ホーン節を内部表現とする自然言語理解システム†

西原典孝†† 森田憲一††

まず，自然言語理解とは，“自然言語の文を何らかの内部表現に変換し，知識として蓄え，疑問文に対して，その知識を基に推論を行い答を出す”ことである，と定める．本研究では，この内部表現として一階述語論理式の subset であるホーン節を用い，PROLOG 上でこの全過程を実行するシステムを実現した．対象言語としては，英語の subset をとった．このシステムは次のことを行う．1）平叙文が入力されれば，それをホーン節に変換し知識として蓄え，2）疑問文が入力されれば，その知識を基に推論を行い論理的答を導出し，3）その論理的答と疑問文の構文情報を基に，正しい答文を生成し出力する．筆者らの目的は，応用的システムを作成することではなく，むしろ，自然言語理解自体の問題点を明確にし，その解決を試みることにあり，本システムはそのような試みを実行するためのツールとして位置づけられる．また本論文では，副詞と指示的な冠詞“the”の取扱い，および答文生成についての問題も考察し，本システムでのそれらの解決方法を述べた．

1. まえがき

計算機による自然言語理解にはいろいろなアプローチがありうるが，ここでは，“自然言語の文を何らかの内部表現に変換し，知識として蓄え，疑問文に対して，その知識を基に推論を行い答を出す”ことである，と定める．そして，本研究では，その内部表現として一階述語論理式の subset であるホーン節を用い，PROLOG 上でこの全過程を実行するシステムを実現した．対象言語としては，英語の subset をとった．

ホーン節を内部表現に用いて自然言語を処理するやり方は，近年よく用いられる手法である．その代表的なものとして Pereira による Chat-80 システムがある[1]．しかし，Chat-80 は，知識ベース（データベース）に対する英語の問合せ文を処理することに主眼を置いたシステムであり，平叙文から知識ベース表現への変換，蓄積を必要としていない．また，内部表現の中で取扱われる個体対象はすべて固有名詞か数値であった．結局，このシステムでは，自然言語理解における，より一般的かつ基本的な問題の多くが省みられることなく残されていたといえる．

筆者らの目的は，自然言語理解に基づく応用システムを作成することではなく，むしろ，自然言語理解自体の問題点を明確にし，その解決を試みることにある．そして，そのような試みを実行するためのツールとして，自然言語から知識ベース表現への変換，蓄積，および疑問文に対して得られた解からの適切な答文の生成等を含めた自然言語理解システムを作成した．

また本論文では，副詞および指示的な冠詞“the”の取扱いについても論じ，本システムでのそれらの解決方法を述べた．

2. システムの構成

システムは，次のような五つの部門からなる（**図1**）．

1) 翻訳部門
2) ホーン節化部門
3) 登録部門
4) 推論部門
5) 答文生成部門

入力された文は 1) により一階述語理論式に翻訳される．対象言語としては英語の subset を用いる．生成された一階述語理論式は 2) により，スコーレム変換され，ホーン節に直される．そして平叙文に対するホーン節は 3) に送られ知識ベースとして蓄えられる．その表現はまったく PROLOG のプログラムと同一である．一方，疑問文に対するホーン節は 4) を起動させる．4) は，送られてきたホーン節を入力とし，知識ベースに基づき論理的推論を行い答を出す．5) は，4) で生成された論理的答と元の疑問文の構文情報に基づき，英語の答文を生成し，出力する．

全システムは，前述したように，PROLOG 上で実現されている．この PROLOG は大阪大学計算機センター ACOS-1000 LISP 1.9 上でインプリメントしたものである．

† Natural Language Understanding System Which Uses Horn Clauses as Its Internal Knowledge Expression by NORITAKA NISHIHARA and KENICHI MORITA (Faculty of Engineering Science, Osaka University).

†† 大阪大学基礎工学部生物工学科

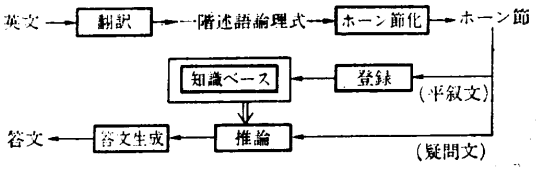

図 1 システムの構成
Fig. 1 System configuration.

