コンピュータと教育　27－3
（１９９３．５．２８）

統合数式エディタの開発と数学教育への応用

内山　靖文
工学院大学　大学院　電気工学専攻

三好　和憲
工学院大学　電子工学科

パーソナルコンピュータ上で、我々が日常使用する表現で数式を表示し、編集できる数式エディタを装備し、入力された数式に対して評価、可視化を行うシステムを開発した。本システムでは、初心者向けに全ての機能を数式エディタに統合し、メニューで呼び出せるようにしている。また、機能をライブラリ化し、ＣＡＩ教材等を作成する際に容易に利用できるようになっている。この報告では本システムの概要と数学教育への応用について述べる。

Development of integrated fomula editor and application for mathematical education

Yasufumi Uchiyama
Graduate school of electrical engineering, Kogakuin Univ.

Kazunori Miyoshi
Department of electronics, Kogakuin Univ.

A formula editor which can show and edit formulae in usual textbook format is developed. This system can evalute and visualize formulea. All functions are integrated on the formula editor, and can be called from a menu for beginners. And because these functions consist libraries, these are useful for development of applications for example CAI system. The outline of this system and application for mathematical education are reported.

１．はじめに

現在、パーソナルコンピュータ上で動作する数式処理システムは、ＲＥＤＵＣＥ、Ｍａｔｈｅｍａｔｉｃａなどいくつか存在する。これらのシステムは、もともとが巨大メモリを備えた大型計算機上において、研究者が各自の研究のために開発したものであって、初心者にとっては、あまりにも操作が難解で、とても手軽に使えるとは思えない。一方、教育目的としてこれらのシステムを使用する場合、ほとんどのユーザーはコンピュータを使用したことがない、あるいはコンピュータにはあまりなじみがないといった初心者であり、教育という目的に至るまでにそのシステム自体を理解しなければならないという問題が生じると思われる。

そこで、これらの問題点を補い、教育などの目的でコンピュータ初心者が数式処理システムを使用するための手助けとなる事を目標とし、以下のようなシステムを開発した。

① 数式を２次元表示し、エディットできる数式エディタ
② エディタにより入力された数式をグラフ化する
③ 多倍長演算による簡単な演算機能
④ 数式の構造を構造木を用いて視覚化する
⑤ 数式をポーランド、逆ポーランド記法に変換して表示

統合数式エディタ「マッスル」はこれらの機能を統合し、メニューにより数式エディタから自由に呼び出すことができる。「マッスル」の開発コンセプトとして最も重視したのは、初心者でも簡単に扱えるよう、次のようにマンマシンインターフェイスに気を配ったことである。

① マウスによる操作が可能
② 機能選択、ファイル選択などは全てメニュー形式による入力
③ パラメータなどの文字列入力のための専用のラインエディタを装備
④ グラフ描画時などのパラメータ設定は出来る限り自動化し、ユーザーの負担を減らす
⑤ ヘルプ機能を装備

また、「マッスル」に組み込まれている各機能はすべてライブラリとして供給されるので、ユーザー（アプリケーション開発者）がアプリケーションを開発する際に容易にその機能を利用できるようになっている。つまり、「マッスル」は既に与えられた実行ファイルの活用のみならず、ライブラリを用いてＣＡＩ教材等を制作する際の手助けにもなりうるのである。

２．「マッスル」の開発環境

「マッスル」はNEC PC-9801上でBorland International社のTurbo Pascal Var6.0aを用いて開発されている。Turbo Pascalは標準PASCALの機能に加え、オブジェクト指向プログラミングなど、ソフトウェアの生産性を向上するためのさまざまな拡張がなされている。「マッスル」は、これらのTurbo Pascalの拡張機能を利用しているので、ユーザーが「マッスル」のライブラリを利用してアプリケーションを開発する場合は必然的にTurbo Pascalを用いることになる。また、ライブラリのほとんどはユニット（Turbo Pascalのライブラリファイル）の中でオブジェクトとして定義されているため、アプリケーションに組み込む場合はオブジェクト指向プログラミングの知識が要求される。

