

解説

自然言語理解の構造—理解のメカニズム:

対話と文脈†



安西 祐一郎†† 神岡 太郎†††

1. 自然言語の特徴

1.1 自然言語の二つの機能

(自然) 言語の機能は、大別して、(1)他者とのコミュニケーションや他者への表現に関与する外的な機能と、(2)概念化や思考に関与する内的な機能に分けることができる。

たとえば、言語発達の観点からは、Vygotsky¹⁾ が他者とのコミュニケーション機能が内在化していったものが内言(心の中で発話される言葉)であると主張したのに対して、Piaget²⁾ は発達段階説に基づいて、認知系や情意系の発達とともに言語系がいわば「内側から」発達するものとした。しかし、基本的には生得的と考えられる言語機能がさらに発達するには、これらの考え方が両方とも重要である。つまり、言語発達には、コミュニケーションと、認知系や情意系のような内的な非言語系の発達が、ともに大切であると考えられる³⁾。

したがって、言語の理解について考える際にも、上にあげた二つの言語機能の両方を取りあげる必要がある。

さらに、これら二つの機能を並行して両方考えることは、人間の言語理解にヒントを得つつ、新しい自然言語理解システムを構築しようとする工学者にとっても、きわめて重要であるように思われる。すなわち、外的機能という点では、自然言語によるコミュニケーション技術を考えるべきこと、内的機能の点からは、自然言語機能と非言語機能の統合化を考えるべきことである。

これらのうち、前者については、自然言語対話処理システムとその要素技術、また、より広く言えば、自

然言語を一部でも用いたいろいろな支援システムの研究が中心となるだろう。一方、後者については、マルチメディア統合化システム、マルチメディアインタフェースの研究が中心となるであろう。こうした研究はもちろんすでに行われているが、今後の自然言語理解技術の研究方向を定めるにあたっても、上に述べた二つの機能分類は、大変重要な示唆を与えるものと考えられる。

1.2 言語のコミュニケーション機能と文脈

それでは、二つの言語機能の第1にあげた、コミュニケーションの機能について、簡単に述べよう。コミュニケーション、とくに対話における情報伝達の定性的仮説としてよく知られているものに、Grice による会話の公準 (Gricean Maxim) がある。これは、次のような4つの公準を満たすように、会話文の形式や内容が決まるという仮説である:

- (1) 量の公準: 必要十分なだけの情報を発話する。
- (2) 質の公準: 真であると思うことを発話する。
- (3) 関係の公準: 会話に関係のあることを発話する。
- (4) 様態の公準: あいまい文を避け、はっきり意味がとおるように発話する。

対話が、話し手と聞き手に共通の目的を達成するために行われるような場合には、グライスの公準がある程度成立する場合が多い。しかし、実際の対話では、どの情報が対話に関係しているのかが、その時点では特定できないことも多く、ほとんどの場合、グライスの公準を厳密に定義することはできない。

とくに、対話における語や文の意味は、対話の文脈によって変わってくる⁴⁾。また、話し手や聞き手の意図によっても、対話文の意味は異なってくる。たとえば、「駅」という一語文でも、その意味は、話し手、聞き手が誰で、どこにいるのか、なにを見ながら話しているのかなどに依存する。コミュニケーションの機能を第1に考えると、語や文の意味を固定的に考えるよ

† Mechanisms for Natural Language Understanding: Dialogue and Context by Yuichiro ANZAI (Department of Electrical Engineering, Keio University) and Taro KAMIOKA (Department of Behavioral Science, Hokkaido University).

