数学の考え方に基づくユーザインタフェース *

7E−2

古田秀和 †
NECソフトウェア中国 ‡

1　はじめに

通常、プログラムは見ることによって理解される。ところがこの方法では、できあがったプログラムの意味を覚えておくことが困難であり、また、プログラムの重要な部分と重要でない部分の区別が明確ではない。これらのことは視覚的プログラミングや個人の創造的な仕事にコンピュータを使うことを困難にしている。そこで[1]では数学を使ってプログラムを理解する方法を提案した。

本稿では、まず第2節で数学の考え方に基づくユーザインタフェース(Mインタフェース)の操作の体系を示し、次にそれを数学と対応させることにより、ユーザインタフェースを数学的に考えていく方法を示す。さらに、プログラミングに対する応用とその特長の概要を述べる。

第3節では、Mインタフェースの例として、数式変形システムのユーザインタフェースを取りあげ、最後にその今後の展望を述べる。

2　数学に基づくユーザインタフェース

2.1　Mインタフェースの操作体系

まず、基本的ないくつかの操作がユーザに与えられる。ユーザはその操作がどういう意味を持つかを知らない。個々の操作に対してコンピュータが反応する。それらの基本的な操作は組み合わせることができるようになっている。ユーザは操作の組み合わせに対するコンピュータの反応を見ることによって、基本的な操作の意味を知ることができる。そのうち、ある種の操作の組み合わせが有用であることにユーザは気づく。ユーザはその組み合わせをもとに、さらに新しい組み合わせを作っていく。組み合わせの操作をいくつも繰り返すと、複雑ではあるが特徴的な操作があることをユーザは発見する。ユーザはその操作のかわりに、ある新しい操作を使うと宣言することができる。このようにして新しく作られた操作は、ユーザの中である操作の組み合わせとして記憶される。ユーザは、新しい操作を作っていく操作を、期待するコンピュータの反応が得られるまで繰り返していく。

2.2　数学との対応

最初に与えられた操作は、ある定義を使うための規則(定義の使用法)に相当する。これらの操作が十分与えられていれば、ある簡単な問題(練習問題)を解くことができる。このようにしてユーザは定義を理解していくことができる。

操作を組み合わせていくことは、数式の計算や証明の記述などに相当する。これらは記号の操作である。数学では、記号を操作することによって、定義の形づくる空間を理解していくことができる。

操作の組み合わせの中で特徴的なものは、数学では定理に相当する。定理は後から引用することができる。定理は通常「ここで定理Aを使う」という形で引用される。定理を引用する段階で、その定理がどういう操作の組み合わせであるのかはユーザは理解している。

定理の中で簡単なものは練習問題となる。また、定理の中で重要と思われるものが最終的な結果となる。

2.3　プログラミングとの対応

最初に与えられた操作は、基本的なプログラムの部品の仕様に相当する。操作を組み合わせていくことは、必要な部品を作成していくことに相当する。このような操作によって、基本的な部品を理解していくことができる。

操作の組み合わせの中で特徴的なものは、重要な部品に相当する。このような部品は何に使うことができるのかを十分に理解した上で使うことになる。

2.4　Mインタフェースの特長

Mインタフェースによるプログラミングは次のような特長を持つ。

(1) 見てわかるということが必要ではない。

(2) 複雑な定義が可能である。

(3) 内容をよく理解しながら作業を進めることができる。

(4) 内容を思い出すことが容易である。

(5) 内容を理解すべきものとそうでないものの区別ができる。

---

*A User Interface based on Mathematics
†Hidekazu FURUTA
‡NEC Software Chugoku

3 数式変形システム

3.1 最初の操作とそれに対する反応

変数を使って書かれた群の式を変形規則

$$0+x \xrightarrow{\alpha} x,\ x \xrightarrow{\alpha^{-1}} 0+x,$$

$$-x+x \xrightarrow{\beta} 0,\ 0 \xrightarrow{\beta^{-1}} -x+x,$$

$$(x+y)+z \xrightarrow{\gamma} x+(y+z),\ x+(y+z) \xrightarrow{\gamma^{-1}} (x+y)+z$$

