数理解析研究所講究録
第453巻 1982年 1-18

浮動小数点表現に代る実数値表現法

日立システム開発研　浜田　穂積

まえがき

実数値の計算機内部における表現形式としての浮動小数点表現は，もちろんこれまでに多大の貢献をしてきている。しかしながら現在の表現形式に対する不満がいくつかでてきた。その第1は，個々の計算機ごとに取決められた仕様が異なり，計算可能な数値の範囲，精度，丸めの方針などの不一致が計算結果を異なるものにする。あるいは，より効率を追求するため，内部表現のままでデータの交換を行なおうとすると，仕様が完全に一致する必要があるのに，現状はそれから程遠い，などの不満がある。これを解決するものとして米国U.C. Berkeley の W. Kahan 等による案をもとにしたIEEE標準案[1,2]が作成された。

IEEE標準案の特徴はまさに上述の不満を解消するためという点に集約されるが，その他にも二三の新しい着想がある。その一つは，丸めの方針をきちんと決めたこと，数でないもの（非数）の表現を導入すべきとしたことなどのほか，単精度では8ビット，倍精度では11ビットと指数部の長さを変

えたことである。ビット数の余裕のある倍精度で多少表現範囲が拡大されるが，それでもなおもう一つの不満であるオーバフロー，アンダフローの発生という問題の根本的解決にはなっていない。

オーバフロー，アンダフローについては，計算の途中で起こるとしても好ましいことではないので，これを防ぐ方法として松井・伊理による案[3]が提出された。これは内部表現上で隣接する仮数部と指数部の境界を動かし得るものとし，指数部の長さを記録するフィールドを持つものである。この表現によると，代数方程式の解法の一つである Gräffe の方法を用いることができて，計算途中で指数の値がかなり大となって仮数部の精度が減少しても，最終結果は表現の本来持つ妥当な精度が得られたと報告されている。

このようにいろいろ新しい表現法が提案されて，従来の形式についての不満が解消したかに見えたが，実は逆にもっと重要な点が満たされなくなってきた。というのは，データの形式がその長さごとに定義されるため，データの長さ，したがってそれの表わす精度が異なるデータ間の相互変換が複雑になってきたことである。大型の電子計算機の場合は余裕があるから，32ビット，64ビットなど処理に好都合なデータ長の種類を少数選んで，その組合せを特別に扱えば何とか解決

できるが，マイクロコンピュータ，その他のディジタル応用機器やAD変換器等，条件の厳しいものの多種の結合を考えると，実数値の表現法を改めて考え直さなくてはならなくなってきた．