†† 慶應義塾大学理工学部電気工学科

††† 北海道大学工学部行動科学科

りも、文脈や意図に依存するという考え方から出発するほうが、言語理解の本質をついたものだとみなすことができよう。近年盛んに研究されてきた状況意味論⁶⁾は、言語へのこのような接近法の一つと考えられるが、言語機能をコミュニケーションの立場から捉えるかぎりでは、人間がどのような情報を文脈情報として捉えうるのかという点を、もっときめ細かく調べ、それを発見的な知識として蓄積していく必要があるように思われる。

1.3 言語の内的機能と言葉の意味

一方、言語の内的機能のほうを考えると、それが情意系や認知系と深い関係にあることは、すでに繰り返して示唆されている⁶⁾。たとえば、生後半年過ぎから現れる、いわゆる「人見知り」(8カ月不安と呼ばれる)のような情意機能や、やはり生後半年くらいから現れる目的・手段の分化のような認知機能は、その後の言語発達に重大な影響を及ぼすと考えることができる。もちろん、人間において言語の内的機能がこのように複雑な構造になっているからといって、計算機による言語理解システムもそうでなければならないという理由はない。しかし、上に述べたことは、概念化や思考に関与する言語の意味が、実際には他の認知機能や情意機能にも依存していることを間接的に示唆するものであり、言葉の意味を理解するという問題を考えるにあたっては、人間、計算機を問わず、つねにつきまとう論点であることを考慮しておく必要があるだろう。

なお、言語理解の問題として、もちろん、Chomskyによって定式化された言語の創造的機能や、言語生成における諸問題も重要であるが、ここでは省略する。

2. 自然言語対話処理システム

2.1 対話処理と文脈

上では、言語の機能として、外的な機能と内的な機能があることを述べた。そこで言語機能というのは、もちろん人間の言語機能のことであって、機能の詳細もわかっていないことが多いし、また、こうした機能をそのまま計算機システム化するには、技術的なギャップが大きすぎる。

このような場合に考えられる研究の方向としては、ある特定の言語機能に焦点をあて、その機能を計算モデルとして実現するという方向がある。とくに、言語機能をコミュニケーションのような外的機能として捉える立場は、内的機能重視の立場に比べて計算モデル

による研究が相対的に少なく、その結果、文脈に依存した意味という考え方が、自然言語処理研究ではまだ多少遅れているように見受けられる。一方、自然言語理解における、文脈に依存した意味という複雑な問題を考える場合、具体的なシステムやアルゴリズムを基にして論じる方が、明確な議論が可能になるように思われる。

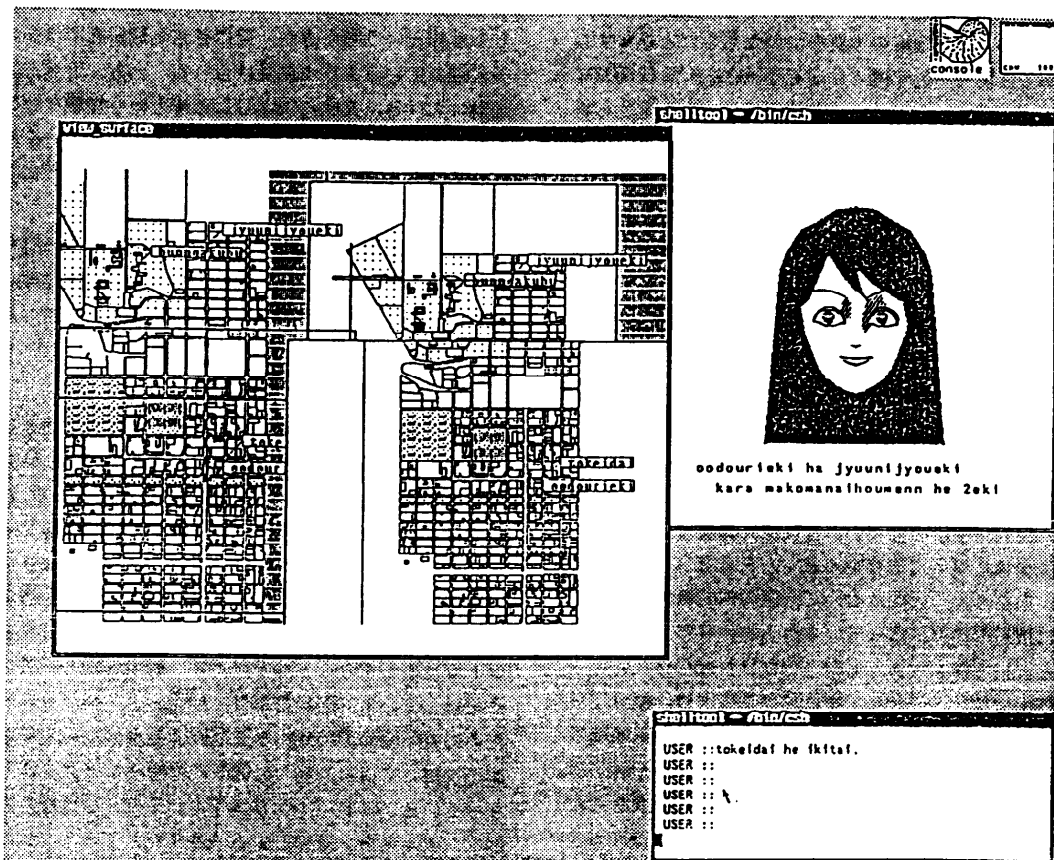
そこで、本論文では以下、このような方向への研究の一例として、われわれが構築した日本語対話処理システム、とくに文脈と発話意図の処理について述べることにする。

