自然言語問合せ文の意味表現方法とその応用

笠 晃一[†] 小林修二[††]
白石正人[†††] 横田将生[††††]

SQL などの形式的な問合せ言語を用いてデータベースにアクセスするのは，初心者にとって決して容易なことではない．そこで，このような問題を解決するために，自然言語を用いてデータベースに問合せできるようなシステムがこれまで数多く開発されてきた．このようなシステムの最も重要な部分は，自然言語の問合せ文を SQL 文のようなデータベース操作コマンドへと変換する過程である．そして，この変換過程で最も問題となるものの一つが，自然言語における省略現象である．すなわち，自然言語の問合せ文においても省略は頻繁に起こるため，省略部分を何らかの形で補完しない限り正確な変換は不可能である．我々は，このような省略の補完を体系的に行うために，自然言語表現と SQL 表現の中間に位置する表現を開発した．この中間的な表現のための言語を，中間意味記述言語 (IML: Intermediate meaning description language) と呼ぶ．一般に，このような中間的な表現は，自然言語表現から，また，SQL 表現へと容易に変換できるものでなければならない．IML 表現は，このような要件も十分満足するものとなっている．我々は，すでに，この IML を用いて，日本語によるデータベース問合せシステムを作成しており，その有効性を確認している．このシステムは，省略を含む自然な日本語を用いて，問合せをすることができる．

A Semantic Representation Method for Natural Language Queries and Its Application

Koichi Ryu,[†] Shuji Kobayashi,[††] Masato Shiraishi[†††] and Masao Yokota[††††]

It is not easy for elementary users to have access to a database by a formal query language such as SQL. There have been developed a lot of systems that allow such users database access by a natural language. One of the most serious problems for such systems is the ellipsis as is usual with natural languages. We have developed an intermediate semantic representation method to complete the meanings of elliptical queries systematically. The intermediate representations are located between a corresponding pair of natural language and SQL representations. The transformation of a natural language query into an SQL statement is performed very easily via the intermediate semantic representation. We have already applied our representation method to a database access system, which has proved to be a fairly good success.

1. まえがき

データベースにアクセスするには，SQL[1] などの形式的な問合せ言語を利用するのが一般的である．しかしながら，形式的な問合せ言語は抽象的かつ複雑であるため，一般人が修得するにはかなりの努力と時間を要する．また，仮に修得できたとしても，検索内容によっては，**図1**の例に示すように，非常に複雑な問合せ文を作成しなければならない．このため，自然言語によってデータベースにアクセスできるようなシステムが，今までに数多く開発されてきた[2)~4)]．

このようなシステムを構築する際に最も問題となるものの一つは，自然言語における省略現象である．理論的には，曖昧性なども「一意的に意味が定まるための情報が省略され（欠落し）ている」という省略の一症例と見なすことができる．そういう意味では，省略が最も重要な問題であると考えられる[6)]．データベースに対する自然言語の問合せ文においても，様々な形の省略が見られ，これらの省略を適切に補完することなしには，自然言語表現から形式言語表現への変換は不可能である．我々は，このような省略を体系的に処理するためには，自然言語表現と形式言語表現の間に

† 日本データワークス株式会社
Japan Data Works Co., Ltd.
†† コバ・テクニカルソフト
KOBA Technical Soft
††† 福岡教育大学教育学部技術科
Fukuoka University of Education
†††† 福岡工業大学言語情報工学研究所
Language and Information Laboratory, Fukuoka Institute of Technology

中間的な表現が必要であるとの認識に立ち，中間意味記述言語（IML：Intermediate meaning description language）を開発した．このIMLは，日本語と形式的問合せ言語SQLの中間に位置するもので，日本語の統語構造と一対一に対応する構造を有している．したがって，自然言語の問合せ文から，構成原理のみを用いてIML表現を組み立てることが可能である．また，IML表現は，目的とするSQL表現にも類似した構造を持っており，比較的容易な操作でSQL表現に変換することが可能である．

我々は，IMLの有効性を確認するために，日本語によるデータベース問合せシステムを試作した[5]．このシステムでは，実用性を考慮し，特に，統語・意味解析時間の短縮および解析に必要な作業領域の節減に留意した．たとえば，DCGで記述された統語・意味規則をSAXアルゴリズム[9]のCプログラムへと変換するトランスレータを開発して利用した．

以下，2章では，データベースに対する問合せ文に出現する省略現象についての考察を行う．問合せ文における省略を定義し，省略が局所的な省略と大域的な省略の二つに分類できることを示す．3章では，省略を適切に補完することのできる言語IMLを提案し，それによる表現が，局所的および大域的な省略の両方に対して有効であることを示す．4章では，試作した問合せシステムの概略とその特徴について述べる．