新しい実数値表現法の満たすべき条件

新しい実数値表現法[4)5)]を設定するにあたって，次の6条件を考慮した．これらのうち次の3条件を必須と考えた．

条件1：2進数としての指数，仮数から本表現へ，またその逆の変換が容易であること．このことは対数関数など，複雑な処理を用いなくてすむことを前提に置いている．

条件2：オーバフロー，アンダフローが事実上起らないこと．

条件3：表現仕様がデータの長さに依存せず，長さの異なるデータ間の相互変換が容易であること．

次の3条件は，必須というほどの必然性はないが，可能ならば満たしたいと考えた．

条件4：形式の取決め事項が不要あるいはなるべく少いこと．取決めのためのパラメタは少い程よく，ある一つの形式に自然に落着くようであれば望ましい．

条件5：固定長と考えたとき，すべてのパタンが異なる数値

に対応すること．従来の表現法には正規形，非正規形があって異なるパタンが同じ値を表わす場合がある方式が多い．こういうパタンの無駄使いをさける．

条件6：値の大小関係が，固定小数点と考えた場合のそれと一致すること．比較演算命令を固定小数点演算用の命令と共用できる．

新実数値表現法

表わそうとする数をxとする．これを

$$x = 2^e \times f \qquad (1)$$

と，2つの数eとfで表現する．ここで値を一意的にするためe，fに次の条件を設ける．

$$e：整数 \qquad (2)$$

$$1 \leq f < 2 \quad あるいは \quad -2 \leq f < -1 \qquad (3)$$

条件1により，この表現法はe，fのビットパタンから簡単な方法で得られるものとする必要がある．条件3を満たす変換法を得るために，より意味のある情報ほど左に記す．すなわち，数を大きく規定する情報ほど左に置き，長いデータへの変換は右に0のビットを補い，短いデータへの変換は左から必要な長さのビット数だけ切り出してくることによって得られるものとする．このときビット0とビット1の意味づけ

を上手に選べば，条件6を満たすことが可能となる．具体的には次の順に置く．

(i) 数xの符号（1ビット）

(ii) 指数部

(iii) 仮数部

仮数の条件を(3)とすれば，xの符号にしたがって

$$\left.\begin{aligned} f>0 \text{ のとき} \quad & f = 0.1 f_1 f_2 \cdots f_j \cdots \\ f<0 \text{ のとき} \quad & f = 1.0 f_1 f_2 \cdots f_j \cdots \end{aligned}\right\} \qquad (4)$$

と表わせる．この表現は小数点の左に2ビットを有する，2進固定小数点（負数は2の補数）表現である．f_jについては全ての組合せが起りうる．小数点の左の情報は符号ビットのみで定まる．したがって

符号部：$x>0$ のとき0，$x<0$ のとき1

仮数部：$f_1 f_2 \cdots f_j \cdots$

とすればよい．これを図示すると図1の通りである．

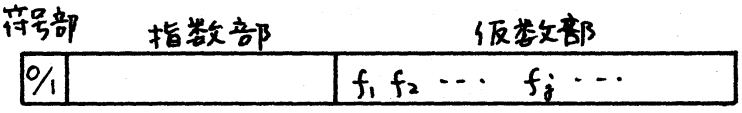

図1．　データの構成

指数部は条件2を満たそうとするとき可変長にせざるを得ない．しかも指数の値に上限を設けたくない．また $x=\pm 1$ すなわち $e\sim 0$ のとき我々にとって重要な値であるから，なる

べく指数部を短くしたい。そこで自分自身で限界を知りうる表現として次の可変長表現を採用する。

まず $x>0$ でかつ $e>0$ の場合，e の値が2進 $i(>0)$ 桁で表わせる範囲は次の通りである。

$$2^{i-1} \leq e \leq 2^i - 1 \tag{5}$$

これに f の範囲も含めて x の範囲として表わせば

$$2^{2^{i-1}} \leq x < 2^{2^i} \tag{6}$$

である。次に $e<0$ のとき，同様に，対称性を考慮すれば

$$2^{-2^i} \leq x < 2^{-2^{i-1}} \tag{7}$$

とすべきであり，これを e の範囲に翻訳すれば

$$-2^i \leq e \leq -2^{i-1} - 1 \tag{8}$$

となる。これは e を2の補数で表わすと解釈できる。このことを用いて，e の2進表現を以下の通りとしよう。

$$\left.\begin{array}{ll} e>0 \text{ のとき} & 0\cdots 0\,1\,e_{i-1}\cdots e_2 e_1 \\ e<0 \text{ のとき} & 1\cdots 1\,0\,e_{i-1}\cdots e_2 e_1 \end{array}\right\} \tag{9}$$

(5),(8) には $e=-1,0$ の場合が含まれないが，これは $i=0$ と考える。ここで定めた i と $e_{i-1}\cdots e_2 e_1$ とから，i の識別ビット列を左に追加して指数部を次の通りに構成する。

$$\left.\begin{array}{ll} e\geq 0 \text{ のとき} & \underbrace{1\cdots 1}_{i+1\text{個}}\,0\,e_{i-1}\cdots e_2 e_1 \\ e<0 \text{ のとき} & 0\cdots 0\,1\,e_{i-1}\cdots e_2 e_1 \end{array}\right\} \tag{10}$$