3．「マッスル」のシステム構成

統合数式エディタ「マッスル」のシステムは以下のようなライブラリから構成され、これらを統合してコンパイル、リンクされて得られた実行ファイルを利用する方法と、ユーザー（アプリケーション開発者）が必要なライブラリを各自のアプリケーションに組み込んで利用する方法がある。

① １つの数式をエディット可能な数式エディタ「ＭＥ」
② 複数の数式をエディット可能な数式エディタ「ＭＥ２」
③ 文字列として与えられた数式をポインタを用いた木構造に変換するライブラリ
④ 数式を２次元グラフ化するライブラリ
⑤ 数式を３次元グラフ化するライブラリ
⑥ 浮動小数点多倍長演算ライブラリ
⑦ 数式を評価するライブラリ
⑧ 数式の構造を構造木として視覚化するライブラリ及び数式をポーランド記法、逆ポーランド記法に変換するライブラリ

図１は実行ファイルをそのまま使用する場合のシステムの構成図である。数式エディタで入力された数式データは、データファイルとして補助記憶装置に出力され、ＲＥＤＵＣＥなどの既存の数式処理システムに渡される。一方、「マッスル」の内蔵機能を使う場合は、数式データは数式木構造化ライブラリにより木構造に変換され、下位オブジェクトとして定義された各ライブラリでそれぞれの機能を実現する。

図２はユーザー（アプリケーション開発者）が「マッスル」で用意されたライブラリを各自のアプリケーションに組み込んで利用する場合のシステム構成である。各ライブラリでは機能ごとにオブジェクトとして定義されているので、ユーザーが簡単に利用、改造が可能となっている。

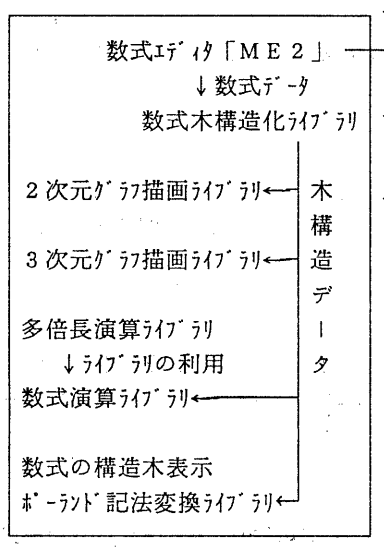

図１　「マッスル」のシステム構成（その１）

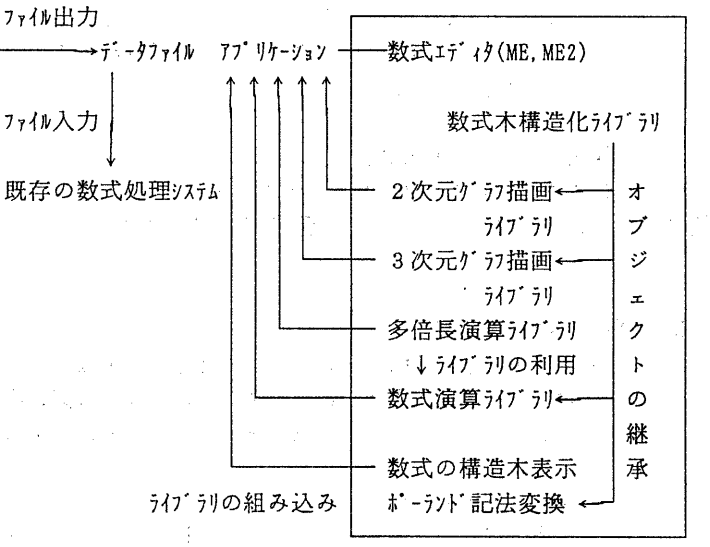

図２　「マッスル」のシステム構成（その２）

4．　数式エディタ

4．1　数式エディタ「ME」

「ME」は1行の数式をエディットできる数式エディタであり、次のような機能を持つ。

① 数式の2次元形式表示でのエディット

数式を2次元形式で表示し、テキストのラインエディタ感覚で自由にエディットできる。2次元形式表示とは、例えば分数や指数ならば

$$\frac{x^2+y^2}{x^2-y^2}$$

というように数式を2次元的に表示する方法である。「ME」はこの2次元形式表示での数式入力のための以下のような機能を持っている。

（1）分数入力
（2）指数入力
（3）ルート（n乗根）の入力

② エディタ終了時のエラーチェック機能

括弧が対応していない、演算子が二つ以上並んでいるなど、文法的な誤りがある場合にはエラー表示し、カーソルをエラーが発生した場所に自動的に移動して再エディットを促す。