2.2 対話処理システムの概要

以下で述べる対話処理システムは、札幌市の地理などに関する、ユーザ(以下対話者ともいうことがある)の日本語文(ローマ字入力)による問合せに対して日本語文(ローマ字出力)で答える質問応答システムである。また、それだけでなく、ユーザが知りたい情報に関連した場所が、システムのもつ地図知識に登録されている場合には、さらに地図を生成してウィンドウに表示することなどにより、道案内ができるようになっている。記述言語は、推論とデータベースのための Quintus Prolog、および画面制御のためのC言語を併用しており、約15,000行程度のプログラムとなっている。以下、「システム」といえば、この対話処理システムを指すものとする。図-1に、システムからの表示例を示す。

さて、上に述べたシステムがユーザの期待する動作を行うためには、対話においてユーザがもつと推測されるゴール構造(ユーザゴールと呼ぶ)に基づいて対話文を解釈する方法が有効であると考えられる。しかし、計算機でこれを実行するうえでの大きな問題として、ユーザゴールの中に文脈をどう表現し、それをどういう形で利用するかという問題がある。

以下では、文脈情報をユーザゴール中の変数に対する制約として表現し、その制約を満たすという条件のもとでシステムからのサービス(ユーザの目的を達成しうる可能性のあるような文や地図などの生成と表示：ただし、以下ではサービスすべき文や地図などを生成するための意味表現などもサービスと呼ぶことがある)を行うという方法について述べる。とくに、最初あいまいであったユーザゴールが次第に具体化されていく過程を、対話の中で動的に生成された制約に対する計算によって表現する方法について述べる。

図-1 システムからの表示例¹⁾

3. システムにおける対話文の理解

3.1 対話文の表現

図-2 に示したように、システムによる対話処理は、対話を管理するマネージャが対話文理解部と発話文生成部を制御する形でなされる。システムのゴールは、ユーザが求めている情報を提供することにあるので、対話文理解の役割は、対話文からユーザの要求している情報が何であることを同定することである。

例として、図-3 に、ユーザの発話文からマネージャの処理に至る対話文理解の流れを、対話文「どのラーメン屋が一番近いですか」に対応させて示した。まず、構文解析によって、入力文に対して成り立つ係り受け関係構造（B表現）を生成し（述部に関しては形態素解析も行う）、次に、このB表現を用いて、入力文の表面的な意味表現であるM表現を生成する。そし

