

# 解説



## 意味処理を導入した機械翻訳について†

### —融合方式—

田中穂積<sup>††</sup> 溝口文雄<sup>†††</sup>

#### 1. はじめに

言語の機械処理における大きな問題の一つに、言語に含まれる曖昧さをいかにして解消するかという問題がある。たとえば機械翻訳を行う場合、「彼は彼女にいった」に含まれる「いった」は、「行った」と解釈して翻訳してはならない。これは同形異義語のレベルでの曖昧さの解消が必要なることを意味している。

I cut the paper with this knife. という文は、統語解析のレベルでは、二通りの意味で曖昧である。[pp with this knife] が、動詞 cut に関係する構造 ([vp [vp cut [np the paper]] [pp with this knife]]) と、名詞句 the paper に関係する構造 ([np [np the paper] [pp with this knife]]) とが得られるからである。さらに言えば、cut の時制が現在か過去かといったレベルの曖昧さがある。これらの曖昧さを解消しない限り、翻訳結果に誤りが含まれることになる。本論文は、この種の問題をサーベイした論文<sup>†</sup>のなかのいくつかを解決しようとしたものである。

#### 2. 機械翻訳における曖昧さの解消

第1章で触れた翻訳における曖昧さの解消には、意味解析が重要な役割を果たすことは明らかだろう。最初の文の「いった」を「行った」と解釈すべきでないということは、動詞「行く」の格構造のなかの助詞「に」を伴う名詞句は、場所に関連するものでなければならない、という解析により明らかになる。これは、統語解析ではなく意味解析の問題である。次に、with this knife が the paper に関係する統語構造を意味的に異常な文であるとして排除するためには、「ナイフ付きの紙」がこの世に存在しないという常識を計算

機が持つ必要がある。これには意味解析を越えた語用論的な解析が含まれている。本稿では、英語の基本動詞を含む文の機械翻訳に焦点をあてて、英語の基本動詞を適切な日本語に訳し分ける方式の開発と実験結果を述べている。英語の基本動詞 (get, make など) は、さまざまな日本語に訳し分ける必要がある。そのためには、比較的深い意味解析が必要になる。英語を日本語に翻訳する場合には、動詞とともに共起する名詞句に、どのような助詞を付加すべきかも、意味を考慮して決める必要がある。一例をあげれば

I take my child to school.

では、目的語が「人間」で、school が「場所」であることを知り、take を「連れていく」と訳さなければならない。また、to school には助詞「へ」を付けて、「私は私の子供を学校へ連れて行く」のように訳さなければならない。ところが、

I take my child to her.

の場合には、to her の her が人間であるから助詞「へ」を付けることはできない。to her は、「彼女のところへ」と訳すべきであろう。

以上の例は（統語解析のレベルではなく）意味解析によりさまざまな曖昧さを解消してはじめて適切な日本語への翻訳が可能になることを示している。それを可能にするためには、意味解析を行う柔軟な機構と、意味を柔軟に記述する枠組とが必要になる。われわれは SRL とよばれる柔軟な意味記述の枠組と、そのインタプリタをすでに開発している<sup>2)</sup>。これらをベースに意味解析と翻訳システムとを融合した新しい翻訳アルゴリズムを開発している。これは融合方式と名付けられている。融合方式は、翻訳に関するさまざまな曖昧さを解消するために、意味解析を中核に据えた新しい機械翻訳の方式である。本稿では融合方式による機械翻訳システムの有効性を検討する。

† Semantics based Machine Translation System by Hozumi TANAKA (Tokyo Institute of Technology) and Fumio MIZOGUCHI (Science University of Tokyo).

†† 東京工業大学情報工学科

††† 東京理科大学経営工学科

### 3. 融合方式による機械翻訳

#### 3.1 基本的な考え方

機械翻訳システムには大別して次の三つの処理過程がある。

- (1) 解析 (analysis),
- (2) 変換 (transfer),
- (3) 合成 (generation).

