ホームページの広場 第39回

浮動小数点数

放射線腫瘍学教室　非常勤講師
（関西福祉科学大学 保健医療学部　教授）　上杉 康夫

1. コンピュータでの数値格納

実数は **可算無限集合** でないため、有限の量を正確に表すことしかできないコンピュータで、実数を正確に表現することは基本的には不可能です。「実数型」という語がありますが、実態は、数値表現の難しさが甘く考えられていた1960年代のプログラミング言語などで見られた歴史的表現、ないし、現在で言うところの浮動小数点数型に対しての実数型と言う命名や、さらには浮動小数点数型とは言い難い場合などにも実数型という名称を使っている場合もあります※1。現在コンピュータの数値格納においては、浮動小数点方式が多用されています。**浮動小数点数**（ふどうしょうすうてんすう、英: floating point number）は、浮動小数点方式と呼ばれる方式によって格納された数のことです※2。

今回は浮動小数点数について記載いたします。また本稿で数値Rの2進法による位取り記数法を $R_{(2)}$ と記載し、特に10進法による位取り記数法を強調する場合は $R_{(10)}$ と記載することにします※3。例えば、「10進法で $13_{(10)}$ は、2進法で $1101_{(2)}$ です。」と記載します。

2. 固定小数点数

小数の表現方法として、浮動小数点数の他に固定小数点数があります。固定小数点数とは数値を表現するときに、整数部と小数部のそれぞれの桁数が決まっている表現方法で、小数点の位置が変わらない表現方法です（図1）※4。

図1：固定小数点数※4

このように固定小数点数は、あらかじめ小数点が何桁目にくるかを決めておく方式です。固定小数点数の特徴として、固定小数点数は、浮動小数点数に比べて表現できる値の範囲は狭いが、高速に処理できるという特徴があります。その一方で表現できる数値の範囲が限られてしまいます。例えば2進数の「$101.01_{(2)}$」を8bit（ビット）の固定小数点数で表現するときに、4桁目の後を小数点の場所と決めると、次のようになります。

0101.0100
（4桁目に小数点が固定されています）

この場合では、最大で「$1111.1111_{(2)}$」までの数字しか表現できなくなり、正負の符号なしで、正の数とすると、10進数で「15.9375」までしか表現できません※5。

その一方で、固定小数点数は小数という言葉がついているものの、コンピュータで整数をあらわす際に多用されています。固定小数点数で、整数を表現するには「小数点を最後」にします。例えば「$10010001_{(2)}$（$=145_{(10)}$）」のような8桁のビット列に対して、最後を小数点以下と決めれば「10010001.」と表現できます※6。

3. 浮動小数点数

浮動小数点数は、現在コンピュータの数値表現において、浮動小数点数が多用されています※2。浮動小数点数は、小数点の位置を一定とせず、数値の正負を示す符号部、有効数字を表示する仮数部、小数点の位置を指定する指数部を用いて表わします。例えば数値aは正負の符号と、

仮数部Mさらにrを底とする冪乗Eの指数部を使って※4

数値 a = (S:符号 + または −) M × r^E

と表します。

2進数での符号部は、正なら 0 で、負なら 1 と表すことにします。また指数部の底は2進数ですから2とします。

10進数 −46.5(10) は
2進数で −101110.1(2) です※7。
−101110.1(2) の符号はマイナスなので、符号部は 1 です。
101110.1(2) の部分は、必ず先頭を「1.」とすると決め、2の冪乗を利用して
101110.1(2) = 1.011101(2) × 2^5(10) と変形し、小数点を移動させ仮数部と指数部を表記します。

そうしますと−46.5(10) は
符号部Sは 1
仮数部Mは1.011101
指数部r^Eは2^5で、冪乗が5乗ですので
101(2)（=5(10)）
となります。

これで、1つの数を **符号部 指数部 仮数部** という3つの情報に分けて、そのいずれも 1 と 0 だけの2進法の情報で示すことが可能となりました※8、9、10。

先程、「小数点を移動させ仮数部と指数部を表記します。」と記載いたしましたが、数値によって小数点位置を左右にずらすことになるので、「浮動小数点」の語源となっています※8。