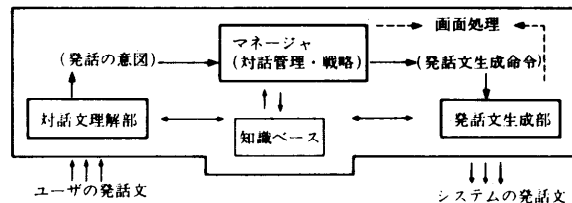


図-2 システムの対話処理構造

て、M表現からユーザゴールの制約を受けたG表現を生成してマネージャに送る。マネージャはG表現とユーザゴールなどを用いて、ユーザに対して行うサービスを決定する。このときのサービスの大半は発話文生成部に送る発話文生成命令に変換される。

以下本論文では、M表現からG表現、そしてマネージャがユーザに対して行うサービスの決定までの過程について述べる。なお、M表現やG表現は属性構造表記によって表現されるが、G表現に関しては、わ

かりやすさのために本論文では簡略化して示すことにする。

3.2 ユーザゴール

システムは、ユーザの発話の動機となるようなゴールのプロトタイプ（プロトタイプゴール）を階層化させており、対話が始まると、このうちの必要な部分を、現在の対話者のユーザゴールとして管理する。ユーザゴールは、対話文の解釈に利用されるのと同時に、対話文の解釈の結果推測された情報が記録される場所にもなっている。

図-4 は、プロトタイプゴールと、システムが対応できるサービスの一部を、太い矢印によって対応づけて示したものである。階層関係の情報は、各ゴール $g[1-9]$ の G_STEP の項に記述され、結合子 “,” は AND ゴールを、“/” は OR ゴールを表している。階層化された各ゴールの中には、ゴール格のスロット・値の組（ゴール格ペアと呼ぶ）によって表現されており（ゴール $g[6-9]$ とサービス $s[1-5]$ は簡略化した述語によって示してある）、たとえば移動のゴール $g3$ の中では、目的地を表すゴール格 loc_f 、出発地を表すゴール格 loc_s の値がそれぞれ $gv:2$, $gv:3$ であることを示している。

なお、 $gv:2$ や $gv:3$ のように gv : ではじまる記号は、対話文の G 表現などにも含まれる大域変数（ゴール変数とよばれる）で、; の右辺はデフォルト値を、← の右辺はそのゴール変数に対して対話が始まる前にすでに設定されている制約（s 制約）を示している。

ユーザゴールのゴール格に対するほとんどの値（ゴール変数）は s 制約をもっているが、この制約は、ゴール変数の値がシステムの知識ベースにおいて具体

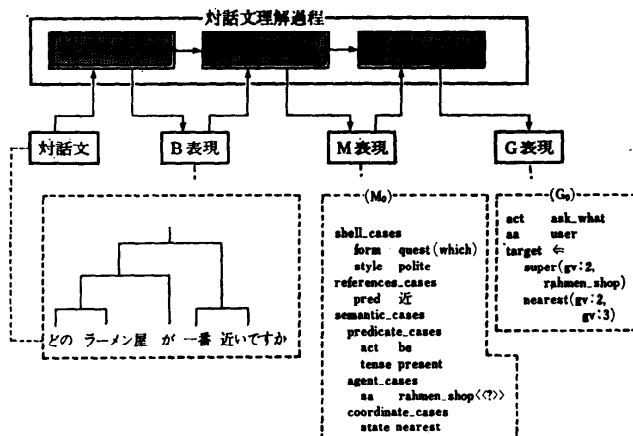


図-3 対話文理解過程と対話文の表現
(なお G 表現 G_i はユーザゴールに $gv:2$ に関する d 制約がないとした場合である。)