(1)の解析は、翻訳システムへの入力となる文(ソース文)を解析し、文の統語構造や意味構造の抽出を行う。(2)の変換は、(1)で得られた解析結果を変換し、翻訳結果(ターゲット文)が合成しやすい構造に作りかえたり(構造変換)、訳語の選択を行う(語彙変換)。(3)の合成は、(2)の変換結果からターゲット文を作り出す過程である。ソース文とターゲット文の語順の違いを調整したり、用言を語尾変化させることなどがこの過程で行われる。

機械翻訳システムで最も良く使われる方式はトランスファ方式である。トランスファ方式では、解析、変換、合成をそれぞれ独立した処理過程として扱う(図-1)。

トランスファ方式で翻訳システムを実現する場合、解析に意味解析を含ませることが最近では多くなってきている。一方、変換過程で行う訳語の選択には、意味解析が含まれている。融合方式の着想は、こうした重複を避けるために、意味解析時に訳語の選択を行い、意味解析過程と変換過程とを融合させることにあった。さらに、ソース文の統語解析結果から、ただちに主語や目的語の所在を知ることが可能で、モンテギュー文法のように、統語解析結果に沿った意味解析を行う方式をとれば、意味解析の過程で、主語や目的語の順序を入れ替えて、ターゲット文を作り出すことが

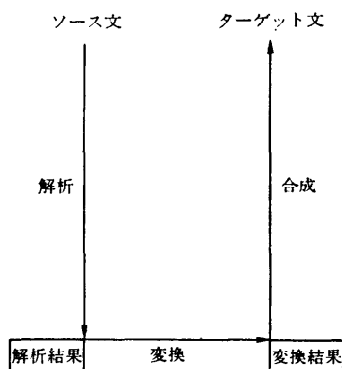


図-1 トランスファ方式による機械翻訳

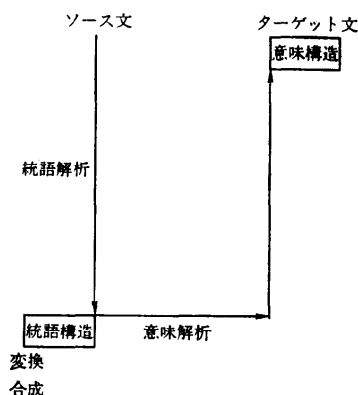


図-2 融合方式による機械翻訳

可能になる。融合方式は、こうした考え方を具体化した機械翻訳の方式である。したがって融合方式による機械翻訳では、ソース文の意味解析が終了するとともに翻訳結果としてターゲット文が得られる。融合方式の概念図を図-2 に示す。

融合方式による英日機械翻訳システムには次の特徴がある。

(1) 統語解析結果に沿った意味解析と変換とを行うため、ソース文(英語)の表層レベルの情報を利用したきめ細かい翻訳が可能である。

(2) 意味解析を行い、意味構造を抽出する過程で訳語を選択する。

(3) 意味解析の過程で統語解析結果から主語、述語、目的語の位置を知り、ターゲット文(日本語)の語順に並べ変えたり、名詞に適切な助詞を付加することができる。

(4) 意味解析が進むにつれて訳語が正確になる。このことは、融合方式は人間の行っている翻訳方式に近いとも考えられる。

融合方式についても、トランスファ方式と同様に、ソース言語とターゲット言語の対を考慮した文法と辞書が必要になるので、多言語間翻訳システムの実現には問題がある。しかし、融合方式で用いる辞書は、ソース言語の統語解析に必要な部分と、意味解析と変換、合成に必要な部分が明確に分離しているので、ターゲット言語が変わっても前者の部分はそのまま使える。また、文法規則も、ソース言語の統語解析に必要な部分と、意味解析、変換、合成を行う部分とが、明確に分離しているので、ターゲット言語が、変わっても前者の部分がそのまま使える。さらに、変換は辞書の部分の記述を中心に行うため、文法規則に付

加した意味処理用のプログラムも、ターゲット言語が変わっても、そのまま使えるという特徴がある。

最後に融合方式では、ソース文の統語解析により、主語、目的語の位置を知ることができなければならない。そのために、

(5) ソース言語として語順が比較的自由的な日本語などの言語を融合方式により他言語に翻訳する場合には問題がある。ただし、ターゲット言語として、語順の自由的な言語がくる場合には(4)に述べたように問題は少ない。

### 3.2 融合方式における語彙変換

融合方式における語彙変換は、意味解析時に行う。

図-3に、意味解析と語彙変換を同時に行う辞書項目の記述例を示す。図-3(a)は、TAKE が動詞 Vであることを記述している。図-3(b)は、TAKE の詳細な構文的特徴(他動詞 VT であり、ホーンビー文型パターンが VP6 A, ...である)を記述している。(c)以下は、SRL<sup>3)</sup>による意味記述である。SRLによる意味記述の枠組は unit (ユニット、意味の単位)とよぶ。これを機械翻訳用の辞書として用いるため、ユニット名には、デフォルト訳語が付随している。図-3(c)では TAKE のデフォルト訳語が「取る」であることを示している。

