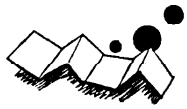
解　説



新標準浮動小数点体系の提案†

一　松　　信††

1. はしがき

現在の電子計算機の主流を占めるのは，IBM 360の流れを汲む十六進 byte 方式である．これが純粋の数値計算の目的には，不便が多いことは，これまで数多く指摘されている．単精度数の精度が低いこと，指数部の幅が狭いこと，精度を保つための工夫に非常な手間と労力と計算時間を要すること，などがその主要な難点であるが，端的にいって，360の基になった SPREAD計画において，科学技術計算に対する吟味が不十分であり，「必要ならば倍長計算をすればよい」の一言で，反対意見をおしきったしわよせが，今日にまで尾をひいている点があるらしい．

数値解析の専門家が3人寄れば，市販の計算機に対するぼやきが語られるといわれていたが，近年では集積回路の進展にともなって，特殊目的用の単能計算機の製造がずっと容易になってきた．計算機がその名に反して事務処理機となり，数値計算がむしろ特殊目的になったのならば，逆に科学技術用の数値計算に適した特別な計算機を設計しようという着想は，当然生じてよいはずである．いわゆる丸め誤差（厳密には計算にともなう誤差）も，厳密に評価するためには，個々の計算機の算術演算の方式にまで立ち入らなければならず，そのための標準化が望まれるとともに，十六進方式では余りに誤差が多すぎて，深い理論を考えるのに値しない，ともいわれている．

他方では，現在のソフトウェア製作者は，ハードウェア製作者の不適切な設計の後始末をさせられている場合が多い．あふれを防ぐために，$\sqrt{x^2+y^2}$ をこの形で直接に計算せず，$x=y=0$ は除いて

$|x|\geqq|y|$ なら $|x|\sqrt{1+|y/x|^2}$

$|x|<|y|$ なら $|y|\sqrt{1+|x/y|^2}$

といった工夫を強いられる，というのが，小さな一例である．民主主義を口にするのなら，ハードウェア作製者，ソフトウェア作製者，ユーザがそれぞれ苦労を分ち合うのが公平であろう．

近年いろいろ能率的な諸算法の研究をしている人々も，現行の計算機ではそれが十分に生かせず，それを活用できるような計算機の設計から考え直す必要があるといいだしている．

以上のような動向を受けて，カリフォルニア大学の W. Kahan らは，新標準浮動小数点体系を提案し[1),2)]，1978年12月 IFIP W.G. 2.5（数学的ソフトウェア）の研究会（Wien 市郊外の Baden）で発表し[3)]，大きな話題となった．その内容は必ずしも革新的なものではないが，重要なのは，十数年前からくりかえされてきた諸不満を吸い上げ，これに基づく新しい現実の計算機の製造が，既に始められているらしい点である．

Kahan 自身が強調しているように，この種の設計案には，唯一最良の解はありえない．すべては費用と利益のバランスであり，現実の計算機にまで育てた者が勝ちであろう．

以下彼らの考え 1),2) の要点を紹介する．筆者自身にも，多くの疑問点があり，読者の方々にもそれが多いと思う．Kahan 自身の述べている検討事項とともに，末尾に若干論じたい．

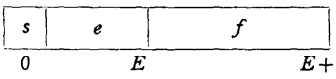
筆者は Kahan に対して，この案に深い関心を有し，日本の関係者と討論して意見をしらせたい旨申しでたが，「アメリカでこれに基づいた計算機が市販されたら，日本のメーカが早速まねをするだろう．」と皮肉をいわれて，ひき下るという苦い経験をした．この言は，日本の計算機産業が外国からどう見られているのかを端的に示しており，我々として素直に受けとめなければならないだろう．

2. 基本データ構造

数は二進絶対値げたばき・けち表現とする（解説後述）．数値語は次の構造を有している．

† Proposal of a New Standard Floating Point Arithmetic System by Sin HITOTUMATU (Research Institute for Mathematical Sciences, Kyoto University).