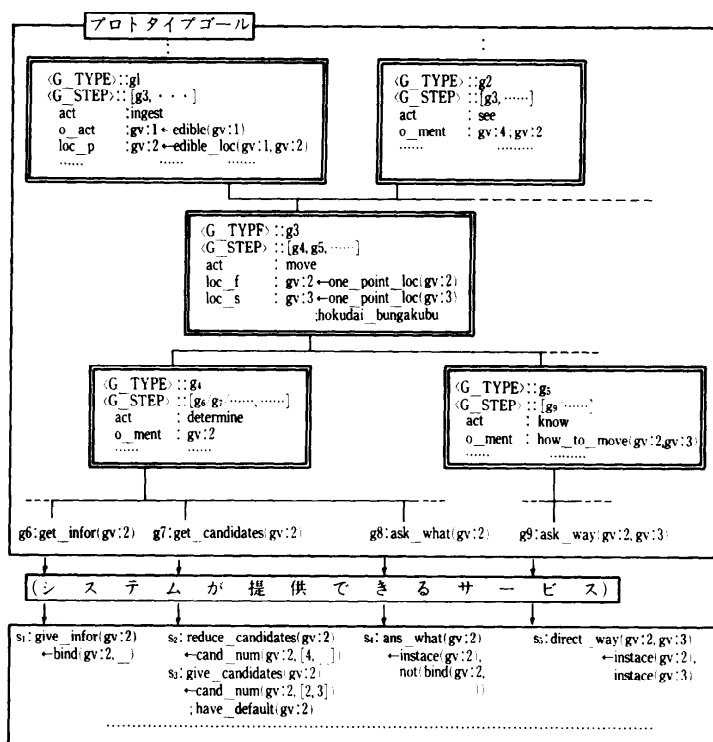


図-4 ユーザのプロトタイプゴール

値（インスタンス）となるように制約するためのもので、特定のユーザゴールに依存しない。

一方、これに対して、図-4 にはまだ現れていない

が、特定の対話に依存して対話中に動的に生成されるゴール変数に対する制約（d制約）があり、このような制約は、< の右辺に示されることになる（図-7 参照）。また、述語に付加された < の右辺はその述語に対応したサービスの適用条件を示している。

システムにおけるユーザゴールの存在は、具体的には、プロトタイプゴール中に記述される、current_goal/1 の引数に含まれるゴールの集合によって示される。この current_goal/1 の引数の更新は以下の(1)-(5)の規則に従う*。ただし、Lは現在の current_goal/1 の引数の内容を、NewGoal はG表現中の act 格ペラに一致する <G_TYPE>（図-4 参照）の値を示している。

(1) $L=[]$ のとき、NewGoal は任意である。

(2) NewGoal が、すでにLに含まれているゴールの少なくとも一つと上位下位関係にある場合、NewGoal をLに含めてよい。たとえば、 $L=[g3]$ に対して、NewGoal=g5 の場合、Lに g5 を加えてよい。

(3) NewGoal が、一つの親ゴールあるいは子ゴールしかもたない場合は、その親（子）ゴールもLに含めてよい。たとえば、 $L=[]$ に対して NewGoal=g7 の場合、g4 と g3 も加えてよい。

(4) NewGoal が、すでにLに含まれているゴールの少なくとも一つと AND ゴール関係にある場合、このとき両者の親ゴールをLに含めてよい。たとえば、 $L=[g5]$ に対して NewGoal=[g4] のとき、g3 も加えてよい。

(5) $\text{NewGoal} \in L$ のとき、Lを更新しない。

一般に、人間の会話では、対話相手のゴールを動的に推測しつつ、その結果を対話文の解釈にも利用している場合があるが、本システムでは、ゴール構造全体を動的に更新するまでには至っていない。しかしゴール変数に対する制約の生成と current_goal/1 の更新は動的に行われており、とくに、更新されるゴール変数の制約は文脈情報を反映したものになっている。

4. ユーザゴールに基づいた対話文の表現

4.1 G 表現の生成

本システムが目的としているような、比較的合理性の強いサービス提供場面での対話では、ユーザの発話文のほとんどがユーザゴールに関わるものと予測でき

る。そこで、さきに述べたように、M表現をG表現、すなわち、ユーザからの入力文の意味表現としてユーザゴールとの関係的な意味が明らかになるような表現に変換することになっている。M表現とG表現の基本的な差は、前者が発話文の表面的な意味を示しているのに対して、後者はユーザゴールと対応づけられた意味表現になっていること、したがって一般にゴール変数を含んでいることである。

