情 報 学 基 礎　66－13
デジタル・ドキュメント 32－13
（２００２．３．１５）

図書検索における自然言語問合わせ方式

新里　圭司　　絹川　博之

東京電機大学　工学部　情報通信工学科
〒101-8457　東京都千代田区神田錦町 2-2
shinzato@cll.im.dendai.ac.jp　kinukawa@im.dendai.ac.jp

パソコンや各種携帯端末の普及により、多くの人がインターネット上のデータベースを利用することが可能になった。しかし、以下に示す課題が残っている.

(1) 情報機器に不慣れなユーザにとって，複雑な論理条件からなる検索要求を指定するのは難しい.

(2) パソコンのディスプレイと比べ携帯端末等の表示画面領域は狭いため、多くの検索条件指定項目を表示することができず、ユーザが満足する検索要求を指定することは難しい.

このような問題を解決する一つの方法として、会話的表現によるデータベース操作可能な自然言語インタフェースが考えられる.

本研究では，図書データベースシステムの検索操作を対象に，パソコンや携帯電話からアクセス可能な自然言語インタフェース処理方式を開発することとした.

Natural Language Query of Books Database

Keiji Shinzato　Hiroshi Kinukawa

Dept. Information and Communication Engineering,
School of Engineering, Tokyo Denki Univ.
Kanda Nishikicho 2-2, Chiyoda-ku, Tokyo, 101-8457, Japan
shinzato@cll.im.dendai.ac.jp　kinukawa@im.dendai.ac.jp

The spread of PC and PDA enables many people to use many databases on the Internet. But following problems are remained.

(1) It is difficult to make retrieval query that is composed of complex logical conditions for naive user who is unfamiliar with information equipment.

(2) Screen of PDA can display less retrieval condition items than that of PC, because former's screen is narrower than the latter.

Natural language interface system that accepts natural language expression of database query can be considered as one of the solution of the above-mentioned problems.

In this study, we developed a natural language interface system that is able to access books database from PC and/or PDA.

1.はじめに

現在，パソコンは家庭に広く普及し，老若男女問わず,多くの人が利用するようになってきている.そのパソコンと人間とのインタフェースとして，GUI ベースのものが広く普及している．GUI ベースのものは便利であるが，その一方で，次の問題を抱えている.

(a) GUI は選択操作が多く，パソコン操作に不慣れなユーザには馴染みにくい.
(b) マウス操作を主とするため，マウス操作の困難なユーザにとっては使い難い.
(c) 多くの GUI を配置した画面や，GUI により階層的に整理された画面は，その意味や構成を，ユーザが一目で把握し難い.

一方，パソコンと同様に，携帯端末機器の普及も著しく,特に携帯電話の普及率は年々増加している．携帯電話の用途も多様化し，今ではインターネットに接続できる機種も多く,携帯電話から,インターネット上で公開されている,各種データベース（以下，DB と略す）の検索サービスを利用することが可能になってきている.しかし,パソコンのディスプレイと比べ,携帯電話の表示画面領域は狭いため,一画面に多くの検索項目を表示することができず,一度に多くの検索条件を指定した DB 検索を行うことはできない．だが，ユーザが求めているデータを得るためには,検索キーワードに対して検索条件を複雑に指定する必要があり,表示画面領域の狭さが問題となっている.

また,携帯電話から検索サービスを利用した場合,検索ページが階層的に構成されていると,ページを順々に追っていく必要があるため,そのつど通信コストがかかり,ユーザの負担となる．そのため，携帯電話から検索サービスを利用する場合,階層的に整理されていない検索画面が求められる.

以上より,パソコン操作が不慣れなユーザや携帯端末を利用するユーザと,計算機とのインタフェースには次の点が求められる.