3. 翻訳部門

本システムで取扱い可能な自然言語の範囲とその論理式への翻訳方法は，統語規則と翻訳規則により定められている．実際の翻訳は，統語規則に翻訳規則を埋め込んだ形をした DCG (definite clause grammar)[2] により実現している．

この部門に英文を入力すると，それは論理式に翻訳され，その結果と入力文の構文情報（平叙文，疑問文の区別，および主語，動詞の内容等）が続く部門に送られる．構文解析に失敗した場合，すなわち，入力文が統語規則に適合しない場合，システムは，“I can't understand.” というメッセージを出力する．

3.1 入力文の範囲と文法的制限

統語規則により定められている入力可能な英文の範囲は，おおよそ次のようになる．まず，形容詞，副詞，代名詞，接続詞，関係代名詞等が許され，時制は現在形のみで，複数形や否定文は許されない．疑問文は YN 疑問文 (Yes, No 疑問文) および，主語ないし目的語に相当するものを尋ねるWH 疑問文 (What 疑問文) である．なお，WH 疑問文の疑問詞は What のみであり，Who, Whom もすべて What で代用している．

しかし，まったく同一の文でも，曖昧性を含む文は複数の読みをもつ．そこで，システムの入出力をスムーズに行うために，文法的制限を設け，一つの文には一つの読みにしかできないようにしている．そのいくつかの例を示す．

文法的制限 1

副詞は直前の動詞のみに係り，前置詞句による副詞句は動詞句全体に係る．

たとえば，次の文では，“slowly” は “talks” のみに係り，“in the park” は “walks and talks” に係る．

John walks and talks slowly in the park.

文法的制限 2

代名詞が指示しているものは，その性と対応し，その代名詞よりも前でかつ最も近くにある文中の名詞句とする．

文法的制限 3

冠詞による限量化の範囲は，文全体とし，左優先とする．

3.2 基本語句の翻訳

基本語句の翻訳方法は，次のようになる．

	英語表現	翻訳
普通名詞…一項述語	man	man(x)
形容詞…一項述語	red	red(x)
固有名詞…個体定数	john	john
副詞…副詞定数	slowly	slowly
前置詞…副詞関数	in	in(x)
自動詞…二項述語	run	run(x, ad)
第1引数：主語に相当する個体変数		
第2引数：副詞リスト		
他動詞…三項述語	love	love(x, y, ad)
第1引数：主語に相当する個体変数		
第2引数：目的語に相当する個体変数		
第3引数：副詞リスト		

英文から論理式の変換における従来の方法では，自動詞を一項，他動詞を二項述語に翻訳している．本システムでは，これに副詞リストという引数を追加し，副詞情報を動詞に対応する述語に埋め込むようにした．これは，副詞により生じる論理の高階性を避けるためである．なお，冠詞 “a” は存在限量化を行うもの，“every” は全称限量化を行うものとして翻訳する．冠詞 “the” については次の節で述べる．

3.3 冠詞 the を伴う名詞句 ([the]＋普通名詞句) の解釈

i）その普通名詞句が関係代名詞節を伴う場合

冠詞 “the” は冠詞 “a” とまったく同一なものとして解釈する．

ii）その普通名詞句が関係代名詞節を伴わない場合

冠詞 “the” は指示的な “the” であり，その名詞句は，それ以前に蓄えられた知識中の，ある個体表現を直接指示しているものと見なす．たとえば，“John loves a woman.” という事実が知識の中に存在するとき，

Bill loves the woman.

の “the woman” は “John loves a woman.” 中の “a woman” と同一のものを指示しているものとする．このような取扱いは次のような手続きにより実現される．

普通名詞句の翻訳を C(x) とすると，←C(x) をゴール節とし，これが知識ベースから導けるかどうかを推論する．もし導けるならば，そのときのxに対するインスタンスを“the woman”の翻訳とする．そのようなインスタンスが複数個あるならば，最初に見つかったもの，すなわち最新に入力された知識に対応するものを採用する．もし導けない場合は，この“the”は“a”と解釈的に同一であると見なし，かつその“the”による限量化は最優先であるとする．

《翻訳例》

(i) john walks slowly in summer.
walk (john, [slowly, in (summer)])