③ エディット範囲指定機能

ユーザー（開発者）がライブラリを利用してアプリケーションに組み込む場合に、画面を分割してアプリケーションと共有する場合を考え、エディタの使用する範囲を1キャラクタ単位で指定できるようになっている。この機能により画面の上半分をエディタ、下半分を他のアプリケーションが用いる、ということも比較的簡単に実現できる。また、このようにエディット画面を狭くすると、数式が1画面では収まらなくなる恐れがあるので、はみ出す場合は自動的にスクロールすることで対応した。

4．2　数式エディタ「ME2」

数式エディタ「ME2」は「ME」オブジェクトのインスタンスを複数個起動、線形リスト構造で連結し、複数行の数式エディットを可能にしたものである。「ME2」は次のような機能を持っている。

① 複数行の数式のエディット

「ME」を複数個起動するので、2次元形式表示など、「ME」そのままの操作環境で複数行の数式のエディットが可能である。また、エディタの複数行化に伴い、行の削除や挿入、コピーなどのスクリーンエディタに必須であると思われる機能を用意した。

② ファイルへの入出力機能

内部形式数式データをファイルから読み込み、またはファイルに保存できる。また、出力形式数式データをファイルに出力することにより、数式処理システムへの数式データの受け渡しは、従来（「ME」）のTurbo Pascalの関数による受け渡しの他に、ファイルによる受け渡しが可能になった。これによって、ユーザー（アプリケーション開発者）が作成する数式処理システムの

他に、ＲＥＤＵＣＥなど既存の数式処理システムへの数式データの受け渡しが可能となった。

③ 出力式の設定機能

②のようにファイルの形で既存の数式処理システム数式データを渡し、何等かの処理をさせる場合、それぞれの数式処理システムによって数式の表現が異なる場合がある（例えば、べき乗なら＾や＊＊がよく使われる）。これに対処するため、特別な数式の出力表現をユーザーが設定できるようにした。

④ ヘルプ機能

「ＭＥ２」に内蔵されたページャーを起動し、別途用意されたヘルプファイルを読み込み、表示する機能を持つ。ヘルプファイルには簡単なキー操作などが記述されているが、これはＭＳ－ＤＯＳのテキストファイルの形で記録されているので、ユーザーが好みに合わせて、その内容をテキストエディタなどで容易に書き換え、追加ができるようになっている。

４．３　数式の内部データと出力データ

数式エディタ内では数式データは文字列として与えられている。数式データの内、１行で表現できる部分についてはそのままの表現となっているが、「ＭＥ」では数式の２次元表示を採用しているため、内部数式データの中にこれを認識するための専用のコードが必要となる。そこで、それを考慮して定義したものが表１のようなデータ表現である。

また、表１には内部データに対応した数式の出力データの表現も示した。ファイルへの出力、プログラム中の関数によるデータの受け渡しにはこの表現が用いられる。

表１　数式のデータ表現

	内部データ	出力データ
べき乗	ｘ＾ｙ」	ｘ＾（ｙ）
分数	＜ｘ｜ｙ＞	（ｘ）／（ｙ）
ｎ乗根	＃ｎ＄ｘ％ ＃＄ｘ％	ｒｏｏｔ（ｎ，ｘ） ｓｑｒｔ（ｘ）
括弧	（｛〔 ）｝〕	（ ）

内部データの例

$$\frac{x+y}{} \rightarrow \text{<x+y|a*b+<c|d>>}$$

$$a*b+\frac{c}{d}$$

｛（ａ＋ｂ）／ｃ｝$^{x+y}$ → {(a+b)/c}^x+y」

出力データの例

<x+y|a*b+<c|d>> → (x+y)/(a*b+(c)/(d))

{(a+b)/c}^x+y」 → ((a+b)/c)^(x+y)

