Dokładny jak komputer

Czy aby na pewno?

Piotr Fulmański

Państwowa Wyższa Szkoła Zawodowa w Płocku
Wydział Nauk Ekonomicznych i Informatyki
piotr@fulmanski.pl
http://fulmanski.pl/zajecia/prezentacje/pwsz_dzien_otwarty_2017/dzien_otwarty_2017_fulmanp.pdf

4 kwietnia 2017

I am a HAL Nine Thousand computer [. . .]
The quick brown fox jumps over the lazy dog [. . .]
Dave – are you still there?
Did you know that the square root of 10 is 3.162277660168379?
Log 10 to the base e is 0.434294481903252. . . correction, that is log e to the base 10 [. . .]
2 times 2 is. . . 2 times 2 is. . . approximately 4.10101010. . . I seem to be having difficulty

HAL, 2001: A Space Odyssey

Spis treści

Problem z dodawaniem

```
1  #include <stdio.h>
2
3  int main(){
4      printf("%.60f\n",0.1);
5      printf("%.60f\n",0.1+0.1);
6      printf("%.60f\n",0.1+0.1+0.1);
7      printf("%.60f\n",0.3);
8
9      if(0.1+0.1+0.1 == 0.3)
10         printf("Rowne\n");
11     else
12         printf("Rozne\n");
13
14     return 0;
15 }
```

```
fulmanp@fulmanp-fusion-lubuntu:/media/fulmanp/dzien_otwarty$ ./a.out
0.1000000000000000055511151231257827021181583404541015625000000
0.2000000000000000111022302462515654042363166809082031250000000
0.3000000000000000444089209850062616169452667236328125000000000
0.2999999999999999888977697537484345957636833190917968750000000
Różne
```

Problem z dodawaniem

```
1  #include <stdio.h>
2
3  int main(){
4      printf("%.60f\n",0.1);
5      printf("%.60f\n",0.1+0.1);
6      printf("%.60f\n",0.1+0.1+0.1);
7      printf("%.60f\n",0.3);
8
9      if(0.1+0.1+0.1 == 0.3)
10         printf("Rowne\n");
11     else
12         printf("Rozne\n");
13
14     return 0;
15 }
```

```
fulmanp@fulmanp-fusion-lubuntu:/media/fulmanp/dzien_otwarty$ ./a.out
0.1000000000000000055511151231257827021181583404541015625000000
0.2000000000000000111022302462515654042363166809082031250000000
0.3000000000000000444089209850062616169452667236328125000000000
0.2999999999999999888977697537484345957636833190917968750000000
Różne
```

Problem z dodawaniem

Liczby podawane przez użytkownika ulegają „zniekształceniu”

Zamiast

0.1

mamy

0.1000000000000000055511151231257827021181583404541015625000000

Zamiast

0.3

mamy

0.2999999999999999888977697537484345957636833190917968750000000

Problem z dodawaniem

Komputer źle dodaje

Przyjmując nawet, że zamiast

```
0.1
```

mamy

```
0.1000000000000000055511151231257827021181583404541015625000000
```

to wówczas

```
0.1 + 0.1
```

powinno być równe

```
0.1000000000000000055511151231257827021181583404541015625000000
0.1000000000000000055511151231257827021181583404541015625000000 +
==================================================================
0.2000000000000000111022302462515654042363166809082031250000000
```

I tak jest!

Problem z dodawaniem

Komputer źle dodaje

Przyjmując nawet, że zamiast

```
0.1
```

mamy

```
0.1000000000000000055511151231257827021181583404541015625000000
```

to wówczas

```
0.1 + 0.1
```

powinno być równe

```
0.1000000000000000055511151231257827021181583404541015625000000
0.1000000000000000055511151231257827021181583404541015625000000 +
==================================================================
0.2000000000000000111022302462515654042363166809082031250000000
```

I tak jest!

Problem z dodawaniem

Liczby podawane przez użytkownika ulegają „zniekształceniu”

Pytanie tylko, dlaczego w takim razie

```
0.1 + 0.1 + 0.1
```

jest równe

```
0.3000000000000000444089209850062616169452667236328125000000000
```

zamiast

```
0.2000000000000000111022302462515654042363166809082031250000000
0.1000000000000000055511151231257827021181583404541015625000000 +
=================================================================
0.3000000000000000166533453693773481063544750213623046875000000
```

```
człowiek
|
0.3000000000000000166533453693773481063544750213623046875000000
0.3000000000000000444089209850062616169452667236328125000000000
|
maszyna
```

Problem z przemiennością

Dodawania to jedno z podstawowych działań arytmetycznych. W dodawaniu możemy stosować prawo **przemienności dodawania**. Pozwala ono – **teoretycznie** – na zmienianie kolejności w jakiej składniki są dodawane bez wpływu na wynik końcowy. Działanie

$$9 + 5 = 14$$

jest równoważne

$$5 + 9 = 14$$

czyli:

$$9 + 5 = 5 + 9$$

Niestety komputer zdaje się o tym nie wiedzieć...