(1) 日常の行動様式に合った操作が可能
(2) マウス等の操作が困難なユーザへの配慮
(3) 階層的でない一元的な構成
(4) 画面の表示領域に依存することなく，サービス提供が可能

このような点を満足する 1 つの方法として，ユーザから簡単な会話的表現による要求（この要求を自然言語問い合わせと称し，以下 NLQ と略す）を入力してもらい，その要求から DB に対するクエリを自動的に生成してくれるような自然言語インタフェース（以下，NLI と略す）が考えられる[1)~9)].

そこで本研究では，先に述べた点を満たす,図書検索用 NLI を作成し，評価することを目標とした.

本稿では，2 章で NLQ からクエリを生成するのに必要となる,自然言語問い合わせ文解析手順について述べ,3 章で自然言語問い合わせ文解析手順を施した結果から,クエリを生成する手順について述べる．その後 4 章で，2 章及び 3 章で述べた各手順の実現方法について，5 章で今回試作した NLI の構成についてそれぞれ述べ，6 章で NLI の評価及び考察を行う.

2. 自然言語問い合わせ文解析手順

NLQ からクエリを生成するにあたり，形態素解析された NLQ を，クエリが生成しやすい形に変換する操作が必要になる．以下，この操作に必要となる処理について,下記の例文を用いて述べる.

例文 1 : 芥川龍之介さんが書いた本について教えてください.
例文 2 : いのちのことば社によって出版されている本が知りたいです.

2.1 品詞情報に基づく形態素の統合処理

NLQ を茶筌[10)]で形態素解析すると，分割されすぎることがある.属性項目名や属性項目データは複合語で表現されることがあるため,分割されすぎた形態素を品詞情報に基づいて統合する処理が必要になる.

形態素の統合処理は次の手順による.

(1) 形態素列から統合処理の対象となる連続した形態素を抽出する.
(2) 抽出された形態素列を統合し、新しい形態素とする.
(3) 統合してできた新しい形態素の品詞情報や読み情報等を変更する.

例文 1 では，形態素解析の結果，属性項目データである「芥川龍之介」は，「芥川」と「龍

之介」のように, 姓と名に分割される. そこで, 隣り合った[名詞-固有名詞-人名-姓]と[名詞-固有名詞-人名-名]を NLQ の中から抽出・統合し，その品詞を[名詞-固有名詞-人名-姓名]に置き換える操作が必要となる.

2.2 属性項目名認定処理

NLQ からクエリを生成するためには，NLQ 中のどの形態素が属性項目名を意味しているかを認定する処理が必要になる.

属性項目名認定処理は、次の手順に従う.

(1) 属性項目名となる，形態素の情報を収めた用語辞書を用意する.
(2) 形態素列から各形態素を順に抽出する.
(3) 抽出された形態素が用語辞書に登録されているようであれば，抽出した形態素に属性項目名を表すフラグを付与する.

例文 1 において，形態素「書い」は，著者名を表す属性項目名である. 茶筌で形態素解析することで，その基本形が容易に得られるため，あらかじめ用意しておく用語辞書には形態素の基本形を登録するようにする．これにより，動詞等の活用に左右されることなく, 属性項目名を表す形態素の認定処理が可能となる.

例文 1 の場合では，動詞「書く」を著者名の属性項目名として登録する.

また，品詞情報も容易に得られるため，品詞情報も用語辞書に登録し, 辞書参照の際の手がかりとする．これにより，基本形のみでは判別つかない，用語の判別が可能になる.

2.3 不要語の認定処理

NLQ はクエリを生成する上で意味を持たない語（以下，不要語と呼ぶ）を含んでいる．そこで，NLQ 中に含まれる，不要語を認定し，削除することで, クエリ生成に必要な手がかり語（属性項目名及び属性項目データ名など）だけを残すことができる.

不要語認定処理は、次の手順による.

(1) 形態素列から不要語となりえる形態素を抽出する.
(2) 抽出された形態素に対し，不要語であることを表すフラグを付与する.