(10)のビット列を符号のない2進数と考えたとき，固定小数点数としての大小順は，eの大小順に一致する．xの大小順は$x>0$のときeの大小順に一致するから，条件6を満たす．

$x<0$のときは，xの大小順とeの大小順は逆であるから，(10)による指数の順を逆にする．これは(10)の指数部の1の補数をとったものとして定める．

新表現法の形式的定義

前節のような慣例的定義法では，条件3,4,5,6が満たされているか否かが直観的に理解しにくい．そこでこれと同じ内容を以下に示す区間分割に基づく方法で形式的に定義する．数学的にはこちらを正式な定義とし，前節のものを従と考える．まず一般論として次の事項から規定する．

任意のビット列Sは実数値の一つの区間に対応する．すなわち

$$S:\{x \mid a\leq x<b\}$$

これを次の記法を用いて示す．なおSが正しく表わすと考える値は区間の下限aであることにする．

$$I(S)=[a,b) \qquad (11)$$

Sの右にビット0を連結したものをS0，ビット1を連結したものをS1と記す．このとき，Sの形およびa,bの値によっ

で定まる第3の値cによって区間I(S)が2分され2，次の対応づけが行なわれる．

$$I(S0) = [a, c), \quad I(S1) = [c, b) \tag{12}$$

上述のことから，条件3, 4, 5, 6が満たされることが理解できよう．

さて，区間の分割を次の4段階の手順で行なう．

(I) 疎分割

$$I(1) = [-\infty, 0) \tag{13-1}$$

$$I(0) = [0, +\infty) \tag{13-2}$$

から出発する．これらを-1, 1で分割して次を得る．

$$I(10) = [-\infty, -1) \tag{14-1}$$

$$I(11) = [-1, 0) \tag{14-2}$$

$$I(00) = [0, 1) \tag{14-3}$$

$$I(01) = [1, +\infty) \tag{14-4}$$

さらにこれらを-2, -0.5, 0.5, 2で分割して次を得る．

$$I(100) = [-\infty, -2) \tag{15-1}$$

$$I(101) = [-2, -1) \tag{15-2}$$

$$I(110) = [-1, -0.5) \tag{15-3}$$

$$I(111) = [-0.5, 0) \tag{15-4}$$

$$I(000) = [0, 0.5) \tag{15-5}$$

$$I(001) = [0.5, 1) \tag{15-6}$$

$$I(010) = [1, 2) \tag{15-7}$$

$$I(011) = [2, +\infty) \tag{15-8}$$

(15)の8式のうち，ビット列の第2ビットから右にみて，0あるいは1の連が止まっている(15-2),(15-3),(15-6),(15-7)については直ちに(IV)の等差分割に進むが，その他の4つの場合は次の二重指数分割を行なう．

(II) 二重指数分割

ここでは $n \geq 2$ の場合について帰納的に次の分割を行なう．

$I(10^n) = [-\infty, -2^{2^{n-2}})$ を $-2^{2^{n-1}}$ で

$I(11^n) = [-2^{-2^{n-2}}, 0)$ を $-2^{-2^{n-1}}$ で

$I(00^n) = [0, 2^{-2^{n-2}})$ を $2^{-2^{n-1}}$ で

$I(01^n) = [2^{2^{n-2}}, +\infty)$ を $2^{2^{n-1}}$ で

これにより，0あるいは1の連の止まっている次を得る．

$$I(10^n1) = [-2^{2^{n-1}}, -2^{2^{n-2}}) \tag{16-1}$$

$$I(11^n0) = [-2^{-2^{n-2}}, -2^{-2^{n-1}}) \tag{16-2}$$

$$I(00^n1) = [2^{-2^{n-1}}, 2^{-2^{n-2}}) \tag{16-3}$$

$$I(01^n0) = [2^{2^{n-2}}, 2^{2^{n-1}}) \tag{16-4}$$

(III) 等比分割

(16)の4つの場合について，次の分割を $n-2$ 回行なう．

$$c = \pm\sqrt{ab} \quad (\text{符号は } a \text{ のそれと同じ}) \tag{17}$$

で分割する．これにより区間の両端の比は2となる．

(IV) 等差分割

最後に次の分割を任意回行なう.

$$c=(a+b)/2 \tag{18}$$

で分割する.

