Politechnika Wrocławska
Wydział Podstawowych Problemów Techniki

Obliczenia Naukowe
Sprawozdanie z Laboratorium

Lista 1

Autor:
Krzysztof Nowak
III Rok INF.
Nr Indeksu: 229807

22 października 2017

1 Zadanie 1

1.1 Opis problemu

Naszym zadaniem było napisane programu w języku Julia, który iteracyjnie wylicza wartości epsilona maszynowego, ety (najmniejszej liczby dodatniej) i maksimum, dla poszczególnych typów arytmetyki zmiennopozycyjnej zgodnej ze standardem IEEE 754: Float16, Float32 oraz Float64, następnie porównać je z zawartymi w Julii stałymi oraz wartościami z biblioteki z dowolnej instancji języka C - float.h.

1.2 Rozwiazanie

Epsilon został wyliczony w następujący sposób:

1. Ustalono $x := 1$
2. Dopóki $fl(1 + x/2) > 1$, wykonujemy $x := x * 2$

Po wykonaniu kroków pod zmienną x będzie zapisany epsilon maszynowy dla danej arytmetyki. Liczbę eta wyliczono w podobny sposób:

1. Ustalono $x := 1$
2. Dopóki $fl(x/2) > 0$, wykonujemy $x := x * 2$

Po algorytmu pod zmienną x będzie zapisana wartość liczby eta dla danej arytmetyki. Maksimum natomiast obliczono tak:

1. Ustalono $x := 1$, $y := 0$
2. Dopóki $fl(x) \neq \infty$, wykonujemy $x := x * 2$ oraz $y := y + 1$
3. Następnie obliczamy $(2 - macheps)2^y$

Po obliczeniu otrzymamy maksymalną wartość dla danej arytmetyki.

1.3 Wyniki

Wartość	Float16	Float32	Float64
Macheps	9.765625 e-4	1.1920928955078125 e-7	2.220446049250313 e-16
Eta	5.960464477539063 e-8	1.401298464324817 e-45	5.0 e-324
Max	65504.0	3.4028234663852886 e38	1.7976931348623157 e308

Podane wartości zgadzają się w języku Julia z odpowiednio funkcjami: eps(), nextfloat(0.0), realmax() oraz z makrami zawartymi w bibliotece float.h języka C.

1.4 Wnioski

W poleceniu do tego zadania postawiono pytanie jaki związek ma macheps z precyzją arytmetyki oraz eta z Min_{sub}. Prezycja arytmetyki to maksmalny błąd względny jaki może zajść podczas wykonywania zaokrąglania liczby rzeczywistej do jej odpowiednika w arytmetyce zmiennopozycyjnej, natomiast Min_{sub} to minimalna liczba w artytmetyce zmiennopozycyjnej, która jest różna od 0. Dzięki wykonanemu zadaniu otrzymano odpowiedź, że precyzja arytmetyki i macheps są równoważne tak jak Min_{sub} i liczba eta.

2 Zadanie 2

2.1 Opis problemu

Naszym zadaniem było sprawdzenie stwierdzenia Kahana, że macheps można otrzymać poprzez obliczenie wyrażenia $3(4/3-1)-1$ w arytmetyce zmiennopozycyjnej.

2.2 Rozwiązanie oraz wynik

Wartość	Float16	Float32	Float64
Macheps	9.765625 e-4	1.1920928955078125 e-7	2.220446049250313 e-16
$3(4/3-1)-1$	-9.765625 e-4	1.1920928955078125 e-7	-2.220446049250313e-16

2.3 Wnioski

W rzeczywistości wartość tego wyrażenia wynosi 0, ale komputer próbując obliczyć 4/3 nie jest w stanie zapisać wyniku dokładnie, więc zaokrągla go do najbliższej w arytmetyce zmiennopozycyjnej. Działanie to generuje pewien błąd. Otrzymujemy zatem po wykonaniu wszystkich działań macheps (ze znakiem dodatnim bądź ujemnym), którego wartość, zgodnie z definicją, jest maksymalnym błędem względnym, który może zajść podczas operacji zaokrąglania. Wyniki te pokazują, że dokładny macheps otrzymamy po wyznaczeniu wartości bezwzględnej z wyniku. Stwierdzenie Kahana jest zatem poprawne.

3 Zadanie 3

3.1 Opis problemu

Naszym zadaniem było sprawdzenie eksperymentalnie w języku Julia, że w arytmetyce Float64 liczby zmiennopozycyjne są równomiernie rozłożone w [1,2] z krokiem $\delta = 2^{-52}$. Następnie sprawdzić rozmieszczenie w przedziałach $[\frac{1}{2},1]$ oraz [2,4]

3.2 Rozwiązanie

Za pomocą wbudowanej funkcji bits() w języku Julia sprawdzono jak prezentują się liczby reprezentacji bitowej w standardzie IEEE dla typu Float64, na podstawie tych obserwacji wyciągnięto wnioski dla 3 sprawdzanych przedziałów.

