

1. Algoritmusok stabilitása, gépi számok

1.1. Feladat*

Tekintsük a következő határozott integrált:

$$T_n := \int_0^1 \frac{x^n}{x+10} dx \quad (n \in \mathbb{N}).$$

Az integrál közelítésére két algoritmust is adunk:

$$\begin{aligned} (1) \quad T_0 &:= \ln(1.1) & T_k &:= \frac{1}{k} - 10T_{k-1} & (k = 1, \dots, n), \\ (2) \quad T_{M+1} &:= 0 & T_k &:= \frac{1}{10} \left(\frac{1}{k} - T_{k+1} \right) & (k = M, \dots, n). \end{aligned}$$

A fenti rekurziókkal megkapható T_n közelítő értéke. Legyen az (1) algoritmus bemenete T_0 , a (2) algoritmusé T_{M+1} , továbbá mindkét algoritmus kimenete T_n . Bizonyítsuk, hogy az (1) algoritmus instabil, a (2) pedig stabil.

A T_k kifejezést átalakítva ellenőrizhető a rekurziók helyessége:

$$\begin{aligned} T_k &= \int_0^1 \frac{x^{k-1} \cdot x}{x+10} dx = \int_0^1 \frac{x^{k-1}(x+10-10)}{x+10} dx = \\ &= \int_0^1 x^{k-1} dx - 10 \int_0^1 \frac{x^{k-1}}{x+10} dx = \\ &= \left[\frac{x^k}{k} \right]_0^1 - 10T_{k-1} = \frac{1}{k} - 10T_{k-1}, \end{aligned}$$

melyből leolvasható az (1) algoritmus alakja. A fenti kifejezést átrendezve T_{k-1} -re k helyére $k+1$ -et írva kapjuk a (2) rekurziót.

Vizsgáljuk először az (1) algoritmust. Változtassuk meg a bemenetét egy kissé, legyen a megváltoztatott bemenet \tilde{T}_0 . A bemenet hibája ekkor

$$|T_0 - \tilde{T}_0|,$$

a kimenet hibája pedig

$$|T_n - \tilde{T}_n|.$$

Ha ez utóbbi mennyiség *nem* becsülhető meg felülről $|T_0 - \tilde{T}_0|$ konstansszorosával, akkor az azt jelenti, hogy az algoritmus instabil. Először vizsgáljuk meg, hogy a rekurzió alapján

mit mondhatunk a hiba terjedéséről. Tegyük fel, hogy a $|T_{k-1} - \tilde{T}_{k-1}|$ értéke ismert, ekkor

$$|T_k - \tilde{T}_k| = \left| \left(\frac{1}{k} - 10T_{k-1} \right) - \left(\frac{1}{k} - 10\tilde{T}_{k-1} \right) \right| = 10 \cdot |T_{k-1} - \tilde{T}_{k-1}|.$$

Ebből egyszerűen látható, hogy

$$|T_n - \tilde{T}_n| = 10^n \cdot |T_0 - \tilde{T}_0|,$$

vagyis az algoritmus nem stabil, ugyanis a 10^n nem becsülhető felülről konstanssal (a 10^n sorozat nem korlátos, ez egy plusz végtelenhez tartó divergens geometriai sorozat):

$$\forall K > 0 \quad \exists n \in \mathbb{N} : |T_n - \tilde{T}_n| > K \cdot |T_0 - \tilde{T}_0|.$$

Teljesen hasonló módon igazolhatjuk, hogy a (2) algoritmus stabil. Legyen a megváltoztatott bemenet most \tilde{T}_{M+1} , és tegyük fel, hogy $|T_{k+1} - \tilde{T}_{k+1}|$ értéke ismert. Ekkor

$$|T_k - \tilde{T}_k| = \left| \frac{1}{10} \left(\frac{1}{k} - T_{k+1} \right) - \frac{1}{10} \left(\frac{1}{k} - \tilde{T}_{k+1} \right) \right| = \frac{1}{10} \cdot |T_{k+1} - \tilde{T}_{k+1}|.$$

Innen már nyilvánvaló, hogy

$$|T_n - \tilde{T}_n| = \left(\frac{1}{10} \right)^{M+1-n} |T_{M+1} - \tilde{T}_{M+1}|.$$

Feltételezésünk szerint $M + 1 - n > 0$, ezért $\left(\frac{1}{10} \right)^{M+1-n} \leq \frac{1}{10}$, ez pedig az algoritmus stabilitását igazolja:

$$|T_n - \tilde{T}_n| \leq \frac{1}{10} \cdot |T_{M+1} - \tilde{T}_{M+1}|.$$

Összefoglalva tehát

$$\exists K > 0 \quad \forall n \in \mathbb{N} \quad (n \leq M) : |T_n - \tilde{T}_n| \leq K \cdot |T_{M+1} - \tilde{T}_{M+1}|.$$