以上の分割法は, ビット列と区間との対応関係の生成法であって, 所望の長さのビット列が得られれば, 必ずしもすべての手順を完了しなくても途中で打切ってよい.

上述の形式的定義によって得られるビット列と, 従来の表現における符号, 指数部, 仮数部との関係はほぼ次の通り.

- (13)によって得られるビット列は符号ビットに対応する.
- (14), (15)と(II), (III)で新たに得られるビット列は指数部に対応する.
- (IV)によって新たに得られるビット列は仮数部に対応する.

非数

この節ではデータの長さはnビット固定として考える. 次の6つの極端な場合を考える.

(i) $10\cdots00 \longrightarrow -\infty$

(ii) $10\cdots01 \longrightarrow -2^{2^{n-3}}$

(iii) $11\cdots11 \longrightarrow -2^{-2^{n-3}}$

(iv) $00\cdots00 \longrightarrow 0$

(v) $00\cdots01 \rightarrow 2^{-2^{n-3}}$

(vi) $01\cdots11 \rightarrow 2^{2^{n-3}}$

0は(v)において$n\rightarrow\infty$とした極限と考えられ，すべてが0の列で表現することは自然である．(13)の定義はこの点を考慮してなされたものである．

オーバフロー，アンダフローを無くしたいが，ビット・パタンの種類は有限であるから，例えば絶対値の小さい数同志の乗算を行なえばもっと絶対値の小なる結果が得られる……と，いずれは表現不可能な値になろう．そこで，表現可能な最も絶対値小のパタンを極限的な値と解釈し，それより絶対値小にはなり得ないことにすればよい．これは結局のところアンダフローと同じであるが，その値が現実の計算ではまず絶対にといっていいほど起りにくいものであって，たとえ起ったとしてもそういう状態であることがわかればよいという程度のものであれば，アンダフローと考えなくてもよいであろう．ちなみに$n=32$としたときの，表現可能な絶対値最小の値は$2^{-2^{29}}$で，10進数で云えば小数点の後に1.61×10^8個あまりの0を並べた後に初めて0でない数の出現するような値である．この場合の表現を，通常の解釈ではなく，特別に状態と考え非数と呼ぶ．非数の概念はIEEE標準案で提案され，松井・伊理方式でも採用されている．両方式とも数のパタンと

は切りはなして独立に定義されているので，数の表現と連続しない。しかるに本表現は以下のように数の表現と連続して自然な解釈が可能である。

$$00\cdots01 \rightarrow +0, \quad 11\cdots11 \rightarrow -0$$

次に対称性を考慮して

$$01\cdots11 \rightarrow +\infty, \quad 10\cdots01 \rightarrow -\infty$$

と定める。こうすると $10\cdots00$ は $-\infty$ ではなく準なる ∞ として扱う方が便利である。$+\infty$ と $-\infty$ はビット・パタンでは近い関係にあり，$10\cdots00$ はその中間にあるからである。以上をまとめて次の通りに定義する。

$$\left.\begin{array}{l} 10\cdots00 \rightarrow \infty \\ 10\cdots01 \rightarrow -\infty \\ 11\cdots11 \rightarrow -0 \\ 00\cdots00 \rightarrow 0 \\ 00\cdots01 \rightarrow +0 \\ 01\cdots11 \rightarrow +\infty \end{array}\right\} \quad (19)$$

表現法に固有の精度に関する評価

多くの浮動小数点表現では，仮数部が数 x の大きさによらず一定の長さをとるので，表現法に固有の精度は相対誤差がほぼ一定という形で述べられ，量としてこうときは用いられ