M表現からG表現を生成するこの過程は、二つの段階、すなわち、(1)M表現を、ユーザゴールの内容に関係なく、そのM表現そのものから推測されるユーザゴールに関する表現（準G表現）に変換する段階と、(2)その準G表現の各要素がユーザゴールと矛盾しないかを確認しながら、ユーザゴールの更新とG表現の生成を行っていく段階に分けられる。

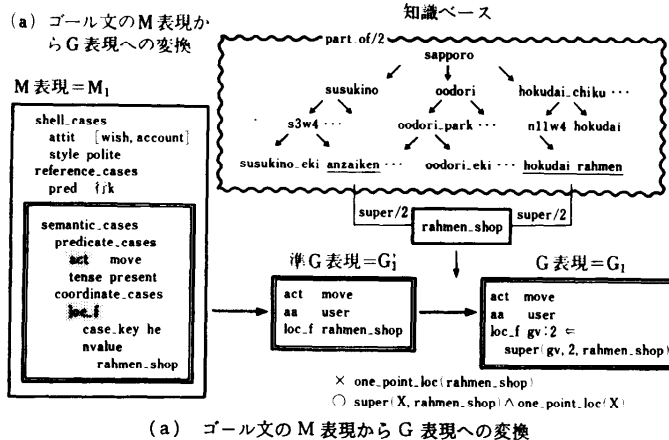
4.2 帰属ルール

システムを用いた対話状況では、ゴールの内容を伝える文（ゴール文と呼び、M表現では態度格 attit の値 wish によって判別する）がしばしばユーザ側から発話されるが、このようなゴール文には、文の表層に直接ユーザゴールに関わる情報が現れているため、M表現から準G表現を生成するのは簡単である。

この場合には、M 表現のパス semantic_cases ≠ predicate_cases に相当する部分からプロトタイプゴールのゴール格に対応しない部分（たとえば図-5(a)では tense 格など）を削除したものが、G表現にそのまま対応する。たとえば、「ススキノへ行きたい」というゴール文のM表現とそれに対応する準G表現は、図-5(a)の G_1' のように表すことができる。

しかし、一般には、ゴールに関係する情報は、対話文の表層の意味を示すM表現には現れない。そこで、ゴール文以外の文のM表現に対しては、準G表現を推測する帰属ルールを用いて、準G表現を生成する。帰属ルールはM表現のパターンによって分類されており、たとえば、図-5(b)の帰属ルールは、地理的關係に関するあるパターンの属性提示文のM表現（たとえば図-3のM表現 M_0 ）に適用される。この帰属ルールにより、 M_0 に対しては、Super で概念が限定され、Prop/geo_rel でさらに地理的属性を設定された対象 gv:2 を知りたい [act, ask_what] という準G表現 G_0 に変換される。また、知識ベース中の geo_rel/2 や upper_rel/2 は、M表現中の属性設定情報を述語形式に変換するメタ述語で、これらのメタ述語の第1引数が属性設定情報を、第2引数が変換された述語形式を示す。

* この5つの規則を用いると、Lの内容は単調増加することになる。現時点では、システムのサービス能力の都合上（残念ながら論理的矛盾を検出してではなく）例外的に、排他的 OR 関係にあるゴールが存在し、その両者がLに含まれる可能性のある場合がある。この場合は、ゴールの選択はユーザに質問して決める。



(a) ゴール文のM表現からG表現への変換

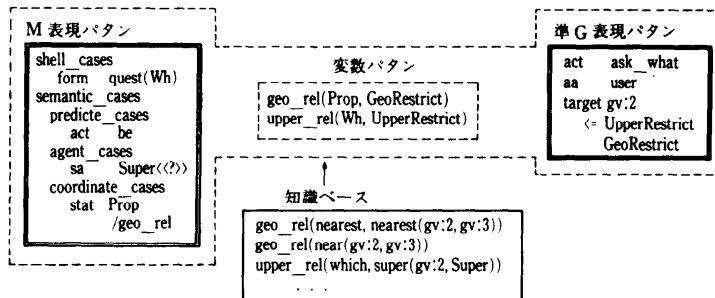
(b) M表現M₀ (図-2) に適用される帰属ルール (生成される準G表現は図-2のG表現G₀に等しい)

