高度なHelp機能の実現法について

4L-9

青柳　龍也　田中　英彦　元岡　達

(東京大学　工学部)

使いやすい計算機を実現するためには、ユーザがその計算機を使うことを助ける様々な機能が必要である[1]。今回はそれらの機能のうちで、コマンドヘルプ(以下helpと略す)と呼ばれる機能の改善について考察する。

1. コマンドヘルプ

helpは、通常オンラインで、その計算機システムのコマンドに関する質問に答える機能である。

オンラインで使うということを考慮すると、helpを使うユーザは初心者でもエキスパートでもない人と考えてよい。まったくの初心者は、紙に書かれたマニュアルを読むなり、計算機や人による教育を受けるべきであるし、コマンドについて熟知しているエキスパートはhelpを使う必要はない。helpを使うのは、例えば、しばらくそのシステムを使わないでいた人や、あまり使わないコマンドを使ってみようとする人などである。結局helpが対象とすべきユーザは、ある程度そのシステムについて知っている人である。

また、コマンドに関する質問に答えるということから、helpが対象とする情報は、コマンドの名前やシンタックス(以下コマンド名とする)と、コマンドの機能や動作に関すること(以下コマンド機能とする)であることがわかる。

まとめると、helpは通常オンラインで動き、ある程度その計算機システムについて知っているユーザが、コマンド名とコマンド機能に関する質問をして情報を得るための機能である。

2. 従来のコマンドヘルプの問題点

ユーザがhelpを使うのは、あるコマンド機能を持つコマンドの存在を知りながらそのコマンド名を知らない(忘れた)場合が多い。逆に、コマンド名は知っているがコマンド機能は知らないという状況は考えにくい。すなわちhelpは、「～するコマンドのコマンド名は何か」という質問に答えられなければならない。

ところが従来のhelpは、コマンド名を入力してコマンド機能を出力するというものだった。そのためユーザは、コマンド名を知らない限りhelpを使うことができなかった。今までのところ、この問題に対する解決法は、

1) コマンド名の一覧表を表示する。
2) コマンド機能を記述するテキスト中をキーワードで探す。

などがあげられるが、1)は、表示されるコマンドの数が少しでも増えると、ユーザは、自分の求めるコマンドを探すのに困難を感じるし、2)は、キーワードの選択が難しく、どちらも使いやすいとはいえない。

以上の考察から、使いやすいhelpは、

1) 「～するコマンドのコマンド名は何か」という質問に答えられる。
2) 「～するコマンド」という部分のコマンド機能の表現は、個々のユーザにとって自然な表現でかまわない。

という特長を持っていなければならない。そのようなhelpを実現するためには、コマンド機能についてのユーザの多様な表現から、どのコマンドのことを質問しているのかを判定しなければならない。以下、ユーザの多様な表現をどのようにしてコマンドに対応させるかについて述べる。

3. 表現・概念・抽象・類似・連想

以下の議論に必要な考え方を簡単にまとめておく。

- 概念は概念名から成る構造である。
- 概念名から概念への写像が存在する。
- 表現から概念名への写像が存在する。
- ある概念の部分構造はその概念の抽象概念である。
- 抽象概念の等しい二つの概念は類似概念である。
- 概念と概念の間に連想関係が存在する。

4. 少数の概念と多様な表現

helpが扱うのはある制限されたコマンドに関するものである。それらのコマンドのコマンド機能に必要な概念はそれほど多くは存在しない。また、かなりはっきりした概念が多い。一方、helpを使うユーザは、ある程度その計算機システムについて知っている人である。計算機システムについて知っているということは、そのコマンド体系に必要な概念をすでにある程度得ていることになる。すなわち、ユーザは、計算機システムについて知ったことにより、そのコマンド体系に内在している少数のはっきりした概念で考えるようになっている。結局、ユーザのコマンド機能に関する表現は多様であるが、概念はそれほど多様ではない。

それでは、表現の多様性はどうして生じるのか。多様性の生じる原因はいくつか考えられる。

まず、ある決まった概念を表現するのに、どのような言葉を選ぶかという表層的な多様性が考えられる。例えば、「ある行を消す」というエディタのコマンドのコマンド機能を表

現することを考えると、「消す」という概念を表現するのに、「削除する」、「消す」、「deleteする」、「なくす」等の様々な言葉を選ぶことが可能である。しかし、それらの表現が表わそうとしている概念はhelpの観点からは一つであるとしていい。