る基数における仮数の桁数でする．仮数部の桁数の変りうる松井・伊理方式とか本表現ではより細く調べてみる必要がある．ここでは誤差として，表現可能な最小単位の半分のものがあるとして，相対誤差Erと絶対誤差Eaについて考察する．

64ビット・データの相対誤差Erについて，代表的な4種の表現についてのものを図2に示す．横軸は表現しようとする値xの，縦軸は相対誤差Erの，それぞれの絶対値の2を底とする対数で示す．図2はかなり複雑に重なりあって見にくい

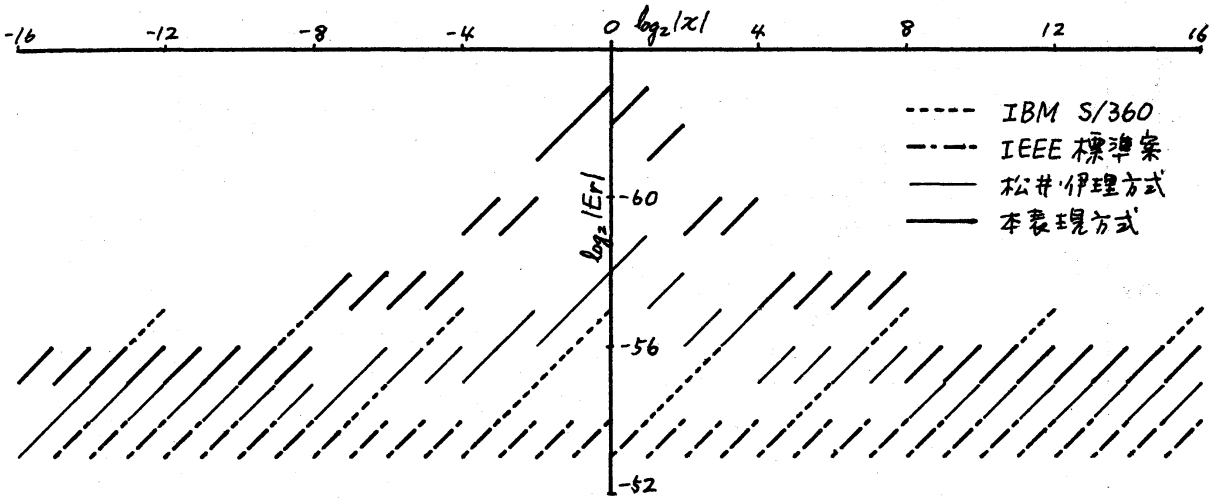

図2．表現法固有の相対誤差

ので少し単純化する．同じ形が左右に並ぶ場合1本の横線で結んだものと置き換えてみる．このような場合最も不利な条件で考えねばならないので，各々の部分の下の縁を連ねなければならない．これを行なうとほぼ左右対称となる．そこで右半分だけをとって，かつ横軸の尺度のさらに2を底とする対数をとる．こうして得られたのが図3である．図4は同様

