

# 大学における計算機環境下での 対話的ヘルプシステムの作成

日笠 亘 古河 雅輝 黒橋 禎夫

Wataru Higasa, Masaaki Kokawa, Sadao Kurohashi

京都大学大学院 情報学研究科 知能情報学専攻

{higasa, kokawa, kuro}@pine.kuee.kyoto-u.ac.jp

## 1 はじめに

計算機と人間との対話についてはこれまでに多くの研究がなされてきたが、ドメインを限ったとしても、いまだに満足のいく対話システムは実現されていない。これは、計算機と人間との対話データが十分に収集されておらず、対話の現象の調査が不十分であることが最大の原因と考えられる。本研究では、計算機と人間との対話が実際におこりえる環境として京都大学における 1200 台の計算機教育環境に着目し、その環境においてアプリケーションなどの初歩的な質問を受け付ける対話的ヘルプシステムを構築した。このシステムを実際に運用していくことによって、対話のデータを収集し、対話システムを徐々に高度なものにしていくプロジェクトについて進捗状況を報告する。

## 2 質問応答の調査とユーザ発話の分類

まず、京都大学総合情報メディアセンター<sup>1</sup>の種々のサービスに関して、実際にどのような質問応答のやり取りがなされているかを次のように調査した。

1. センターのホームページ<sup>2</sup>にまとめられている FAQ(約 100 項目)
2. ユーザからセンターに届けられた質問メール (約 150 通)
3. センターで実際に行われているユーザと TA (Teaching Assistant) との質疑応答を録音・文書化 (約 20 時間)

上記の調査のから以下のようなことが分かった。

<sup>1</sup> 学内に 1,200 台のパーソナルコンピュータ/ワークステーションを配置し、情報リテラシー教育の支援を行うことを目標とした機関。

<sup>2</sup> <http://www.media.kyoto-u.ac.jp/>

1. ユーザの質問にはバリエーションが少ない (FAQ にまとめられている程度)。
2. FAQ を見れば解決する質問であっても、ユーザは自分で調べることは少なく、TA に質問する傾向が強い。

そこで、FAQ 程度の内容について対応可能なシステムを作成・運用すれば、ユーザとの対話はある程度可能になり、また実際上有用であると考えられる。

また、上記の調査の結果、ユーザの発話を以下のように分類することができた。

### How 型

方法を問うもの。以下のような幾つかのパターンがある。

- mule で日本語入力する方法を教えてください。
- mule で日本語入力する方法は？
- mule で日本語入力したいのですが？
- mule で日本語入力できますか？

### What 型

事実を問うもの。

- mule とはどんなものですか？
- mule って何ですか？

### 症状提示型

ユーザが症状を提示し、その対処法を求めるもの。

- 個人用のフォルダにアクセスできません。
- 画面が真っ白になってしまいます。

### Request 型

ユーザがセンターに対して、ソフトのインストール等の要求を行う場合。

- Visual-C をインストールして下さい。
- ファイルの使用制限を増やして下さい。

## Assert 型

システムが質問を行った場合のユーザの返答。

- WINDOWS です。
- ファイルの拡張子は lzh です。

## 3 対話的ヘルプシステムの構成

2 節で述べた調査に基づき、以下の要素から成る対話的ヘルプシステムを試作した。

1. 知識データベース
2. ユーザの自然言語入力を計算機処理可能な中間表現に変換する入力解析部
3. 入力解析部の解析結果を利用し、ユーザの質問に対する処理を行う質問処理部

以下では各モジュールについて述べる。

### 3.1 知識データベース

一般に、知識データベースは、拡張・管理・メンテナンス等に非常にコストがかかる。そのため、例えば FAQ テキストそのものを知識ベースとして利用し、ユーザの入力文と FAQ テキストのマッチングによって擬似的対話を行うようなシステムがある [島津, 伊藤 98]。

しかし、本研究の目的は、メンテナンスのしやすい対話システムを構築することではなく、対話のモデルについての基礎的研究を行うことにある。そのためには構造化された知識ベースを用意することが必要である。そこで、FAQ の内容について、図 1 のような知識データベースを構築した。これによって、上位・下位概念への移動・関連要素間の移動が可能となり、興味深い対話の現象を扱うことが可能となる。

### 3.2 入力解析

入力解析部では、ユーザの自然言語入力に対して構文解析を行い、その解析情報を用いて入力の分類とキーワード (自立語) の列を中間表現として出力する。構文解析ツールには KNP [黒橋 97] を用いた。例えば、ユーザの入力が

「mule で日本語入力する方法は？」 (1)

である場合は以下のように中間表現を出力する。

Type:How Content:mule 日本語入力 (2)