2. 問合せ文における省略

一般に，日本語は省略が生じやすい言語だと言われている．日本語の問合せ文においても同様の現象が数多く見られる．この章では，まず省略というものを定義した上で，省略が局所的省略と大域的省略の二種類に分類できることを示す．さらに，これらはそれぞれ細分類されて取り扱われる．これらの細分類は，我々がこれまでに収集した自然言語文データ（約1000件）から，そのような目的で抽出したものであり，将来の改善の余地を残している．

2.1 問合せ文における省略の定義

省略[6]というのは，何か完全なものがあって，それから何かが脱落しているという意味である．そこで，問合せ文の場合，完全なものとして，問合せ文に対応するSQL文を考えることにする．SQL文と問合せ文を比較して，SQL文にはあるが問合せ文にはないような情報が存在するとき，問合せ文において省略が起こっているとみなすのである．すなわち，問合せ文における省略は次のように定義される．

［定義1］ 問合せ文における省略

自然言語による問合せ文Xが与えられたとき，Xの意図するSQL文をYとする．このとき，YにはあるがXにはないような情報が存在するならば，そしてこのときに限り，Xには省略があるという．また，Yにおけるそのような情報をXに関する省略要素という．

問合せ文の省略について考える場合，対応するSQL文から省略のない完全な問合せ文を作成してみると理解しやすい．たとえば，次のS1のような問合せ文を考えてみよう．

（S1） 身長が平均より低い男の人の名前は？

これは，C1のようなSQL文に対応していると考えられる．ただし，ここでは関係表は一つだけしかないと仮定し，from節を省略している．

```
（C1） select 名前
       where 身長 < (select avg(身長))
       and 性別="男"
```

納入業者

業者番号	業者名	所在地
S1	大崎商会	福岡
…	…	…

部品

部品番号	部品名	色
P1	ボルト	青
…	…	…

納入表

業者番号	部品番号	個数
S1	P2	2000
…	…	…

日本語問合せ文：

すべての部品を納入している業者は？

ＳＱＬ問合せ文：

```
select 業者名
from 納入業者
where not exists
  (select *
   from 部品
   where not exists
     (select *
      from 納入表
      where 業者番号＝納入業者.業者番号
        and 部品番号＝部品.部品番号))
```

図1 自然言語検索文と対応するSQL文の例
Fig. 1 A retrieval sentence and its corresponding SQL statement.

この SQL 文を省略のない完全な問合せ文に変換してみると，F1 のようになるだろう．

（F1） 身長が平均身長より低い性別が男の人の名前は？

ここで，S1 と F1 を比較することにより，S1 では F1 の下線部が省略されていることがわかる．したがって，これらの下線部と対応する C1 の下線部が S1 に関する省略要素ということになる．

2.2 局所的省略

問合せ文における省略が，他の問合せ文から独立しているとき，これを局所的省略と呼ぶことにする．前節で述べた例は，局所的省略の一種である．我々は現在のところ，局所的省略をさらに下のように細分類して取り扱っている．

①主語としての属性名が省略される場合
②属性値の間に論理的結合がある場合
③組を表す句との比較を含む場合
④集計関数を表す句との比較を含む場合

以下，各場合について説明する．

2.2.1 主語としての属性名が省略される場合

次の S2 および S3 のような問合せ文では，主語としての属性名が省略されていると考えることができる．

（S2） 福岡の人の名前は？
（S3） 150 センチより低い人の名前は？

これらの文に対応する SQL 文は，それぞれ C2 および C3 のようになる．

（C2） select 名前
where 住所="福岡"
（C3） select 名前
where 身長<150

これらの SQL 文に対応する完全な問合せ文は，それぞれ F2 および F3 のようになると考えられる．

（F2） 住所が福岡の人の名前は？
（F3） 身長が 150 センチより低い人の名前は？

これらの文，F2 および F3 を，それぞれ，S2 および S3 と比較することによって，F2 および F3 の下線部が省略されていることがわかる．つまり，主語としての属性名が省略されているのである．なお，C2 と C3 の下線部は対応する省略要素である．

2.2.2 属性値の間に論理的結合がある場合

属性値，あるいは，それに比較を表す名詞（「以上」や「以下」など）が連接したものが論理的結合を表す助詞や接続詞で結ばれている場合にも省略が起こっているとみなすことができる．ただし，我々の文法では，同じタイプの属性値が論理的結合を表す助詞や接続詞で結ばれて句が構成されると仮定している．たとえば，次のような文 S4 および S5 を考えてみよう．