4. IEEE 754

1980年代頃まではこの浮動小数点の方式には標準がなく、コンピューターメーカーがおのおの勝手にフォーマットを決めていましたが、1985年にIEEE（Institute of Electrical and Electronics Engineer：《米国》電気電子学会）がIEEE 754と呼ばれる標準規格（IEEE 754-1985）を定めて、各社もこれに順ずる形で実装するようになりました。したがって最近はよほど特殊な要求でない限り、浮動小数点のフォーマットは共通です。そのIEEE 754ですが、その後2008年にも改定されIEEE 754-2008-Standard for Floating-Point Arithmetic（以後IEEE 754-2008）が公開されました。昨今のCPUはすべてこのIEEE754-2008に準じて数値格納をしていると考えていいと思われます。IEEE754-2008では、数字の表現について表1の8つの形式名を持つフォーマットを定めています※11、12、13、14。

5. IEEE 754-2008の浮動小数点数の実際

フォーマットIEEE 754-2008のBinary32（単精度）を例として記載します。IEEE 754-2008のフォーマット（表1）では桁・ビット数（p）23+1と記載されています。Binary32の内部フォーマットは図2の様になります※15。符号部（S）は1ビット、指数部（E）は8ビット、仮数部（M）は23ビットと定義されています。IEEE 754-2008のフォーマット（表1）の桁・ビット数（p）が仮数部（M）に相当します。

形式名	一般名	基数 (b)	桁・ビット数 (p)	指数最小値 (emin)	指数最大値 (emax)	備考	十進換算桁数	十進換算 emax
binary16	半精度	2	10+1	−14	15	交換形式であって、基本形式ではない	3.31	4.51
binary32	単精度	2	23+1	−126	127		7.22	38.23
binary64	倍精度	2	52+1	−1022	1023		15.95	307.95
binary128	四倍精度	2	112+1	−16382	16383		34.02	4931.77
binary256	八倍精度	2	236+1	−262142	262143	交換形式であって、基本形式ではない	71.34	78913.2
decimal32	十進単精度	10	7	−95	96	交換形式であって、基本形式ではない	7	96
decimal64	十進倍精度	10	16	−383	384		16	384
decimal128	十進四倍精度	10	34	−6143	6144		34	6144

表1：IEEE 754-2008のフォーマット※11、12

仮数部（M）は図2では23ビットとされているのに、IEEE 754-2008のフォーマット（表1）では桁・ビット数（p）が23+1と記載されているのは、後述するHidden Bitの仕組みを組み込んでいるためです。

A) 仮数部のHidden Bit

仮数部のHidden Bitの設定により精度につき1ビットを稼ぐことが可能です。このHidden Bitについて、Binary32を例として記載します。

先程の10進数　$-46.5_{(10)}$ を例にして説明いたします。

10進数 $-46.5_{(10)}$ は
2進数で $-101110.1_{(2)}$ です。
$-101110.1_{(2)}$ の符号はマイナスなので、符号部は 1 です。
次に仮数部にあたる$101110.1_{(2)}$ の部分の表記を検討しますと指数部の冪乗との組み合わせで

$101110.1_{(2)} = 101110.1_{(2)} \times 2^{0}{}_{(10)}$
$101110.1_{(2)} = 10111.01_{(2)} \times 2^{1}{}_{(10)}$
$101110.1_{(2)} = 1011.101_{(2)} \times 2^{2}{}_{(10)}$
$101110.1_{(2)} = 101.1101_{(2)} \times 2^{3}{}_{(10)}$
$101110.1_{(2)} = 10.11101_{(2)} \times 2^{4}{}_{(10)}$
$101110.1_{(2)} = 1.011101_{(2)} \times 2^{5}{}_{(10)}$
$101110.1_{(2)} = 0.1011101_{(2)} \times 2^{6}{}_{(10)}$
$101110.1_{(2)} = 0.01011101_{(2)} \times 2^{7}{}_{(10)}$