図-5 G表現の生成

次に、このような帰属ルールを適用することによって帰属されたゴール、すなわち準G表現の act 格の値が、current_goal/1 の引数に登録できるかどうかを、3.2 に述べた方法を用いてチェックする。たとえば上の例では、プロトタイプゴール中の g8 について、current_goal/1 の引数に登録できるかどうかチェックされる。もし登録できない場合は別のM表現の生成を試み、さらに適当な準G表現が生成できない場合は、対話文理解過程を終了させてマネージャに判断を委ねる。

なお、一つのM表現に対して複数の帰属ルールが適用できる場合は、生成される仮説が多い説明ほど優先して解釈する。これは、すでに発話してわかっている情報をユーザが繰り返して発話することはないだろうという考えに基づく発見的方法である。ただし、それでも選択できない場合は、マネージャの判断で、どちらの解釈が正しいかを、ユーザに対しての発話によって確かめることができる。

4.3 G表現とゴール変数に対する制約の生成

さて、次の段階では、M表現の変換によって得られた準G表現を、その各要素がユーザゴールと矛盾しないG表現に変換するとともに、このG表現がユーザゴールを含意するようにユーザゴールの更新を行う。たとえば、図-5(a)のG₁'のようにゴール格ペアで示される準G表現では、そのうちのゴール格 act 以外の任意のゴール格ペア [Case, Val] が、図-6のような計算によってG表現の要素に変換され、その過程でユーザゴールの更新も行われる。

図-6のⅠはすべての制約を満たす場合に、Ⅱはこれまでの対話文処理の結果生成されたd制約と、いま解析の対象となっている対話文との間で矛盾が生じる場合に、またⅢは対象となっているゴール格のs制約は満たさないが、Valを制約とみなせばユーザゴールに矛盾しない場合に対応する。このアルゴリズムで実際にd制約を生成するのはⅢである。たとえば、図-5(a)の準G表現G₁'のloc_f格の値susukinoは、

帰属ルールによって変換された G 表現の各任意の要素 [Case, Val] をこのまま G 表現あるいはユーザゴールに登録するかどうかを計算する。ユーザゴールにおける Case に対応するゴール変数 GV の s 制約集合を S_Restrict, current_goal/1 に登録されているゴールの範囲で GV に対する d 制約集合を D_Restrict とする (ゴール変数の値がすでにある値 Val' に束縛されている場合はそれを特殊な d 制約 bind(GV, Val') として D_Restrict に含める)。

I Val が S_Restrict, D_Restrict とともに満たす場合, GV の値を Val に束縛する (D_Restrict = [bind(GV, Val)]).

II Val が S_Restrict を満たすが D_Restrict を満たさない場合, 矛盾 [Case, [Val, D_Restrict]] を出力する。

III Val が S_Restrict を満たさないが, 推移関係 Rel (概念の上位下位関係 super/2 や部分全体関係 part_of/2) によって Val に含意されるような値が S_Restrict を満たす場合, Rel(GV, Val) を新しい制約として算出する。

(III-1) Rel(GV, Val) $\wedge$ D_Restrict $\wedge$ R_Restrict を満たす解釈が一つも存在しない場合, 矛盾 [Case, [Rel(GV, Val), D_Restrict]] を出力する。

(III-2) 推移律をたどることによって, $\exists$ Upper. (Rel(GV, Upper) $\in$ D_Restrict) [Rel(Val, Upper)] が成り立つような場合, 新しい d 制約集合を D_Restrict $\cup$ (Rel(GV, Val)) - {Rel(GV, Upper)} とする。

(III-3) 推移律をたどることによって, $\exists$ Upper. (Rel(GV, Upper) $\in$ D_Restrict) [Rel(Upper, Val)] が成り立つような場合, これまでの d 制約集合は更新しない。