†† 京都大学数理解析研究所



s（1ビット）は符号部で1が負，0が正．eはE桁の指数部で，$B=2^{E-1}-1$のげたをはかせた表現とする．fはF桁の仮数部で，正規化された数は$1.f$（fの最上位に小数点があるとして表わした二進小数）で表わすが，この1.を陽に表現しない（これを「けち」表現とよんだ）．全語長は$E+F+1$であり，表-1のような値を標準とする．数値は

$$(-1)^s 2^{e-B}\times(1.f)$$

となる．

注意すべき点は，倍長，四倍長数では，仮数部だけでなく，指数部も増加していることである．けち表現は二進法に特有の表現法であり，じっさい1ビット分だけ精度が向上することが，実験的にも確かめられている．十進表示（具体的には10ビット千進表示）や，補数表示などの対案を採らなかった一つの原因は，けち表現による1ビット分の向上を重く考えたためらしい．なお四倍長拡大型の案は未決定である．

もちろんすべての機械で表-1のすべての型を許すわけではなく，たとえば単長と単長拡大型のみという設計もありうる．拡大型はアキュムレータその他で中間的に使用することを主眼としており，通常の記憶装置への格納方式は未定義である．

この拡大型は，たとえば$\sqrt{x^2+y^2}$において，x,yの指数部が許容限界の半分を僅かに超えており，答は許容限界内にあるのに，中間で僅かなあふれを生ずる場合に，算法の工夫をせずとも，直接に正しく計算できることを，一つの目標にしている．その基礎には，上下へのあふれは，めったに生じない例外的異常現象ではなくて，科学技術計算には日常茶飯的に生ずる現象であるから，無用の労力を避けられるようにしようという自覚ないし哲学があるように思われる．

指数部の整数値eが$0<e<2^E-1=2B+1$の範囲にある数は，上記のような正規化された数とするが，$\times e=0$の数は**不正規化** (denormalized) 数とし，$(-1)^s 2^{-B+1}(0.f)$を表わすものとする．とくに$e=0, f=0$を正規化された**0**とし，$+0$と-0とを区別する．これは僅かの下へのあふれを認めて，支障なく計算しようという工夫である．

表-1

	E	F	全長	B	指数部の範囲
単長	8	23	32	127	$10^{\pm 38}$
単長拡大	11	≧32	≧44	1023	$10^{\pm 307}$
倍長	11	52	64	1023	同上
倍長拡大	15	≧64	≧80	16383	$10^{\pm 4932}$
四倍長	15	112	128	16383	同上

他方$e=2B+1$（全ビットが1）の数は特別な値とする．このうち$f=0$のものは**無限大**を表わす．当然$+\infty$と$-\infty$の区別があるが，符号なしの$0,\infty$という「射影表現」も，モードの切換の形で許す．eが最大で$f\neq 0$である語は**非数** (not a number；略してNAN) とする．この場合fの情報は多くはごみであるが，それの生じたポインタなどの情報を保持する場合もあるらしい．

値の大小比較は，符号以外はビットの辞書式順序による．指数部が上位にあれば，あふれの検出が容易である他に，このような簡略比較ができる利点がある．この結果，正の非数は$+\infty$よりも大きく，負の非数は$-\infty$よりも小さいと判定される．$-0<+0$であり，正の不正規化数は$+0$より大きく，正の正規化数より小さくなるから，実用上に支障はない．

ただしこの比較命令を使うためには，仮数部が完全に0になったときには，指数部も0にするという正規化を正しく実行しなければならない．（けち表現のため，2^nは見掛け上$e\neq 0, f=0$の形で表現されるが，それは差支えない．）その昔この点に不備のあった計算機で，指数部が大きい非正規化0が，それより指数部の小さい正の数よりも大きいと判定され，誤りの原因追求に苦労した経験がある．

3. 丸　め

丸めは，下記の4種の命令，またはモードを考える．ただしこの4種全部を常備するのでなく，下記の記号のRNのみとし，整数商などを求めるためにRZを部分的に用意する，という設計も考えられている．もちろん標準はRNであり，RP, RMは区間解析の便宜用に考えられたものである．

値zに対して，所要の桁数で正しく表現できる数のうち，zよりも（代数的に）小さい数のうち最大のものをZ1，zよりも大きい数のうち最小のものをZ2とする．

RN (Round to Nearest) Z1とZ2の近いほうを採る．ただし機械的な0捨1入でなく，ちょうど中間のときは，末位のビットが0のほうをとる．これは現行JIS規格の四捨五入の規約と同じであり，このほうが機械的0捨1入や，末位のビットを1にする規則よりも優れていることを示す例があげられてい

る.

