数式処理 *J.JSSAC (2002)*
Vol. 9, No. 1, pp. 32 - 37

特集「いろいろな数式処理システム」

黒板アプレット

出口 博章

神戸大学 発達科学部*

1 はじめに

黒板アプレット [1] は 1997 年に当時大学院生であった松嶋純也によって修士論文において作成された。

松嶋は 1997 年当時の数式処理システムの問題点を「複雑な CUI[1)] 入力」と「OS 依存」であると考え、その問題点の解決の糸口として「わかりやすい」「手軽である」の二点を挙げ、「使いやすい数式処理システム」のサンプルとして黒板アプレットを提示した [1]。

松嶋の「分かりやすい」とは『利用者が即座に正しい操作方法を推測しやすく作られていること [1]』であり、「数式処理のボタン化」と「マウス操作による数式への代入」などの GUI(Graphical User Interface) 強化によって実現を試みている。また松嶋は「手軽である」とは『システムの更新が容易であることと、すぐに使える環境にあること [1]』として、Web ブラウザの利用が有効であるとしている。

松嶋によって黒板アプレットが作成されたのは 1997 年であるが、2001 年に筆者が松嶋よりソースコードを譲り受け改良を継続していくこととなった。本稿執筆時点では JDK1.0 対応から JDK1.1 対応への移行に伴う変更と、グラフウィンドウ内の操作の追加などを行なっている。以下、本稿ではオリジナル版と改良版の違いについては特に触れず、改良版に基づいて述べる。

2 システムの特徴

2.1 直感的なユーザインタフェース

黒板アプレットの最大の特徴は、マウス利用を前提とした GUI において一般的である「クリック」や「ドラッグ＆ドロップ」などの操作に「数式の処理」という意味を割り当ててい

*deg@kobe-u.ac.jp

1) Character User Interface：文字入力中心のユーザインタフェース。

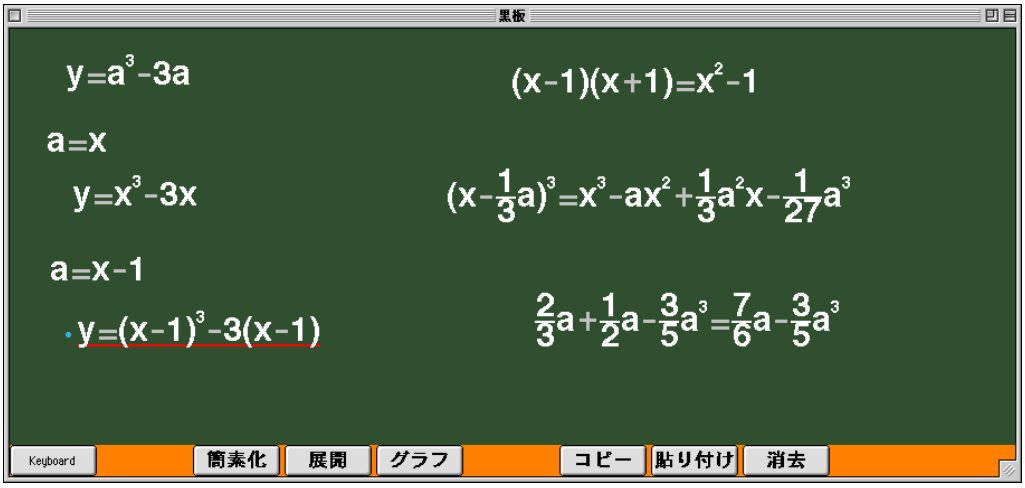

図 1: 黒板ウィンドウ

ることである。黒板アプレットでは、数式の展開を行なう際に `Expand[]` や `expand()` などのコマンドを利用するのではなく、「展開ボタンをクリック」や「ドラッグして展開ボタンにドロップ」のように GUI における操作を利用する。