と様々な表記が可能です。

ここで仮数部Mは必ず先頭を「1.」として表記すると決めますと、

$101110.1_{(2)} = 1.011101_{(2)} \times 2^{5}{}_{(10)}$

の記載方法が一意的に定まります。

上記のようにすれば、どのような数値であっても必ず先頭が「1.」で始まる2進法の小数による仮数部と2の冪乗による指数部の2つを使って数値格納が可能となります。

さて、必ず先頭を「1.」として仮数部を格納するとしますと、全ての数値の仮数部で小数点の左側に"1"があることになります。従って全ての数値で仮数部の先頭の「1.」は暗黙（hidden）に存在することになり、仮数部の数値に対しては、小数点以下を格納しておけば十分と言うことになります。格納された数値を計算に使用する際やディスプレーに表示する際といった時に、仮数部として格納された数値の先頭に「1.」を加えればよいと言えます。すなわち数値の最上位ビットは必ず"1"になるので、データフォーマット上ではこの"1"をHidden Bitとして省略し、1ビット分の精度を稼ぐことが可能となっています（表2）※16。

仮数部（M）は23ビットと定義される一方で、IEEE 754-2008のフォーマット（表1）の桁・ビット数（p）が23+1と記載されているのは上記のHidden Bitの組み込みによるためです。

このHidden Bitの効果が大きいのは、23bitでは10進相当で6.923桁分で、つまり有効数字が7桁になりません。ところがHidden Bitにより1ビット分の精度を稼ぎ実質24bitにすると7.224桁分で、有効数字が7桁になります※15。

Binary32での演算		
仮数部の表現	内部での扱い方	値
00000000000000000000000	10000000000000000000000	1.0
10000000000000000000000	11000000000000000000000	1.5
11000000000000000000000	11100000000000000000000	1.75
11100000000000000000000	11110000000000000000000	1.875
：	：	：
11111111111111111111111	11111111111111111111111	1.99999988079071

表2：仮数部のBinary32での処理例※15
仮数部は23ビットであるが、Hidden Bit により1ビット分の精度を稼ぎ、実質24ビットとなっている。

B) 指数部のexcess

指数部の8ビットはexcess（エクセス）※17もしくはbias（バイアス）、下駄履き表現、オフセット・バイナリ（offset binary）と呼ばれる操作を行った形式が使用されています。

IEEE 754-2008 Binary32（単精度）では、指数部の8ビット（=256(10)=128(10)×2(10)）を格納する際にexcess 127（127=128−1=256/2−1）の操作を行います。

このexcess 127は指数部の実際の値に、固定値（127）を加算したものです（表1、図2）。このような表現にしているのは浮動小数点数同士の大小比較を容易にするためです。指数部は大きな値も小さな値も表せるようにしていますが、1より小さな数値では負の値になります。これを単に2の補数で表すと、全体の符号とは別に指数部も符号を持つことになり、単純な数値の大小比較が困難となります。そのため、指数部に固定値（127）を加算し常に正の値となるような形式で格納することにします。この操作をexcessと言います。

Binary32（単精度）では−126 ～ +127に127を加えて、1 ～ 254としています。この表現により「指数が正の数」「指数が1の数」「指数が負の数」を、この順に自然に並べることができます。浮動小数点数を解釈するときは、excessを減算して実際の指数を求めます※18、19。

指数部の8ビットが00000000（0(10)）の場合は数字そのものが0、もしくは非正規数（0ではないけど、限りなく0に近い小さな数字）を意味し、11111111（255(10)）の場合は無限大もしくはNaN（Not a Number：数字ではない）を意味します※15。

C) Binary32で格納可能な数値

最大は

$2^{127}=1.701411834604692\times10^{38}=1.701411834604692\times$（100澗）

澗（かん）は10^{36}を示します（表3）。

最小は

$2^{-127}=1.175494350822288\times10^{-38}$

です。

最小の数値に最も近い命数は10^{-24}としての涅槃寂静（ねはんじゃくじょう）が記載されています。またディラック定数（Dirac's constant）$\hbar=1.054571817...\times10^{-34}$J・Sが最小の数値に近い数値として挙がります※20、21、22、23。

6. Excelへの影響

Office製品のトラブルシューティング / Excelの所に「浮動小数点演算が Excel で不正確な結果をもたらす可能性がある」と掲載されています。Excel製品の内 Excel 2010、Excel 2013、Excel for Microsoft 365、Microsoft Excel for Mac 2011、Excel for Mac for Microsoft 365はIEEE 754仕様に基づいて設計されました。これらのExcel は、倍精度（表1参照）を使用して数値を格納しています。