５．数式木構造化ライブラリ

数式木構造化ライブラリは、図３のように与えられた数式をポインタの鎖を用いた木構造データに変換するものである。

```
{ 構造木へのポインタ }
TreePtr = ^BinTree;
{ 数式の構造木 }
BinTree = Record
  Node : String;
  Left, Right : TreePtr;
end;
```

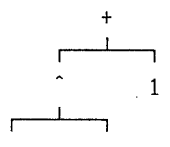

図3　構造木のデータ形式及び、その例として数式 x^2+1 を木構造化したもの

「マッスル」における数式木構造化ライブラリを中心にしたオブジェクト構成は以下のとおりである。各下位ライブラリは継承された数式木構造化ライブラリの機能を用いて数式を木構造化し、それをもとにそれぞれの機能を実現する。

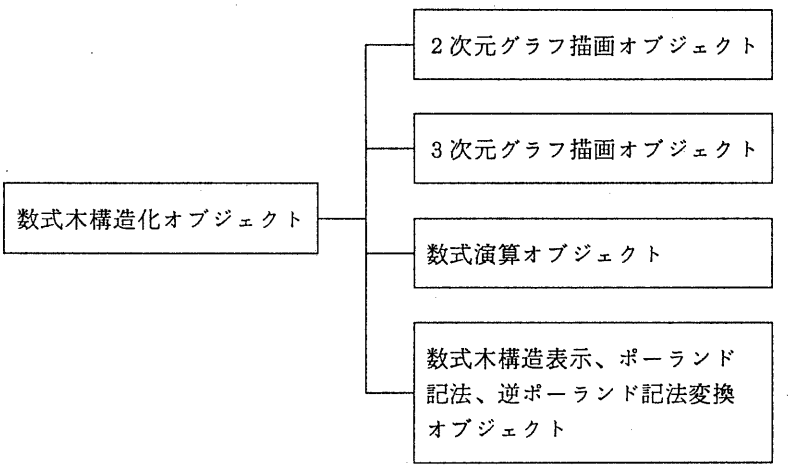

図4　数式木構造化ライブラリを中心としたオブジェクト構成

6．グラフ描画ライブラリ

2次元グラフ描画ライブラリ及び3次元グラフ描画ライブラリは与えられた数式（1変数関数，2変数関数）をグラフ化するものである。これらのライブラリではメニュー形式により対話的にパラメータを指定し、グラフを描くことが出来る。パラメータの設定は出来る限り自動化し（もちろん手動による設定も可能）、ユーザーの負担を減らしている。

6．1　グラフ描画ライブラリの機能

① 指定した数式のグラフを描く

② ループを用いてグラフを描く（2次元グラフ）

指定した数式の中に変数が2つ以上含まれている場合、主変数以外にもう一つ変数を指定し、その変数の初期値、終了値、ステップを設定することでＢＡＳＩＣのＦＯＲ～ＮＥＸＴ文のような感覚でくり返しグラフを描くことが出来る。例えば、y=a*sin(x)という関数が与えられて、主変数をxとし、aを初期値1、終了値3、ステップ1とすると、画面上には

```
y=1*sin(x)
y=2*sin(x)
```

y=3*sin(x)

の３種類のグラフが重ねて描かれる。

③ 計算結果の解析（２次元グラフ）

グラフ表示範囲内の任意の関数値が表示できる。

④ グラフの拡大表示（２次元グラフ）

グラフ表示範囲内の任意の場所を拡大表示できる。これは⑧の表示領域の設定でも可能であるが、こちらはボックスを表示して、視覚的に領域指定できる。

⑤ 変数の設定

主変数の設定をする。

⑥ 定数の設定

数式に２つまたは３つ以上の変数が含まれている場合は主変数以外は全て定数とみなす。ここではこの定数に対して値を設定する。

⑦ スケールの間隔設定（２次元グラフ）

グラフに表示されるスケールの間隔設定を行う。通常は自動設定となっているが、三角関数を含む場合などは自動設定では思うように設定できないことがあるので、手動で指定する。

⑧ 表示領域の設定

グラフの表示領域を設定する。２次元グラフの場合は、自動スケーリング機能を使用するかどうかユーザーに聞き、使用する場合は横軸の範囲のみ、使用しない場合は横軸、縦軸の範囲の設定を促す。