「コマンド + 数式」の形で処理を行なうためには、数式を処理するためのコマンドを覚える必要があり、入門者にとってのハードルとなる。このハードルでつまずく原因の一つとして、プログラミングの経験がないユーザや OS の GUI 部分にしか触れたことのないユーザにとっては、同じ文字列として並んで記述されているコマンドと数式を、別のものとしてとらえることが困難なのではないかということが考えられる。

そこで、黒板アプレットではこのハードルでつまずくユーザに焦点をあてて「数式は黒板上に表示されている文字列」であって「数式の処理は GUI 操作」であるというように表現を明確に区別することを試みている。結果として、直感的に数式を処理するユーザインタフェースを備えているということが黒板アプレットの特徴となっている。

また、現在のところ黒板アプレットで試みている GUI 強化とは、数式の入力についてではなく数式を処理するという操作についてである。黒板ウィンドウ (図 1) 下部にツールバーが配置されており、そのツールバー上に「簡素化」「展開」「グラフ」などの操作ボタンが配置されている。

2.2 メンテナンス負荷の軽減

黒板アプレットは Java 言語で記述されており、JDK 1.1[2)]以上の Java 実行環境で動作するように作られている。OS に依存しないとはいえ JDK 1.1 以上の Java 実行環境が動作す

[2)] Java(TM) Development Kit 1.1.x：`http://java.sun.com/products/jdk/1.1/`

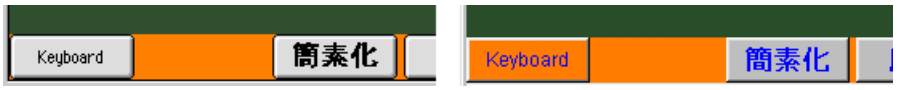

図 2: 外観の違い (左：Mac OS 9, 右：Windows 98)

る OS[3)] を使用していることが前提となる。

本稿執筆時点における黒板アプレットは Java の AWT(Abstract Windowing Toolkit) を使用しているため、ボタンなど GUI パーツの外観が OS によって異なる (図 2)。GUI パーツの表示や振る舞いに差異があるものの、一度バイトコードを用意すれば異なる OS 上においても同じコードが利用可能であり、OS 毎に分けて実行ファイルを用意する必要がない。そのため、一度のコンパイルとファイル更新によって、複数の OS に対してのバージョンアップが可能となりメンテナンスのための負荷が軽減される。

メンテナンスの際の負荷は、教室などのように端末が多くある場所においては、一台当たりの負荷ではなくその場所での作業全体の負荷を考えなければならない。例えば CD-ROM などのメディアを利用してインストールを行なうのであれば、一台当たりの負荷はそれほど大きくなくとも、数十台の端末一台一台に対して行なうことになるため、作業全体の負荷はかなり大きなものとなる。

アプレットは、利用する際に Web サーバから利用者の端末へダウンロードされるので、管理者側で行なうバージョンアップは、サーバに置くファイルの更新のみで良い。また、Java バイトコードであるため、複数の OS が混在している場合であっても OS 毎にファイルを用意する必要はない。現在のところ、黒板アプレットのサイズは 80K バイトほどであるため、アプレット利用時のダウンロードに多くの時間を取られるということもない。

3 システムを用いた事例

3.1 システムの構成

黒板アプレットは黒板ウィンドウ (図 1) に加えて、キーボードウィンドウとグラフウィンドウ (図 3) で構成されている。黒板ウィンドウは数式の表示を行なう黒板スクリーン部と処理ボタンの並ぶツールバー部に分かれる。その他、OS によって表示される場所は異なるがメニューバーを備えている。