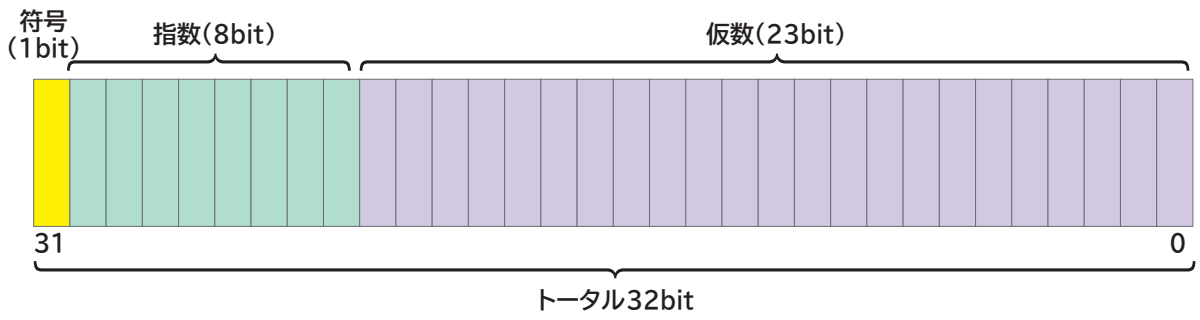

図2：Binary32（単精度）の内部フォーマット※15
十進換算の桁数は $p \times \log_{10} b$ で得られる十進での桁数の近似値である。
十進換算の emax は $emax \times \log 10\ b$ で得られる十進での指数最大値である。

Excel は$1.79769313486232\times10^{308}$ から$2.2250738585072\times10^{-308}$ までの数値を格納できますが、精度は15桁で格納します。この制限は、Excelの制限ではなくIEEE 754仕様に厳密に従ったことによるIEEE 754の制限によるものです※24。

具体例が説明されていましたので、そのうちのいくつかを記載いたします。

A) 非常に大きな数値を使用する例

新しいブックに次の情報を入力します。

A1: 1.2E+200

B1: 1E+100

C1: =A1+B1

セル C1 の結果の値は、セル A1 と同じ値である 1.2E+200 になります。実際、IF(A1=C1) などの IF 関数を使用してセル A1 と C1 を比較すると、結果は TRUE になります（図3）。これは、有効桁数15桁の精度のみ

図3：Excelで非常に大きな数値を使用する例

数詞	読み	数	10000^m	一（いち）	十（じゅう）	百（ひゃく）	千（せん）	補足
				10^0	10^1	10^2	10^3	
万	まん	10^4	10000^1	一万 10^4	十万 10^5	百万 10^6	千万 10^7	
億	おく	10^8	10000^2	一億 10^8	十億 10^9	百億 10^{10}	千億 10^{11}	
兆	ちょう	10^{12}	10000^3	一兆 10^{12}	十兆 10^{13}	百兆 10^{14}	千兆 10^{15}	
京	けい	10^{16}	10000^4	一京 10^{16}	十京 10^{17}	百京 10^{18}	千京 10^{19}	（きょう）
垓	がい	10^{20}	10000^5	一垓 10^{20}	十垓 10^{21}	百垓 10^{22}	千垓 10^{23}	
秭	じょ	10^{24}	10000^6	一秭 10^{24}	十秭 10^{25}	百秭 10^{26}	千秭 10^{27}	杼（し）
穣	じょう	10^{28}	10000^7	一穣 10^{28}	十穣 10^{29}	百穣 10^{30}	千穣 10^{31}	
溝	こう	10^{32}	10000^8	一溝 10^{32}	十溝 10^{33}	百溝 10^{34}	千溝 10^{35}	
澗	かん	10^{36}	10000^9	一澗 10^{36}	十澗 10^{37}	百澗 10^{38}	千澗 10^{39}	
正	せい	10^{40}	10000^{10}	一正 10^{40}	十正 10^{41}	百正 10^{42}	千正 10^{43}	
載	さい	10^{44}	10000^{11}	一載 10^{44}	十載 10^{45}	百載 10^{46}	千載 10^{47}	
極	ごく	10^{48}	10000^{12}	一極 10^{48}	十極 10^{49}	百極 10^{50}	千極 10^{51}	
恒河沙	ごうがしゃ	10^{52}	10000^{13}	一恒河沙 10^{52}	十恒河沙 10^{53}	百恒河沙 10^{54}	千恒河沙 10^{55}	
阿僧祇	あそうぎ	10^{56}	10000^{14}	一阿僧祇 10^{56}	十阿僧祇 10^{57}	百阿僧祇 10^{58}	千阿僧祇 10^{59}	
那由他	なゆた	10^{60}	10000^{15}	一那由他 10^{60}	十那由他 10^{61}	百那由他 10^{62}	千那由他 10^{63}	
不可思議	ふかしぎ	10^{64}	10000^{16}	一不可思議 10^{64}	十不可思議 10^{65}	百不可思議 10^{66}	千不可思議 10^{67}	
無量大数	むりょうたいすう	10^{68}	10000^{17}	一無量大数 10^{68}	十無量大数 10^{69}	百無量大数 10^{70}	千無量大数 10^{71}	