(ii) john eats the fish that a woman finds.
$(\exists V_1)(\exists V_2)$ (eat (john, V_1, nil)∧
fish (V_1)∧find (V_2, V_1, nil)∧
woman (V_2))

ここで“eat”，“find”には副詞が係っていないので，副詞リストは nil (=空リスト) となっている．

4. ホーン節化部門

この部門では，生成された一階述語論理式を，まずスコーレム変換して，スコーレム標準形に直し，さらに節表現に直す．そして，その結果であるホーン節リストは，入力文が平叙文ならば登録部門に送られ，入力文が疑問文ならば推論部門に送られる．

《文からホーン節への翻訳例》

(i) john eats the fish that a woman finds.
fish (c_0).
woman (c_1).
eat (john, c_0, nil).
find (c_1, c_0, nil).

(ii) the woman loves john.
love (c_1, john).
“the woman”が指示しているものが c_1 となっている．

(iii) does john eat a fish?
← eat (john, V_1, nil), fish (V_1)

(iv) what does john eat?
← eat (john, ANS, nil).
“ANS”は“what”に相当する個体変数を表す．

5. 登録部門

登録部門では，ホーン節化部門から送られてきた平叙文に対するホーン節，および答文生成のための名詞句情報を知識ベースへ登録する．すべての登録が終了すると，システムは“I see.”というメッセージを出力する．

5.1 ホーン節の登録

平叙文に対するホーン節集合は知識として蓄えられる．否定文が許されていないため，このホーン節集合は，宣言節ないし規則節から成り，ゴール節は含まれていない．これらのホーン節は PROLOG プログラムとまったく同一であり，事実，システムは PROLOG のプログラムとしてこれらを登録する．

5.2 名詞句情報の登録

本部門では，答文生成のために，存在限量詞化されている変数に対して与えられたスコーレム定数が具体的に何を指すのかの情報の登録も行う．これについてはここでは述べず，7章で詳述することにする．

6. 推論部門

疑問文に対するホーン節はこの部門に送られてくる．推論部門は，このホーン節に対して，知識ベースを基に推論を行い論理的答を導出する．

6.1 推論手続き

推論部門において行うべきことは，(『知識ベース』→『疑問文が尋ねている事実』) を証明することであり，疑問文が WH 疑問文ならば，この式を成り立たせしめる what に相当する変数のインスタンスを求めることである．一方，本システムでは，自然言語に対する内部表現としてホーン節を使用している．このため，ホーン節集合=PROLOG プログラム，ホーン節集合に対する推論=PROLOG プログラムの実行，という性質により，本部門での推論は，知識ベースと疑問文に対するホーン節集合をそのまま PROLOG プログラムとして実行することで実現できる．

疑問文に対するホーン節集合は，一般に次のようなものになる．

P_1.
⋮
P_n. $n \geq 0$ P_i は宣言節ないし規則節，
← G. ← G はゴール節

これらのホーン節に対する推論の手続き（今後，推論手続きと呼ぶ）は，次のようなものである．

$P_1, \cdots, P_n$ を知識ベースに追加登録する．この新たな知識ベースであるホーン節集合とゴール節“←G”を PROLOG プログラムとして実行する．もし“←

G" が成功すれば，推論手続きは成功したと呼ぶことにし，成功しなければ，推論手続きは失敗とする．ゴール節の評価を終えると，$P_1, \cdots, P_n$ を知識ベースから削除し，推論手続きは終了する．

6.2 付加推論規則

推論部門では，導出原理の適用だけでは不十分な点を，付加推論規則でおぎない，推論能力を高めている．

たとえば次の文について考えてみる．

john runs slowly.

does john run ?

これらの文に対するホーン節はそれぞれ，

run (john, [slowly]).

← run (john, nil).