RZ (Round to Zero) |Z1|, |Z2| の小さい方を採る. すなわち絶対値切捨てである.

RP (Round to Plus ∞) つねに Z2 を採る. つまり正なら切上げ, 負なら絶対値切捨てとなる.

RM (Round to Minus ∞) つねに Z1 を採る. つまり正なら切捨て, 負なら絶対値切上げとなる. 0 は −0 となる (他の丸めでは, 0 は +0 となる).

将来, 値の対を INTERVAL という語の型にして, 区間解析式の丸めの操作を直接実行する案も, 検討事項になっている.

4. 算術演算

最低限度, 必要な型の数について, 次のような演算が必要とされている.

ADD (加法), SUB (減法), MUL (乗法), DIV (除法), REM (整数除法の余り), CMP (比較), MOV (転送), SQRT (平方根).

他に型の変換命令として次のようなものがあげられている (説明省略).

INT, FINT, INTF, BINDEC, DECBIN, INTEEC, STRBIN.

型の混合演算は原則として禁止されるが,

(単×単→倍)+倍→倍

の類は, 実用上にも必要が高い. 内積計算 $\sum a_i b_i$ において, 中間の積 $a_i b_i$ をすべて倍長精度で保持し, 最後の結果を単長に丸めれば, 事実上全部を倍長で計算したのに匹敵する精度がえられることは, Wilkinson[4]が詳しく解析したとおりである. この操作が, 手廻し計算器や初期の電子計算機では自動的にできたのに, 現在の電子計算機では特別に凝ったプログラムを書き, 余分の時間をかけないと実行できないのが, 不満の種になっている. したがってこの種の計算が自然に機械語および自然な言語表現で可能なように配慮することが, 強調されている. 単長数同志の演算結果が, アキュムレータに単長拡大型のデータとして保持されているだけでも, 随分改良されることが期待される.

Kahan の論文[1]の付録には, 多項式, 連分数, 消去法など, 数値計算で基礎的な演算のいくつかについて, 実際に普通に実行されるループの形に変形し, どのような配慮をすれば, プログラマが苦労しなくても精度が保持しやすいか, が解析されている.

四則演算については, 被演算数に関して詳しい仕様が 2) に与えられている. 加法, 乗法, 除法の要点を**表-2,3,4** に示した. ここで V は通例の数であり, 不正規化数でもありうる. V同志の数の演算で, 上下のあふれが生じたときの処置についても細かく規定されているが, ここでは省略する. ±0 同志の積は 0 に正しい符号をつけたものである. ±0 同志の和は, 同符号なら同じ符号の 0, (+0)+(−0) なら丸めのモードに応じて −0 (RM) または +0 (他の丸め) とする. 非数同志の演算では, 仮数部同志を比較して小さい方の非数を答とするのが原則である.

表-2 $X+Y$

$X\backslash Y$	±0	V	±∞	非 数
±0	±0	Y	Y	Y
V	X	$X+Y$	Y	Y
+∞	X	X	(†)	Y
非 数	X	X	X	非 数

(†) (+∞)+(+∞)=+∞, (−∞)+(−∞)=−∞
(+∞)+(−∞) は非数 (不定)

表-3 $X*Y$

$X\backslash Y$	±0	V	±∞	非 数
±0	±0	±0	非 数	Y
V	±0	$X*Y$	±∞	Y
±∞	非 数	±∞	±∞	Y
非 数	X	X	X	非 数

(±0)*(±∞) は非数 (不定)

表-4 X/Y

$X\backslash Y$	±0	不正規	正 規	±∞	非 数
±0	非 数*	非 数*	±0	±0	Y
V	±∞*	非 数*	$X\div Y$	±0	Y
±∞	±∞*	±∞	±∞	非 数*	Y
非 数	X	X	X	X	非 数

* をつけた場合には, 不合理な演算という信号 (わりこみ) がでる.