（S4） 住所が福岡か熊本である人の名前は？
（S5） 年齢が 20 以上かつ 30 以下の人の名前は？

これらの文では，「か」「かつ」のような論理的結合を表す助詞や接続詞が用いられており，対応する SQL 文は C4 および C5 のようになる．

（C4） select 名前
where 住所="福岡" or 住所="熊本"
（C5） select 名前
where 年齢>=20 and 年齢<=30

これらの SQL 文に対する完全な問合せ文を考えると，F4 および F5 のようになるだろう．

（F4） 住所が福岡か住所が熊本である人の名前は？
（F5） 年齢が 20 以上かつ年齢が 30 以下の人の名前は？

これらの文において，下線部は，S4 や S5 において省略された部分を表しており，したがって，C4 と C5 の下線部が省略要素であることがわかる．

2.2.3 組を表す句との比較を含む場合

次に示す問合せ文 S6 は，組を表す句「名前が宮崎である人」との比較を含んでいる．

（S6） 名前が宮崎である人より給与とボーナスの和が多い人の名前は？

この文は，「給与とボーナスの和」を「名前が宮崎である」という条件を満足する組の「給与とボーナスの和」と比較していると考えられ，対応する SQL 文は C6 のようになる．

（C6） select 名前
where 給与＋ボーナス>
(select 給与＋ボーナス
where 名前="宮崎")

この C6 に対する完全な問合せ文は，F6 のようになるであろう．

（F6） 名前が宮崎である人の給与とボーナスの和より給与とボーナスの和が多い人の名前は？

下線部は，S6 において省略された部分であり，C6 の下線部が S6 に関する省略要素となる．

2.2.4 集計関数を表す句との比較を含む場合

集計関数を表す句との比較を行っている問合せ文に

も省略が見られることがある．たとえば，次のような文 S7 を考えてみよう．

（S7） 給与とボーナスの和が平均の2倍より多い人の名前は？

ここでは，「平均」という名詞が集計関数を表しており，対応する SQL 文は C7 のようになる．

（C7） select 名前
where 給与＋ボーナス〉
(select avg(給与＋ボーナス)＊2)

この SQL 文に対する完全な問合せ文は次の F7 のようになるだろう．

（F7） 給与とボーナスの和が給与とボーナスの和の平均の2倍より多い人の名前は？

ここに，下線部は S7 において省略されている部分であり，C7 の下線部が省略要素ということになる．

2.3 大域的省略

問合せ文における省略が，他の問合せ文に依存しているとき，これを大域的省略と呼ぶことにする．つまり，他の問合せ文の意味を考慮しないと，その問合せ文の省略内容が補完できないとき，大域的省略と呼ぶのである．大域的省略と考えられるものの中から，いくつかを次に示す．

2.3.1 検索条件が追加されるもの

前文の検索条件に新たに条件を追加して，検索を要求する場合がある．条件の追加は，連言的になされる場合と選言的になされる場合とに分けられる．まず，条件が連言的に追加される場合を例文 S8 および S9 を用いて考えてみよう．

（S8） 年齢が20以上の人は？

（S9） そのうち趣味が読書であるのは？

これらに対応する SQL 文は次の C8 および C9 のようになるだろう．

（C8） select ＊
where 年齢 〉= 20

（C9） select ＊
where 年齢 〉= 20 and 趣味 = “読書”

ここで，C9 の下線部の年齢が 20 以上という条件は S9 を解析するだけでは得られず，前文の S8 に対応する C8 を参照する必要がある．したがって，S9 では大域的省略が起こっており，C9 の下線部が省略要素だと考えることができる．

次に，検索条件が選言的に追加される場合の例 S10 および S11 を考えてみよう．

（S10） 趣味が読書の人は？

（S11） 趣味が音楽の人も表示せよ．

これらの文に対応する SQL 文は，C10 および C11 のようになる．

（C10） select ＊
where 趣味 = “読書”

（C11） select ＊
where 趣味 = “読書” or 趣味 = “音楽”

ここでも，C11 の下線部の条件は，S11 を解析するだけでは得られず，大域的省略が起こっている．

2.3.2 表示項目が追加されるもの

次の問合せ文 S12 および S13 は，表示項目が追加されるような例であるが，ここにも大域的省略が見られる．

（S12） 年齢が20歳以上の人の名前は？

（S13） 趣味も表示せよ．

これらに対する SQL 文は，次の C12 および C13 のようになるだろう．

（C12） select 名前
where 年齢 〉= 20

（C13） select 名前，趣味
where 年齢 〉= 20

ここで，S13 に出現している情報は「趣味」だけであり，対応する C13 を完成させるためには，これ以外の情報を C12 から補完する必要がある．つまり，C13 の下線部を省略要素と考えることができる．

3. 中間意味記述言語 IML

この章では，局所的省略と大域的省略の両方を適切に補完することのできる中間意味記述言語を提案する．この中間的な言語を我々は，IML (Intermediate meaning description language) と呼んでいる．IML 表現から SQL 表現への変換は，後述する変換関数を用いて行われる．省略要素の補完はこの変換処理の段階で行われる．

