数理解析研究所講究録
1169巻 2000年 97-101

Handwriting Interface for Computer Algebra System

—数式処理システムの手書き入力インターフェイス—

九州大学大学院　数理学研究科
岡村 博文
金堀 利洋
福岡教育大学　教育学部
叢 偉

1　はじめに

今日のネットワークやコンピュータの加速的な進歩に伴い、その用途はあらゆる分野に広がっている。数学においても例外ではなく、単なる計算だけに留まらず様々な可能性が考えられている。例えば、数式処理システムや数式のデータベースに接続した電子ボードなど数学の授業や講義の新しいスタイルを生み出すかもしれない。また、それに歩調を合わせるかのように、TEXやMathMLといった各種数式フォーマット、Mathematicaのような数式処理システム、各種ワープロに付属の数式入力機能など、コンピュータ上で数式を扱う環境も開発されて来ている。

しかし、現在、これらの環境を用いてもコンピュータ上で数式を扱うのはまだまだ簡単であるとは言えない([1][2])。先ず、数式の入力方法に関して言えば、Mathematicaや各種ワープロに付属の数式入力機能などは、メニューボタンなどによる入力操作を繰り返さなければならず、その煩わしさがスムーズな入力を妨げ、ユーザの思考の流れを中断させる可能性が高い。TEXやMathMLなどは数式をスムーズに入力できるが、使いこなせるようになるまで相当な修練が必要とされる。また、入力された数式の意味を一目見ただけで把握することが難しく、入力の誤りを数式を見ながら訂正することができないという事も挙げられるだろう。

また、これらのアプリケーション間の互換性の問題も挙げられる。TEXによる数式をMathMLに変換しネットワーク上で処理したり、MathMLによる数式を数式処理システムで計算することも必要となるだろう。TEXやMathMLの自動点訳、自動読み上げのアプリケーションも目の不自由な人たちにとって必要不可欠なものになっている。しかし、このような数式変換プログラムを様々な数式フォーマットの組み合わせの数ほど開発するには大変な時間と労力が必要となる。

我々は、この問題を解決するために、手書き数式入力インターフェイスを開発した([4])。このシステムはリアルタイムで手書き数式を認識し、自動的に書き換えを行うインタラクティブな手書き数式インターフェイスと、様々なフォーマットに対応し、数式の視覚的な編集を可能とするエディタからなる。そして、その効果を見るために、数式処理システムであるMathematicaにMathLinkを通して接続したシステムを開発した。

以下、そのシステムについて述べる。

2　本システムについて

本システムの基本的な機能は以下のとおりである。

- ユーザーは、マウス、データタブレット、ペンディスプレイなどを用いて手書きで数式入力することができる。(図 2-①)
- 手書きで入力された数式は、OK ボタンを押すことにより構文認識され、本システムのメインボードに活字体で表示される。(図 2-②)
- メインボードに表示された数式は、切り取り、コピー、貼り付けといった基本的な編集機能により、簡単に編集することができる。(図 2-③)
- メインボードに表示された数式を Mathematica で計算することができる。(図 2-④) この機能は、MathLink を用いたデータの受け渡しにより実現している。計算のメニューは、それぞれ Mathematica のコマンドに対応している。
- Mathematica による計算結果はメインボードの至る所に貼り付けることができ、表示されている他の数式と同じように編集できる。(図 2-⑤)
- メインボードに表示された数式の正確なグラフを表示することができる。(図 2-⑥)
- 本システムは、手書き入力だけでなく、TEX、MathML による数式フォーマットからの入出力もサポートしている。

本システムは、これらの機能を持つことにより、数式処理システムで扱う数式を手書きで入力したり、TEX や MathML フォーマットによる数式を手書きで編集したりすることが可能となる。また、その過程で用いられる編集中の数式を計算させたり、その結果を再度編集したりすることも可能となる。

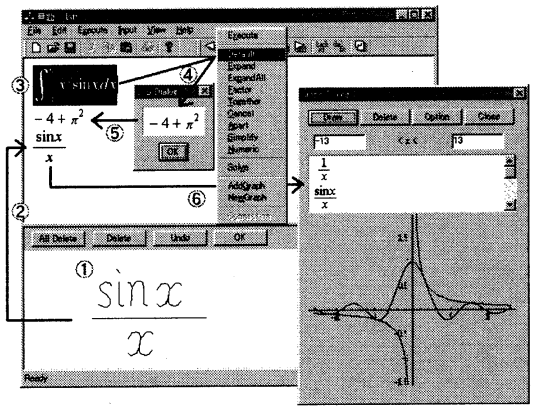

図 1: 本システムの全体図

3 手書き数式入力インターフェイス

ユーザーは、本システムに手書きで数式を入力することができる。この章では、この手書き数式入力インターフェイスについて述べる。ここで用いている文字認識の手法については、論文 [3] を見られたい。