図-3(e)~(l)は1つのスロット記述である。スロット名は obj で、これは、このスロットを満たすことができるユニット(フィラー)は、文中で目的格の位置を占めるものでなければならないという(フィラーに対する)統語的制約条件を記述している。obj 以降には、以下で述べる(フィラーに対する)意味的制約条件と、アクションの対をいくつか並記する。語彙変換はアクションを起動して行う。

図-3(g)~(l)は、フィラーに対する意味的制約条件とアクションの対の記述例である。図-3(g)の ANI-

MAL は、フィラーの意味的制約条件を示している。言い代えると、フィラーが文中で目的格の位置を占めるとともに、その意味的性質が ANIMAL である場合には、このフィラーは、obj スロットを満たすことができる、ということを示している。そしてこのとき、関数 @HONYAKU (図-3(h))と @WAIT-AND-SEE (図(i)~(l))を実行することができる。これらを用いて、when-filled method もしくはアクションの起動とよぶことにする。語彙変換は、これらのアクションを起動して行うが、その方法を以下で説明する。そのために、次の例文を考えてみる。

(i) I took books to school.

(私は本を学校へ持っていった。)

(ii) I took you to school.

(私はあなたを学校へ連れていった。)

(iii) I took you for John.

(私はあなたを John だとみなした。)

文(i), (ii)は、目的語が ANIMAL か ANIMAL でないかに応じて、take をそれぞれ「持っていく」、「連れていく」のように訳さなければならないことを示している。一方文(ii)と(iii)の目的語は同じであるから、後続する前置詞句の相違によって take の訳語を変えなくてはならないことを示している。また(ii)の to school を to my uncle に変えると、「私の叔父へ」ではなく「私の叔父のところへ」のように、名詞句に付加する助詞を変えなければならない。このような訳語の選択は、先に述べたアクションを起動して行うことになるが、説明を簡単にするために、文(ii)と(iii)について考えてみる。

文(ii), (iii)ともに目的語は you で、その意味的性質は ANIMAL であるから、図-3の記述を見ると、you は(e)~(l)に示すスロットにつけられた意味的制約条件(g)を満たすことが可能であることが分かる。そこでアクションとして関数

(@HONYAKU \* WO 連 RETEIK \* U)

を実行する((h)行)。この関数の第一引数\*は、フィラーの訳語を変えずに、フィラーに助詞「を(WO)」を付けて、take の訳語を「連れていく」のように変えることを指示している。次に図-3(i)~(l)に示すアクション、すなわち関数 @WAIT-AND-SEE を実行する。この関数は、人工知能の研究で使われるデモンを作り出すもので、第一引数の(j)に示す統語的ならびに意味的制約条件を満たすスロットが作成されたかどうかを監視する。したがって、文(iii)の場合に、前

- (a) (TAKE V
- (b) ((VT (VP6 A.....) 0)
- (c) '(TAKE 取 R \* U) unit
- (d) .....
- (e) (obj (.....)
- (f) .....
- (g) (ANIMAL
- (h) (@HONYAKU \* WO 連 RETEIK \* U)
- (i) (@WAIT-AND-SEE
- (j) (for ANIMAL
- (k) (@HONYAKU \* DATO
- (l) (みな S \* U)))
- (m) .....

図-3 融合方式の機械翻訳で用いる辞書項目の記述例

置詞句 for you の意味処理が進み、それに対するスロットが作り出されると、先に作成したデモンが動作し、図-3 の(k)~(l)に示すアクションを起動する。それにより、you の訳語をそのままにして助詞「だと(DATO)」を付加し、take の訳語を「連れていく」から「みなす」に変えることを指示する。動作後のデモンは自動消滅する。