⑨ 視点の設定（３次元グラフ）

グラフを描く際の視点の再設定を行う。視点の設定は縦方向の角度と横方向の角度を指定することで行う。

⑩ 計算間隔の設定

グラフを描く際の計算間隔の設定を行う。計算間隔を大きく取ると描画速度は速くなるが、グラフが粗くなる。

⑪ 隠線処理の設定（３次元グラフ）

グラフ表示の際に、最大最小法による隠線処理を行うかどうかを設定する。隠線処理を行わない場合は多少描画速度が速くなる。

⑫ グラフの表示形態の設定（３次元グラフ）

グラフの表示形態（網目状、線状）を設定する。線状の場合、ｘ軸方向のみのラインで、網目状の場合、ｘ軸方向とｙ軸方向のラインで網目状になるように描く。

６．２　アプリケーションからの利用のための機能

グラフ描画ライブラリは次のような、ＣＡＩ教材などのアプリケーションから利用する際に便利な機能を持っている。

① グラフの表示位置設定

グラフの表示位置を設定する。２つの点の座標を指定し、これを結ぶ線を対角線とする領域にグラフが表示される。

② グラフ上に補助線を引く

表示された関数のグラフに重ねて直線分を引くことが出来る。グラフ上の任意の点から任意の点を結ぶ線分を引くメソッド、関数上の点から任意の点を結ぶ線分を引くメソッド、関数上の点から関数上の点を結ぶ線分を引くメソッドが用意されている。

③ グラフの色設定

グラフの表示色を設定することが出来る。効果的なＣＡＩ教材を作成する際は、全体の色のバランスも重要な要素であると考えて、任意に色設定できるようこの機能を追加した。

７．浮動小数点多倍長演算ライブラリ

浮動小数点多倍長演算ライブラリは、浮動小数点表現を用いて多倍長演算を行うライブラリである。このライブラリはすべてPascalで記述されているので、アセンブリ言語で記述されたライブラリに比べると、処理速度が遅く、実用的に使用するにはかなりのＣＰＵパワーが必要となる。そこで、この点に少しでも対処するため、本ライブラリでは、πやｅといったよく使う数値で、計算に時間のかかるものをあらかじめ計算しておき、内部で定数として保有している。また、演算の有効桁数をユーザーが設定出来るようにすることで、必要に応じてかなりの演算時間の短縮が出来るようになっている。

７．１　浮動小数点表現を用いた多倍長数

本ライブラリで用いた多倍長数のレコードは右のようになっている。Lenは仮数の長さ、Signは正負の符号、Expは指数部、MPは仮数部である。。

```
Type
  { 仮数部分のレコード }
  MPArray = Array[1..ArrayMax] of Byte;
  { 多倍長数のレコード }
  MPRec = Record
    Len  : Word;    { 仮数の長さ }
    Sign : Integer; { 符号 }
    Exp  : Integer; { 指数 }
    MP   : MPArray; { 仮数 }
  end;
```

８．数式演算ライブラリ

数式演算ライブラリは、数式木構造化ライブラリにより木構造化された数式を評価（演算）するものである。本ライブラリは前節で解説した浮動小数点多倍長演算ライブラリを用いているので、大きな有効桁数の演算が可能である。

８．１　数式の書式

本ライブラリで扱う数式の書式は、数式エディタの出力する数式と同様である。等号が含まれている数式を評価した場合、等号の左側の変数に右側の構造木が代入される。等号の左側が１つの変数でないときは、数式評価オブジェクト内のフィールド（変数）ErrorにTrueが代入され、処理を終える。変数への代入が値ではなく、構造木となっているのは、以下のように変数の値を変更した後でも、正しく数式を演算出来るようにするためである。

```
                                  { プログラム中のＸは多倍長数 }
E.SetExp('x=a+b'); E.Eval(X);     { ｘにａ＋ｂの構造木が代入される }
E.SetExp('a=10');  E.Eval(X);     { ａに１０が代入される }
E.SetExp('b=10');  E.Eval(X);     { ｂに１０が代入される }
E.SetExp('x');     E.Eval(X);     { ｘの値としてａ＋ｂが演算される }
Write('a+b=');     MPWrite(X);    { 計算結果の２０を表示 }
E.SetExp('a=20');  E.Eval(X);     { ａの値を２０に変更 }
E.SetExp('b=30');  E.Eval(X);     { ｂの値を３０に変更 }
E.SetExp('x');     E.Eval(X);     { ｘの値としてａ＋ｂが演算される }
WriteLn('a+b=');   MPWrite(X);    { 計算結果の５０を表示 }
```