3.1 IML の定義

中間意味記述言語 IML は，SQL への変換を容易にするため，これによく似た構文を持っている．ただし，自然言語表現との対応づけが容易になるように，省略をうまく表現できるような工夫がなされている．**図2**は，IML の定義を BNF 記法で行ったものである．この定義に従って，自然言語表現を IML 表現へ変換してみよう．たとえば，前章の局所的省略を含んでいる問合せ文 S5 および S6 は，それぞれ IML 表現 M5 および M6 に変換される．

(S5) 年齢が20以上かつ30以下の人の名前は？

(S6) 名前が宮崎である人より給与とボーナスの和が多い人の名前は？

(M5) select 名前
where 年齢：(〉=20 and 〈=30)

(M6) select 名前
where (給与＋ボーナス)：
〉tuples(one, 名前："宮崎")

また，大域的省略を含んでいるS8およびS9の場合は，IML表現M8およびM9に変換される．

(S8) 年齢が20以上の人は？

(S9) そのうち趣味が読書であるのは？

(M8) select ＊
where 年齢：〉=20

(M9) select ＊
where (pcond and 趣味："読書")

これらのIML表現を，対応するSQL表現C5, C6, C9と比較すれば明らかなように，IML表現はもとの自然言語表現の構造をそのまま反映しており，自然言語表現において省略された情報は，補完されないままになっている．

3.2 変換関数τによる省略要素の補完

中間意味記述言語IMLによる表現では，問合せ文において省略された情報はそのままになっている．そのような省略情報は，τと呼ぶ一変数関数によってIML表現がSQL表現に変換される段階で補完される．この変換関数τの一部を**図3**に示す．この図において，関数τでは不可能な文脈依存の変換を行うため，二変数関数αが補助的に使用されている点に注意されたい．以下では，前章で述べた省略の分類に従って，それぞれの省略要素が変換関数τによってどのように補完されるのかを示す．

3.2.1 主語としての属性名の補完

主語としての属性名が省略された問合せ文の例として，前出のS2について考えてみよう．

(S2) 福岡の人の名前は？

この文のIML表現は，次のM2のようになる．

(M2) select 名前
where nil："福岡" [住所]

ここに，「福岡」という属性値の直後の鉤括弧の中の「住所」は，この属性値が関係表において属している属性を表している．ここでは，このような属性値を属性名付き属性値と呼ぶ．このIML表現の条件部は以下のように，SQL表現へ変換される．なお，各式の末尾につけられた鉤括弧内の記号は，その式を導出するのに使われた図3の規則を表している．

```
(G1)  <属性名>::=名前|身長|性別|住所|年令|給与|ボーナス|趣味|...
(G2)  <属性値>::="男"|"福岡"|"熊本"|"宮崎"|"読書"|"音楽"|2|20|30|150|...
(G3)  <算術演算子>::= +|-|*|/
(G4)  <集計関数子>::= sum|avg|max|min
(G5)  <量化子>::= one|all|some
(G6)  <論理演算子>::= and|or
(G7)  <比較演算子>::= =|!=|>|>=|<|<=
(G8)  <属性名付き属性値>::=<属性値>[<属性名>]
(G9)  <属性値式>::=<属性値>|<属性名付き属性値>|(<属性値式><算術演算子><属性値式>)
(G10) <属性名式>::=<属性名>|(<属性名式><算術演算子><属性名式>)|
                    (<属性名式><算術演算子><属性値式>)|(<属性値式><算術演算子><属性名式>)
(G11) <集計関数>::=<集計関数子>(<属性名式>)
(G12) <集計式>::=<集計関数>|<集計関数子>|(<集計式><算術演算子><集計式>)|
                  (<集計式><算術演算子><属性値式>)|(<属性値式><算術演算子><集計式>)
(G13) <被選択式>::=<属性名式>|<集計式>
(G14) <選択式>::= <量化子>(select <被選択式>)|<量化子>(select <被選択式> where <全条件式>)
(G15) <組集合>::= tuples(<量化子>,<全条件式>)
(G16) <基本式>::=<属性値式>|<属性名式>|<集計式>|<選択式>|<組集合>
(G17) <基本条件式>::=<基本式>|(not <基本条件式>)|(<基本条件式><論理演算子><基本条件式>)
(G18) <半条件式>::=<比較演算子><基本条件式>|
                    (not <半条件式>)|(<半条件式><論理演算子><半条件式>)|
                    (<半条件式><論理演算子><基本条件式>)|(<基本条件式><論理演算子><半条件式>)
(G19) <被比較式>::=<属性名式>|nil
(G20) <全条件式>::=<被比較式>:<基本条件式>|<被比較式>:<半条件式>|pcond|
                    (not <全条件式>)|(<全条件式><論理演算子><全条件式>)
(G21) <属性名式リスト>::=<属性名式>|<属性名式リスト>,<属性名式>
(G22) <集計式リスト>::=<集計式>|<集計式リスト>,<集計式>
(G23) <追加リスト>::= append(<属性名式リスト>)| append(<集計式リスト>)
(G24) <選択リスト>::=<属性名式リスト>|<集計式リスト>|<追加リスト>
(G25) <選択文>::= select <選択リスト>| select <選択リスト> where <全条件式>
```