に32ビット・データの場合である。松井・伊理方式については，原論文では32ビット・データを定義していないが，容易に類推可能のためそれによった。

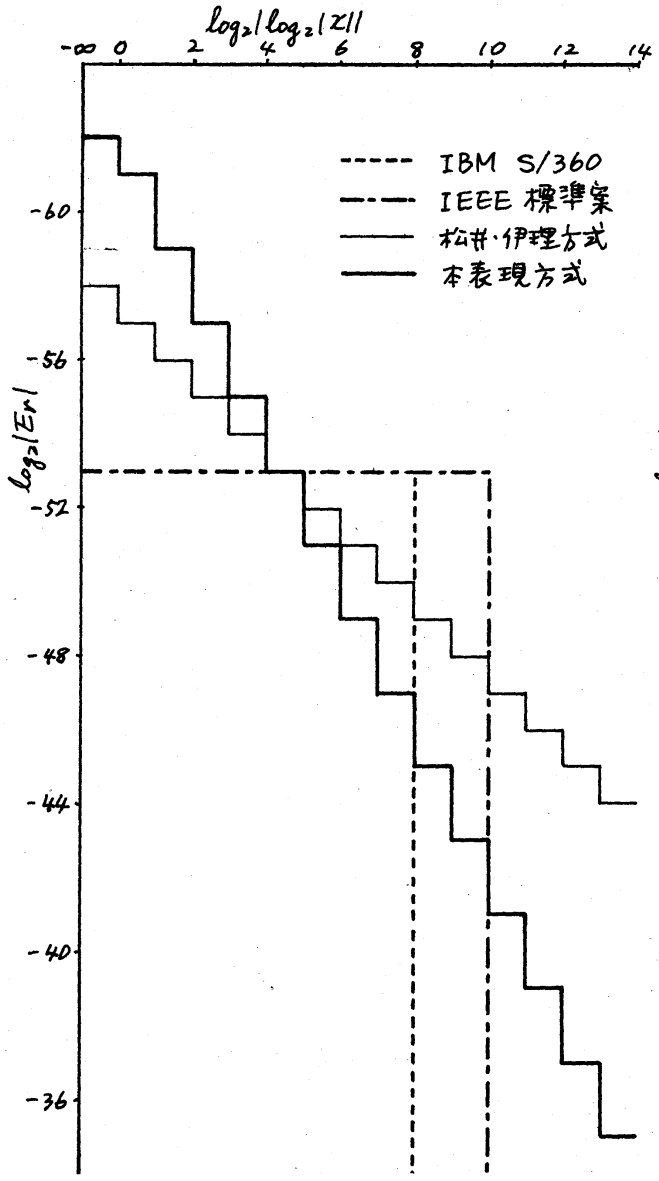

図3．表現範囲と相対誤差
（64ビット・データ）

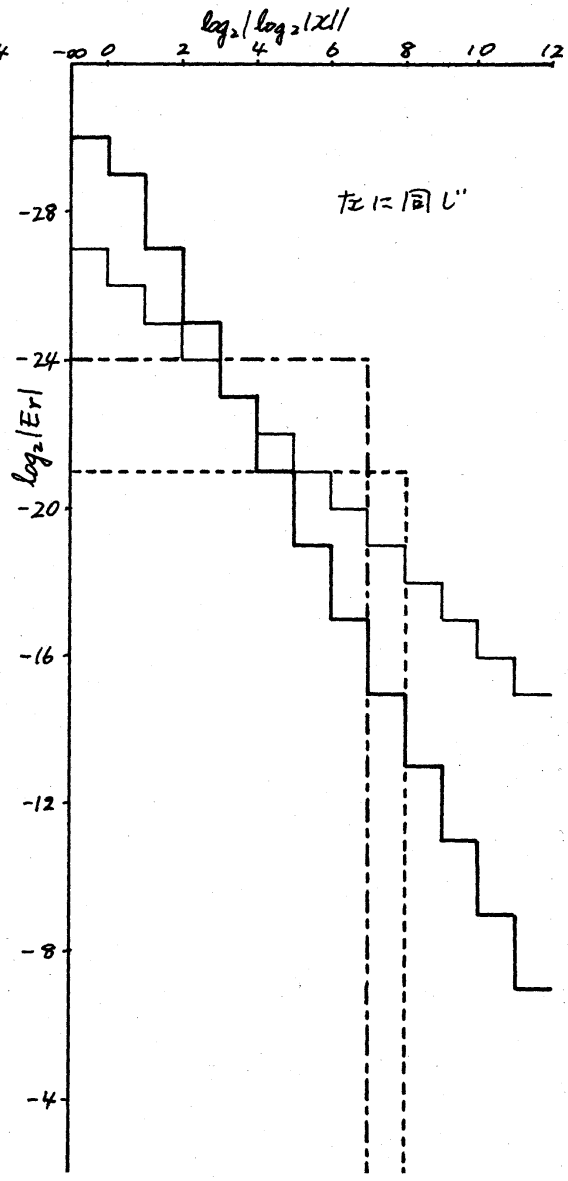

図4．表現範囲と相対誤差
（32ビット・データ）

次に64ビット・データの場合の絶対誤差を図5に示す。な

おこれには松井・伊理方式に代えて，64ビットで $-2 \leq x < 2$ の範囲を表わす固定小数点表現の場合を追加して記してある．これによると，本表現が固定小数点表現と比べて不利であるのは3個所だけで，しかもいずれにおいても1ビットの不利