Problem z przemiennością

```
1  #include <stdio.h>
   #include <math.h>
3
   int main(){
5      int i;
       int n = 100;
7      double suma1, suma2;

9      suma1 = 0;
       for(i=1; i<=n; ++i){
11         suma1 += pow(i,-5.0);
       }
13
       suma2 = 0;
15     for(i=n; i>=1; --i){
           suma2 += pow(i,-5.0);
17     }

19     printf("suma1=%.65f\nsuma2=%.65f\n", suma1, suma2);
   }
```

Problem z przemiennością

```
fulmanp@fulmanp-fusion-lubuntu:/media/fulmanp/dzien_otwarty$ ./a.out
suma1=1.03692775269295545115255663404241204261779785156250000000000000000
suma2=1.03692775269295323070650738372933119535446166992187500000000000000
```

Problem z przemiennością

$$\sum_{i=1}^{n} \frac{1}{i^5}$$

```
suma1=1.036927752692955451152556634042412042617797851562500000000
```

$$\sum_{i=n}^{1} \frac{1}{i^5}$$

```
suma2=1.036927752692953230706507383729331195354461669921875000000
```

Reprezentacja informacji

Informacja z punktu widzenia komputera

- **Z punktu widzenia komputera KAŻDA informacja to ciąg zer i jedynek.**
- Ten sam ciąg zer i jedynek raz może być zdjęciem naszego przyjaciela innym razem naszą ulubioną MP3 a jeszcze innym razem listem do cioci.
- To my, czyli użytkownik, mówimy jak interpretować dany ciąg zer i jedynek.
- Od sposobu interpretacji zależy co tak naprawdę odczytamy.
- To nie plik graficzny informuje nas o tym, że jest plikiem graficznym, ale to my plik interpretujemy jak gdyby był plikiem graficznym.

Reprezentacja informacji

Informacja z punktu widzenia komputera

- **Z punktu widzenia komputera KAŻDA informacja to ciąg zer i jedynek.**
- **Ten sam ciąg zer i jedynek raz może być zdjęciem naszego przyjaciela innym razem naszą ulubioną MP3 a jeszcze innym razem listem do cioci.**
- **To my, czyli użytkownik, mówimy jak interpretować dany ciąg zer i jedynek.**
- **Od sposobu interpretacji zależy co tak naprawdę odczytamy.**
- **To nie plik graficzny informuje nas o tym, że jest plikiem graficznym, ale to my plik interpretujemy jak gdyby był plikiem graficznym.**

Reprezentacja informacji

Informacja z punktu widzenia komputera

- **Z punktu widzenia komputera KAŻDA informacja to ciąg zer i jedynek.**
- **Ten sam ciąg zer i jedynek raz może być zdjęciem naszego przyjaciela innym razem naszą ulubioną MP3 a jeszcze innym razem listem do cioci.**
- **To my, czyli użytkownik, mówimy jak interpretować dany ciąg zer i jedynek.**
- **Od sposobu interpretacji zależy co tak naprawdę odczytamy.**
- **To nie plik graficzny informuje nas o tym, że jest plikiem graficznym, ale to my plik interpretujemy jak gdyby był plikiem graficznym.**

Reprezentacja informacji

Informacja z punktu widzenia komputera

- **Z punktu widzenia komputera KAŻDA informacja to ciąg zer i jedynek.**
- **Ten sam ciąg zer i jedynek raz może być zdjęciem naszego przyjaciela innym razem naszą ulubioną MP3 a jeszcze innym razem listem do cioci.**
- **To my, czyli użytkownik, mówimy jak interpretować dany ciąg zer i jedynek.**
- **Od sposobu interpretacji zależy co tak naprawdę odczytamy.**
- **To nie plik graficzny informuje nas o tym, że jest plikiem graficznym, ale to my plik interpretujemy jak gdyby był plikiem graficznym.**

Reprezentacja informacji

Informacja z punktu widzenia komputera

- **Z punktu widzenia komputera KAŻDA informacja to ciąg zer i jedynek.**
- **Ten sam ciąg zer i jedynek raz może być zdjęciem naszego przyjaciela innym razem naszą ulubioną MP3 a jeszcze innym razem listem do cioci.**
- **To my, czyli użytkownik, mówimy jak interpretować dany ciąg zer i jedynek.**
- **Od sposobu interpretacji zależy co tak naprawdę odczytamy.**
- **To nie plik graficzny informuje nas o tym, że jest plikiem graficznym, ale to my plik interpretujemy jak gdyby był plikiem graficznym.**

Liczby rzeczywiste w systemie dziesiętnym i binarnym

Mówiąc *liczba rzeczywista* mamy na myśli liczbę złożoną z części całkowitej i ułamkowej.