3.2 操作方法

まず、入門者にとってハードルが低いと思われる基本的な操作方法、次にショートカットなどの少し高度な操作方法を述べる。

[3)] Java(TM) Platform Ports：`http://java.sun.com/cgi-bin/java-ports.cgi`

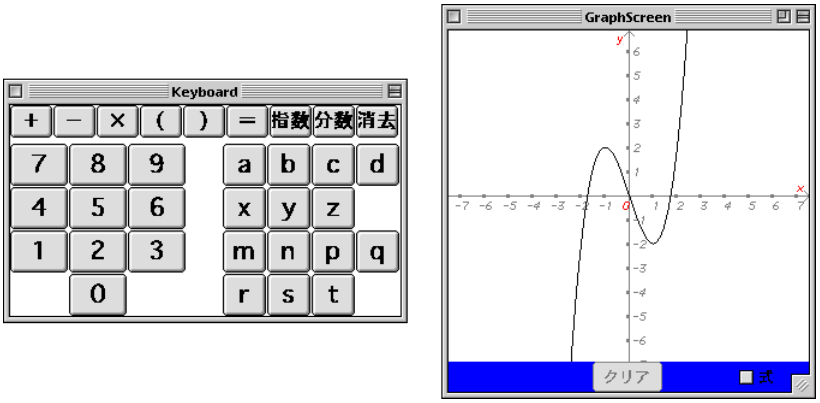

図 3: キーボードウィンドウとグラフウィンドウ

3.2.1 基本操作

数式の入力はキーボードウィンドウから行なう。指数や分数を入力する場合はそれぞれ指数ボタン、分数ボタンをクリックしてから数字の入力を行なう。現在のところは指数と分数の入力に利用できるのは数字のみである。

数式の左にある水色のカーソルと数式下の赤い線は、その数式が現在選択されていることを示す。式を選択した状態で黒板ウィンドウ下部のボタンをクリックすると「選択されている式」に対して「クリックしたボタンに応じた処理」を行なう。

グラフは一変数多項式のみ描画可能となっているが、グラフウィンドウには同時に三つまでのグラフを表示することが可能である。一度グラフに表示した数式を編集して再度グラフ処理を行なうと再描画が行なわれ、新たに別の数式に対してグラフ処理を行なうとグラフウィンドウに新たなグラフが追加される。

3.2.2 メニュー

黒板アプレットのメニューバーが表示される位置は OS によって異なり、ディスプレイ画面の上部 (Mac OS など) や黒板ウィンドウの上部 (Windows など) に表示される。現在のところは、上で述べた基本操作全ては、黒板ウィンドウ下部のボタンをクリックする代わりに、メニューバーからメニューを選択することによっても可能となっている。

メニュー表示においてコマンドの右側にショートカットが表示されているメニューは、キーボードショートカットによっても実行可能である。この場合のキーボードとは、コンピュータに接続されている装置としてのキーボードを指す。例えば数式のコピーは、数式を選択した状態でキーボードのショートカットキー[4)]と「c」キーを同時に押すことによって実行される。同様にコピーした数式の貼り付けは、ショートカットキーと「v」キーを同時に押すこと

[4)] Mac OS ではコマンドキー。Linux や Windows では CTRL キーなど。

によって、黒板上の水色のカーソルの位置に対して実行される。

数式のない場所への貼り付けはコピーされた数式の複製を行ない、数式上への貼り付けは数式の連結や代入を行なう。

3.2.3 ドラッグ＆ドロップ

黒板上に表示された数式の上にマウスカーソルを移動しクリックすると、その数式が選択される。クリックとはマウスボタンを押し、その同じ場所でマウスボタンを離すことをいうが、離さずに押し続けたままマウスを動かすことをドラッグという。ドラッグして移動した後に、押していたマウスボタンを離すことをドロップという。詳しく書くと「マウスボタンを離した瞬間にマウスカーソルのあった場所へのドロップ」となる。

ドラッグ＆ドロップは、マウスボタンを押し続けている時には指に力が入っており、マウスボタンを離す時には指から力が抜けることから、物を「掴んで」動かし「離して」落とすという動作を連想させる直感的な操作であるといえる。「どこにドロップされたのか」によって処理が分岐することが一般的であり、黒板アプレットにおいても数式のドロップ先によって処理が分かれる。

ドロップ先が黒板スクリーン内であって、さらに数式などのない場所であれば数式をドロップ先の位置へ移動する。ドロップ先が黒板スクリーン内であっても他の数式上であれば、ドラッグされた数式の「コピー」処理とドロップされた数式への「貼り付け」処理を連続して実行した場合と同様である。