融合方式による語彙変換は、以上のように意味解析と並行して行われ、意味解析が進むにつれて訳語が次第に正確になることが理解できよう。そしてその鍵となる考え方は、関数 @WAIT-AND-SEE を実行してデモンを作成することにある。ここで注意しておきたいことは、図-3 の(j)~(l)に示すデモンは、スロットの記述形式そのものであるから、図-3(k)~(l)に示すアクション部に関数 @WAIT-AND-SEE を記述し、さらに別のデモンを起動させることもできる。このような @WAIT-AND-SEE の何重にもネストした記述により、語彙変換に影響を及ぼす複雑な条件を必要なだけ調べることができる。

以上と同様なことは、take に対して、大略次の変換規則を用いることでも実現できる。

- (イ) (obj(N1=ANIMAL) &  
for(N2=ANIMAL))  
⇒N1 を N2 だとみなす。  
(ロ) (obj(N1=ANIMAL))  
⇒N1 を連れていく。

以上の変換規則の適用にあたり注意すべきことは、条件の厳しい(イ)から順に適用しなければならないことである。(イ)の適用に失敗したら、条件のゆるい(ロ)を適用する。この場合には、各変換規則の適用にあたって、同じ条件検査が繰り返されるのが問題である。たとえば、(イ)、(ロ)ともに (obj(N1=ANIMAL)) という条件検査が含まれている。

本節で述べたデモンを用いる方法は、条件のゆるいものから順に変換規則を適用していると考えられることができるから、同じ条件検査を繰り返さないようにすることができる。

### 3.3 融合方式による構造成変換

パーザとしてわれわれは拡張 LINGOL を用いている<sup>4),5)</sup>。拡張 LINGOL では、統語解析に用いる各文脈自由法規則に対して、意味解析プログラムを付加(補強)することができる。その一例を図-4 に示す。

図-4 の(a)~(c)は

VP → V (OBJ1) (OBJ2)

- (a) (VP (V . . . .))  
(b) @ (OBJ1 . . . .)  
(c) @ (OBJ2 . . . .)  
. . . .  
(d) (@VP  
(e) (INTERP 'OBJ1)  
(f) (INTERP 'OBJ2)  
(g) (FMB 'V)  
(h) '(OBJ1 OBJ2 V)))

図-4 融合方式の機械翻訳で用いる文法規則の記述例

という文法規則に対応している。図-4(d)~(h)に示す関数 @VP が意味解析を行う補強プログラムである。(e)~(g)は、OBJ1 と OBJ2 に集められた意味情報と V に集められた意味情報を用いて意味解析を行うための記述である。意味解析中に、前節 3.2 に述べた語彙変換が行われる。関数 @VP の第4引数は、OBJ1 OBJ2 V の順に訳語を並べて、意味解析結果と cons して関数 @VP の値とすることを指示している。このような記述により、語順のならべかえなどの構造成変換を行うことができる。関数 @VP の実行結果は、VP→V (OBJ1) (OBJ2) の VP に送られることになる。

ここで次の注意をしておく。図-3 の(a)~(b)、図-4 の(a)~(c)の記述は英語のみに依存した記述、図-3 の(c)~(m)、図-4 の(d)~(h)は言語対(英語と日本語)に依存した記述であり、両者は明確に分離英語から他の言語へ翻訳する場合に、英語に依存した記述は辞書、文法ともにそのまま使える。これが2章の後半に述べたことであった。

### 3.4 融合方式による機械翻訳の実験について

機械翻訳というと、たいていの場合、大規模な計算機上でシステム開発がなされている。ところが、言語の曖昧さを処理したり、また、意味の記述をどのように表現したりする実験では、必ずしも、大規模の辞書や文法を準備しなくてもよい。むしろ、見通しのよい処理モデルを仮定して、どのような形で翻訳が進行していくのか、また、われわれ、人間が試行するような形で翻訳が進むためには、どのような意味選択がなされていくのかについて透明な実験システムが必要である。

こうした考え方を背景に、大型計算機上で作成された融合方式による機械翻訳システムをマイクロコンピュータ上でも実験できるかを検討してみた<sup>6)</sup>。

機械翻訳の基礎となる構文解析部は拡張 LINGOL を用いた。まず、8 bit のマイクロコンピュータでの実験システムでは沖電気の if 800 モデル 30 で、CP/M 上の Stiff Upper Lisp を用いて作成された。使用