3.1 仕様

現在、本システムでは、分数、根号、極限、級数、積分などを含む、高校または大学初年度の授業で扱われる数式を認識対象としている。使用する文字や記号の種類もその範囲でよく使われるものに限定している。アルファベット、数字、使用頻度の多いギリシャ文字、積分記号などの特殊文字、演算子などが認識対象文字となる。

また、高校でも行列は使われるが、今回は認識対象外とする。表示があまり複雑でない限り、分数、添え字などの入れ子の構造にも対応している。

3.2 数式認識 (逐次認識方式)

本システムの最大の特徴は、全体の数式を書き終えてから文字認識や数式認識を行うのではなく、ペンを離す度に文字認識を実行し、文字（或いは記号）として認識したら直ちに、整形された文字に書き直し、

位置を調整して表示してしまう点にある。これを逐次認識方式と呼ぶ。

数式認識では、入力ストローク列の文字・記号の単位の切り出しや文字認識の誤り、数式構文中の位置判定の誤りが数式全体の認識を破壊してしまうことが多く、その修正が著しく困難である。また、数式の書き直しや整形に神経を裂かれることは、書き手の思考の中断を生み出すことになり、スムーズな入力の大きな障害になる。

こうした困難に対する解答が上述の逐次認識方式である。

この方式により、以下の効果が望める。

1. 文字が書かれる毎に、きれいな文字に位置も調整して書き直されるので、誤認識が起きたときには直ちに修正することができる。また、添え字は、小さく添え字位置に表示されるため、添え字判定の誤りも直ちに修正できる。修正方法としては、削除による書き直しだけでなく、1クリックによる第3位までの認識候補の切り替えも用意している。
2. 逐次、文字の大きさや位置が調整されて書かれていくため、数式構文認識の誤りが非常に少なくなる。

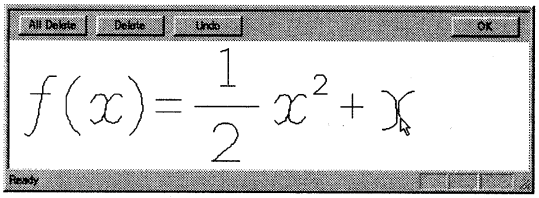

図 2: 逐次認識方式による自動書き換え

3.3 入力ルール

対象文字	入力ルール
C, S(大文字)	書き始めに縦線を付ける
U, V, W(大文字)	上に横線を付ける
P(大文字)	下に横線を付ける
X(大文字)	上下に横線を付ける
Z(大文字)	書き始めと終わりに縦線を付ける
q(小文字)	書き終わりにはねを付ける
1(数字)	書き始めに斜線を付ける

本システムでは、同じ形を持つ文字の識別(C、cなど)に関して、次の表のような入力規則を設けている。表の下段の小文字のq、数字の1は、それぞれ数字の9、絶対値記号と識別するためのものである。

また、数字の0、大文字のO、小文字のoに関しては、常に0を認識結果の第1候補、Oを第2候補、oを第3候補として出力するようにしている。

4 汎用入出力インターフェイス

本システムでは、独自に開発した内部形式で表された数式を画面表示することにより、数式を視覚的に扱うことが可能である。また、各種数式フォーマットからの入出力機能(インタプリタ、コンバータ)を備えている。インタプリタは数式フォーマットによる数式を内部形式に変換し、コンバータは内部形式を数式フォーマットによる数式に変換する。それらは、汎用性のある構造になっており、文法を記述したリソースと簡単な仮想関数の書き換えによって、様々な数式フォーマットに対応することが可能である。各種数式フォーマットに対応したインタプリタやコンバータを必要に応じてサポートしていくことにより、数式フォーマット間の変換アプリケーションの開発コストの削減につながる(図3)。