5. 検討事項

前記のような仕様に対して, Kahan 自身もいくつかの検討事項を示している. たとえば補数表示や十進法の体系である.

そのほか次のような点が指摘されている (1),2)):

1° 不正規化数による除法は, 0による除法に準じて不合理な演算としているが, 答が正しく表現できる範囲にあれば, 実行するようにしてはどうか?

2° 0*非数, 非数/∞ を非数としているが, 0とすべきか?——これらを0とすれば, プログラムが簡易化される実例もあげられている. しかし, そうしてしまうと, 不合理な演算が消えてしまい, かえって深刻な誤りを生ずる危険性も考えられる. 0にも, あらゆ

るものに掛けて0としてしまう「絶対的な0」と，下へあふれたエプシロン的な0とを区別しようという考えもあるが，そうしても五十歩百歩かもしれない．

3° 負数 $-A$ の平方根を $-\sqrt{A}$ とするのが原案だが，これを不合理な演算として非数とすべきか？

4° 多倍長数の完全な商および剰余が必要か？——整数論のためには望まれるが，そのためにはむしろ無制限多倍長整数の演算用ソフトウェアが作りやすい体系を考えるべきではないか？

5° 少しばかりの下へのあふれは不正規化数の形で保持するが，丸め誤差ばかりとなったときには，注意信号をだす．この限界の判定をどうするか？ Coonen は一つの案をだしているが，それでよいか？ 代案はどうか？

6° 射影モードと通例のアフィンモードの切換えに対して，$+0, -0$，符号なしの0（∞も同様）の3種併用のほうがよくないか？

二進絶対値表現を採用した根拠らしいものは前に述べたが，数値解析の研究用には，純十進の計算機がほしくなる．千進法の形にすれば，ビットのむだは大幅に改善できるし，早い二進十進変換法も考えられており，実験用に試作してみる価値があるかもしれない．

なお言語については，何も述べていないが，このような意図を生かした計算機の金物ができた場合，それを生かすために当然それにふさわしい言語体系が必要になる．そのことは Kahan 自身も，言語の専門家と協同して，次の研究課題であることを強調していた．それは恐らく RORTRAN を基調とした言語になるだろう．ただし従来の言語が共通性・互換性を重んじて，機械の内部表現を無視していたのに対して，必要ならば機械の内部表現にたち入った操作（たとえば指数部をとりだすなど）も可能なことが望まれるし，前述の $\sum_{i=1}^{n} a_i b_i$ の計算を SIGMA $(A(I), B(I), N)$ とでも書けば，特に指定しない限り，中間は自動的に倍長演算でやってくれることが望まれる．

いま一つ，標準外部関数（指数関数，三角関数など）も加えることが考えられており，大型機では，マイクロプログラム方式による演算命令（たとえば，5)）を加えることも，当然検討に値する．

6. 批 判

以上が Kahan らの提案の大要であるが，以下に筆者個人並びに二，三の方々から伺った批判的意見を記す．

まず単長型（32ビット）が必要かという問題がある．科学技術計算専用機ならば，CDC 系の多くの計算機がそうであるように，倍長型（64ビット）を標準語長として，必要ならその倍長が可能なようにすれば十分ではないか？

これに対する Kahan の答は，彼らのねらっている計算機が，超高速機でなく，むしろ各研究室に1台という小型ミニコンであり，既製のプログラムとの互換性を重視した，というのであろう．中大型機になれば，当然倍長ないしは四倍長演算の金物をも有し，制御カード1枚のさしかえで，単長，倍長が切り換えられるという方式が望まれる．そのような場合には，むしろ倍長が標準であり，いわゆる単長は，低精度で簡易計算をする，むしろ半長語と考えられるだろう．