図2 中間意味記述言語IMLの定義
Fig. 2 Definition of intermediate meaning description language, IML.

τ(nil : "福岡" [住所])
= τ(nil : ="福岡" [住所]) [R4]
= (住所 = "福岡") [R16]

また，次の S3 は IML 表現 M3 へと変換される．

(S3) 150センチより低い人の名前は？
(M3) select 名前
where nil : 〈= 150 [身長]

S3 において，「150センチ」のような名詞句は，「センチ」という接尾語から，たとえば「身長」という属性名が推定され，属性名付き属性値に変換されている．そして，M3 の条件部は，次のようにして SQL 表現へと変換される．

τ(nil : 〈= 150 [身長])
= (身長 〈= 150) [R16]

このように，主語としての属性名が省略されている場合には，属性名付き属性値によって属性名が補完される．

3.2.2 論理的結合がある場合の補完

前出の S5 は，属性値の間に論理的結合が存在する例である．この文の IML 表現 M5 の条件部は，次のようにして SQL 表現に変換される．

τ(年齢 : 〉=20 and 〈=30)
= (τ(年齢 : 〉=20) and τ(年齢 : 〈=30)) [R19]
= (年齢 〉= 20 and 年齢 〈= 30)
[R1, R2, R5]

この例から推察されるように，属性値の間に論理的結合が存在する場合，省略要素は分配法則によって補完される．

3.2.3 組や集計関数を表す句に対する補完

前出の S6 は，組を表す句との比較を含んだ問合せ文の例である．この文の IML 表現 M6 の条件部は，規則 (R15) などを用いることで，省略のない SQL 表現へと変換される．

τ((給与＋ボーナス) :
〉 tuples (one, 名前 : "宮崎"))
= τ((給与＋ボーナス))
〉 τ(one) (select τ((給与＋ボーナス))
where τ(名前 : "宮崎")) [R15]
= (給与＋ボーナス)
〉 (select (給与＋ボーナス)
where 名前 = "宮崎")
[R1, R2, R3, R4, R5, R11]

また，S7 は集計関数を表す句との比較を含んでいる問合せ文であるが，これは M7 のような IML 表現へと変換される．ここでも，IML 表現が自然言語の問合せ文の構造を反映している点に注意されたい．