Pytanie

W jaki sposób zapisywać liczby rzeczywiste na ciągach znaków o określonej długości?

Liczby rzeczywiste w systemie dziesiętnym i binarnym

Mówiąc *liczba rzeczywista* mamy na myśli liczbę złożoną z części całkowitej i ułamkowej.

Pytanie

W jaki sposób zapisywać liczby rzeczywiste na ciągach znaków o określonej długości?

Liczby rzeczywiste w komputerze

Odpowiedź nie jest prosta

Opowieść od dwóch naukowcach: p. J. Pantofelek i p. Z. Galaktyce.

Notacja naukowa (dziesiętna)

Notacja naukowa

Zapis liczby składa się z następujących elementów:

- znak liczby;
- mantysa (liczba ułamkowa) znormalizowana, to znaczy mieszcząca się w przedziale [1,10);
- mała lub wielka litera E;
- wykładnik (liczba całkowita) nazywany też cechą.

Wartość liczby rzeczywistej, zapisanej w notacji naukowej można odczytać, stosując się do poniższego wzoru

$$x = M \cdot 10^E$$

gdzie:

- M (ang. *mantissa*) – mantysa
- E (ang. *exponent*) – wykładnik (cecha)

Notacja naukowa (dziesiętna)

Notacja naukowa

Zwarty i oszczędny sposób zapisu liczb dużych i małych

$$10000000000 = 1 \cdot 10^{+10} = 1.0E + 10$$

$$0.0000000001 = 1 \cdot 10^{-10} = 1.0E - 10$$

Notacja naukowa (dwójkowa)

Notacja naukowa

Zapis liczby składa się z następujących elementów:

- znak liczby;
- mantysa (liczba ułamkowa) znormalizowana, to znaczy mieszcząca się w przedziale [1,2);
- mała lub wielka litera E;
- wykładnik (liczba całkowita) nazywany też cechą.

Wartość liczby rzeczywistej, zapisanej w notacji naukowej można odczytać, stosując się do poniższego wzoru

$$x = M \cdot 2^E$$

gdzie:

- M (ang. *mantissa*) – mantysa
- E (ang. *exponent*) – wykładnik (cecha)

Notacja naukowa (dwójkowa)

Notacja naukowa

Zwarty i oszczędny sposób zapisu liczb dużych i małych

$$1024 = 1 \cdot 2^{+10} = 1.0_2 E + 10$$

$$0.0009765625 = 1 \cdot 2^{-10} = 1.0_2 E - 10$$

Liczby zmiennoprzecinkowe

Przyjmując notację naukową za punkt wyjścia, próbujemy określić jak ciąg zer i jedynek (skończonej i zawsze takiej samej długości) może opisywać liczbę rzeczywistą. Przyjmujemy następującą konwencję

```
znak liczby
|
Z M M ... M C C ... C
  |       | |       |
  | cyfry | | cyfry |
  |mantysy| | cechy |
```

A gdzie znak cechy?

```
+11 = +3 = 6-3
+10 = +2 = 5-3
+01 = +1 = 4-3
+00 = +0 = 3-3
-00 = -0 = 3-3
-01 = -1 = 2-3
-10 = -2 = 1-3
-11 = -3 = 0-3
```

Przyjmijmy następujące znaczenie bitów: pierwszy bit od lewej oznacza znak liczby, kolejne 3 mantysę zaś ostatnie 4 bity będą cechą. Jako wartość stałej K_C (odejmowanej od cechy) przyjmijmy 7. Tak więc liczby dające się zapisać w tym formacie są postaci

$$z_m M \cdot 2^{C-K_C},$$

gdzie z_m to znak mantysy, M – mantysa, C – cecha.

Najmniejszą dodatnią możliwą do reprezentowania liczbą jest 0.0078125 zapisane jako 0000000 ($1.000000 \cdot 2^{-7}$), zaś największa dodatnia to 480.0 zapisane jako 1111111 ($1.875000 \cdot 2^{8}$); łącznie mamy oczywiście 128 różnych liczb dodatnich.