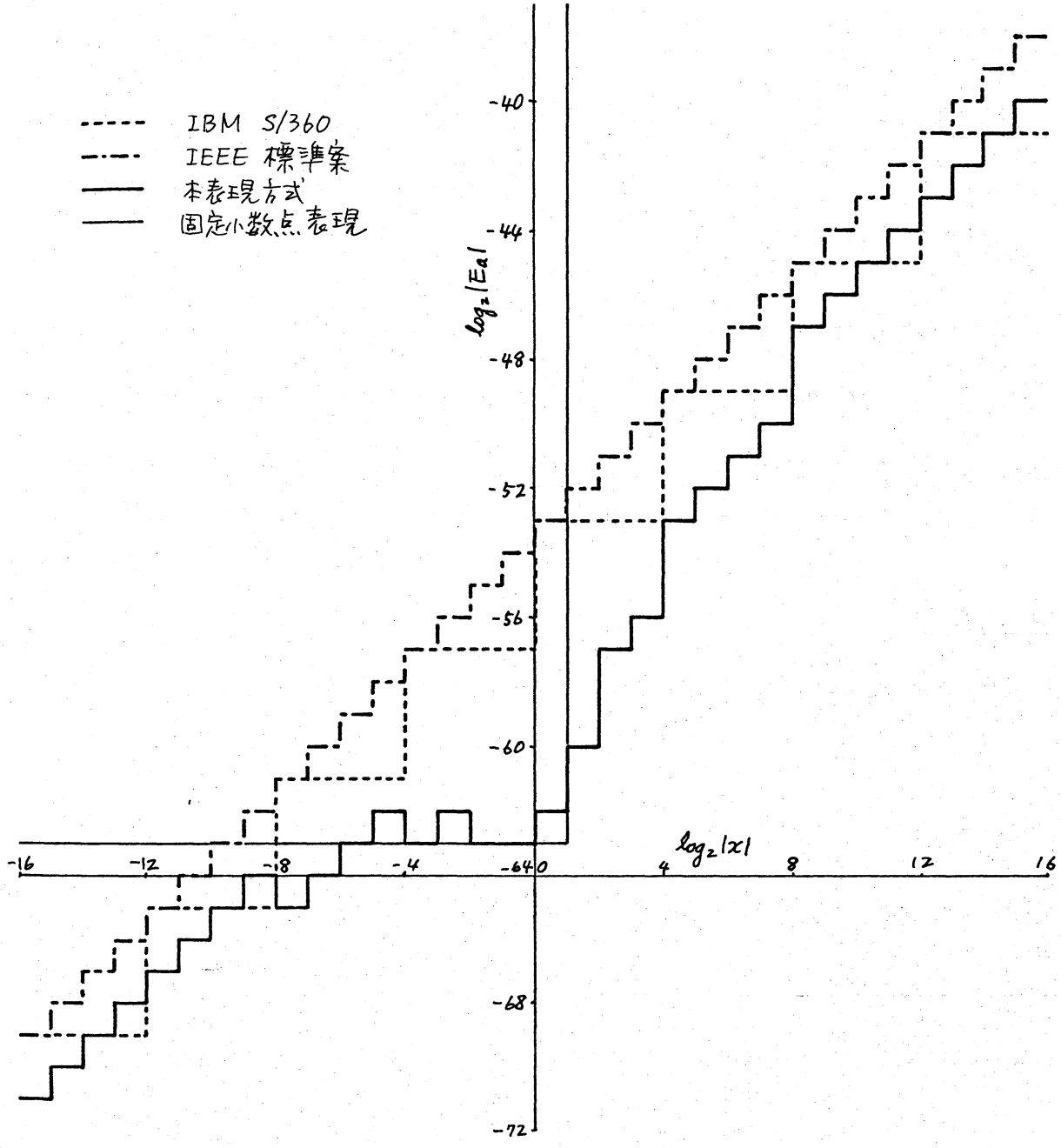

図5. 表現法個有の絶対誤差（64ビット）

にすぎず，なおかつ浮動小数点表現の持つ長所，すなわち大きい数を表現でき，小さい数はより細かく表現可能という長所を併せ持つことがわかる。このことは別の観点から見るとき，ビット数の小なるデータもかなりの表現能力を持つことを示している。

以上の考察により，本表現は汎用大型計算機での実数値計算のみならず，ミニコンピュータ，マイクロコンピュータにも使え，また広義のアナログ・ディジタル変換におけるディジタル側表現にも適する。これによりすべてのディジタル機器における実数値表現が一つの表現法で統一できるから，これら機器を有機的に結合して，自由な計算処理が可能となる。

あとがき

現行の浮動小数点表現法に対して指摘されている欠点をほぼすべて取除くことのできる新しい実数値表現法を考案した。本表現法は最初に設定した6条件をすべて満たすことができた。これらを含めて，次の8項目が本表現法の特長である。

(i) 2進数としての指数と仮数との間で容易に相互変換が可能である。

(ii) オーバフロー，アンダフローが事実上発生しない。

(iii) 表現の仕様がデータの長さによらず一様で，そのため

長さの異なる系を相互に簡単に結合できる。

(iv) 形式の取決め事項が不要である。データの特性を定めるパラメタはデータの長さのみである。

(v) データの長さを固定するとき，すべてのパタンは異なる数値に対応し，無駄がない。

(vi) 値の大小関係が，固定小数点数と考えた場合のそれと一致する。

(vii) 固定小数点表現と比べ，絶対誤差はたかだか1ビットの不利でしかない。

(viii) データの延長で，任意の実数値を限りなく近似可能である。

