23 bits
0	10000100	1010000111...0

Παρατηρήστε ότι το πρώτο μηδέν αντστοιχεί στο πρόσημο, καθώς επίσης ότι δεν αποθηκεύεται η πρώτη μονάδα του χανονικοποιημένου αριθμού. Έτσι το πεδίο των 23 bits χρησιμοποιείται για την αποθήκευση μιας mantissa 24 bits. □