3 M legkisebb pozitív eleme:

$$\varepsilon_0 = [100 \dots 0 | k^-] = \frac{1}{2} \cdot 2^{k^-} = 2^{k^- - 1}$$

4 M -ben az 1 után következő gépi szám és 1 különbsége:

$$\varepsilon_1 = [100 \dots 01 | 1] - [100 \dots 00 | 1] = 2^{-t} \cdot 2^1 = 2^{1-t}$$

5 M legnagyobb eleme:

$$\begin{aligned} M_\infty &= [111 \dots 11 | k^+] = 1.00 \dots 00 \cdot 2^{k^+} - 0.00 \dots 01 \cdot 2^{k^+} = \\ &= (1 - 2^{-t}) \cdot 2^{k^+} \end{aligned}$$

6 M elemeinek száma (számossága):

$$|M| = 2 \cdot 2^{t-1} \cdot (k^+ - k^- + 1) + 1$$

1.2. Feladat

Tekintsük az $M(6, -10, 10)$ gépi számhalmazt!

- (a) Számítsuk ki M nevezetes számait $(\varepsilon_0, M_\infty, \varepsilon_1, |M|)$!
- (b) Határozzuk meg $\text{fl}(137)$ értéket! Mennyi $\Delta_{\text{fl}(137)}$, a számábrázolás hibájából származó abszolút hibakorlát?

- (a) ε_0 a lehető legkisebb pozitív gépi szám M -ből. Kiszámításához a lehető legkisebb mantisszát, és a lehető legkisebb karakterisztikát kell választanunk. Mivel a mantissza első tagja mindig 1 (a normalizálás miatt), ezért a keresett mantissza az 100000. Hozzávéve a legkisebb megengedett -10 -es karakterisztikát kapjuk az

$$\varepsilon_0 = [100000 \mid -10] = \frac{1}{2} \cdot 2^{-10} = 2^{-11} = \frac{1}{2048}$$

számot. M_∞ a lehető legnagyobb gépi szám M -ből. Az előzőek alapján most a lehető legnagyobb mantisszát kell választanunk a lehető legnagyobb karakterisztikával. A keresett gépi szám így:

$$M_\infty = [111111 \mid 10] = \left(\frac{1}{2} + \frac{1}{4} + \frac{1}{8} + \frac{1}{16} + \frac{1}{32} + \frac{1}{64} \right) \cdot 2^{10} = 1008$$

Az ε_1 , úgynevezett gépi epszilon a számábrázolás relatív hibájával kapcsolatos mennyiség. Úgy tudjuk kiszámítani, hogy kivonjuk az 1-et az 1 rákövetkezőjéből. Mivel $1 = [100000 \mid 1]$, illetve a rákövetkező gépi szám $1 = [100001 \mid 1]$, ezért

$$\varepsilon_1 = [100001 \mid 1] - [100000 \mid 1] = \left(\frac{1}{2} + \frac{1}{64} \right) \cdot 2 - 1 = 1 + \frac{1}{32} - 1 = \frac{1}{32}.$$

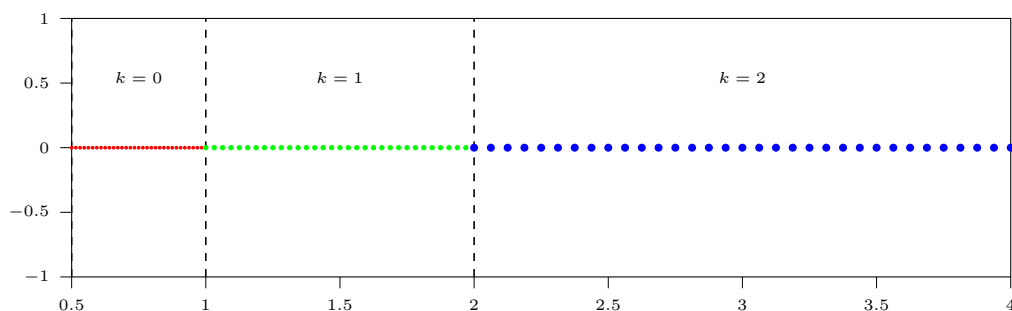
1.1. Megjegyzés

A ε_1 mennyiség éppen a $k = 1$ karakterisztikán ábrázolt szomszédos gépi számok távolsága. Eszerint tetszőleges k karakterisztikán a szomszédos gépi számok távolsága $2^{k-1} \cdot \varepsilon_1$.

Az M gépi számhalmaz számosságát a következő képpen számolhatjuk ki:

$$|M| = \underbrace{2}_{\text{előjel}} \cdot \underbrace{2^5}_{\text{mantissza}} \cdot \underbrace{(10 - (-10) + 1)}_{\text{karakterisztika}} + \underbrace{1}_{\{0\}} = 1345.$$

Itt 2 a lehetséges előjelek, 2^5 a mantissza lehetséges szabad tagjainak, 21 a lehetséges karakterisztikáknak a száma, 1 pedig a gépi számhalmazhoz hozzávett $\{0\}$ egyelemű halmaz számosságát jelöli. Az $M(6, -10, 10)$ gépi számhalmaz egy részlete megtekinthető a következő ábrán.