を使って変形することを考える。最初の操作としては

$$\alpha, \alpha^{-1}, \beta, \beta^{-1}, (1+\sigma)\circ\gamma\circ(\tau+1), (\sigma+1)\circ\gamma^{-1}\circ(1+\tau)$$

が与えられる。ここで、$\circ$ は変形の合成、1 は変形しないこと、$\sigma+\tau$ は $x+y$ の形の式の x を規則 σ で変形し、y を規則 τ で変形したものを表す。ユーザは、これらの変形規則に対応したある動作 (fig.1) を入力する。それによってその変形規則に対応した記号 (fig.2) が表示される。また、指示すれば、数式の変形の様子が動画で表示される (fig.3)。

たとえば β^{-1} に対応する動作を入力すれば、β^{-1} に対応する記号が表示され、

$$0 \longrightarrow -x+x$$

という変形の様子が表示される。これは、最初は 0 に対応する図が表示され、次第に $-x+x$ に対応する図に変形する様子が表示されるものである。

3.2 組み合わせの操作

β^{-1} につづいて $(\sigma+1)\circ\gamma^{-1}\circ(1+\tau)$ に対応する動作を連続して入力すれば、$(\sigma+1)\circ\gamma^{-1}\circ(1+\tau)$ に対応する記号が表示され、これらを組み合わせた変形規則

$$\gamma^{-1}\circ(1+\beta^{-1})$$

に対応する変形

$$x+0 \longrightarrow (x+-y)+y$$

の様子が表示される (fig.3)。

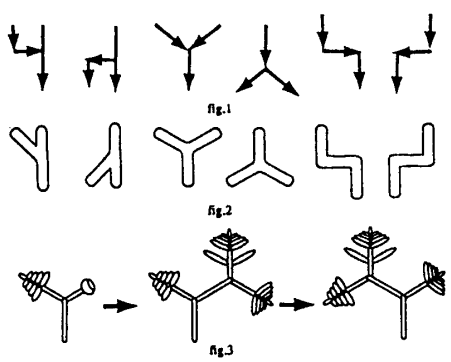

fig.1

fig.2

fig.3

3.3 定理の登録

組み合わせ

$$\alpha\circ(\beta+1)\circ\gamma^{-1}\circ(1+\beta^{-1})$$

による変形

$$--x+0 \longrightarrow x$$

が定理にあたると思ったとき、この変形に対応する動作と記号を定義することができる。この変形を使用するときは、新しく定義した動作を用いる。

3.4 特長

このシステムの特長は、数式の変形の様子を見せるということ、入力は動作によって行うということ、およびそれらの間に論理的な関連があるということである。

これによってユーザは自分の動作とその意味する結果とのつながりを理解することができる。したがって、変形を動作として理解することができ、重要な変形 (定理) と重要でない変形を区別することができる。また、数式の変形の様子を見ることができるため、定理の内容を思い出すことも容易である。

4 おわりに

第 3 節述べたユーザインタフェースは、ある数学の理論の一部を抽象化したものと考えることができる。このように考えると、このユーザインタフェースが、あるプログラミングの理論を抽象化したものになっている場合、プログラミングに応用することができる。

また、このユーザインタフェースが、別の数学の理論を抽象化したものになっていれば、その理論を研究するためにこのユーザインタフェースを使うことができる。さらに、その理論で得られた結果をプログラミングに応用することもできる (fig.4)。

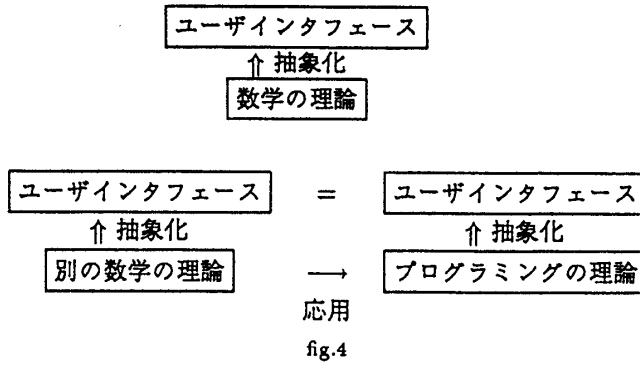

fig.4

参考文献

[1] 古田他: 数学的プログラミング環境, 情報処理学会第 45 回全国大会論文集 (5), pp.11-12, 1992.