(R1) τ(<属性値>) = <属性値>, τ(<属性値>[<属性名>]) = <属性値>
(R2) τ(<属性名>) = <属性名>
(R3) τ((<属性名式>$_1$<算術演算子><属性名式>$_2$)) = (τ(<属性名式>$_1$)<算術演算子>τ(<属性名式>$_2$))
(R4) τ(<被比較式>:<基本式>) = τ(<被比較式>: =<基本式>)
(R5) τ(<属性名式>:<比較演算子><属性値式>) = τ(<属性名式>)<比較演算子>τ(<属性値式>)
(R6) τ(<属性名式>$_1$:<比較演算子><属性名式>$_2$) = τ(<属性名式>$_1$)<比較演算子>τ(<属性名式>$_2$)
(R7) τ(<属性名式>:<比較演算子><集計式>) = τ(<属性名式>)<比較演算子>(select α(<属性名式>, <集計式>))
(R8) α(<属性名式>$_1$, <集計関数子>(<属性名式>$_2$)) = <集計関数子>(τ(<属性名式>$_2$))
(R9) α(<属性名式>, <集計関数子>) = <集計関数子>(τ(<属性名式>))
(R10) α(<属性名式>, (<集計式><算術演算子><属性値式>))
= (α(<属性名式>, <集計式>)<算術演算子>τ(<属性値式>))
(R11) τ(one) = <空文字列>, τ(all) = all, τ(some) = some
(R12) τ(<属性名式>:<比較演算子><選択式>) = τ(<属性名式>)<比較演算了>α(<属性名式>, <選択式>)
(R13) α(<属性名式>$_1$, <属性名式>$_2$) = τ(<属性名式>$_2$)
(R14) α(<属性名式>, <量化子>(select <被選択式> where <全条件式>))
= τ(<量化子>)(select α(<属性名式>, <被選択式>) where τ(<全条件式>))
(R15) τ(<属性名式>:<比較演算子>tuples(<量化子>, <全条件式>))
= τ(<属性名式>)<比較演算子>τ(<量化子>)(select τ(<属性名式>) where τ(<全条件式>))
(R16) τ(nil:<比較演算子><属性値>[<属性名>]) = <属性名><比較演算子><属性値>
(R17) τ(<被比較式>:(<基本条件式>$_1$<論理演算子><基本条件式>$_2$))
= (:(<被比較式>:<基本条件式>$_1$)<論理演算子>τ(<被比較式>:<基本条件式>$_2$))
(R18) τ(<被比較式>:<比較演算子>(<基本条件式>$_1$<論理演算子><基本条件式>$_2$))
= (τ(<被比較式>:<比較演算子><基本条件式>$_1$)<論理演算子>τ(<被比較式>:<比較演算子><基本条件式>$_2$))
(R19) τ(<被比較式>:(<半条件式>$_1$<論理演算子><半条件式>$_2$))
= (τ(<被比較式>:<半条件式>$_1$)<論理演算子>τ(<被比較式>:<半条件式>$_2$))
(R20) τ(pcond) = <直前の問合せ文に対するSQLの条件部>
(R21) τ((<全条件式>$_1$<論理演算子><全条件式>$_2$)) = (τ(<全条件式>$_1$)<論理演算子>τ(<全条件式>$_2$))
(R22) τ(<属性名式リスト>, <属性名式>) = τ(<属性名式リスト>), τ(<属性名式>)
(R23) τ(append(<属性名式リスト>)) = <直前の問合せ文に対するSQLの選択リスト>, τ(<属性名式リスト>)
(R24) τ(select <選択リスト> where <全条件式>) = select τ(<選択リスト>) where τ(<全条件式>)

図 3 IML から SQL への変換関数 τ (部分)
Fig. 3 Some part of definition of the function τ which transforms IML into SQL.

(S7) 給与とボーナスの和が平均の2倍より多い人の名前は？

(M7) select 名前
where (給与＋ボーナス) : 〉(avg ＊ 2)

そして，M7の条件部は規則 (R7) や (R9) などを用いることによって，SQL 表現へと変換される．

τ((給与＋ボーナス) : 〉(avg ＊ 2))
= τ((給与＋ボーナス)) 〉
(select α((給与＋ボーナス), (avg ＊ 2)))
[R7]
= (給与＋ボーナス) 〉
(select (α((給与＋ボーナス), avg)＊ τ(2)))
[R2, R3, R10]
= (給与＋ボーナス) 〉
(select (avg(給与＋ボーナス)＊2))
[R2, R2, R3, R9]

3.2.4 大域的省略における補完

まず，検索条件が追加される次の問合せ文について考えてみよう．これらは，M10 および M11 の IML 表現に変換される．

(S10) 趣味が読書の人は？
(S11) 趣味が音楽の人も表示せよ．
(M10) select ＊
where 趣味 : “読書”
(M11) select ＊
where (pcond or 趣味 : “音楽”)

このうち，M10 は C10 のような SQL 表現へと変換される．

(C10) select ＊
where 趣味 = “読書”

そして，M11 の条件部は C10 の条件部を用いて，次のように変換されることになる．

τ((pcond or 趣味 : “音楽”))
= (τ(pcond) or τ(趣味 : “音楽”)) [R21]
= (趣味 = “読書” or 趣味 = “音楽”)
[R1, R2, R4, R5, R20]

ここでは，図3の規則 (R20) が重要な役割を果たしている．

次に，表示項目が追加される例を考えてみよう．S12 および S13 は，それぞれ，M12 および M13 のような IML 表現へと変換される．

(S12) 年齢が20歳以上の人の名前は？
(S13) 趣味も表示せよ．
(M12) select 名前
where 年齢 : 〉=20
(M13) select append(趣味)
where pcond

このうち，M12 は C12 のような SQL 表現に変換される．

(C12) select 名前
where 年齢 〉= 20

M13 は C12 を用いて，次のように変換される．

τ(select append(趣味) where pcond)
= select τ(append(趣味)) where τ(pcond)
[R24]
= select 名前，趣味
where 年齢 〉= 20 [R2, R20, R23]

この変換では，図3の規則 (R23) が重要な役割を果たしている．

4. 日本語問合せシステムへの応用

我々の試作した日本語によるデータベース問合せシステム[5]の統語・意味規則は DCG 記法[7]を用いて記述されている．そこで，DCG 記法で書かれた規則を実行可能なプログラムへと変換するトランスレータも併せて開発した．本章では，試作したシステムの概要とこのトランスレータについて述べる．