例文 1 において，検索対象「本」より後ろに現れる各形態素は, クエリ生成の手がかりにならない．また，「芥川龍之介」の直後に現れる敬称「さん」及び助詞「が」，助動詞「た」も手がかりにならない. そこで，これらの形態素を不要語として認定し, 必要に応じて除去することで, クエリ生成に必要な手がかり語だけを得ることが可能となる.

また，DB 操作を検索に限った場合，例文 1 や例文 2 のように, 検索対象となる形態素「本」が，NLQ 中に明示的に含まれている場合は，それより後に表れる形態素を不要語として扱うことで，NLQ の文末に表れる，願望・依頼を表す言い回しや, 命令表現などに影響を受けなくなる.

2.4 属性項目名前後にある形態素の統合処理

属性項目名を意味する形態素の前後には, その属性項目名に対する属性項目データが, 連続した形態素として現れる場合が多い. そのため, 属性項目名前後に現れる形態素列を統合する処理が必要になる.

各形態素が属性項目名を表すか否かという情報は, 用語辞書を参照することで得られるため, 属性項目名前後に現れる形態素列を統合することが可能になる.

属性項目名前後にある形態素の統合処理は次の手順に従う.

(1) 属性項目名を表す形態素を中心とし，その前後にある形態素列から統合処理の対象となる連続した形態素を抽出する.
(2) 抽出された形態素列を統合し、新しい形態素とする.
(3) 統合されてできた新しい形態素の品詞情報や読み情報等を変更する.

出版社の中には，例文 2 の「いのちのことば社」のように，社名の中に助詞を含むものも少なくない. 基本的に，助詞は不要語として扱われるが，この場合のように，属性項目名「出版社」を表す形態素「出版」の前に現れる，助詞を含む形態素列は, 属性項目データ「出版社名」である可能性があるため, 本来ならば不要語である助詞を不要語とせず, 他の形態素と統合することで, 茶筌の用語辞書に登録されていない出版社を特定できるようになる.

2.5 属性項目データ認定処理

NLQ からクエリを生成するためには，NLQ 中のどの形態素が属性項目データを表してい

るかを認定する処理が必要になる．

この処理を行う場合，属性項目データごとに用語辞書に登録していたのでは，未登録の用語が現れた時に対処できなくなるため，形態素に含まれる要素をもとにして，属性項目データの認定処理を行う．

この処理は次の手順による．

（1）形態素中の各要素をもとにして，属性項目データとなりえる形態素を抽出する．

（2）抽出された形態素に，属性項目名に対する属性項目データであることを表す，フラグを付与する．

例文 1 においては，「芥川龍之介」は著者項目データである．そこで，「[名詞-固有名詞-人名]は著者項目データである」ということを認定する処理が必要になる．

以上が，NLQ をクエリが生成しやすいように変換する操作の大別である．各操作はさらに細かい複数の処理から構成される．

3. クエリの生成手順

本研究では NLQ を，クエリ生成の手がかりとなる語と，不要語から構成されるものと考えた．そのため，NLQ の中から不要語を除去することで，属性項目名，属性項目データ等を意味する語が得られる．

また，対象システムへのクエリを生成するためには，対象システムに関する情報が必要である．対象システムの情報は，

（a）DB の構成情報

（b）対象システムの受け付けるクエリ情報

の 2 種類に分類することができる．

DB の構成情報とは，対象システムがどのようなテーブルを持ち，各テーブルはどのような要素を持つかといった情報や，テーブル間のリレーション情報などのことであり，対象システムの受け付けるクエリ情報とは，対象システムがどのような DB 操作言語で操作されるかを表す情報である．