次に、ユーザの知識の量による多様性が考えられる。ユーザは、あるコマンドを表現するのに、自分の知っている他のコマンドと区別できるように表現する。例えば、「次の単語の最初の文字へカーソルを動かす」コマンドと「次の単語の最後の文字へカーソルを動かす」コマンドがあったときに、前者しか知らないユーザは、「次の単語へカーソルを動かす」と表現する。しかし、両方のコマンドを知っているユーザは、両者が区別できるように、「次の単語の最初の文字へカーソルを動かす」と表現する。

さらに、省略による多様性が考えられる。文脈などによって、一意に決まる部分は表現しないのが普通である。例えば、今カーソルを動かすことが話題になっている場合には、「次の単語の最初の文字へカーソルを動かす」コマンドは、「次の単語の最初の文字へ」と表現される。また、カーソルが常に文字の上に移動するならば、「次の単語の最初へ」と表現されるかもしれない。

5. 表現の多様性の吸収

ユーザの表現から、ユーザのコマンド機能に関する概念（以下user概念と呼ぶ）を構成し、その概念とhelp内に記述されているコマンド機能の概念（以下system概念と呼ぶ）とを比較して対応するコマンドを決定することを考える。表現の多様性は次のようにして吸収することができる。

まず、表層的な多様性は、表現から概念名への写象で吸収する。例えば、前にあげた例ならば、「消す」という概念を表わす概念名に、「削除する」、「消す」、「deleteする」、「なくす」等のすべての表現を対応させればいい。

次に、ユーザの知識の量による多様性と省略による多様性は、どちらの場合もuser概念がsystem概念の抽象概念になっていると考えられるので、user概念が抽象概念となるようなコマンドを対応させればいい。例えば、「次の単語へカーソルを動かす」、「次の単語の最初へ」等のユーザの表現から構成されるuser概念は、「次の単語の最初の文字へカーソルを動かす」ということを表わすsystem概念の抽象概念になっている。

このようにして、表現から概念名への写像と、抽象という考え方により、ユーザの表現の多様性は吸収できる。

6. 動作

以上述べてきた考え方に基づくhelpの動作について考察する。

ユーザから入力される表現は単語の列である。単語の列を、表現から概念名への写像を使って概念名の列に変換する。次に、概念名の列からuser概念（概念名から成る構造）を構成する。さらに、help内に記述されているsystem概念を取り出して、user概念がsystem概念の抽象概念になっているすべてのコマンドを集める。

以上の探索で求まったコマンドの個数が少数ならば、そのコマンド名を答えればいい。しかし、コマンドの個数が多かった場合とコマンドが一つもみつからなかった場合は、なんらかの対策を考える必要がある。

コマンドの個数が多かったというのは、ユーザの表現が曖昧だった場合である。そのときは、求まったコマンドの集合の要素をそれぞれ区別するために必要な情報を考慮して、ユーザに聞きかえせばいい。

コマンドが一つも求まらなかった場合は、類似のコマンドと連想されるコマンドを答える。多くの場合それによりユーザは有用な情報を得ることができる。例えば、「次の単語を消す」というコマンドが存在しないのにユーザが質問してきた時には、類似のコマンドとして「単語を消す」コマンドを連想されるコマンドとして「次の単語へカーソルを動かす」コマンドを答える。

類似のコマンドはuser概念を抽象して、user概念の類似概念であるsystem概念を探せばいい。例えば、「次の単語を消す」というユーザの表現から構成されるuser概念を抽象して「単語を消す」という類似概念を作り対応するコマンドを探す。ただし、user概念を無制限に抽象すると、すべてのコマンドが類似したコマンドになってしまうので、この場合の抽象には、価値判断がはいることになる。「次の単語を消す」から「単語を消す」を作るのは、「次の」の部分の重要度が低いという価値判断による。

連想されるコマンドは、概念間の連想関係から得られる。例えば、エディタで、あるものを消すという概念は、あるものの所へカーソルを動かすという概念と連想関係にあるということが与えられていれば、「次の単語を消す」から「次の単語へカーソルを動かす」という概念が得られ、その働きをするコマンドを答えることができる。

7. Prolog による実装

エディタのコマンドに関するhelpをProlog を用いて実装した。表現は日本語とし、「の」、「を」、「から」、「まで」等の助詞により概念を組立てるものとした。概念は、Prolog の論理変数による束縛とリスト構造を使って表わした。また、表現から概念名への写像や連想関係等は、Prolog のファクトとして記述されている。

<参考文献>

[1] 青柳、田中、元岡、「Intelligent Manual System に関する一考察」、情報処理学会第29回全国大会.