本表現が数値解析的な実際の応用においてどのような結果をもたらすかについては，さらに研究を待たねばならない。これについては残念ながらここで言及できない。今後に残された問題点と考える。

なお文献[4]で規定した表現法と，文献[5]で規定した表現法との間には細部で相異がある。これは文献[4]を投稿した後で，計算機の論理回路構成を容易にするために変更した方がよいと考えたので，その結果を文献[5]に記したものである。結果的には理解の点でも後者が優れていた。そこで，本報告書も文献[5]に規定したものによっている。

参考文献

[1] J. Coonen, et al : A Proposed Standard for Binary Floating Point Arithmetic, ACM SIGNUM Newsletter Special Issue. (Oct. 1979), pp. 4-12.

[2] D. Stevenson, et al: A Proposed Standard for Binary Floating-Point Arithmetic, Draft 8.0 of IEEE, COMPUTER (Mar. 1981), pp. 51-62.

[3] 松井正一・伊理正夫：あふれのない浮動小数点表示方式，情報処理学会論文誌，Vol. 21, No.4, (1980年7月), pp. 306-313.

[4] 浜田穂積：二重指数分割に基づくデータ長独立実数値表現法，情報処理学会論文誌，Vol. 22, No. 6, (1981年11月), pp. 521-526.

[5] 浜田穂積：浮動小数点表現に代る実数値表現法，情報処理学会第23回全国大会予稿集（1981年10月），pp. 151-152.