となる．この場合，疑問文に対するゴール節は第2引数が異なるため成功しない．しかし，一般には，この疑問文に対しては Yes と答えるのが妥当であろう．そこで次の付加推論規則を設ける．

付加推論規則

動詞名 ($x_1, \cdots, x_n$, 副詞リスト')

→

動詞名 ($y_1, \cdots, y_n$, 副詞リスト)　$n=1, 2$

が成り立つのは，

動詞名 ($x_1, \cdots, x_n$) → 動詞名 ($y_1, \cdots, y_n$)

が成り立ち，かつ

副詞リスト'⊇副詞リスト

のときである．

この付加推論規則を実現するには，疑問文に対するゴール節，

← $Q_1, Q_2, \cdots, Q_n$.　　($n \geq 1$)

の述語 Q_i が，"動詞名 ($x_1, \cdots, x_n$, ad)" である場合，Q_i の代りに二つの述語，"動詞名 ($x_1, \cdots, x_n, AD_1$), include (AD_1, ad)" をゴール節に挿入し，これを新たなゴール節とすればよい．ここで "include (x, y)" は x⊇y を意味する述語である．

6.3 推論部門の動作

YN 疑問文に対する推論

YN 疑問文に対するホーン節集合に対して，推論手続きを行い，この手続きが成功したならば，答は Yes となり，失敗したならば No となる．このように推論は閉世界の立場で考え，疑問文が知識ベースから証明できない場合，すなわち，知識として知らない事柄を尋ねられた場合，No と答える．

WH 疑問文に対する推論

WH 疑問文に対するゴール節を "←G (ANS)" とおく．"ANS" は "what" に相当する変数である．このゴール節が成功するような "ANS" に対するインスタンス（今後，これを解と呼ぶ）をすべて求めなければならない．そこで，ゴール節を次のように変更し，推論手続きにはいる．

← G (ANS), regist-ans (ANS, P), false.

ここでPは，G (ANS) を構成している述語 $Q_1, \cdots, Q_n$ を要素とするリスト $[Q_1, \cdots, Q_n]$ である．regist-ans (ANS, P) が呼出された時点では，ANS には解がユニファイされ，Pには，P中の各変数に具体的な値がユニファイされている．このPは答文生成に必要とされる．regist-ans (ANS, P) は，ANS とPの値をグローバルに蓄えておく述語である．この述語は，述語の定義や削除を行う組込み述語を使用して構成している．一つの解が見つかると述語 false により，強制的にバックトラックが起こり別の解の探索にかかる．この手順が繰返され，最終的に推論手続きは失敗に終る．しかしすべての解は，解のリストとして別に蓄えられている．

6.4 実 行 例

every man walks.

john loves mary.

john loves sarry.

これらの文に対するホーン節は，

walk (V_1, nil) ← man (V_1).

love (john, mary, nil).

love (john, sarry, nil).

である．いま，これらのホーン節が知識ベースに蓄えられているとする．

(i) "does every man walk ?" に対するホーン節，

man (a_1).

← walk (a_1, nil).

が推論部門に送られてきた場合，推論手続きは成功し，答えは Yes となる．

(ii) "what does john love ?" に対するホーン節，

← love (john, ANS, nil).

が送られてきた場合，解は，mary と sarry，となる．

7. 答文生成部門

答文生成部門では，推論部門の結果と，翻訳部門で

生成された疑問文の構文情報を基に，疑問文に対する答えを正しい英文で出力する．YN 疑問文に対しては，推論部門の推論手続きが成功したならば Yes，失敗したならば No と答える．WH 疑問文に対しては，推論部門で求められた解のリストを用いて，答文を生成する．ただし解のリストが nil ならば，システムは“I don't know.”と出力する．

7.1 スコーレム定数に対する名詞句生成

平叙文“bill eats a fish.”に対するホーン節，

eat (bill, c_0, nil).

fish (c_0).

が知識ベースに蓄えられているとき，疑問文“what does bill eat?”に対する解のリストは [c_0] である．ここで c_0 は，スコーレム定数である．

答文生成のためには，この c_0 が具体的に指す名詞句をまず生成しなければならない．スコーレム定数に対する名詞句生成は，登録部門ですでに登録されている名詞句情報というものを使用して行う．名詞句情報とは，スコーレム定数が発生したごとに，登録部門で登録されるそのスコーレム定数に関する情報で，これは，

c_i ([c_i に関するホーン節集合]).

c_i はスコーレム定数

の形式をした宣言節の形で登録されている．ここで，c_i に関するホーン節集合は，次のように定義する．

スコーレム定数に関するホーン節集合 1

そのスコーレム定数が，もともと，冠詞“a”を伴う名詞句に対応するものならば，その名詞句が含まれている文に対応するホーン節集合の中で，そのスコーレム定数を引数として含む宣言節の集合．

スコーレム定数に関するホーン節集合 2

そのスコーレム定数が，もともと，冠詞“the”と関係詞を伴う名詞句に対応するものならば，その名詞句に対応するホーン節集合の中で，そのスコーレム定数を引数として含む宣言節の集合．

《スコーレム定数に対する名詞句情報の例》

(i) bill eats a fish.