クエリ生成処理は次の手順で行う．

（1）不要語と認定された形態素を NLQ 中から削除する．

（2）残った形態素を収集し，対象システムの情報をもとにクエリを生成する．

4. NLI の実現方法

NLI を設計する場合において考慮しなくてはならない問題としてシステムの移行性がある．システムの移行性は，以下の 2 つに分けられる．

（a）応用分野移行性

（b）対象システム移行性

応用分野移行性とは，システムが他の応用分野に，容易に移行できるかを示すものであり，対象システム移行性とは，システムが異なるスキーマの DB に容易に移行できるかを示すものである．NLI は異なる分野の DB や，異なるスキーマを持つ DB に容易に移行できなくてはならない．

この 2 つ問題を解決するために，本研究では簡易的なプロダクションシステムを作成し，2 章及び 3 章で述べた各処理を，プロダクションルール（以下，ルールと呼ぶ）により，実装することにした．そのため，ルールの記述を変更することにより，他のシステムに容易に移行することが可能になり，先に挙げた（a），（b）の 2 つの問題の解決が期待できる．

4.1 ルールの実装

ルールを実装するために，本研究ではスクリプトを作成し用いることにした．

ルールの記述形式は以下のとおりである．

〈ルール〉:=

〈条件部記述スクリプト〉〈行動部記述スクリプト〉|
〈条件部記述スクリプト〉〈ルール〉

条件記述スクリプトの条件部を記述するためには，以下の点を満たす必要があると考えた．

（1）形態素の連続を容易に記述することができる

（2）特定の条件にある形態素の抽出処理を簡便に行うことができる

これらの条件は正規表現により容易に満たすことができる．そのため，スクリプトの条件指定部分には正規表現を使用した．

条件の記述に正規表現を用いるため，各形態素をテキスト形式で表現しなくてはならず，その内部表現形式を XML 記述とした．図 1 に形態素「書い」の例を挙げる．

```
<morpheme>
    <surface_form>書い</surface_form>
    <base>書く</base>
    <reading>カイ</reading>
    <part_of_speach>動詞-自立</part_of_speach>
</morpheme>
```

図 1　XML 表現された形態素「書い」の例

4.2 スクリプトの種類

本研究では 2 章及び 3 章で挙げた各処理をルールで実装するため，表 1，2 に示す 12 種類のスクリプトを作成した. スクリプトは条件部記述用と行動部記述用に分けることができ，条件部記述用スクリプトで形態素解析後の NLQ から形態素を抽出し，行動部記述スクリプトで抽出した形態素に対して操作を施す.

4.3 ルールの記述方法

各ルールは上記の記述方法に従い，表 1，2 で定義されたスクリプトを用いて記述される. 2.1，2.3，2.4，2.5 節の各処理を，実際にルールで記述した例を，図 2 に (1)，(2)，(3)，(4) で順に示す.

2.2 節で述べた「属性項目名認定処理」は，ルール記述により実装することも可能であるが，属性項目名の同義語数が多くなると，それに比例しルールの数も多くなり，ルールの管理が難しくなるため，本研究ではルールによる実装を行わなかった.

5. NLI の構成

NLQ からクエリを生成するために，下記に述べる 5.1 から 5.3 の各処理を順次実行する. クエリ生成処理の流れを図 3 に示す.

5.1 入力文整形処理

ユーザより得た NLQ を以降の処理で操作しやすいように整形する。茶筌の標準辞書である「パトリシア木辞書」では、半角の英数字文字は受け付けられず、未知語として扱われるため、これらの文字を全角に変換する必要がある.

5.2 形態素解析

入力文整形処理を施された NLQ を茶筌により形態素列に分解する. 形態素の表層語、基本形、品詞などといった情報は CSV 形式で得られるため、それを図 1 に例示した XML 表現の形態素列に変換する.