表3：塵劫記（寛永11年版）での命数（日本の現行方式）※20

を格納するという IEEE 仕様が原因です。上記の計算を格納するには、Excel で少なくとも 100 桁の精度が必要です。

B）非常に小さい数値を使用する例

新しいブックに次の情報を入力します。

A1: 0.000123456789012345

B1: 1

C1: =A1+B1

セル C1 の結果の値は、1.000123456789012345 ではなく 1.00012345678901 になります（図4）。これは、有効桁数15桁の精度のみを格納するという IEEE 仕様が原因です。上記の計算を格納するには、Excelで少なくとも19桁の精度が必要です。

C）10進法表記で循環小数ではない小数が2進法表記では循環小数となる例

2進形式の浮動小数点数の格納に影響を与えるもう一つの問題は、10進表記で循環小数ではない小数が2進表記では循環小数となる点です。

この最も一般的な例は、数値 $0.1_{(10)}$ です。この数値は10進表記で循環小数ではありませんが、2進法表記では循環小数となります。

$0.1_{(10)}=$

$0001100110011001100110011\cdots_{(2)}$

数値が10進表記で循環小数ではない小数も2進表記ではその多くが循環小数になり有限桁では正確に表せません（表4）※25。

IEEE 754仕様では、仮数に格納できる値が格納され、残りは切り捨てられます。このため、2進表記で循環小数となる場合は格納時に誤差が発生します。

	A	B	C
1	0.000123456789012345	1	1.00012345678901

図4：Excelで非常に小さい数値を使用する例

このことをExcelで観てみます。Microsoft Visual Basic for Applications での次の例について考えてみます。

```
VB (Visual Basic)
  Sub Main ()
    MySum = 0
    For I% = 1 To 10000
      MySum = MySum + 0.0001
    Next I%
    Debug.Print MySum
  End Sub
```

上記をWindows 8.1 Pro上のExcel 2013で実行しても、$1_{(10)}$ は出力されませんでした。数値$0.0001_{(10)}$ を$10000_{(10)}$ 回加算するのですから$1_{(10)}$ になるはずですが、実行を試みたところ$0.9999999999999906_{(10)}$ が出力されました。一般的な小数といえる$0.0001_{(10)}$ でさえ、2 進数で正確に表すことはできません。数値$0.0001_{(10)}$ は、104ビットの周期を持って繰り返す2進の循環小数

$0.0000000000000011010001101101110001011101011000111000100001100101101\cdots_{(2)}$

となるためです。

十進表現	二進表現
0.1	0.000110011001100…
0.2	0.001100110011001…
0.3	0.010011001100110…
0.4	0.011001100110011…
0.5	0.1
0.6	0.100110011001100…
0.7	0.101100110011001…
0.8	0.110011001100110…
0.9	0.111001100110011…

表4：小数の10進表記と2進表記※25

本例は仮数部に2進数として格納するときに切捨てられたことによる誤差が、合計することにより顕著となっています。

7. 10進数ベースの浮動小数点フォーマット

米軍のパトリオット(または ペトリオット：MIM-104 Patriot)地対空ミサイル※26が、湾岸戦争当時の1991年2月25日にイラク軍のスカッドミサイルの迎撃に失敗し、サウジアラビアの米国軍兵舎がスカッドミサイルにより攻撃を受け破壊されました※27。

この原因は、パトリオットの制御に利用していたプログラムが、内部で24bitのカウンターを利用して時間を測定していたのですが、2進/10進の変換の誤差が蓄積した結果、タイミングが0.34秒ずれたことでした※15。