4.1 問合せシステムの構成

問合せシステムは，形態素解析部，統語・意味解析部，意味トランスレータ，エコーバック生成部，および，知識獲得部から構成されており，それら相互の関係およびデータベース管理システムとの関係は，**図4**に示すようになっている．この図において，形態素解析部と統語・意味解析部は分離して描かれているが，処理的には，この二つの部分は同時並行的に動作する．以下，各部分について簡単に説明する．

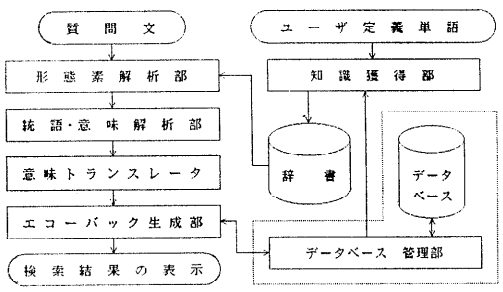


図4 データベース問合せシステムの全体構成
Fig. 4 Configuration of the database query system.

（1） 形態素解析部

ここでは，入力された日本語問合せ文を最長一致法を用いて単語に分解し，各単語の統語的・意味的情報を次の統語・意味解析部に引き渡している．解析用辞書の構造としては，TRIE 構造[11]に改良を加えたものを使っている．

（2） 統語・意味解析部

ここでは，形態素解析部から引き渡された情報をもとにして，中間意味記述言語 IML による表現を作成する．我々の場合，IML 表現は計算機上では素性構造によって表現されている．作成された IML 表現の例を**図5**に示す．

（3） 意味トランスレータ

ここでは，統語・意味解析部で得られた IML 表現を変換関数 τ を用いて SQL 表現へと変換する．もとの問合せ文において省略されている情報は，この段階で補完される．問合せ文に曖昧性が検出された場合，推論処理によって解消が試みられる．それが失敗するものに対しては，次のエコーバック生成部でユーザに問い返すことによって解決する．

（4） エコーバック生成部

意味トランスレータによって得られた SQL 文を曖昧性のない日本語に変換してユーザに提示するのが，エコーバック生成部の働きである．エコーバック生成部は，入力文が正しく解釈されたかどうかをユーザに確認させることと曖昧性の解消の二つの役割を持っている．

（5） 知識獲得部

システムは，知識獲得部を通して，いろいろな知識を外部から学習することができる．知識には，大別して，データベースから得られる知識とユーザから得られる知識の二種類がある．

4.2 SAX-C トランスレータ

我々は，統語・意味解析にかかる時間を短くし，解析に必要な作業領域を小さくするため，次のような工夫を行った．

（a） 統語・意味規則の記述に DCG 記法[7]を用いた．このことにより，統語処理と意味処理が同時並行的に行えるようになり，統語解析の段階で発生する曖昧性による組合せ的爆発を抑制することができるようになった．

（b） DCG 記法を実行可能なプログラムへと変換するのに SAX トランスレータ[9]を用いた．これは，得られる実行可能プログラムの実行効率やメモリ効率が，DCG トランスレータや BUP トランスレータ[8]を用いた場合よりも優れているからである[10]．

（c） SAX トランスレータが出力する実行可能プログラムの記述言語としてC言語を採用した．もともとの SAX トランスレータは，出力される実行可能プログラムを記述するために，論理型言語 Prolog を用いているが，実行効率やメモリ効率をさらに上げるために，C言語を用いるように改造した．

この結果完成したのが SAX-C トランスレータである．これは，DCG で記述された統語・意味規則を SAX アルゴリズムのCプログラムへと変換する．なお，SAX-C の出力するCプログラムと SAX の出力する Prolog プログラムについて比較実験を行った．その結果，前者の方が実行速度において 10 倍程度速いこと，また，作業領域においては後者の約半分程度で済むことがわかった．

```
質問文：身長が平均より低い宮崎さんは
[1]
[where: [cat: [main: tuples
               sub: complex]
         quan: one
         cond: [op: and
                cat: [main: logop]
                arg1: [arg1: [cat: [main: a_name
                                    sub: factor]
                              type: number
                              value: 身長]
                       cat: [main: colon]
                       arg2: [arg: [cat: [main: statis
                                          sub: functor]
                                    op: avg
                                    type: number]
                              op: lt
                              cat: [main: singop]]]
                arg2: [arg1: [cat: [main: a_name
                                    sub: factor]
                              value: 名前]
                       cat: [main: colon]
                       arg2: [cat: [main: a_value
                                    sub: factor]
                              type: char
                              value: 宮崎]]]]
 cat: [main: select-statement]]

SQL:
select *
where (身長 < (select avg(身長)) and 名前 = "宮崎")

number of objects: 17
ambiguity        : 1
```

図5 統語・意味解析部の出力例と対応する SQL 文
Fig. 5 An output example of the syntax and semantics analysis part and its corresponding SQL statement.