(b) A szokásos, tízes számrendszerbeli egész- vagy törtszámok ábrázolásához meg kell határoznunk a konvertálandó szám kettes számrendszerbeli alakját. A számok egészrészének és törtrészének átváltásakor két különböző algoritmust használunk. Az egészrész átváltásához a következő algoritmus használható:

- Induljunk ki x -ből. Ismételjük a következőt.
- Minden lépésben maradékosan osszuk el x -et 2-vel és írjuk fel a maradékot.
- Az eljárás leáll, ha $x = 0$.
- Ekkor fordított sorrendben összeolvasva a maradékokat éppen x bináris alakját kapjuk.

A következő táblázatban a függőleges vonaltól balra található a maradékos osztások eredményei, a jobb oldalon pedig a maradékok.

137		
68		1
34		0
17		0
8		1
4		0
2		0
1		0
0		1

A táblázat jobb oldalát letről felfelé olvasva kapjuk tehát a 137 szám kettes számrendszerbeli alakját:

$$137_{10} = 10001001_2.$$

A tízes számrendszerhez hasonlóan, kettes számrendszerben is az egyes számjegyek a rendszer alapszáma hatványainak szorzótényezőit jelentik. Azaz a legjobboldalibb 1-es számjegy a 2^0 szorzója, az azt követő 0 a 2^1 szorzója és így tovább. Ellenőrzésképpen visszaválthatjuk 10-es számrendszerbe a bináris számunkat:

$$10001001_2 = 2^0 + 2^3 + 2^7 = 1 + 8 + 128 = 137_{10}.$$

Ha nem okoz félreértést, akkor a továbbiakban nem tüntetjük fel alsó indexben, hogy 10-es, vagy 2-es számrendszerbeli számról beszélünk.

Az $\text{fl}(137)$ meghatározásához keresnünk kell két olyan gépi számot, melyek közrefogják a 137-et. Ezt könnyen megtehetjük a kettes számrendszerbeli alak alapján.

1.2. Megjegyzés

Tegyük fel, hogy x egész szám, melyre $\varepsilon_0 \leq |x| \leq M_\infty$. Ekkor

- $\text{fl}(x)$ mantisszája a bináris alak első t bitje (kerekítéssel),
- karakterisztikája pedig az x bináris jegyeinek száma.

A 137 bináris jegyeinek száma 8, ezért $\text{fl}(137)$ karakterisztikája 8, mantisszája pedig 100010, azaz

$$\text{fl}(137) = [100010 \mid 8]$$

Az fl függvény definíciója szerint a legközelebbi gépi számot rendeli a 137-hez. Honnan tudjuk, hogy a fenti gépi számnál nincs közelebbi a számhalmazban? A válasz az, hogy a bináris átváltás és a kerekítési szabály helyes alkalmazása miatt biztosak lehetünk benne, hogy a fenti gépi szám a legközelebbi. Tekintsük a következő magyarázatot. A 137 bináris alakja 8 számjegyű, viszont az $M(6, -10, 10)$ gépi számhalmaz számainak értékes jegyeinek maximális száma 6. Tehát a bináris alak első 6 értékes jegyét tarthatjuk meg, ezért kerekítenünk kell. A kerekítés irányát (akárcsak a tízes számrendszerben) egy t -jegyű szám esetén a $(t+1)$ -edik számjegy határozza meg. Ha a $(t+1)$ -edik számjegy 1, felfelé, ha 0, lefelé kerekítünk. Esetünkben $t+1 = 7$, a 7. bit pedig 0, ezért lefelé kerekítünk:

$$\underline{100010}\boxed{0}1 \approx \underline{10001000}.$$

Az aláhúzott bitsorozat a szám mantisszája. Győződjünk meg róla, hogy tényleg ez van a 137-hez a legközelebb. Számítsuk ki a gépi szám értékét!

$$[100010 \mid 8] = \left(\frac{1}{2} + \frac{1}{32}\right) \cdot 2^8 = 2^7 + 2^3 = 136 \leq 137.$$

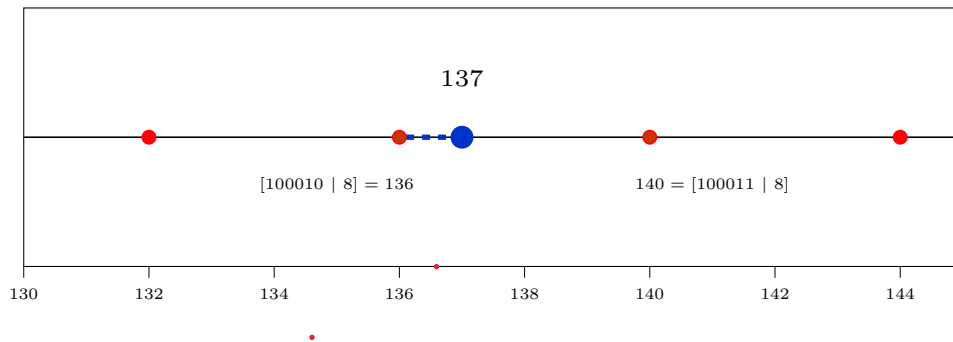
A fenti szám a lefele kerekítés miatt kisebb, mint a 137. Nézzük meg, hogy mi a következő, eggyel nagyobb gépi szám értéke:

$$[100011 \mid 8] = \left(\frac{1}{2} + \frac{1}{32} + \frac{1}{64}\right) \cdot 2^8 = 2^7 + 2^3 + 2^2 = 140 \geq 137$$

Megkerestük tehát azt a két egymást követő gépi számot, amelyek közrefogják az ábrázolandó számot:

$$[100010 \mid 8] = 136 \leq 137 \leq 140 = [100011 \mid 8].$$

Minthogy a 137 a bal oldaléhoz esik közelebb, $\text{fl}(137) = [100010 \mid 8]$.



A számábrázolásból eredő hibakorlát meghatározásához azt kell meggondolnunk, mekkora a hibát követünk el *legfeljebb*, amikor az fl függvényt alkalmazzuk, azaz mivel becsülhető felülről $|\text{fl}(137) - 137|$?

Általánosan, az fl függvény definíciója szerint x az $\text{fl}(x)$ és egy azzal szomszédos gépi szám közé esik. A legnagyobb hiba, amit elkövethetünk tehát, az $\text{fl}(x)$ és alsó/felső szomszédjának *távolságának a fele*. Vegyük észre, hogy ez csak a szám ábrázolásakor használt karakterisztikától függ! A ε_1 definíciója szerint éppen a $k = 1$ karakterisztikán ábrázolt szomszédos számok távolságát jelenti. Ezért tetszőleges k karakterisztika esetén ez a távolság a 2^{k-1} -szeresére nyúlik/zsugorodik, vagyis:

$$|\text{fl}(x) - x| \leq \frac{1}{2} \cdot \varepsilon_1 \cdot 2^{k-1} = 2^{-1} \cdot 2^{1-t} \cdot 2^{k-1} = \frac{1}{2} \cdot 2^{k-t} =: \Delta_{\text{fl}(x)}.$$

Esetünkben tehát a következő becslés adható:

$$|\text{fl}(137) - 137| \leq \frac{1}{2} \cdot 2^{8-6} = 2.$$

③ M legkisebb pozitív eleme:

$$\varepsilon_0 = [100 \dots 0 \mid k^-] = \frac{1}{2} \cdot 2^{k^-} = 2^{k^- - 1}$$

④ M -ben az 1 után következő gépi szám és 1 különbsége:

$$\varepsilon_1 = [100 \dots 01 \mid 1] - [100 \dots 00 \mid 1] = 2^{-t} \cdot 2^1 = 2^{1-t}$$

⑤ M legnagyobb eleme:

$$\begin{aligned} M_\infty &= [111 \dots 11 \mid k^+] = 1.00 \dots 00 \cdot 2^{k^+} - 0.00 \dots 01 \cdot 2^{k^+} = \\ &= (1 - 2^{-t}) \cdot 2^{k^+} \end{aligned}$$

⑥ M elemeinek száma (számossága):

$$|M| = 2 \cdot 2^{t-1} \cdot (k^+ - k^- + 1) + 1$$

1.3. Feladat

Tekintsük az $M(8, -5, 5)$ gépi számhalmazt!

- (a) Számítsuk ki M nevezetes számait $(\varepsilon_0, M_\infty, \varepsilon_1, |M|)$!
- (b) Próbáljuk meghatározni az $\text{fl}(1/6)$ gépi számot a 0.17 és 0.167 decimális közelítés alapján! Határozzuk meg a gépi számot az $1/6$ pontos értékének használatával is!
- (c) Számítsuk ki $\text{fl}(3.14)$ értékét is, majd végezzük el az $\text{fl}(3.14) + \text{fl}(1/6)$ gépi összeadást! Adjunk abszolút hibakorlátot az eredmény *számábrázolásból* eredő hibájára!

- (a) Számítsuk ki a nevezetes számokat az előbbiek szerint!

$$\begin{aligned}\varepsilon_0 &= [10000000 \mid -5] = \frac{1}{2} \cdot 2^{-5} = \frac{1}{64} \\ M_\infty &= [11111111 \mid 5] = (1 - 2^{-8}) \cdot 2^5 = \frac{255}{8} \\ \varepsilon_1 &= [10000001 \mid 1] - [10000000 \mid 1] = 2^{1-8} = \frac{1}{128} \\ |M| &= 2^8 \cdot (5 - (-5) + 1) + 1 = 2817\end{aligned}$$

- (b) Ahogy azt korábban említettük, a számok törtrészének és egészrészének két külön algoritmussal keressük meg a kettes számrendszerbeli alakját. A törtrész esetén a következőképpen járunk el. Tegyük fel, hogy $\varepsilon_0 < x < 1$ tetszőleges (tizedes) törtszám.