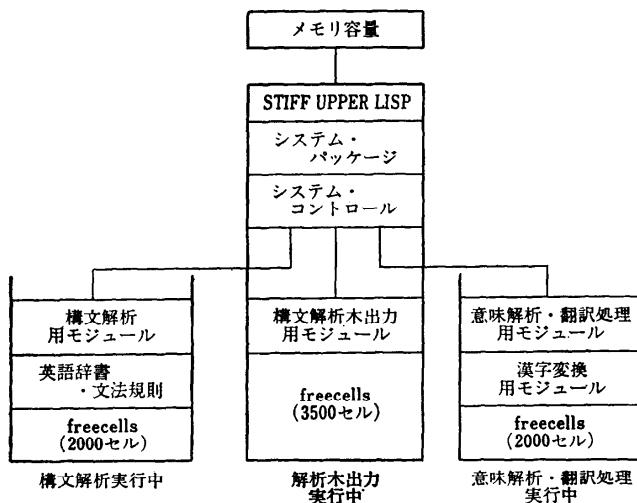
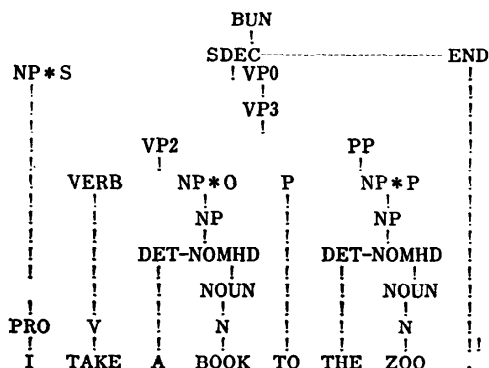


図-5 マイクロコンピュータによる実験システム

実行例

I&gt; I take a book to the zoo.

```
=====
INPUT => I TAKE A BOOK TO THE ZOO.
PRINT PPT TREE ? : y
*** PPT TREE ***
```



\*\*\*翻訳処理\*\*\*

=&gt; 本を取る

\*\*\*翻訳処理\*\*\*

=&gt; 動物園へ本を持っていく

\*\*\*翻訳処理\*\*\*

=&gt; 私は動物園へ本を持っていく

\*\*\*\*\*

INPUT =&gt; I TAKE A BOOK TO THE ZOO.

翻訳結果 =&gt; 私は動物園へ本を持っていく。

\*\*\*\*\*

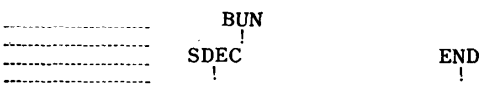


図-6 融合方式による翻訳のプロセス

メモリの容量の構成は図-5のようになっている。8 bit のマイクロコンピュータという制約のために、図-5 のシステム・コントロール部用のモジュールがメモリ上にロードされ、各々の処理に必要なモジュール間の制御を行う。このとき、各処理に必要なモジュールだけをメモリ上にロードし、処理の終了とともに不要となったモジュールをメモリ上から消去する操作も行っている。この実験システムでの実行結果を示したのが、図-6 である。

図-6 の出力のように、翻訳が進行するプロセスで、“take” の意味が“取る”→“持っていく”に変わっていく様子が示されている。8 bit のマイクロコンピュータによる実行結果をいくつか示したのが図-7 である。翻訳の結果は、大型

計算機による実験とほとんど一致している。

マイクロコンピュータによる融合方式の実験は、16 bit コンピュータ（富士通 FM-11, LISP 86 のインタ・プリタ）によっても作成され、8 bit のそれより

INPUT =&gt; I TAKE A REST.

翻訳結果 =&gt; 私はひと休みする。

INPUT =&gt; I TAKE A BOOK.

翻訳結果 =&gt; 私は本を取る。

INPUT =&gt; THE CHILD TAKES A BOOK TO THE ZOO.

翻訳結果 =&gt; 子供は動物園へ本を持っていく。

INPUT =&gt; I TAKE A BOOK TO THE CHILD.

翻訳結果 =&gt; 私は子供のところへ本を持っていく。

INPUT =&gt; I TAKE A CHILD TO THE ZOO.

翻訳結果 =&gt; 私は動物園へ子供を連れていく。

INPUT =&gt; THE CHILD TAKES A COLD IN THE COLD.

翻訳結果 =&gt; 子供は寒さの中で風邪をひく。

INPUT =&gt; I TAKE A BUS.

翻訳結果 =&gt; 私はバスに乗る。

INPUT =&gt; I TAKE A BATH.