3.3 Wyniki

Dla przedziału [1,2]

Wartość	Reprezentacja binarna
1.0	0011111111110000000000000...0000000000
$1+\delta(=2^{-52})$	0011111111110000000000000...0000000001
$1+\delta*2$	0011111111110000000000000...0000000010

Dla przedziału [$\frac{1}{2}$,1]

Wartość	Reprezentacja binarna
$\frac{1}{2}$	0011111111100000000000000...0000000000
$\frac{1}{2}+\delta(=2^{-52})$	0011111111100000000000000...0000000010
$\frac{1}{2}+\delta*2$	0011111111100000000000000...0000000100
$\frac{1}{2}+\delta/2$	0011111111100000000000000...0000000001

Dla przedziału [2,4]

Wartość	Reprezentacja binarna
2.0	0100000000000000000000000...0000000000
$2+\delta(=2^{-52})$	0100000000000000000000000...0000000000
$2+\delta*2$	0100000000000000000000000...0000000001
$2+\delta*4$	0100000000000000000000000...0000000010

3.4 Wnioski

W standardzie IEEE 754 liczby są rozmieszczone równormienie dopóki ich cechy są równe. Gdy cecha zwiększa się o 1 to odległość między liczbami zostaje podwojona. Odwrotnie natomiast następuje gdy cecha zostaje pomniejszona o 1, wtedy to odległoć między liczbami zostaje podzielona przez 2. Wynika to z faktu, że liczb w każdym takim przedziale zapiszemy 2^t-1, więc liczby muszą zostać rozłożone mniej lub bardziej gęsto.

4 Zadanie 4

4.1 Opis problemu

Naszym zadaniem było znalezienie eksperymentalnie w arytmetyce Float64 zgodnej ze standardem IEEE 754 liczby zmiennopozycyjnej z przedziału $1 < x < 2$ takiej, że $fl(x * fl(\frac{1}{x})) \neq 1$.

4.2 Rozwiązanie

Wykorzystując funkcję nextfloat(x) sprawdzano liczby od 1.0 do napotkania najmniejszej takiej, która spełnia założenie.

4.3 Wynik

Pierwszą i najmniejszą liczbą dla której zachodzi założenie jest liczba 1.000000057228997.

4.4 Wniosek

W arytmetyce zmiennoprzecinkowej Float64 zgodnej ze standardem IEEE 754 dochodzi do tego, że $x * \frac{1}{x} \neq 1$ ponieważ nie jesteśmy zdolni przedstawić każdej liczby $\frac{1}{x}$ dokładnie.

5 Zadanie 5

5.1 Opis problemu

Naszym zadaniem było napisanie 4 algorytmów realizujących obliczanie iloczynu skalarnego dwóch wektorów

5.2 Rozwiązanie oraz wynik

Algorytm	Float32	Float64	Float32 bład wzg.	Float64 bład wzg.
a.	-0.4999443	1.0251881368296672e-10	4.96680577e10	11.1849553
b.	-0.4543457	-1.5643308870494366e-10	4.5137965e10	14.541186645
c.	-0.5	0.0	4.9673591e10	-
d.	-0.5	0.0	4.9673591e10	-

5.3 Wniosek

Żaden z algorytmów nie dał satysfakcjonującego wyniku. Wyniki z pojedynczej prezycji są bezwartościowe z powodu rozmiaru błędu tam występującego. Podwójna precyzja dała nam przybliżenia poprawnej wartości w przypadku dwóch pierwszych algorytmów. Pozostałe nie zadziałały z powodu źłośliwych"danych wprowadzonych do tych algorytmów.

6 Zadanie 6

6.1 Opis problemu

Naszym zadaniem było sprawdzenie dwóch równoważnych funkcji i okrelenie która daje bardziej wiarygodne wyniki w arytmetyce zmiennopozycyjnej

6.2 Rozwiązanie i wynik

x	f(x)	g(x)
8^{-1}	0.0077822185373186414	0.0077822185373187065
8^{-2}	0.00012206286282867573	0.00012206286282875901
8^{-3}	1.9073468138230965e-6	1.907346813826566e-6
8^{-4}	2.9802321943606103e-8	2.9802321943606116e-8
8^{-5}	4.656612873077393e-10	4.6566128719931904e-10
8^{-6}	7.275957614183426e-12	7.275957614156956e-12
8^{-7}	1.1368683772161603e-13	1.1368683772160957e-13
8^{-8}	1.7763568394002505e-15	1.7763568394002489e-15
8^{-9}	0.0	2.7755575615628914e-17

6.3 Wnioski

Bardziej wiarygodną funkcją jest g(x) jest to spowodowane, że od pewnego momentu funkcja f(x) za każdym razem zaczyna zwracać 0. Dzieje się to prawdopodobnie przez użyte w tej funkcji odejmowanie, które to redukuje cyfry znaczące, przez co funkcja szybciej traci na swojej dokładności.

7 Zadanie 7

7.1 Opis problemu

Naszym zadaniem było obliczenie pochodnej w punkcie dla danej funkcji przy zmienianiu parametru h.

7.2 Rozwiązanie i wynik

h	f'(x)
2^{0}	2.0179892252685967
2^{-1}	1.8704413979316472
2^{-2}	1.1077870952342974
2^{-3}	0.6232412792975817
2^{-4}	0.3704000662035192
2^{-5}	0.24344307439754687
2^{-6}	0.18009756330732785
2^{-7}	0.1484913953710958
2^{-8}	0.1327091142805159
2^{-29}	0.11694222688674927
2^{-30}	0.11694216728210449
2^{-31}	0.11694216728210449
2^{-32}	0.11694192886352539
2^{-33}	0.11694145202636719
2^{-34}	0.11694145202636719
2^{-35}	0.11693954467773438
2^{-47}	0.109375
2^{-48}	0.09375
2^{-49}	0.125
2^{-50}	0.0
2^{-51}	0.0
2^{-52}	-0.5
2^{-53}	0.0
2^{-54}	0.0

Pochodna funkcji $sin(x) + cos(3x)$ to w rzeczywstości $cos(x) - 3sin(3x)$. Wynik dla x = 1 wynosi 0.116942281. Widać więc, że zmniejszanie h od pewnego momentu przestaje przybliżać, a zaczyna generować większy błąd. Prawdopodobnie jest to wynik działań na bliskich sobie liczbach w arytmetyce zmiennopozycyjnej.