- Induljunk ki x -ből. Ismételjük a következőt.
- Minden lépésben szorozzuk meg x -et 2-vel. Ha az így kapott szám 1-nél nagyobb (túlcsordul), vonjunk le x -ből 1-et, majd jegyezzünk fel egy 1-et. Ha x nem csordul túl, jegyezzünk fel egy 0-t.
- Az eljárást véges számú lépés után leállítjuk.
- Sorrendben összeolvasva a túlcsordulásokat éppen x bináris alakját kapjuk.

A fenti algoritmussal egy szám kettesdespont utáni számjegyeit határozhatjuk meg. Ha x nem írható fel 2-hatványok összegeként, akkor a bináris alakja nem véges, ezért a fenti algoritmust manuálisan állítjuk le, amikor elegendő értékes bitet határoztunk meg az ábrázolandó számból. Az első 1-est követő $t + 1$ bitre van szükségünk, a t

bit alkotja a mantisszát, a $(t + 1)$ -edik bit pedig a kerekítés irányát határozza meg. A (részleges) bináris alak ismeretében most is egyszerűen meghatározható $\text{fl}(x)$.

1.3. Megjegyzés

Tegyük fel, hogy x olyan szám, melyre $\varepsilon_0 \leq |x| < 1$. Ekkor

- $\text{fl}(x)$ mantisszája a bináris alak első t értékes bitje (kerekítéssel),
- karakterisztikája pedig a kettedes pont és az első 1-es számjegy közötti nullák számának -1 -szerese.

Határozzuk meg először az $\frac{1}{6}$ két tizedesjegy pontosságú decimális közelítésének megfelelő gépi számot!

17		$0.17 =$	$0.00\underline{10101110}\boxed{0} \dots \approx$
0 34		\approx	$0.00\underline{10101110}0 =$
0 68		$=$	$[10101110 \mid -2]$
1 36		\Downarrow	
0 72		$\text{fl}(0.17) =$	$[10101110 \mid -2]$
1 44			
0 88			
1 76			
1 52			
1 04			
0 08			
0 16			
\vdots			

Fontos megjegyezni, hogy a fenti szám a 0.17-hez legközelebbi gépi szám, de nem biztos, hogy az $\frac{1}{6}$ -hoz is ez van legközelebb!

Lássuk most a három decimális jeggyel való közelítéshez tartozó gépi számot!

$$\begin{array}{r|l}
& 167 \\
0 & 334 \\
0 & 668 \\
1 & 336 \\
0 & 672 \\
1 & 344 \\
0 & 688 \\
1 & 376 \\
0 & 752 \\
1 & 504 \\
1 & 008 \\
0 & 016 \\
\vdots & \vdots
\end{array}
\quad
\begin{array}{l}
0.167 = 0.00\underline{10101011}\boxed{0}\dots \approx \\
\approx 0.00\underline{10101011}0 = \\
= [10101011 \mid -2] \\
\downarrow \\
\text{fl}(0.167) = [10101011 \mid -2]
\end{array}$$

Vegyük észre, hogy $\text{fl}(0.167)$ és $\text{fl}(0.17)$ még csak nem is szomszédos gépi számok! Keressük most meg, hogy melyik az $\frac{1}{6}$ -hoz legközelebbi gépi szám, induljunk ki az $\text{fl}(0.167)$ számból:

$$\begin{aligned}
\text{fl}(0.167) &= [10101011 \mid -2] = \left(\frac{1}{2} + \frac{1}{8} + \frac{1}{32} + \frac{1}{128} + \frac{1}{256} \right) \cdot 2^{-2} = \\
&= \frac{128 + 32 + 8 + 2 + 1}{1024} = \frac{171}{1024} = \frac{513}{3072} > \frac{512}{3072} = \frac{1}{6}.
\end{aligned}$$

Ahhoz, hogy eldöntsük az $\text{fl}(1/6)$ értékét, meg kell találnunk azt a két szomszédos gépi számot, amely közrefogja az $\frac{1}{6}$ -ot. Mivel a fentiek szerint $\text{fl}(0.167) > \frac{1}{6}$, vizsgáljuk meg $\text{fl}(0.167)$ alsó szomszédját. Az $\text{fl}(0.167)$ számot az utolsó bitnek megfelelő értékkel kell csökkentenünk:

$$[10101010 \mid -2] = \frac{171}{1024} - \frac{1}{1024} = \frac{170}{1024}.$$

Ez a szám már valóban kisebb, mint $\frac{1}{6}$, ugyanis

$$[10101010 \mid -2] = \frac{170}{1024} = \frac{510}{3072} < \frac{512}{3072} = \frac{1}{6}.$$