表 1　条件部記述スクリプト一覧

条件部記述スクリプト	説明
match(*正規表現パターン*)	*正規表現パターン* にマッチする部分を抜き出す
pre-match(*正規表現パターン*)	*正規表現パターン* にマッチする部分より前方にある文字列を抜き出す
post-match(*正規表現パターン*)	*正規表現パターン* にマッチする部分より後方にある文字列を抜き出す
not-match(*正規表現パターン*)	*正規表現パターン* にマッチしない部分を抜き出す
while(*正規表現パターン*)	*正規表現パターン* にマッチする部分を抜き出し続ける

表 2 行動部記述スクリプト一覧

行動部記述スクリプト	説明
union( )	抜き出された形態素列を結合する.
delete( )	抜き出された部分をワーキングメモリから削除する.
enclose(*文字列*)	抜き出された部分を<*文字列* >…</ *文字列* >タグで囲む.
replace(*文字列*)	抜き出された部分を指定された *文字列* に置き換える.
append(*文字列*)	抜き出された部分の最後尾に *文字列* を付け足す.
setAttribute(*属性名, 属性値*)	抜き出されたタグに対して， *属性名=“属性値”* を設定する.
print( )	抜き出された部分をコマンドプロンプトへ表示する.

（1）隣接する名詞-固有名詞-人名-姓, 名を結合し名詞-固有名詞-人名-姓名とするルール

```
// 隣接する姓と名を抽出する
match((<morpheme.*>(.*¥s){4}.*名詞-固有名詞-人名-(姓|名).*¥s</morpheme>¥n){2}){
    union();
    // 結合後の品詞の書き換え処理
    match(<part_of_speach>.*</part_of_speach>){
        replace("<part_of_speach>名詞-固有名詞-人名-姓名</part_of_speach>");
    }
}
```

（2）最後の名詞以降に現れる形態素列を不要語と認定するルール

```
// 最後に現れる名詞以降の形態素列を抽出する
post-match((<morpheme(.*¥s){5}</morpheme>¥n)+
                <morpheme.*>(.*¥s){4}.*名詞.*一般.*¥s</morpheme>¥n ){
    // 不要語を表す属性をセット
    match(<morpheme.*>){
        setAttribute("isDelete", "true");
    }
}
```

（3）用語辞書未登録の出版社名を認定するルール

```
// 属性項目「出版社」を表す形態素の前にある，出版社名となりえる形態素列を抽出する
match((<morpheme.*>(.*¥s){4}.*>
        (名詞|未知語|記号|助詞-(並立|連体化)).*¥s</morpheme>¥n)+
            (<morpheme.*>(.*¥s){4}.*接尾-人名.*¥s</morpheme>¥n)?
                <morpheme.*>(.*¥s){4}.*(カラ|ガ)格.*¥s</morpheme>¥n
                    <morpheme.*code="ATTRIBUTE-PUBLISHER".*>(.*¥s){5}</morpheme>¥n ){
    // 出版社名となりえる形態素列を抽出する
    pre-match((<morpheme.*>(.*¥s){4}.*接尾-人名.*¥s</morpheme>¥n)?
                <morpheme.*>(.*¥s){4}.*(カラ|ガ)格.*¥s</morpheme>¥n
                    <morpheme.*code="ATTRIBUTE-PUBLISHER".*>(.*¥s){5}</morpheme>¥n ){
        union();
        // 結合後の品詞の書き換え処理
        match(<part_of_speach>.*</part_of_speach> ){
            replace("<part_of_speach>名詞-固有名詞-組織</part_of_speach>");
        }
    }// end of pre-match
}// end of match
```

（4）人名を表す形態素に対し属性項目「著者」のデータであることを意味するコードを付与するルール

```
// 品詞が［名詞-固有名詞-人名］で始まる形態素を抽出する
match(<morpheme.*>(.*¥s){4}.*>名詞-固有名詞-人名.*¥s</morpheme>¥n){
    // 著者データを表すことを意味するフラグを付与する
    match(<morpheme.*>){
        setAttribute("code", "DATA-AUTHOR");
    }
}
```