原因がわかったことで、システムを再起動して誤差の蓄積を解消させることを行い現場での対処が可能になりましたが、煩雑な作業となりました。根本的には2進数をベースにするとどうしても誤差を避けられないということで、こうした用途向けに10進数ベースの浮動小数点フォーマットとしてdecimal32(十進単精度)、decimal64(十進倍精度)、decimal128(十進四倍精度)が追加されることになりました(表1)※15。

今回は、浮動小数点数について記載いたしました。

参考文献

※1：実数型 - Wikipedia
https://ja.wikipedia.org/wiki/実数型

※2：浮動小数点数 – Wikipedia
https://ja.wikipedia.org/wiki/浮動小数点数

※3：位取り記数法 – Wikipedia
https://ja.wikipedia.org/wiki/位取り記数法

※4：>it-shikaku.jp - 基礎理論 - 1.基礎理論 - 1.離散数学 - 2.数値の表
http://www.it-shikaku.jp/top30.php?hidari=01-01-02.php&migi=km01-01.php

※5：固定小数点数と浮動小数点数の違いを調べよう！｜ITの学び
https://itmanabi.com/fixed-floating/

※6：固定小数点数とは - ITを分かりやすく解説
https://medium-company.com/固定小数点数/

※7：小数点つき10進数を2進数に変換する方法とツール - 具体例で学ぶ数学
https://mathwords.net/syosuu2sin

※8：基本情報でわかる 浮動小数点「3つの情報で1つの数を表す仕組みを知れば、浮動小数点数がわかる
https://www.seplus.jp/dokushuzemi/ec/fe/fenavi/mastering_tech/float/

※9：ieee
https://wwws.kobe-c.ac.jp/deguchi/sc180/calc/ieee.html

※10：ieee2
https://wwws.kobe-c.ac.jp/deguchi/sc180/calc/ieee2.html

※11：IEEE 754 - Wikipedia
https://en.wikipedia.org/wiki/IEEE_754

※12：IEEE 754 - Wikipedia
https://ja.wikipedia.org/wiki/IEEE_754

※13：754-2008 - IEEE Standard for Floating-Point Arithmetic｜IEEE Standard｜IEEE Xplore
https://ieeexplore.ieee.org/document/4610935

※14：ASCII.jp：いまさら聞けないIT用語集　浮動小数点演算の単精度と倍精度って？ (1/3)
https://ascii.jp/elem/000/001/713/1713959/

※15：ASCII.jp：いまさら聞けないIT用語集　浮動小数点演算の単精度と倍精度って？ (2/3)
https://ascii.jp/elem/000/001/713/1713959/2/#eid1713964

※16：コンピュータアーキテクチャの話(93) ヒドン(Hidden)ビットで精度を1ビットを稼ぐ｜TECH+(テックプラス)
https://news.mynavi.jp/techplus/article/architecture-93/

※17：Offset binary - Wikipedia
https://en.wikipedia.org/wiki/Offset_binary#Excess-127

※18：符号付数値表現 - Wikipedia
https://ja.wikipedia.org/wiki/符号付数値表現

※19：バイアス - Wikipedia
https://ja.wikipedia.org/wiki/バイアス

※20：命数法 - Wikipedia
https://ja.wikipedia.org/wiki/命数法

※21：浮動小数点数 - ゲーム制作に使う数学を学習しよう #4
https://www.youtube.com/watch?v=u7C7KdY8-bc

※22：数字の単位一覧(少数も)|億・兆・京・垓・秭から無量大数より大きい数字まで|年収ガイド
https://nenshuu.net/magazine/pages.php?pages_id=269

※23：ディラック定数 - Wikipedia
https://ja.wikipedia.org/wiki/ディラック定数

※24：浮動小数点演算が Excel で不正確な結果をもたらす可能性がある - Office
https://docs.microsoft.com/ja-jp/office/troubleshoot/excel/floating-point-arithmetic-inaccurate-result

※25：1より小さい数を含む二進数表現
https://www.seiai.ed.jp/sys/text/csc/ch12/c12a300.html

※26：パトリオットミサイル – Wikipedia
https://ja.wikipedia.org/wiki/パトリオットミサイル

※27：湾岸戦争時のペトリオットミサイルが撃墜失敗 - takumi296's diary
https://takumi296.hatenablog.com/entry/2013/02/25/033525