オーバーフローのない数値計算環境の実現(その2)

6K－1

小嶋　卓

(静岡大学　工学部 情報知識工学科)

１．まえがき

計算機内部の浮動小数点数の表現はパソコンやワークステーションを中心にＩＥＥＥ規格が定着し、ここしばらくはＩＥＥＥ規格の時代が続きそうである。倍長浮動小数点数の指数部は従来のものより若干長くなったものの、依然としてオーバーフローやアンダーフローを起こす問題には弱い。指数部１５ビットの拡張精度型もあるが一部のマシンの一部のコンパイラがサポートしているのみである。またオーバーフローやアンダーフローの解消を目的とする新しい浮動小数点数の表現として、松井・伊理の方式や浜田の方式も提案されているが残念ながら普及するには至っていない。筆者は現行のハードウェアを前提として、ソフトウェアでオーバフローやアンダーフローのない実用的な計算環境を実現するにはどうしたら良いかについて考察し、以下に述べるような効率の良い実現をみた。言語はＣおよびＣ＋＋を使用している。前回[1][2]の報告より 対応機種が増え、いくつかの関数の計算速度の向上と高精度化がなされた。

２．指数部３２ビットの浮動小数点演算パッケージ

ここで定義する新しい浮動小数点数とは仮数部として通常の倍長浮動小数点数を、また指数部として３２ビット整数を用い、これらを組み合わせたものである。
このパッケージの特徴をまとめると次のようになる。

① 指数部が３２ビットあり、オーバーフローやアンダーフローがほとんど起きない。
最大正実数 は 1.761613e+646456993 である。

② 比較的高速である。（通常の倍長浮動小数点数の1.3倍～3倍）浮動小数点数と整数の並行処理性が高いマシンほど効率がよい。

③ 非数（-∞,-0,+0,+∞,NaN および±∞）への対応。定数設定、四則、比較、各関数、入出力に使える。

④ 指数関数、Γ関数などオーバーフローしやすい基本関数を用意。（表１参照）

⑤ 特に精度の指定がなければ "仮数部＋指数部" が一定幅の出力をする。仮数部15～7桁,指数部1～9桁。

⑥ ２進系のマシンであれば移植は容易。
８機種に移植ずみ。ＩＥＥＥ規格でなくても良い。

⑦ マクロやクラスの使い分けによって高速化可能。

⑧ ＩＥＥＥ規格のマシンでは丸めの制御が可能。

Ｃ版のマクロ（３２種）

符号反転	e_neg(a,b)	など	２種
絶対値	e_abs(a,b)	など	２種
四則	e_add(a,b,c)	など４種×３＝１２種	
比較	e_eq(a,b)	など６種×２＝１２種	
型変換	e_d2e(a,b)	など２種×２＝	４種

Ｃ＋＋版の演算子（３６種）

クラス　e_double　（関数で実現）

符号反転	-
絶対値	e_abs()
四則	+, -, *, /, +=, -=, *=, /=
比較	==, !=, >, >=, <, <=
型変換	(e_double),(double)

クラス　e_double1　（一部は inline 関数）

符号反転	-
絶対値	e_abs()
四則	+, -, *, /, +=, -=, *=, /=
比較	==, !=, >, >=, <, <=
型変換	(e_double1),(double)

Ｃ版およびＣ＋＋版に共通な関数と定数

e_double 型関数（２４種）
$2^x, e^x, 10^x, \ln, \log_2, \log_{10},$
$\sinh, \cosh, \tanh, \sinh^{-1}, \cosh^{-1}, \tanh^{-1},$
$\sqrt{}, a^b, \Gamma(x), !, !!, nPr, nCr$, e_class,
e_atof, e_scanf, e_printf, e_form

double 型関数（１４種）
$2^x, 10^x, \log_2, \ln(1+x), \sinh^{-1}, \cosh^{-1}, \tanh^{-1},$
sin, cos, tan, nextafter, f_lsb, drem, f_lsb

e_double 型定数（１１種）
-∞, -0, 0, +∞, ±∞, NaN, 1., 2., 10., Rmax, Rmin

double 型定数（２３種）
-∞, -0, 0, +∞, ±∞, NaN, Rmax, Rmin,
$e, \log_2 e, \log_{10} e, \ln 2, \ln 10, \pi, \pi/2, \pi/4,$
$1/\pi, 2/\pi, \sqrt{\pi}, 1/\sqrt{\pi}, 2/\sqrt{\pi}, \sqrt{2}, \sqrt{2}/2$

丸めの制御　round

表１．用意されているマクロ、演算子、関数、定数

A realization of the Overflow Free Computing Environment (Part 2)
Takashi KOJIMA
Shizuoka University

実現したマクロや演算子や関数や定数は表１.にまとめてある。また演算の速度を計測するためのベンチマークテストを行った。乗算、加算、平方根をそれぞれ２万回のループで計測し、通常のdouble型と新しいe_double型について、３つの総和を秒で表２.に示してある。また、e_double型のdouble型対する比も掲げてある。最近のＲＩＳＣチップを搭載したマシンは速度も向上しているが、浮動小数点数と整数の並行処理性にすぐれ、新しい浮動小数点数の速度は通常の浮動小数点数の速度に比べてあまり大きな速度低下をもたらさない。