ホーン節：eat (bill, c_0, nil).

fish (c_0).

名詞句情報：

c_0 ([eat (bill, c_0, nil), fish (c_0)])

(ii) john eats the fish that he finds.

ホーン節：eat (john, c_1, nil).

find (john, c_1, nil).

fish (c_1).

名詞句情報：

c_1 ([find (john, c_1, nil), fish (c_1)])

ただし，現システムでは，スコーレム関数は取扱えない．たとえば次のような文では，

every man loves a woman.

love (V_1, f_1 (V_1), nil) ← man (V_1).

woman (f_1 (V_1)) ← man (V_1).

love の目的語が“a woman”に対するスコーレム関数 f_1 (V_1) となっている．ある疑問文に対して，このような f_1 (α) が解となった場合 (α は V_1 のあるインスタンスとする)，名詞句生成はできず，現段階では答として f_1 (α) をそのまま返している．

7.2 名詞句生成の手続き

推論部門で導出された解に対する名詞句を生成する手続きは，次のようなものである．ここで，その解を q とし，それに対して生成される名詞句は，その名詞句を構成する単語のリスト NL で表すことにする．

q が元の疑問文中にある場合，q の性と格に従った代名詞に変換する．そうでない場合，

q＝固有名詞　の場合　NL＝[q]

q＝スコーレム定数　の場合，次のようにする．

まず，そのスコーレム定数に対するホーン節集合を取出す．このようなホーン節は，そのスコーレム定数を述語名とする述語の引数の形で，すでに知識ベース中に蓄えられている．それらを基に名詞句 NL を生成する．ここで，各ホーン節は宣言節，すなわち一つの述語表現となっている．ホーン節の中に述語名が動詞であるものが存在するときには，関係代名詞節で名詞句を構成する．さらにその述語の各引数に対しては，この名詞句生成手続きを再帰的に呼出し，各引数を名詞句に直す．ただし副詞情報は名詞句生成に用いない．

たとえば，7.1 節で例示した c_1 に対する名詞句 NL は次のようになる．

NL＝[the, fish, that, john, finds]

7.3 名詞句生成における冗長性の除去

7.2 節で例示した平叙文に対するホーン節が知識ベース中にあるとき，疑問文“what does john finds?”(そのホーン節は，← find (john, ANS, nil).) に対する解は c_1 である．c_1 に対して生成される名詞句は“the fish that john finds”であり，“that john finds”の部分は，冗長なものになる．

さらに次の例について考えてみる．

john seeks the woman that finds a unicorn.
seek (john, c_2, nil).
find (c_2, c_3, nil).
woman (c_2).
unicorn (c_3).
c_2 ([find (c_2, c_3, nil), woman (c_2)])
c_3 ([find (c_2, c_3, nil), unicorn (c_3)])

c_2 に対する名詞句 NL は,

NL=[the, woman, that, finds, {c_3 に対する名詞句}]

c_3 に対する名詞句 NL は,

NL=[the, unicorn, that, {c_2 に対する名詞句}, finds]

となる．このように，c_2 に対する名詞句には，c_3 に対する名詞句を含み，かつその逆も成り立っているので，名詞生成手続きは，無限ループに陥ってしまう．

このような冗長性は除去しなければならない．推論部門で導出されたおのおのの解は，[解, P] の形で蓄えられている．冗長性を除去するために，このPを利用し，名詞句生成手続きを次のように変更する．

まず，除去すべきホーン節集合を表すリスト DL を用意し，初めに DL=P とおいて，名詞句生成手続きに入る．そして，q=スコーレム定数の場合，それに対するホーン節集合の中で，述語名が動詞でかつその述語と同一のものが DL 中に存在するかをチェックし，もし存在すれば，その述語は名詞句生成に用いないようにする．さらに，名詞句生成手続きを再帰的に呼出す際に，その名詞句生成手続きに対する DL には，この DL と，最初に選び出されたホーン節集合を連接したリスト DL′ を使用する．

このように変更した名詞句生成手続きにより生成される名詞句 NL は，次のようになる．

c_1 に対して,
NL=[a, fish]
c_2 に対して,
NL=[the, woman, that, finds, a, unicorn]
c_3 に対して,
NL=[the, unicorn, that, a, woman, finds]

7.4 答文生成例

(i) mary kills the man that loves her.
kill (mary, c_0, nil).
love (c_0, mary, nil).
man (c_0).
what does mary kill ?
← kill (mary, ANS, nil).