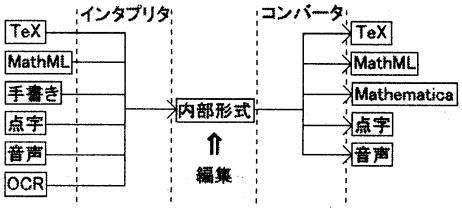

図 3: システムの概要

4.1 数式の内部形式

数式の内部形式は、数式の再帰構造を実現する独自のクラス構造で表された形式である。このクラス設計には、オブジェクト指向ソフトウェアを設計する上で頻繁に再利用されるデザインパターンの１つであるCompositeパターンを適用させている([6])。Compositeパターンは、オブジェクトを再帰的な木構造に組み立てることを可能にするデザインパターンであるため、数式の内部形式の実現に適していると考えられる。この木構造の実現により数式全体とその部分数式とを一様に扱うことが可能になるので、編集、数式フォーマットへの変換などがより簡単に行えるようになる。

また、新しい数式の構造を簡単に追加できる設計になっているため、今回未対応である行列の構造を追加することも容易に行うことができる。

4.2 コンバータ

コンバータは、内部形式により表される数式の木構造から各種数式フォーマットによる数式文字列への変換を行う。

このクラス設計には、デザインパターンの１つであるBuilderパターンを適用させている([6])。Builderパターンは、変換の過程を各種数式フォーマットに依存せずに行うことができるため、汎用性のあるコンバータを実現するために採用した。これにより、新しい出力フォーマットへの対応が、各文字、記号の変換を記述したリソースといくつかの数式の構造を変換する仮想関数の書き換えによって実現することができる。

4.3 インタプリタ

インタプリタは、数式フォーマットによる数式の構文を解釈し、その結果生成された構文木から内部形式への変換を行う。

数式の文法を文脈自由文法で表し、その文脈自由文法を解釈するプッシュダウンオートマトンをインタプリタの実現に適用させている([5])。文脈自由文法は、一般的に言語モデルとして使われ、数式のような再帰的な構造を表すことができるので、構文を記述する文法として採用した。

インタプリタは、ある数式フォーマットの文法を記述したリソースを読み込むことにより、その数式フォーマットを解釈するオートマトンに変化する。これは、文脈自由文法からプッシュダウンオートマトンを生成する方法に従って、数式構文解釈オートマトンが自動生成される仕組みになっているからである。そのため、新しい入力フォーマットへの対応は、そのフォーマットの文法を記述したリソースと各フォーマットに依存するいくつかの仮想関数の書き換えによって実現することができる。現時点では、TeX、MathML、Mathematicaとの数式のデータ通信を可能にするインタプリタ、コンバータが実現しており、目下、点字インタプリタ、コンバータも作成中である。

5 最後に

我々は、手書き数式入力インターフェイスを持った数式エディタを開発し、MathLinkにより、数式処理システム（Mathematica）へのアクセスを実現した。本システムでは、ペンによる数式の作成、編集ができ、必要であればMathematicaで計算することが可能である。

更に、コンピュータ上の数式を視覚的に編集する機能、各種数式フォーマット間の変換機能も実現した。インタプリタ、コンバータは、必要な情報を記述したリソースと簡単な仮想関数の書き換えによって、様々な数式フォーマットへの対応を可能にする汎用性のあるクラス設計になっている。

参考文献

[1] D. Blostein and A. Grbavec, "Recognition of Mathematical Notation", *Handbook of Character Recognition and Document Image Analysis*, (1997) 557-582

[2] T. Sakurai, Y.Zhao, H. Sugiura and T. Torii, "A Front-end Tool for Mathematical Computation and Education in a Network Environment", *Proc. 3rd. Asian Technology Conference in Mathematics*, Springer (1998) 197-205

[3] R. Fukuda, I. Sou, M. Xie, F. Tamari and M. Suzuki, "A Technique of Mathematical Expression Structure Analysis for the Handwriting Input System", *Proc. 5th. International Conference on Document Analysis and Recognition*, (1999) 131-134

[4] H. Okamura, T. Kanahori, W. Cong, R. Fukuda, F. Tamari and M. Suzuki, "Handwriting Interface for Computer Algebra Systems", *Proc. 4th. Asian Technology Conference in Mathematics*, December (1999) 291-300

[5] 都倉 信樹 著, "オートマトンと形式言語", 昭晃堂 (1995)

[6] Erich Gamma, Richard Helm, Ralph Johnson 著, 本位田 真一, 吉田 和樹 監訳, "オブジェクト指向における再利用のためのデザインパターン", ソフトバンク株式会社 (1995)