Opis testu

Biorąc wszystkie możliwe kombinacje dwóch spośród otrzymanych 128 liczb, sprawdzamy czy ich suma jest jedną z tych 128 liczb. Mówiąc inaczej, sprawdzamy czy suma dwóch spośród owych 128 liczb daje się wyrazić w tym formacie.

Wyniki testu

Łacznie mamy $128 \cdot 128 = 16384$ możliwości. 434 sumy są poza zakresem obejmowanym przez tak przyjęty format (np. $480 + 480$ daje 960). Trudno o to mieć jednak pretensje – ograniczona ilość bitów skutkuje ograniczonym zakresem. Pozostaje więc 15950 sum. Z tego 14232 nie dają się wyrazić w tym formacie. Oznacza to, że jedynie 1718 sum jest prawidłowych. Reszta to będą przybliżenia. Zatem jedynie 9.28% wyników jest poprawna.
Gdyby opuścić wszystkie sumy, które się powtarzają, czyli nie rozróżniać czy dodajemy x_1 do x_2 czy na odwrót: x_2 do x_1, wówczas mamy 8256 różnych sum, z których 221 jest poza zakresem naszego formatu a 7121 nie daje się wyrazić w tym formacie. Oznacza to, że jedynie 914 sum jest prawidłowych, czyli 11.37%.

Liczby zmiennoprzecinkowe

Test 1

L.p.	x	y	$y - x$
1	0.0078125000	0.0087890625	0.0009765625
8	0.0146484375	0.0156250000	0.0009765625
9	0.0156250000	0.0175781250	0.0019531250
16	0.0292968750	0.0312500000	0.0019531250
17	0.0312500000	0.0351562500	0.0039062500
24	0.0585937500	0.0625000000	0.0039062500
25	0.0625000000	0.0703125000	0.0078125000
32	0.1171875000	0.1250000000	0.0078125000
33	0.1250000000	0.1406250000	0.0156250000
40	0.2343750000	0.2500000000	0.0156250000
41	0.2500000000	0.2812500000	0.0312500000
48	0.4687500000	0.5000000000	0.0312500000
49	0.5000000000	0.5625000000	0.0625000000
56	0.9375000000	1.0000000000	0.0625000000
57	1.0000000000	1.1250000000	0.1250000000
64	1.8750000000	2.0000000000	0.1250000000

L.p.	x	y	$y - x$
64	1.8750000000	2.0000000000	0.1250000000
65	2.0000000000	2.2500000000	0.2500000000
72	3.7500000000	4.0000000000	0.2500000000
73	4.0000000000	4.5000000000	0.5000000000
80	7.5000000000	8.0000000000	0.5000000000
81	8.0000000000	9.0000000000	1.0000000000
88	15.0000000000	16.0000000000	1.0000000000
89	16.0000000000	18.0000000000	2.0000000000
96	30.0000000000	32.0000000000	2.0000000000
97	32.0000000000	36.0000000000	4.0000000000
104	60.0000000000	64.0000000000	4.0000000000
105	64.0000000000	72.0000000000	8.0000000000
112	120.0000000000	128.0000000000	8.0000000000
113	128.0000000000	144.0000000000	16.0000000000
120	240.0000000000	256.0000000000	16.0000000000
121	256.0000000000	288.0000000000	32.0000000000
127	448.0000000000	480.0000000000	32.0000000000

Wniosek

Praktycznie każdy wynik operacji jest przybliżeniem wyniku dokładnego.

Pytanie

Czy to jednak istotnie jest takim problemem?

Wniosek

Praktycznie każdy wynik operacji jest przybliżeniem wyniku dokładnego.

Pytanie

Czy to jednak istotnie jest takim problemem?

Przypadek Lorenza i jego motyla. Problem dotyczy powstawania i akumulacji błędów zaokrągleń i związany jest z niemożnością przewidywania w deterministycznych układach ze sprzężeniem zwrotnym w tym także w matematycznych modelach, które wykorzystywano do długoterminowych analiz pogody. Jak to zwykle bywa na problem ten Lorentz natrafił przez przypadek...