図 2 ルールの記述例

5.3 ルール実行処理

形態素解析され生成された XML 表現の形態素列を，プロダクションシステムに入力として与え､各ルールを適用する.ルール実行処理は，自然言語問い合わせ文解析処理とクエリ生成処理の 2 つに分けることができる．各処理はスクリプトで記述されたルールであり，システム内のルールインタプリタにより解釈･実行される．

ルールはシステムと独立しているため，新しい処理の追加や，既存の処理の削除・変更は容易にできる．また，NLI の実装対象となるシステム毎にルールファイルを変更することで，異なるシステムに対応することが可能である．

5.3.1 自然言語問い合わせ文解析処理

この処理では、2.1 から 2.5 で述べた各処理を行う．3.3 節で述べたように，属性項目名認定処理はルール実装されていないため，この処理に関しては対象となるシステムに依存してしまっている．そのため，対象システム毎に用語辞書の準備が必要である．

5.3.2 クエリ生成処理

自然言語問い合わせ文解析処理により、属性項目名，属性項目データ名等を表す属性が付与された形態素を収集し、ユーザから得た NLQ に対応するクエリを生成する．

クエリを生成する際に，次の条件を使用した．

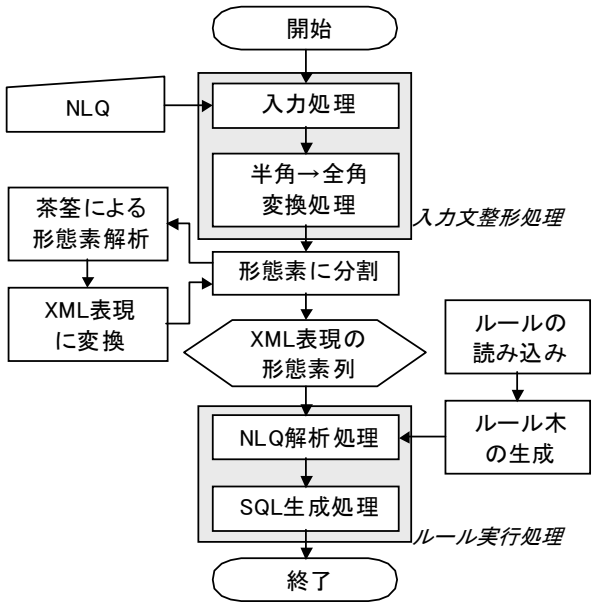

図 3　NLI の処理の流れ

(1) 属性項目名を表す語は，属性項目データを表す語と隣接している．
(2) 属性項目名を表す語は，属性項目データを表す語と不要語を挟んで隣接している．
(3) 属性項目名を表す語は省略され，属性項目データを表す語のみが現れる．

クエリ生成処理に入る前に，不要語となりえる形態素を削除することで，(1) と (2) は同等に扱うことが可能になる．(3) に関しては，属性項目名が省略される可能性の高い属性項目データを，あらかじめ用語辞書に登録し，辞書参照により，属性項目データ認定処理を行い，属性項目名の省略を対処した．

6. NLI の評価及び考察

6.1 受理可能な NLQ

今回は，図書検索における NLI を試作した．本システムで受理可能な NLQ は以下の形式のものである．

```
<NLQ>:= {(
  <属性項目名><属性項目データ>          |
  <属性項目データ><属性項目名>          |
  <属性項目名><不要語><属性項目データ>  |
  <属性項目データ><不要語><属性項目名>  |
  <属性項目データ> )<不要語>? }+
```

?:0 又は 1 回出現
+:1 回以上の連続

6.2 システムの評価

(1) 本研究では，NLQ からクエリ生成の過程を，ルールを用いて実装した．これにより次の効果が得られた．
(a) ルールを変更するだけで，他のシステムに容易に移行することできる．
(b) ルールの条件部に正規表現を用いているため，複雑な条件を含む繰り返しの指定を簡潔に記述できる．
また, この他にも 1 章で述べた (1), (2), (3), (4) の問題点も克服できる見通しを得る事ができた．
(2) その反面，次の問題点もある．
(a) ルールがその実行順序に依存しているため，ルールの変更が，他のルールに影響する可能性がある．

