例 1 の解法

例 1 の誤差は 10 進数と β 進数の相互変換:純小数 出した $(0.1)_{10}$ の 16 進数と 2 進数で IBM 方式のときと IEEE 方式のときになるようにすることで理解できる. **(??)** 式で小数点以下 7 桁目を切り捨てる.

$$(0.1)_{10} = (0.199999)_{16} \times 16^0.$$

同様に **(??)** 式で小数点以下 25 桁目を 0 捨 1 入すると,

$$(0.1)_{10} = (0.11001100110011001100\underline{1101})_2 \times 2^{-3} \quad = (0.CCCCC\underline{D})_{16} \times 2^{-3}$$

となる. これらをそれぞれ 10 進数に戻す. IBM 方式の方は **(??)** 式より

$$\begin{aligned}(0.199999)_{16} &= (199999)_{16} \times 16^{-6} \\ &= (((((1 \times 16 + 9) \times 16 + 9) \times 16 + 9) \times 16 + 9) \times 16 + 9) \times 16^{-6} \\ &= \frac{1677721}{16777216} \\ &\approx (0.09999996424)_{10}\end{aligned}$$

となって例 1 の値になる. IEEE 方式も同様に

$$\begin{aligned}(0.CCCCC\underline{D})_{16} \times 2^{-3} &= (CCCCC\underline{D})_{16} \times 2^{-27} \\ &= (((((12 \times 16 + 12) \times 16 + 12) \times 16 + 12) \times 16 + 12) \times 16 + 13) \times 2^{-27} \\ &= 13421773 \times 7.450580597 \times 10^{-9} \\ &\approx (0.1000000015)_{10}\end{aligned}$$

となる.

*

例 2 の解法 $\sum_{n=1}^{10000} 0.01$ の計算において n 項目までの部分和の大きさが $0.01n$ であり, ε の相対誤差が毎回生じたとすると

$$\begin{aligned}\sum_{n=1}^{10000} 0.01n\varepsilon &= 0.01\varepsilon\frac{(10000)(10000+1)}{2} \\ &\cong 0.01 \times \frac{(10000^2\varepsilon)}{2} \\ &= 5 \times 10^5\varepsilon\end{aligned}$$

ほどの誤差が累積する. IBM 方式では $\varepsilon = 6 \times 10^{-8} \sim 10^{-6}$ の間なので, この値は $0.03 \sim 0.5$. IEEE 方式では $\varepsilon = 3 \times 10 - -8 \sim 6 \times 10^{-8}$ として, この値は $0.015 \sim 0.03$ ほどとなる. よって, IBM 方式では $100 - 0.5 = 99.95$,IEEE 方式では $100 + 0.03 = 100.003$ で例 2 の結果とほぼ一致する.

*

例 3 の解法 IBM 方式の場合に $0.1 = (0.199999)_{16} \times 16^0$ を 10 個足すと

$$
\begin{array}{rr}
 & 0.199999 \\
+ & 0.199999 \\
\hline
 & 0.333332 \\
+ & 0.199999 \\
\hline
 & 0.4CCCCB \\
+ & 0.199999 \\
\hline
 & 0.666664 \\
+ & 0.199999 \\
\hline
 & 0.7FFFFD \\
+ & 0.199999 \\
\hline
 & 0.999996 \\
+ & 0.199999 \\
\hline
 & 0.B3332F \\
+ & 0.199999 \\
\hline
 & 0.CCCCC8 \\
+ & 0.199999 \\
\hline
 & 0.E66661 \\
+ & 0.199999 \\
\hline
 & 0.FFFFFA
\end{array}
$$

となり, 例 1 と同様に 10 進法表示にすると

$$
\begin{aligned}
(0.FFFFFA)_{16} &= (FFFFFA)_{16} \times 16^{-6} \\
&= (((((15 \times 16 + 15) \times 16 + 15) \times 16 + 15) \times 16 + 15) \times 16 + 10) \times 16^{-6} \\
&\approx (0.9999996424)_{10}
\end{aligned}
$$

となり 1 より小さいので while の条件は満たされている.

同様に IEEE 方式でも同様の計算を行う. 各計算で桁を浮動小数点表示の 25 桁目を 0 捨 1 入する.

1 回目

$$\begin{array}{rr} & 0.11001100110011001101 \\ + & 0.11001100110011001101 \\ \hline & 1.10011001100110011010 \end{array}$$

答えは 1 桁増えたので最後の 0 を削る, また, その値に合わせて足すほうも削る.

2 回目

$$\begin{array}{rr} & 1.10011001100110011010\not{0} \\ + & 0.11001100110011001100\not{0} \\ \hline & 10.01100110011001100110011 \end{array}$$

同様に 1 桁増えたので最後を繰り上げる. その値に合わせて足すほうも削る.

$$\begin{array}{rr} & 10.01100110011001100110\not{0}_1\ \not{1}_0\not{1} \\ + & 0.11001100110011001100\ 1\quad 1\not{0} \\ \hline & 11.00110011001100110011\ 0\quad 1 \end{array}$$

桁が増えなかったのでこのままの答えを使って計算する.

3 回目

$$\begin{array}{rr} & 11.00110011001100110011001101 \\ + & 0.11001100110011001100110011 \\ \hline & 100.00000000000000000000000 \end{array}$$

答えは 1 桁増えたので最期を削る. また, 足すほうも最期を繰り上げる.

4 回目

$$\begin{array}{rr} & 100.00000000000000000000\ \ 0\not{0} \\ + & 0.11001100110011001100\not{0}_1\not{1}_0\not{1} \\ \hline & 100.11001100110011001101\ \ 0 \end{array}$$

答えは最後まで繰り上がらないのでこれ以降は続けて書く.

$$
\begin{array}{rr}
 & 100.11001100110011001100110010 \\
+ & 0.11001100110011001100110010 \\
\hline
 & 101.10011001100110011001100100 \\
+ & 0.11001100110011001100110010 \\
\hline
 & 110.01100110011001100110011110 \\
+ & 0.11001100110011001100110010 \\
\hline
 & 111.00110011001100110011101000 \\
+ & 0.11001100110011001100110010 \\
\hline
 & 1000.00000000000000000000000010
\end{array}
$$

となる. ここで小数点以下 25 桁以上は 0 捨 1 入すると 0.10000000000000000000000001 となりこれを 10 進に直すと,

$$
\begin{aligned}
(0.10000000000000000000000001)_2 \times 2^1 &= 1 \times 2^1 \times 2^{-1} + 1 \times 2^{-27} \times 2^1 \\
&\approx (1.000000119)_{10}
\end{aligned}
$$

となり while の条件が満たされなくなる.