ここで Type は入力の分類、Content はキーワード列である。

## 3.3 質問処理

質問処理部では入力解析部が出力した中間表現と知識データベースを用いて質問の処理を行う。

処理の基本は、(2) に示したようなキーワード列から、知識中のフレームを探索・決定することである。知識内の各キーワード (object や function 等) はデータベース化されており、2つのキーワード「mule」と「日本語入力」でこのデータベースを探索すると、「mule」のフレーム中の「function:日本語入力」が発見される。その他に「mule」と「日本語入力」を含むフレームはないので、このフレームが唯一の探索結果となる。

その後、それぞれのタイプの発話に対して以下のように処理を行う。

**How 型:** ユーザが要求することに対して、それが可能な場合、すなわちフレーム内に use もしくは usefunction がある場合はそれを説明する。不可能な場合、すなわちフレーム内に use が無い場合は false\_reason を表示する。

例えばユーザの入力が (1) の場合、前述のように「mule」フレームが検索され、また「日本語入力」に対応する使用法の usefunction が検索フレーム内に存在するので

「C-\\(Ctrl キーを押しながら\\キーを押す)  
として下さい」 (3)

と表示する。

**What 型:** ユーザが要求するものについてのフレームが存在すれば、その description を表示する。

**症状提示型:** ユーザが提示する症状に対する対処法 (フレーム内の error) が存在すればそれを表示する。

このタイプは **Assert 型** の発話との区別が難しいことと、症状には複数の事例が存在し、特定が難しいことから、現在は一部を除いて未対応である。

**Request 型:** ユーザの要求が既に満たされていれば (フレーム内の use が存在する場合) その旨を表示し、そうでなければセンターにその内容のリクエストメールを出す。

### 3.4 対話の構造

前節の入力文 (1) に対して (3.3) を表示した例は、知識内のフレームが一意に決定される最も単純なケース

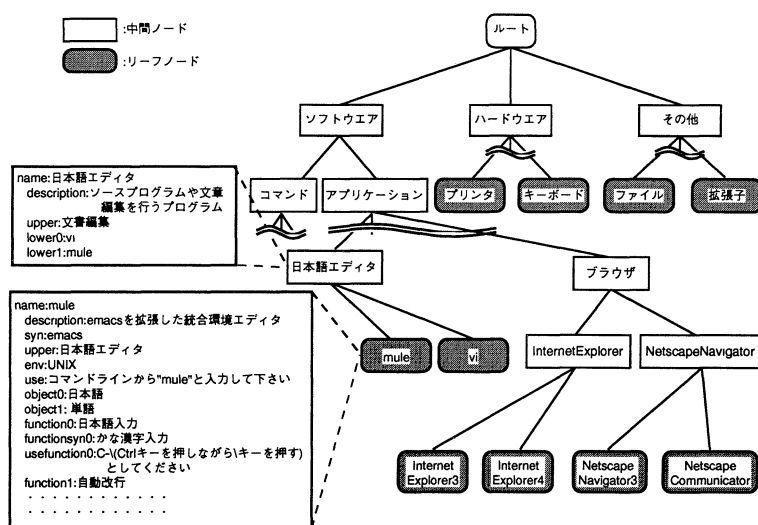


図 1: 計算機内の知識フレーム

であった。実際にはユーザの最初の質問に満足できる応答が返されるまでには、ユーザとシステムの間で何度かのやり取りが行われる。

この構造は図2のようにまとめられる。ここで、ユーザの質問は  $Q_u$ 、システムの質問は  $Q_s$ 、システムの回答は  $A_u$ 、ユーザの回答は  $A_s$  である。また、 $Q-A$  の組は同一の段のものが対応する。

システムは必ずユーザの直前の発話に対して対応する。その際に、ユーザ発話に対して回答が可能な場合(前述(1)のような例)と、不可能である場合がある。回答が不可能な場合にはシステムはユーザに質問を行う。システムからの質問は次のような場合に起こる。

1. 知識内の検索で何も見つからない場合  
→ 知識データベース中に存在しないキーワードをユーザに聞き返す。
2. 複数のフレームが見つかる場合  
→ 候補を表示し、ユーザにその指定を依頼する。

上記の処理を行っても、システムが質問に回答が出来なかった場合は、その内容をセンターにメールとして送ることに対応する<sup>3</sup>。

一方、ユーザはシステムのどの発話に対しても質問・回答することが可能である。そのため、システムはユーザの各発話が以前のどの発話に対して行われた

ものなのかを理解しなければならない。

このような前提をうまく扱い、前述のシステムの聞き返しや候補の指定の発話を極力行わず、複雑な  $Q-A$  の入れ子構造に陥らないようにするために必要な機能としては以下のものが考えられる。