Összefoglalva, találtunk két *szomszédos* gépi számot, melyek közrefogják az $\frac{1}{6}$ -ot:

$$[10101010 \mid -2] \leq \frac{1}{6} \leq [10101011 \mid -2].$$

Az előzőek alapján az is világos, hogy a nagyobbik szám van közelebb $\frac{1}{6}$ -hoz, hiszen

$$\left| \frac{1}{6} - [10101010 \mid -2] \right| = \frac{2}{3072},$$

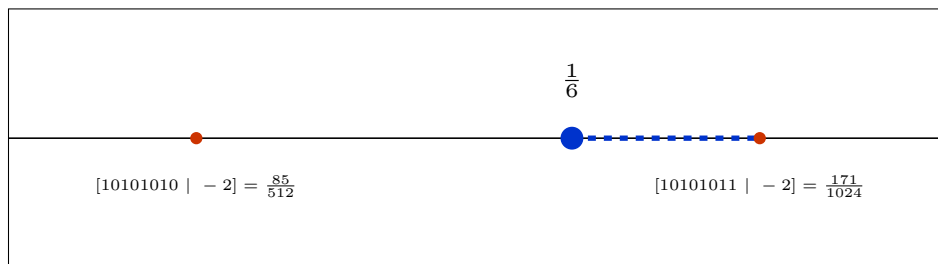
$$\left| \frac{1}{6} - [10101011 \mid -2] \right| = \frac{1}{3072}.$$

Ezért

$$\mathfrak{fl}(1/6) = [10101011 \mid -2].$$

Végül megjegyezzük, hogy az algoritmusunk akkor is működik, ha közösleges tört alakokkal dolgozunk, így az $\frac{1}{6}$ a szokásos kerekítési szabályok alkalmazásával egyetlen lépésben konvertálható a gépi számhalmazba.

	$1/6$	$1/6 = 0.00\underline{10101010}\boxed{1}\dots \approx$
0	$1/3$	$\approx 0.00\underline{10101011}0 =$
0	$2/3$	$= [10101011 \mid -2]$
1	$1/3$	\Downarrow
0	$2/3$	$\mathfrak{fl}(1/6) = [10101011 \mid -2]$
1	$1/3$	
0	$2/3$	
1	$1/3$	
0	$2/3$	
1	$1/3$	
0	$2/3$	
1	$1/3$	
\vdots	\vdots	



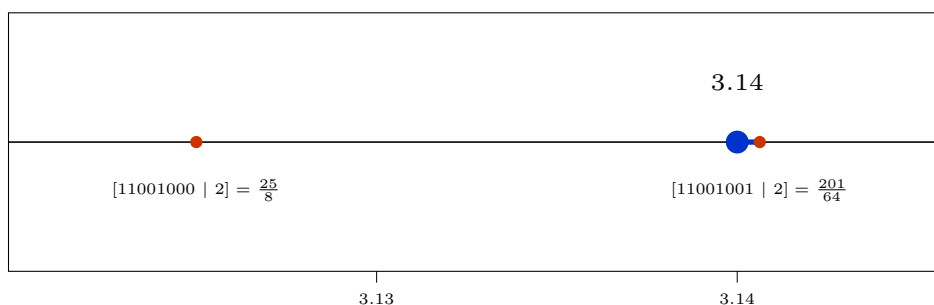
- (c) Ha az ábrázolandó szám egészrészt és törtrészt is tartalmaz, akkor a két részt külön-külön konvertáljuk bináris számmá. Ugyanúgy, ahogy a decimális számok esetében is, a kettedespont az egészrész és a törtrész közé kerül.

3		14		3.14 = 11.001000 $\boxed{1}$... \approx
1	1	0	28	\approx 11.0010010 =
0	1	0	56	= [11001001 2]
		1	12	\Downarrow
		0	24	$\text{fl}(3.14) =$ [11001001 2]
		0	48	
		0	96	
		1	92	
		\vdots	\vdots	

1.4. Megjegyzés

Tegyük fel, hogy $x \in \mathbb{R}$ olyan szám, melyre $\varepsilon_0 \leq |x| \leq M_\infty$. Ekkor

- $\text{fl}(x)$ mantisszája a bináris alak első t értékes bitje (kerekítéssel),
- karakterisztikája pedig a kettedespont és az első értékes bit „előjeles távolsága” (azaz hányszor kell jobbra vagy balra mozgatni a kettedespontot, hogy az közvetlenül az első értékes bit elé kerüljön).



Végezzük el az $\text{fl}(3.14) + \text{fl}(1/6)$ gépi összeadást! Az összeget csak úgy tudjuk kiszámítani, ha a számok azonos karakterisztikájúak, azonban ez most nem igaz. Ilyen esetben mindig a nagyobb karakterisztika lesz a közös, hiszen így minimalizálhatjuk a művelet során elkövetett hibát. A legegyszerűbb az, ha először bináris alakban adjuk össze a számokat, ezután a kapott bináris számot ábrázoljuk gépi számként.