5. むすび

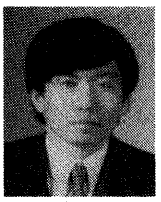
自然言語の問合せ文における省略を適切に補完するための中間意味記述言語 IML とその応用システムについて述べた．応用システムは，現在，NEC のパーソナル・コンピュータ PC-9801 RA 上にインプリメントされ，本論文中の例文のような標準的問合せ文を数秒以内で処理することが確認されている．また，省略を含んだ自然な問合せ文を受理することができ，このシステムをデータベースに関しては全くの初心者である 20 名ほどの大学生に使用してもらったが，使用感はおおむね好評であった．しかしながら，このシステムは現在のところ一つの関係表に対する問合せしか行えず，複数の表に対しても問合せが行えるようにすることが将来の課題として残っている．また，大域的省略の取扱いが不十分であり，IML をこの部分に関して拡張する必要もあるだろう．

謝辞 最後に，この研究を行う機会を与えてくださった日本データワークス株式会社の今西社長をはじめ，研究を御助力いただいた方々に深く感謝いたします．

参考文献

1) Date, C. J.: *An Introduction to Database Systems*, Addison-Wesley (1975).
2) 藤崎，間下，諸橋，渋谷，鷺尾：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. 1-24 (1979).
3) 服部，宮本，吉川，前田，千葉：データベースの日本語インターフェースにおける対話処理について，電子通信学会技術研究報告，Vol. 86, No. 216, pp. 19-23 (1986).
4) 伊藤，高橋：データベース用自然言語インタフェース「dBmate」，第 38 回情報処理学会全国大会論文集，No. 2, pp. 981-982 (1989).
5) 笠，小林，横田：日本語データベース処理システムの試作，情報処理学会自然言語処理研究会，81-6, pp. 41-48 (1991).
6) 横田将生：省略を含む自然言語談話の理解処理について，第 3 回情報処理学会九州支部研究会報告 (1989).
7) Pereira, F. C. N. and Warren, D. H. D.: Definite Clause Grammars for Language Analysis: A Survey of the Formalism and a Comparison with Augmented Transition Networks, *Artif. Intell.*, Vol. 13, pp. 231-278 (1980).
8) Matsumoto, Y., Tanaka, H., Hirakawa, H., Miyoshi, H. and Yasukawa, H.: BUP: A Bottom-Up Parser Embedded in Prolog, *New Generation Computing*, Vol. 1, No. 2, pp. 145-158 (1983).
9) 松本，杉村：論理型言語に基づく構文解析システム SAX，コンピュータソフトウェア，Vol. 3, No. 4, pp. 4-11 (1986).
10) 杉村，奥西，松本，田村，上脇，田中，清野：ロジックプログラミングをベースにした自然言語解析システムの比較，情報処理学会自然言語処理研究会，57-2 (1986).
11) Aho, A. V., Hopcroft, J. E. and Ullman, J. D.: *Data Structures and Algorithms*, Addison-Wesley (1983).

(平成 4 年 8 月 27 日受付)
(平成 5 年 2 月 12 日採録)

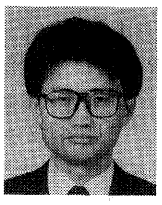


笠　晃一（正会員）

昭和 56 年九州大学工学部電子卒業．昭和 58 年同大学院修士課程修了．昭和 61 年(株)日本データベースネットワーク研究所入社．平成 4 年日本データワークス(株)入社，現在に至る．自然言語処理，神経回路網，カオスなどの研究に従事．日本認知科学会会員．

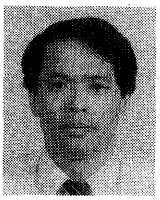
小林　修二

昭和 53 年有明工業高等専門学校電気工学科卒業．昭和 63 年(株)日本データベースネットワーク研究所に入所．同所で自然言語処理の研究に従事し特に辞書，形態素解析を担当している．



白石　正人（正会員）

昭和 59 年九州大学総合理工学研究科修士課程修了．自然言語処理および機械学習の研究に従事．現在福岡教育大学助手．



横田　将生（正会員）

昭和 47 年九州工業大学工学部電子卒業．昭和52年九州大学大学院博士課程修了．同年同大工学部助手．昭和54年同大医学部附属病院講師．昭和 63 年福岡工業大学工学部電子工学科教授．現在，同大工学部情報工学科教授．言語および図形の意味理解，医療・生体情報処理，思考心理実験の研究に従事．工学博士．電子情報通信学会，人工知能学会，認知科学会各会員．