- ユーザの発話が、以前の発話のどの部分に対して行われたものかを推定する機能
- ユーザが以前に指定した環境を保持し、新しいユーザ発話に対してどのように情報を補完するのかを決定する機能
- ユーザのタイプミス等に対する聞き返しを避けるための推定機能
- ユーザの質問発話に対する質問発話を、システムは2回以上行わない(図2の<9>のような状態に陥らない)ようにする機能

以上の機能の一部は、以下に示す対話例のようにシステムで実現されているが、多くの部分については今後の課題である。

### 3.5 対話例

システムで実際に動作する対話例の一部を以下に挙げる。対話例中の括弧内は発話のタイプと図2中の状態を表している。

<sup>3</sup> このような機能を設けておくことで、ユーザにとってシステムの利用価値が高まり、利用度が増すと期待される

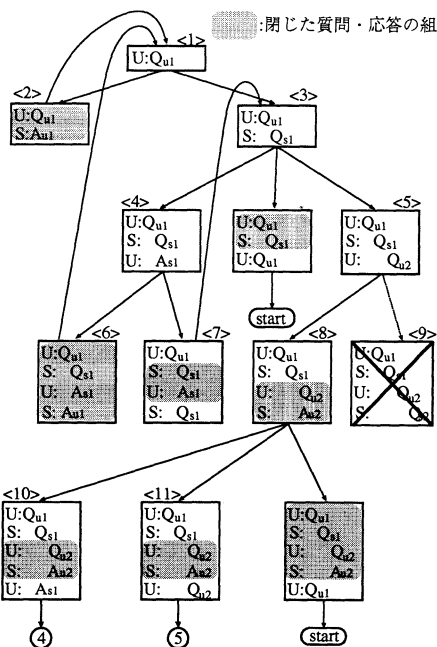


図 2: 現在のシステムの Q-A の構造

#### 対話例 1

User: IE は使えますか? (How 型:< 1 >)  
System: InternetExplorer3 が使えます。(< 2 >)

このときユーザが直接質問した「IE」は図 1 の中間ノードにあたるため、下位概念が複数存在する。このため下位概念から実際に使えるものだけを表示する。

#### 対話例 2

User: モザイクは使えますか? (How 型:< 1 >)  
System: 「モザイク」とは何ですか? (< 3 >)  
User: ブラウザです。 (Assert 型:< 4 >)  
System: ブラウザでしたら  
InternetExplorer3, NetscapeNavigator3  
が使えます (< 6 >)

この場合、ユーザが直接質問した「モザイク」が見つからなかったため、システムはユーザに問い直す。ユーザが「ブラウザ」と答えれば、まず「ブラウザ」フレームを検索し、次にブラウザの下位概念を探索して使用可能なものの表示を行う。

#### 対話例 3

User: 圧縮ファイルの解凍はできますか?  
(How 型:< 1 >)  
System: lzh・gz・tar の中から拡張子を指定して下さい。  
(< 3 >)  
User: 拡張子とは何ですか? (What 型:< 5 >)  
System: 「ファイル名のピリオドより後ろの部分」です。  
もう 1 度、拡張子を指定してください。  
(< 8 >)  
User: 拡張子は lzh です。 (Assert 型:< 4 >)  
System: 解凍できます。コマンドラインで  
LHA a [filename]  
と入力してください。 (< 6 >)

ユーザの「圧縮ファイルの解凍」に対する質問 (How 型) は、ファイルの拡張子によって解凍に用いるアーカイバ (図 1 の中間ノードにあたる) が異なるので、システムはユーザにその指定を求める (図 2 の< 3 >の状態)。しかし、ユーザは「拡張子」に対する質問 (What 型) を行う (図 2 の< 5 >)。そこで、システムはその質問に答えることに処理を移してそれが終了すると、元の拡張子の指定 (Assert 型の入力) を期待する状態に戻る (図 2 の< 8 >)。ユーザが拡張子の指定 (Assert 型の入力) を行う ((図 2 の< 4 >) と、それに対応する回答の表示を行う (図 2 の< 6 >)。

## 4 今後の課題

本稿では、対話的ヘルプシステムの構築について述べた。現在、このシステムは研究室内で試験運用されているが、今後、京都大学の 1200 台の計算機環境で実際に運用していくことを予定している。このことによって、対話のデータを収集し、対話システムを徐々に高度なものにしていくことが目標である。

## 参考文献

- [Uehara 89] Kuniaki UEHARA: An Intelligent On-Line Help System: ASSIST, Future Generation Computer System 5 (1989) 11-20
- [島津, 伊藤 98] 島津秀雄, 伊藤慎: ヘルプデスク支援システムの最新動向, 情報処理 39 巻 9 号 (1998)
- [黒橋 97] 黒橋禎夫: 日本語構文解析システム KNP 使用説明書 ver. 2.0b3, 京都大学工学部 (1997)