また、ドラッグされた数式を黒板ウィンドウ下部のボタン上にドロップすると、上の基本操作で述べた「数式を選択してからボタンをクリック」と同様の処理を行なう。

グラフウィンドウへのドラッグ＆ドロップも可能であり、グラフウィンドウが表示されているのであれば、ドラッグされた数式をグラフウィンドウにドロップすることによってグラフ描画処理 (数式を選択して「グラフ」ボタンと同様の処理) が行なわれる。

3.3 実行例

図 1 の左側に並んでいる $y = a^3 - 3a$ から $y = (x-1)^3 - 3(x-1)$ までの式のうち下から二つ $a = x-1$ と $y = (x-1)^3 - 3(x-1)$ について述べる。これらは、$y = a^3 - 3a$ に $a = x-1$ を代入して $y = (x-1)^3 - 3(x-1)$ が得られるということを示している。$y = (x-1)^3 - 3(x-1)$ の位置には $y = a^3 - 3a$ があったものとして以下に具体的に述べる。

まず、$a = x-1$ を選択して、「コピーボタンを押す」「メニューバーのメニューからコピーを選ぶ」「ポップアップメニューからコピーを選ぶ」「ショートカットキー (CTRL など) と『c』キーを同時に押す」または「$a = x-1$ をドラッグしてコピーボタンへドロップ」のいずれかを行なう。

次に、$y = a^3 - 3a$ を選択して、「貼り付けボタンを押す」「メニューバーのメニューから貼り付けを選ぶ」「ポップアップメニューから貼り付けを選ぶ」「ショートカットキーと「v」キーを同時に押す」のいずれかを行なう。貼り付け実行後にダイアログが表示された場合は

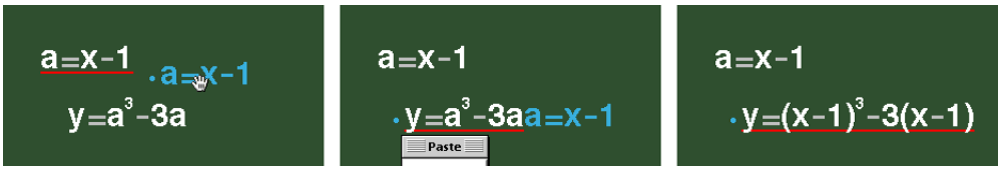

図 4: ドラッグ&ドロップ (左：ドラッグ, 中：ドロップ, 右：実行後)

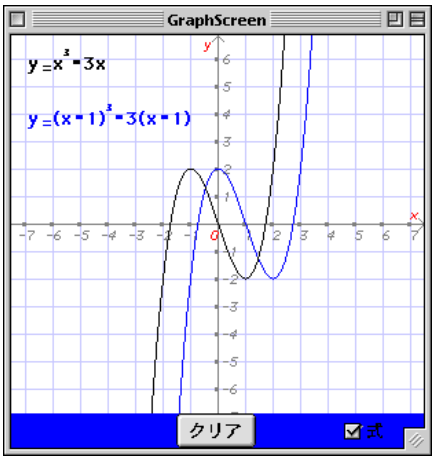

図 5: $y = x^3 - 3x$ と $y = (x-1)^3 - 3(x-1)$

「代入」を選択する。

あるいは、$a = x - 1$ をドラッグして $y = a^3 - 3a$ にドロップし、ダイアログが表示された場合は「代入」を選択するという方法 (図 4) も可能である。

このようにして作成した $y = (x-1)^3 - 3(x-1)$ と、同様の方法で作成した $y = x^3 - 3x$ の二つを同じグラフウィンドウに表示させたものが (図 5) となる。この図では右下の「式」チェックボックスをチェックし、グラフの式を表示させている。

4 入手先

黒板アプレットは Web 上で公開している。URL は以下のとおりである。

`http://doc.h.kobe-u.ac.jp/BlackBoard/`

参 考 文 献

[1] 松嶋純也: Java を用いた使いやすい数式処理システム, 神戸大学大学院教育学研究科 修士論文, 1998.