９．ライブラリの利用例

「マッスル」のライブラリの利用例として以下にサンプルプログラムを示す。

```
{ ２次元グラフ描画ライブラリサンプルプログラム }
Program Test;
Uses Graph, MGraph; { ユニット使用の宣言 }
Var
  G: GraphObj;        { ２次元グラフ描画オブジェクトのインスタンス生成 }
begin
  G.Init('sin(x)'); { ｸﾞﾗﾌ化する数式を設定 }
  G.DrawGraph;        { 対話形式でﾊﾟﾗﾒｰﾀを入力し、グラフを描く }
end.
```

```
{ ２次元グラフ描画ライブラリサンプルプログラム２ }
Program Test2;
Uses Graph, MGraph;                    { ユニット使用の宣言 }
Var
  G: GraphObj;                         { ２次元グラフ描画オブジェクトのインスタンス生成 }
begin
  G.Init('sin(x)');                    { グラフ化する数式を設定 }
  G.SetViewX(-5, 5);                   { 横軸の表示領域を(-5, 5)に設定 }
  G.SetCrtArea(200, 100, 500, 350);    { 画面上のグラフの表示位置を設定 }
  G.SetEvalSpace(2);                   { 計算間隔を２ドット毎に設定 }
  G.SetScalingMode2(off);              { スケール間隔自動設定機能を使用しない }
  G.SetScaleStr('pi/2', '0.5');        { ｽｹｰﾙ間隔を横軸π/2, 縦軸0.5に設定 }
  G.DrawGraph2;                        { グラフを描く }
  G.Line(-5, 0.5, 5, 0.5);             { グラフに重ねて直線 y=0.5 を描く }
  G.LineToFunc(3.14/2, 3.14/2, 0);     { (3.14/2, sin(3.14/2))と(3.14/2, 0)を
                                         結ぶ線分をグラフに重ねて描く }
end.
```

１０．まとめ

統合数式エディタ「マッスル」は、初心者でも比較的容易に扱える多機能なシステムを目指して開発された。開発コンセプトの１つとして、マンマシンインターフェイスの充実という点を重視したために、初心者でも手軽に扱えるシステムが出来たと考える。特に、数式を２次元的に表示し、エディットできる数式エディタは、コンピュータ独特の１ラインによる表現ではなく、教科書等に掲載されているような表現でエディットできるので、コンピュータに馴染みのない初心者でも気軽に利用できると思われる。「マッスル」の第２の開発コンセプトとして、全ての機能をライブラリ化し、ユーザーの作成するアプリケーションから容易に利用できるようにした。例えば、グラフ表示などを用いたＣＡＩ教材などを作成する際に、「マッスル」のライブラリを利用すると、プログラミングの手間が省けて非常に便利である。

今後の課題としては以下のようなものを考えている。

・数式エディタの機能を充実させる。
　現段階では２次元表示でエディットできる数式の種類が限られている（中学生程度が扱う数式）ため、これを増やしていく。

・数式の評価に本格的な数式処理を適用する。
　現段階では数式処理を行いたい場合は、ファイルを介して外部の数式処理システムに渡す必要があるが、これを「マッスル」内部で行えるようにする。

・「マッスル」のライブラリを利用できるＣＡＩ教材のオーサリングシステムを開発し、プログラミングをすることなしに、数式エディタやグラフ描画などの「マッスル」の機能を利用したＣＡＩ教材を開発できるようにする。

参考文献

[1] 奥村 晴彦　Ｃ言語による最新アルゴリズム辞典　技術評論社
[2] 宮地 利雄　データ構造とプログラミング　昭晃堂
[3] R.BRENT　MP(A MULTIPLE-PRECISION ARITHMETIC PACKAGE)ソースリスト