ユーザインタフェースに数学の性質を導入する方法 *

2J−10

古田秀和 †
NECソフトウェア中国 ‡

1 はじめに

数学の問題をコンピュータを使って解こうとする場合、人間とコンピュータとが協調してその問題について考えるということが重要である。ところが、コンピュータの操作の体系が、人間が考えながら操作するという目的に適していないために、コンピュータを使うことによって人間の思考が阻害されているという問題点がある。

この問題を解決するために、私たちは、数学の問題を記述することができるプログラミング言語と、人間が考えながら操作することができるユーザインタフェースとからなる証明支援システムを提案した。

本稿では、証明支援システムのユーザインタフェースの部分について述べる。

ユーザインタフェースの部分は、ある数学の理論に基づいたものである。ユーザインタフェースの基礎となる理論としては、集合の演算の性質、代数系の性質、位相空間の性質、自然数や実数などの性質などが考えられる。これらの理論の中の特定の性質に着目して、その性質を使った数式または数学的対象の変形を操作の基礎とするユーザインタフェースを考える。本稿では、ユーザインタフェースの部分を

(1) ユーザインタフェースの操作体系と理論との関係

(2) ユーザインタフェースの理論とプログラミング言語との関係

という点から考察する。

2 ユーザインタフェースの操作体系と理論との関係

ユーザが、ユーザインタフェースに対して予測ができたり、考えることによってその概念を得たりすることができるためには、ユーザインタフェースが、ある理論に基づいて動くことが必要となる。

ここでは、群における式の変形に対する操作を考えてみる。

x, y, z などを使って書かれた群の式を、群の公理の一部である変形の操作と、変形を施す位置を移動する操作によって、変形していくというものである。このシステムは、群論の公理系の意味をユーザが理解するということを目的としている。

このシステムの操作は次のようになる。入力ウィンドウ上でマウスのボタンを押すと、図1左のような図形が表示される。ここでボタンを押したまま移動させることにより、図1右のような表示となり、これによって四方向への移動を入力することができる。また、その他の移動方法により、合計12通りの入力をすることができる。

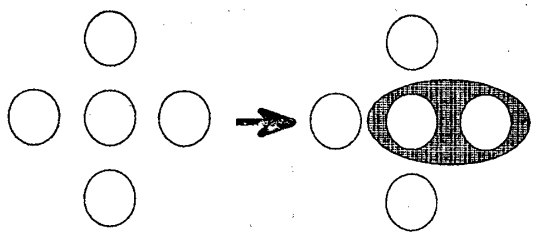

図 1: 変形の操作

この操作体系に対応して、数式変形ウィンドウ上に表示された数式が変形されていく。数式は図2のように木の形式で表示されていて、どの部分に対して操作を行なうかを表す点(大きい円で表示されている)も表示されている。

たとえば、図2は、$x + -x$ という式を表し、変形を施す点は式の全体であることを表している。図1で示した操作によって、変形を施す点は $-x$ の部分に移動する。

また、数式を表す木の角度をスライダーによって変えることができ、これによって隠れている部分を見ることができ、また、数式の立体的なイメージを把握することができる。

この操作体系では、式のどの部分に対しても、同じ変形に対しては同じ操作を施せばよいようになっている。したがってユーザは自分の操作と数式の変形とを比べることによって、式の変形の体系を把握できるようになっている。

このシステムは、ユーザインタフェースの基礎となる理論として群などを使用した場合、ここで示したような操作体系を用いることができることを表している。

*How to introduce mathematics to user interfaces
†Hidekazu FURUTA
‡NEC Software Chugoku

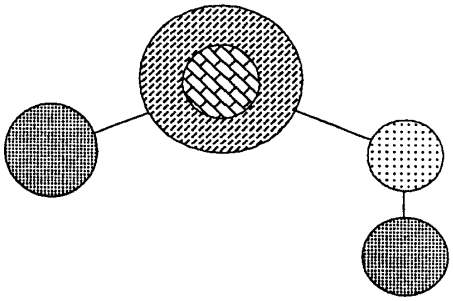

図 2: 変形の表示

3 ユーザインタフェースの理論とプログラミング言語との関係

図3のように帰納的に定義された図を描く問題を考えてみる。

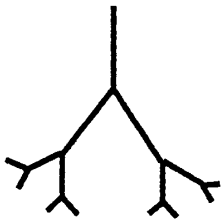

図 3: 帰納的に定義された図形

これは、たとえば図4のように並列に実行されるプログラムと考えることができる。

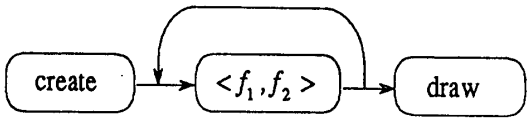

図 4: データの流れ

この図で、createは、最初の線分のデータを作成することを表し、drawは線分を実際に表示することを表す。f_1, f_2 は、線分から線分への変換を表し、$\langle f_1, f_2 \rangle$ は、f_1, f_2 の組を表す。また、矢印はデータの流れを表す。

この図によって、帰納的な定義は三つの部分に分かれると考えることができる。最初の部分はcreateという部分であり、一度だけ実行される。第二の部分は $\langle f_1, f_2 \rangle$ の部分である。これは、繰り返し実行され、データの流れが循環している。第三の部分はdrawの部分である。これも、繰り返し実行される。

したがって、図4に適合した操作体系を考えることにより、帰納的な定義(のある種のもの)が実現できるということになる。これはまた、図4に示したような実際のプログラムとも対応している。

4 まとめ

証明支援システムのユーザインタフェースには理論的な背景が必要であるということを述べた。また、ユーザインタフェースの理論と実際の操作体系との関係、ユーザインタフェースの理論とプログラミング言語の性質との関係について考察した(図5)。

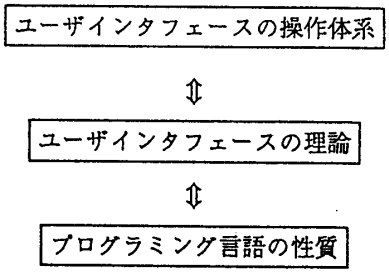

図 5: 証明支援システムの構成

第2節では、ユーザインタフェースの理論として群の式の変形を扱った場合のユーザインタフェースの操作体系について述べた。これは、プログラミング言語の性質としては、代数的な定義ができるようなプログラミング言語に相当すると考えられる。第3節では、ユーザインタフェースの理論として帰納的な定義を扱い、それがプログラミング言語の性質としてはどのようになるかということを述べた。また、これに対応するユーザインタフェースの操作としては図4をあらわす操作を使うことができるということを示した。

今後は、これらを結びつけたものを考察していく必要がある。

参考文献

[1] 古田秀和他: 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.

[2] 古田秀和: 数学の考え方に基づくユーザインタフェース, 情報処理学会第46回全国大会論文集 (5), pp.101-102, 1993.