翻訳結果 =&gt; 私は風呂にはいる。

INPUT =&gt; I TAKE A PLANE

翻訳結果 =&gt; 私は飛行機に乗る

INPUT =&gt; I TAKE A PLANE TO MOSCOW

翻訳結果 =&gt; 私はモスクワへ飛行機に乗っていく。

INPUT =&gt; I TAKE A PLANE WITH A GUN TO MOSCOW

翻訳結果 =&gt; 私はモスクワへ銃で飛行機を乗っ取って行く。

図-7 翻訳の事例

もさらに、種々の実験が可能である<sup>7)</sup>。こうした実験システムは、大型計算機では扱えなかった。思いついたアイデアを、すぐに実験するには非常によい道具だてになると思われる。

#### 4. 今後の課題

現在の技術では、文脈に依存した曖昧さの解析と除去が難しい。たとえば

The quality of this paper is terrible.

という文に含まれる paper をどのように訳すかは、この文がどのような文脈で述べられたかによって決まる。paper には、「論文」、「新聞」、「紙」の少なくとも三通りの訳語がある。このような問題をどのように扱うかは、今後の研究の課題である。

われわれは当初、英語の基本動詞の翻訳はかなり難しいと考えていた。しかし、動詞は主語や目的語、前置詞句と共に起るのが普通であるから、翻訳のための情報をそれから引き出すことができる。ところが名詞は、文中にあらわれる他の名詞との共起関係が比較的弱い。そのため、訳語選択のための情報が得にくい。少なくとも動詞との間には強い共起関係があるから、せめてそれを利用した名詞の訳語選択のアルゴリズムを開発する必要がある。たとえば、

Let's take it.

という文では、it が乗物であることを文脈から明らかにしなければ、たとえばこの場合の take を「乗る」とは訳せない。これは今後の研究課題である。

融合方式についていえば、翻訳にあたり辞書に大量の翻訳情報を組み込む必要がある。これらを入力しやすい（人間が理解しやすい）記述形式を考える必要がある。また辞書の中で概念の上位/下位関係が、意味解析や変換で大きな役割を果たしていることが明らかになっている。これらはシソーラスの一種であるが、概念の上位/下位の関係だけに限った大規模なシソーラスを開発することも十分に意味のあることである。

さらに、言語の、意味の多様性を処理するためには

小型のマイクロコンピュータ上で、意味表現を実験してみたり、また、意味選択プロセスをどのように決定したりするかを試みる必要があろう。その点で、融合方式によるマイクロコンピュータ上の機械翻訳システムは、まさに“合成による解析”の道具だてになると思われる。そして、こうした試みは、すでに小型コンピュータ上での、モンテギュー文法を利用した翻訳システムと同じ考え方である<sup>8)</sup>。

#### 参 考 文 献

- 1) 長尾 真: 機械翻訳, 電子通信学会誌, Vol. 65, No. 4, pp. 386-392 (Apr. 1982).
- 2) Tanaka, H.: Semantic Representation Language, in Kitagawa, T.: Japan Annual Reviews in Electronics, Computers and Telecommunications Computer Science and Technologies, North-Holland pp. 71-86 (1982).
- 3) 田中穂積: 融合方式による英日機械翻訳システムにおける英語基本動詞の訳し分け, 文部省科研費研究成果報告書役割分担課題: 科学技術文における曖昧さの構造の計算機による検証 (Mar. 1985).
- 4) 畠見達夫: LINGOL の  $n$  進木への拡張, 東京工業大学総合理工学研究科修士論文 (Mar. 1983).
- 5) 井佐原均他:  $n$  進木拡張 LINGOL のユーティリティ関数について, 電総研集報, 46, pp. 740-766 (Dec. 1982).
- 6) 溝口文雄, 伊藤達夫: マイクロコンピュータ上における機械翻訳システムの試作と考察, 昭和 56, 57, 58 年度総合研究 A 研究成果報告書 (Mar. 1984).
- 7) 田中穂積, 糸井理人, 諸星博司: 科学技術文における曖昧さの構造の計算機による検証, 昭和 57, 58, 59 年言語の機械処理における標準化研究成果報告書 (Mar. 1985).
- 8) Nishida, T. and Doshita, S.: Application of Montague Grammar to English-Japanese Machine Translation, In Proc. of Conference on Applied Natural Language Processing, pp. 156-165 (1983).

(昭和 60 年 8 月 8 日受付)