(b) ルールの適用が“照合ー行動”のサイクルで実行されるため，ルール数の増大により，実行効率悪化の可能性がある．

6.3 今後の課題

6.3.1 NLI としての課題

今後の課題として以下の点が残っている．

(1) 6.1 で挙げた形式以外の NLQ への対応
(2) 検索以外の問合せへの対応
(3) NLQ からクエリへの変換精度の評価
(4) インタフェースの移行性の評価
(5) インタフェースの操作性の向上

その他，ユーザとの対話管理機能による，時間軸を考慮した DB 操作の実現や，問い合わせ要求が満足できない場合における，代案提示機能の実現，問い合わせ文の曖昧性解消に有効な確認文生成機能の実現等が考えられる．

6.3.2 図書検索用 NLI としての課題

本研究では NLI を図書検索に適用したが，書籍名の認定処理は未実装である．図書検索用 NLI において，NLQ に含まれる書籍名の認定処理は重要であると考えられる．一般に書籍名中には，不要語である助詞を多く含んでいる場合が多い．そのため，任意の助詞が，書籍名中で使用されているのか，それとも NLQ を構成する上での助詞として使用されているのかを判別する必要がある．

7. おわりに

本稿では，NLI の実装方法として，プロダクションシステムによる方法を提案した．自然言語による問合せ文から DB 操作クエリを生成する手続きを，ルール記述により実装しているため，クエリ生成手順がシステムと分離でき，ルール変更により，NLI の移行性の問題に対応できた．

また，パソコン操作に不慣れなユーザ及び，携帯端末のように表示画面領域の狭い機器を利用しているユーザを対象とした DB 検索用インタフェースとして，GUI より NLI の方が有効である見通しを得ることができた．

今後は，作成した NLI の評価実験を行い，6.3 節で述べた課題に対処していく予定である．

謝辞

本研究を行うにあたり，自然言語問合せ文収集にご協力頂いた計算言語学研究員の諸氏に深く感謝いたします．

参考文献

[1]藤崎哲之助，ほか：データベース照会システム「ヤチマタ」と名詞句データ模型，情報処理学会論文誌，Vol.20，No.1，pp.77-83(1979)

[2]絹川博之：表階層モデルに基づく自然語インタフェース処理方式，情報処理学会論文誌，Vol.27，No.5，pp.499-509(1986)．

[3]杉山健司，ほか：自然言語にもとづく情報検索システム IRIS，情報処理学会研究報告 86-NL-58-8 (1986)．

[4]鳥居肖史：曖昧な入力を可能とする自然言語インタフェース，情報処理学会研究報告 94-NL-100-7 (1994)．

[5]中川優，ほか：日本語データベース検索システムにおける意味理解方式，情報処理学会論文誌，Vol.27，No.11，pp.1069-1076(1986)．

[6]牧之内顕文，ほか：移行性のあるデータベース自然言語インタフェース，情報処理学会論文誌，Vol.29，No.8，pp.749-759(1988)．

[7]渡辺日出雄，ほか：複数のアプリケーションと応答可能な自然言語インタフェース・システム，情報処理学会研究報告 88-NL-65-3 (1988)．

[8]笠晃一，ほか：日本語データベース処理システムの試作，情報処理学会研究報告 91-NL-81-6 (1991)．

[9]森本由起子，ほか：自然語インタフェースにおける対話誘導機能の設計と評価，計測自動制御学会ヒューマンインタフェース部会研究論文集，Vol.4，No.2，pp.107-114(1995)．

[10]奈良先端科学技術大学院大学 松本研究室 URL:http://chasen.aist-nara.ac.jp/index.html.ja

[11] 菅原研次：人工知能，森北出版，1997