じつのところ筆者は，まだ48ビット語に郷愁を感ずる．単長拡大型を48ビットにして標準の語長とする構想は，2の累乗でないための支障が大きいのだろうか？ もっとも速度が向上すれば，64ビットを標準とするのでも文句はない．ただしその場合，指数部が最低11ビット，できれば15ビットほしい．

第2の疑問は，不正規化数により下へのあふれを防ぐのが，どれほど有効かという問題である．初期の計算機でこの案が随分論ぜられたが，けっきょく労多くして功少なしとして捨てられた感がある．ただし当時は一般に非正規化数を全般的に許す方針であり，この場合のように，普通にはありえない（あるとすればむだになっている）$e=0, f\neq 0$ の数に特別な意味をもたせるという構想ではなかった．このような情報に意味をもたせ，その処理に大して費用がかからないものであるのならば，当面「ないよりまし」なことはたしかだから，有効性の実例が十分にないとしても，頭から無用の長物視するのはやめて，つけておき，有効な活用策を考えよう，というのが Kahan の哲学のように思われる．そうであるならば，筆者としても，そのゆき方に賛成である．

第3には，$\pm\infty$，非数などのデータが読みだされたときの処理である．この場合，必要に応じてとめられる注意信号がでるように設計されていれば，ソフトウェア作成者の手間は大幅に減るし，無制限倍長演算などのプログラム作成にも活用できるであろう．ただしこれは計算機設計段階の問題であって，浮動小数点体系自体の問題ではないかもしれない．

その他別の体系として，たとえば平野菩保の桁落ち分を情報としてつけるような案（たとえば 6)）など，

いろいろの変形が考えられる．ただこの桁落ち分の情報は，ある種の数値解析の実験には有用であっても，実用上の計算には，むしろ「足手まとい」になる可能性さえあるように思われる．

初めにも述べたとおり，この種の案は，紙上の提案のみでは意味が薄く，計算機として実現されなければならない．Kahan が強調するように，この計算機の仕様は，見掛けほど繁雑でなく，数値計算の品質保証は，これまでの計算機よりもずっと精密かつ容易にできそうであり，大いに期待している．

現在の電子計算機科学は大発展し，専門化が著るしい．しかし，本当のおもしろさは，やはり自分の研究目的にふさわしい計算機体系を（硬件も軟件も含めて）手作りするところにきわまるのかもしれない．オーストラリアの計算群論の専門家達が，有限群計算の専用機の提案をしているのもその一例である．これはもちろんなまやさしい仕事でないし，また金のもうかる話でもない．しかし規模の経済，大集中化の神話がすでに崩壊した現在，当然試みられてよい方向である．この提案の紹介の労をとったのも，ひとえにそれを述べたかったからである．

参考文献

1) Kahan, W.: New Standard Floating Point Arithmetic, Technical Report, (Nov. 1978).
2) Coonen, J. T.: Specification for a Proposed Standard for Floating Point Arithmetic, Technical Report, Revised Memorandum, (Dec. 1978).
3) Foskick, L. D. edi.: Performance Evaluation of Numerical Software, Proc. of IFIP TC-2, W. G. 2.5 Working Conference, North Holland, (1979).
4) Wilkinson, J. H.: Rounding Errors in Algebraic Processes. Her Majesty's Stationary Office, London, 1963——日本語訳，一松・四条訳，丸め誤差解析，培風館，(1974).
5) 一松　信：初等関数の数値計算，新曜社 (1974).
6) 平野菅保：区間代数データと誤差消失，数理解析研究所研究集会，1979年3月23日；同講究録に発表予定.

(昭和54年6月4日受付)

追　　加

Vol. 20, No. 9, pp. 793—797 掲載：一松信解説「新標準浮動小数点体系の提案」に追加コメントがあります.

プレプリントの形で引用した Kahan および Coonen らの論文が，この提案に対する批判的意見とともに，下記に公表された.

The Proposed IEEE Floating-Point Standard, SIGNUM Newsletter Secial Issue, 1979 Oct. 全 32ページ.

相当に白熱した紙上討論が展開されているが，全体としての印象は否定的な意見が強いようである.