Mivel

$$\begin{aligned}\text{fl}(3.14) &= 11.001001, \\ \text{fl}(1/6) &= 0.0010101011,\end{aligned}$$

így

$$\begin{array}{r} 1 \ 1. \ 0 \ 0 \ 1 \ 0 \ 0 \ 1 \\ + \quad 0. \ 0 \ 0 \ 1 \ 0 \ 1 \ 0 \ 1 \ 0 \ 1 \ 1 \\ \hline 1 \ 1. \ 0 \ 1 \ 0 \ 0 \ 1 \ 1 \ 1 \ 0 \ 1 \ 1. \end{array}$$

Az eredményt gépi számmá konvertálva a következőt kapjuk:

$$11.010011\boxed{1}011 \approx 11.010100 = [11010100 \mid 2].$$

1.5. Megjegyzés

Megjegyezzük, hogy a bináris összeadás ugyanolyan elven végezhető, mint a decimális összeadás. Számjegyenként adunk össze (jobbról balra), ha a művelet eredménye nem ábrázolható egy számjeggyel, akkor maradékot képzünk. Azaz pl. "1 + 1 egyenlő 0, maradt az 1".

Látható, hogy az $\text{fl}(1/6)$ utolsó 4 jegyéből elegendő lett egyet meghagyni a kerekítés irányának helyes meghatározásához. Éppen ezért eljárhatunk a következőképpen is. Írjuk fel az $\text{fl}(1/6)$ -nak megfelelő számot a nagyobbik $k = 2$ karakterisztikával:

$$\begin{aligned}\text{fl}(1/6) &= [10101011 \mid -2] = \\ &= (0.\underline{10101011}_2) \cdot 2^{-2} = \\ &= (0.\underline{01010101}\boxed{1}_2) \cdot 2^{-1} = \\ &= (0.\underline{00101010}\boxed{1}_2) \cdot 2^0 = \\ &= (0.\underline{00010101}\boxed{0}11_2) \cdot 2^1 = \\ &= (0.\underline{00001010}\boxed{1}011_2) \cdot 2^2 \approx \\ &= (0.\underline{00001011} \ 0 \ 000_2) \cdot 2^2 = [00001011 \mid 2].\end{aligned}$$

Ha tehát a karakterisztikát eggyel növeljük, miközben a mantissza jegyeit eggyel jobbra toljuk (shifteljük), akkor a szám értéke (elvileg) változatlan (gyakorlatilag a kerekítés miatt megváltozhat). Hasonló a helyzet, ha a karakterisztikát eggyel csökkentjük miközben a mantissza jegyeit eggyel balra toljuk. **A 2-vel való szorzás/osztás (karakterisztika növelése/csökkentése) tehát kiváltható a bitek eltolásával (bitshift).** Az így kapott szám ugyan nem normalizált, de műveletet tudunk vele végezni. A művelet elvégzése után kapott számot később normalizáljuk.

Tekintsük meg így is a gépi összeadást:

$$\begin{array}{r} [11001001 \mid 2] \\ + [00001011 \mid 2] \\ \hline [11010100 \mid 2]. \end{array}$$

Látható, hogy ezzel a módszerrel is ugyanazt kaptuk, mint korábban. Becsüljük most az operandusok és az eredmény számábrázolásából származó hibáját:

$$\begin{aligned} \Delta_{\text{fl}(1/6)} &= \frac{1}{2} \cdot 2^{-2-8} = 2^{-11} = \frac{1}{2048}, \\ \Delta_{\text{fl}(3.14)} &= \Delta_{\text{fl}(3.14)+\text{fl}(1/6)} = \frac{1}{2} \cdot 2^{2-8} = 2^{-7} = \frac{1}{128}. \end{aligned}$$

1.4. Feladat*

Tekintsük az $M(6, -4, 4)$ gépi számhalmazt, legyen továbbá $a = \text{fl}(8)$, valamint $b = \text{fl}(1/16)$.

- (a) Mennyi az $(a + b) + b$ értéke?
- (b) Mennyi az $a + (b + b)$ művelet eredménye?
- (c) Mennyi az $a + a$ művelet eredménye?

- (a) Mivel a és b is 2-hatvány, könnyen ábrázolhatóak a gépi számhalmazban:

$$a = \text{fl}(8) = 2^3 = \frac{1}{2} \cdot 2^4 = [100000 \mid 4],$$

$$b = \text{fl}(1/16) = 2^{-4} = \frac{1}{2} \cdot 2^{-3} = [100000 \mid -3].$$

A két számot a közös, nagyobb karakterisztikára kell hoznunk, hogy összeadhatók legyenek, ezért hozzuk b -t a 4-es karakterisztikára:

$$\begin{aligned} b &= [100000 \mid -3] = (0.1_2) \cdot 2^{-3} = \\ &= (0.01) \cdot 2^{-2} = \dots = \\ &= (0.\underline{000000} \boxed{0} 1_2) \cdot 2^4 \approx [000000 \mid 4]. \end{aligned}$$