マシン	double	e_double	比率（倍）
DG AV300(MC88100)	21.9s	29.4s	1.3
SUN4/280s(SPARC)	19.6s	31.7s	1.6
NEWS-821(+MC68881)	45.7s	79.6s	1.7
PC9801RA+i80387	45.6s	104.7s	2.3
VAX11/780+ACC	64.3s	159.5s	2.5
PC9801m+i8087	143. s	435. s	3.0

表２．C版でのベンチマーク結果

３．二重指数関数型数値積分への応用

両端点が∞になるような関数の積分にも有効な手法として二重指数関数型数値積分公式（ＤＥ公式）[3]が知られている。ここでは森の例題[3] $I=\int_{-1}^{1}1/(1-x^2)^{1-\varepsilon}dx$, $0<\varepsilon<1$ を取り上げよう。通常のやり方では変換後の因子 cosh(C*sinh t) がオーバーフローを起こすが、この限界が VAX-d型で 4.72, IEEE倍長で 6.80, 本方式で 21.36である。通常のやり方ではIEEE倍長で $\varepsilon=10^{-2}$ まで、本方式でも、$\varepsilon=10^{-7}$までしか正しく求まらないことがわかった。そこで、次のような対策をして、$\varepsilon=10^{-306}$ まで求まるようにした。①オーバーフロー対策 。因子 cosh(C*sinh t) に対して $\log_2(\cosh(C*\sinh t))$ を考え、$t\leqq$const. と $t>$const.の部分に分けて計算し、後で 2^x を引用する。② 効率の対策。ε をどんどん小さくしていくと、変換後の関数のピークがどんどんｔの大きいほうへ移動する。従って、ε を小さくすると、積分を打ち切ってもよい位置もｔの大きい方へ移動する。このままでは演算時間もどんどん増大してしまう。ε が 10^{-7}より小さいところでは変換の定数Cを ε に応じて大きくすると、一定のところで打ち切っても良いようになった。この結果を表３.に示す。 この厳密解はΓ関数で表現でき、これから計算されたものも同時に示してある。極めて良い一致をみた。また丸めの影響をみるために round +∞ （切上げ） と round nearest even と round -∞ （切捨て）の３つのモードで計算した。$\varepsilon=10^{-306}$ の場合についてのみ表４.に示す。 丸め誤差の累積は極めて少ないことがわかる。

参考文献

(1)小嶋 卓 オーバーフローのない計算のための浮動小数点演算パッケージ 情報処理学会第37回（昭和63年後期）全国大会講演論文集 5D-9 p59-60

(2)小嶋 卓 他 オーバーフローのない数値計算環境の実現 情報処理学会第38回（昭和64年前期）全国大会講演論文集 4B-6 p62-63

(3)森 正武 「数値解析法」 p67-p72 (1984) 朝倉書店

rn	eps	by DE formula	by gamma function	error	tmax
	10^-01	1.13230869752158e+01	1.13230869752158e+01	3.137584e-16	5.625
	10^-02	1.01379510335044e+02	1.01379510335044e+02	-4.205245e-16	7.875
	10^-03	1.00138561090034e+03	1.00138561090034e+03	0.0	10.125
	10^-04	1.00013862259640e+04	1.00013862259640e+04	1.818737e-16	12.500
	10^-05	1.00001386287521e+05	1.00001386287521e+05	2.910343e-16	14.750
	10^-06	1.00000138629368e+06	1.00000138629368e+06	-1.164152e-16	17.125
	10^-07	1.00000013862943e+07	1.00000013862943e+07	3.725290e-16	19.375
	10^-08	1.00000001386294e+08	1.00000001386294e+08	-2.980232e-16	19.375
	10^-09	1.00000000138629e+09	1.00000000138629e+09	7.152557e-16	19.375
	10^-10	1.00000000013863e+10	1.00000000013863e+10	-3.814697e-16	19.375
	10^-11	1.00000000001386e+11	1.00000000001386e+11	0.0	19.375
	10^-12	1.00000000000139e+12	1.00000000000139e+12	7.324219e-16	19.375
	10^-13	1.00000000000014e+13	1.00000000000014e+13	0.0	19.375
	10^-14	1.00000000000001e+14	1.00000000000001e+14	0.0	19.375
	10^-15	1.00000000000000e+15	1.00000000000000e+15	6.250000e-16	19.375

表3. DE公式による計算例

	eps	by DE formula	by gamma function	error	tmax
rp	10^-306	1.0000000000000e+306	1.0000000000000e+306	4.989601e-15	21.250
rn	10^-306	1.0000000000000e+306	1.0000000000000e+306	0.0	19.375
rm	10^-306	1.0000000000000e+306	1.0000000000000e+306	-2.806650e-15	19.375

表4. 丸めモード違いの影響