この場合，解は c_0 で，生成される c_0 に対する名詞句と答文は，それぞれ次のようになる．

the man that loves her
she kills the man that loves her.

(ii) the woman that finds a red fish eats it.
finds (c_1, c_2, nil).
eat (c_1, c_2, nil).
woman (c_1).
red (c_2).
fish (c_2).
what eats the fish ?
← eat (ANS, c_2, nil).

この場合，解は c_1 で，生成される c_1 に対する名詞句と答文は，それぞれ次のようになる．

the woman that finds it.
the woman that finds it does.

8. むすび

知識ベースへの知識の登録，スコーレム定数に対する名詞句生成等を含めた，英文の入力から出力までの首尾一貫した自然言語理解を行う基本システムを作成できた．さらに，本来，ホーン節だけでは記述が困難であった対象，たとえば，副詞や，冠詞“the”を伴う名詞句に対して，付加推論規則を設けたり，手続き的な取扱いを織り混ぜることによって，その取扱いの実現に成功した．また，本システムは，知識表現として論理式をとっているため，論理的評価，考察が容易であるとともに，柔軟性と拡張性に富んだものとなっており，より能力のある自然言語理解システム開発のためのコアシステムになると考えられる．

しかし，現システムで取扱える英文の範囲は，まだ狭いものである．とくに，否定文を取扱えない．これは，PROLOG においては，否定の表現であるゴール節が，質問ととらえられ，否定の表現ができないからである．否定文の取扱いを含めて，より枠の広い自然言語の取扱いを実現するために，メタロジカルな機能の強化，ないし内部表現として用いる論理体系自体の拡張等を現在検討中である．

参考文献

1) Pereira, F.: Logic for Natural Language Analysis, Ph. D. Thesis, Univ. of Edinburgh (1982).
2) Pereira, F. and Warren, D.: Definite Clause

Grammar for Language Analysis—A Survey of the Formalism and a Comparison with Augmented Transition Networks, *Artif. Intell.*, Vol. 13, No. 1, 2, pp. 231-278 (1980).
3) 西原典孝：ホーン節を内部表現とする自然言語理解システム，大阪大学修士論文 (1984).

付録　システムの実行例

+JOHN RUNS SLOWLY IN SUMMER.
I SEE.
+DOES JOHN RUN SLOWLY?
YES, HE DOES.
+DOES JOHN RUN FAST?
NO, HE DOESN'T.
+JOHN LOVES MARY AND SHE LOVES BILL.
I SEE.
+MARY IS A WOMAN.
I SEE.
+DOES JOHN LOVE THE WOMAN THAT LOVES BILL?
YES, HE DOES.
+JOHN FINDS A RED LARGE UNICORN AND EATS IT.
I SEE.
+THE WOMAN THAT TOM LOVES KILLS HIM.
I SEE.
+WHAT DOES JOHN EAT?
HE EATS THE RED LARGE UNICORN THAT HE FINDS.
+WHAT FINDS A RED UNICORN?
JOHN DOES.
+WHAT KILLS TOM?
THE WOMAN THAT HE LOVES DOES.
+SARRY LOVES THE MAN THAT RUNS IN THE PARK.
I SEE.
+THE MAN LOVES KATE.
I SEE.
+BEN FINDS A BLUE FISH.
I SEE.
+MARY EATS THE FISH.
I SEE.
+WHAT DOES THE MAN THAT RUNS IN THE PARK LOVE?
HE LOVES KATE.
+WHAT DOES BEN FIND?
HE FINDS A BLUE FISH.
+WHAT DOES MARY EAT?
SHE EATS THE BLUE FISH THAT BEN FINDS.

(昭和59年10月1日受付)
(昭和60年3月20日採録)