Tehát

$$\begin{array}{r} [100000 \mid 4] \\ + [000000 \mid 4] \\ \hline [100000 \mid 4]. \end{array}$$

Vagyis

$$(a + b) = a \implies (a + b) + b = a + b = a = [100000 \mid 4].$$

- (b) A b szám önmagával könnyen összeadható, hiszen az operandusok (b és b) karakterisztikája megegyezik:

$$\begin{array}{r} [100000 \mid -3] \\ + [100000 \mid -3] \\ \hline (\underline{1.000000})_2 \cdot 2^{-3} = [100000 \mid -2]. \end{array}$$

A fenti példában a mantisszák legelső jegyének összeadásakor maradék keletkezett, melyet eggyel nagyobb helyiértékre kellett átvinnünk (túlcsordulás). Ezért

az eredmény már nem ábrázolható az eredeti karakterisztikán. Ilyen esetben az eredmény első t bitjét tartjuk meg, a karakterisztikát pedig eggyel növeljük. Az előzőek alapján látható, hogy a gépi összeadáshoz az (akár különböző karakterisztikájú) operandusok mantisszájának összesen $2t$ bitjén kívül 2 további bit szükséges (1 a kerekítéshez, 1 a túlsordulás jelzésére).

Megjegyezzük, hogy mivel $b + b = 2b$, az eredményt megkaphattuk volna egyszerűen úgy is, ha b karakterisztikáját eggyel növeljük:

$$b + b = 2b = [100000 | -3 + 1] = [100000 | -2].$$

Adjuk most össze a $b + b$ összeget a -val. Először is ábrázoljuk $(b + b)$ -t a 4-es karakterisztikán:

$$\begin{aligned} b + b &= [100000 | -2] = (0.1_2) \cdot 2^{-2} = \\ &= (0.\underline{000000}\overline{1}_2) \cdot 2^4 \approx [000001 | 4]. \end{aligned}$$

Így $a + (b + b)$ -re:

$$\begin{array}{r} [100000 | 4] \\ + [000001 | 4] \\ \hline [100001 | 4]. \end{array}$$

Ez az előbbi eredményünkkel együtt azt jelenti, hogy a gépi összeadás nem asszociatív, mivel

$$a + (b + b) = [100001 | 4] \neq [100000 | 4] = (a + b) + b.$$

Ez nem meglepő, azok után, hogy a nullelem sem egyértelmű. Ezt jelenti ugyanis korábbi eredményünk:

$$a + b = a \not\Rightarrow b = 0.$$

A gépi összeadás viszont kommutatív, vagyis

$$a + b = b + a$$

mindig teljesül.

(c) Az $a + a$ értéke nem ábrázolható a számhalmazban, mivel

$$a + a = 2a = 2 \cdot [100000 | 4] = [100000 | 5] \notin M,$$

és $5 > k^+ = 4$. Ez a már korábban megismert túlsordulás (overflow) jelensége, a művelet eredménye ekkor nem ábrázolható. Egy szabványos (IEEE 754) lebegőpontos aritmetikát megvalósító rendszerben egy ehhez hasonló művelet eredménye egy, a $+\infty$ -nek megfelelő extrémális érték. Az így kapott végtelennel akár további is műveleteket végezhetünk (ahogy azt analízisből tanultuk), pl.

$$(+\infty) + 3 = +\infty, \quad (+\infty) \cdot (-1) = -\infty.$$

(d) A $b \cdot b$ művelet eredménye:

$$b \cdot b = (2^{-4})^2 = 2^{-8} = \frac{1}{2} \cdot 2^{-7} = [100000| - 7] \approx 0,$$

mivel $-7 < k^- = -4$. Ez a jelenség pedig az alulcsordulás (underflow), a számot ekkor nullára kerekítjük. Egy szabványos (IEEE 754) lebegőpontos aritmetikát megvalósító rendszer ugyan tartalmaz úgynevezett denormalizált számokat (ezek esetében a mantissza első bitje 0), melyek használatával a 0-hoz igen közeli számok is ábrázolhatók, az alulcsordulás jelensége viszont ezzel a módosítással sem kerülhető meg. Az alulcsordult számot nullára kerekítjük, annak viszont előjelét megtartjuk, így egy szabványos gépi számhalmaznak eleme a $+0$ és a -0 is. Ezek segítségével szintén analízisből ismerős műveleteket vezethetünk be, pl:

$$1/(+0) := +\infty, \quad 1/(-0) := -\infty.$$

Az előzőekben említett $\pm\infty$ és ± 0 szimbólumokon kívül a szokásos gépi számhalmazokban még egy extrémis szimbólum használatos, a "nem szám" (*not a number* = NaN). Ezt olyan műveletek esetében használjuk, melyhez nem tudunk értéket rendelni:

$$(+\infty) + (-\infty) := \text{NaN}, \quad 0 \cdot (+\infty) := \text{NaN}.$$