Opis testu

Jako temat dalszych rozważań wybieramy tak zwany *model logistyczny* a więc rekurencyjne wyrażenie następującej postaci

$$p_{n+1} = p_n + rp_n(1 - p_n) \tag{1}$$

oraz jego „drugą" postać otrzymaną przez zastosowanie elementarnych przekształceń algebraicznych (tak więc z matematycznego punktu widzenia oba te wyrażenia, jeśli nawet nie są takie same, to dają takie same wyniki)

$$p_{n+1} = (1 + r)p_n - rp_n^2. \tag{2}$$

W powyższych równaniach r jest pewną stałą, natomiast n i $n + 1$ to indeksy odpowiednio poprzedniego i nowo obliczanego wyrazu. Jako wartość stałej r przyjęto 3.0 natomiast jako wyraz a_1 przyjęto wartość 0.01.

Liczby zmiennoprzecinkowe

Test 2

Iteracja	Według wzoru 1	Według wzoru 2	Różnice pomiędzy wzorami
1	0.009999999776483 0.010000000000000 -0.000000000223517	0.009999999776483 0.010000000000000 -0.000000000223517	0.000000000000000 0.000000000000000 0.000000000000000
257	0.070538848638535 1.189384903904431 -1.118846055265897	0.070538848638535 1.189384903904431 -1.118846055265896	0.000000000000000 0.000000000000000 -0.000000000000000
258	0.267228215932846 0.513630266710465 -0.246402050777619	0.267228215932846 0.513630266710467 -0.246402050777621	0.000000000000000 -0.000000000000001 0.000000000000001
300	0.099408328533173 0.967600411062598 -0.868192082529425	0.099408328533173 0.961800737412491 -0.862392408879318	0.000000000000000 0.005799673650107 -0.005799673650107
399	0.856545627117157 1.298726103232332 -0.442180476115175	0.856545627117157 0.008244384236384 0.848301242880773	0.000000000000000 1.290481718995948 -1.290481718995948

Opis testu

Na zakończenie jeszcze jeden test – test wrażliwościowy. Sprawdzimy jaki wpływ na wyniki obliczeń ma zakłócenie warunków początkowych. To całkiem tak jak przy pogodzie – zrobiliśmy pomiar temperatury powietrza i „trochę" się pomyliliśmy. Tabela 1 prezentuje część wyników. Pierwszy wiersz dla każdej iteracji policzony został dla współczynnika $r = 3.0$, drugi dla $r = 3.000000000001$, trzeci dla $r = 3.0000000000001$, czwarty dla $r = 3.00000000000001$.

Iteracja	Współczynnik	Wyniki
1	3.0	0.039700000000000
	3.0+10E-12	0.039700000000010
	3.0+10E-13	0.039700000000001
	3.0+10E-14	0.039700000000000
	3.0+10E-15	0.039700000000000
2	3.0	0.154071730000000
	3.0+10E-12	0.154071730000075
	3.0+10E-13	0.154071730000008
	3.0+10E-14	0.154071730000001
	3.0+10E-15	0.154071730000000
3	3.0	0.545072626044421
	3.0+10E-12	0.545072626044783
	3.0+10E-13	0.545072626044457
	3.0+10E-14	0.545072626044425
	3.0+10E-15	0.545072626044422

Tabela: Wpływ niedokładności na otrzymywane wyniki

Iteracja	Współczynnik	Wyniki
4	3.0	1.288978001188801
	3.0+10E-12	1.288978001189313
	3.0+10E-13	1.288978001188852
	3.0+10E-14	1.288978001188806
	3.0+10E-15	1.288978001188801
5	3.0	0.171519142109176
	3.0+10E-12	0.171519142106890
	3.0+10E-13	0.171519142108948
	3.0+10E-14	0.171519142109153
	3.0+10E-15	0.171519142109174
6	3.0	0.597820120107100
	3.0+10E-12	0.597820120100453
	3.0+10E-13	0.597820120106437
	3.0+10E-14	0.597820120107033
	3.0+10E-15	0.597820120107094

Tabela: Wpływ niedokładności na otrzymywane wyniki

Iteracja	Współczynnik	Wyniki
6	3.0	0.597820120107100
	3.0+10E-12	0.597820120100453
	3.0+10E-13	0.597820120106437
	3.0+10E-14	0.597820120107033
	3.0+10E-15	0.597820120107094
7	3.0	1.319113792413797
	3.0+10E-12	1.319113792411292
	3.0+10E-13	1.319113792413548
	3.0+10E-14	1.319113792413772
	3.0+10E-15	1.319113792413795
51	3.0+10E	0.074892694909774
	3.0+10E-12	0.462398200088556
	3.0+10E-13	0.056394912819080
	3.0+10E-14	1.280547749402646
	3.0+10E-15	0.178875826166109

Tabela: Wpływ niedokładności na otrzymywane wyniki

Pytania