(III-4) 新しい d 制約集合を D_Restrict $\cup$ (Rel(GV, Val)) とする。

IV それ以外の場合は矛盾 [Case, [Val, D_Restrict]] を出力する。

なお, (III-2) (III-3) (III-4) では G 表現における [Case, Val] は [Case, Rel(GV, Val)] と書き換える。また準 G 表現の段階で Val がすでに d 制約述語形式となっている場合には直接 (III) を行えばよい。

図-6 準 G 表現を G 表現へ変換するための計算手続き

one_point_loc/1 という s 制約を満たしていないが, part_of 関係をたどることによって, この s 制約を満たす具体値が知識ベース上に少なくとも一つは存在するので, loc_f 格の値 gv:2 の d 制約として, part_of (gv:2, susukino) が生成される。この制約が III の中の 4 つの場合のどれに相当するかは, その時点におけるユーザゴールの状態, すなわち gv:2 に対する制約に依存する。

以下では, この 4 つの場合を, ゴール変数 gv:2 を対象として, システムとユーザの実際の対話例 (図-7) を用いて説明することにする。

〈対話例 1〉では, ①の解釈の結果ユーザゴールに登録されていた gv:2 が寿司屋のインスタンスであるという制約 super(gv:2, susi_shop) に対して, ③を解釈して得られた gv:2 がラーメン屋のインスタンスであるという制約 d1 を加えた場合, この両者を

同時に満たす gv:2 の値が存在しないので矛盾が生じる。矛盾はこのほかにもいろいろなレベルで存在する。たとえば, すでに述べた current_goal/1 に登録できないゴールに登録しようとした場合も矛盾であると解釈できる。

矛盾を検出すること自体, 対話処理においては重要なことで, 本システムによる対話処理の過程では, 検出された矛盾を推論に利用することを試みている。たとえば, 〈対話例 1〉の場合は, 矛盾をユーザゴールの変化であると見なしている*。

〈対話例 1〉で生成された d1 と異なり, 〈対話例 2〉以降で生成される制約 d2, d3 (図-7 にはない), d4, d5, d6 はユーザゴールに含まれる制約と矛盾しないので, G 表現の d 制約となりうる。しかし, これらの d 制約には, ユーザゴールに含まれる d 制約の更新のされ方に違いがある。

たとえば, 〈対話例 2〉では, ⑥で生成されてユーザゴールに登録されている part_of(gv:2, oodori) を, ⑦によって生成された d2 が合意している (oodori_park は oodori の一部である) ので, ユーザゴールの gv:2 の制約から part_of(gv:2, oodori) を除いて d2 を加える。

また, 〈対話例 3〉では, ⑩から d 制約 d3 と d4 を生成する (⑩の M 表現には図-5 (b) の帰属ルールが適用できる) が, d3 は⑨から生成された gv:2 に対する制約 super(gv:2, rahmen) に含意されるので, ユーザゴールの制約には d4 だけが加えられる。

〈対話例 4〉では, ⑬と⑭によってユーザゴールの更新がなされているが, ⑯から生成された制約 d5 と d6 はすでにユーザゴールに登録されている gv:2 に対する制約と含意関係にないので, d5, d6 とともに gv:2 の制約として登録される。

このようなゴール変数に対する制約は, current_goal/1 に含まれるゴールに関してまとめて管理し, どのゴールのどのゴール格かがわかれば, そのゴール格に対する制約はすぐに取り出せるようにしている。たとえば, current_goal ([g3, g4]) ならば, gv:2 に対してはユーザが決めた値 (g4 の o_ment 格) と希望する移動の目的地 (g3 の loc_f 格) などに対する制約が同時に与えられることになる。ゴール変数は, G 表現の中などのように, ゴール以外にも出現するので, このような管理を行った方が便利である。

*ただし, 一般には矛盾を解消する解釈を生むための非単調計算に時間がかかり, また矛盾をつねにユーザゴールの変化とみなしていいとは限らないなど, 問題点が多い。

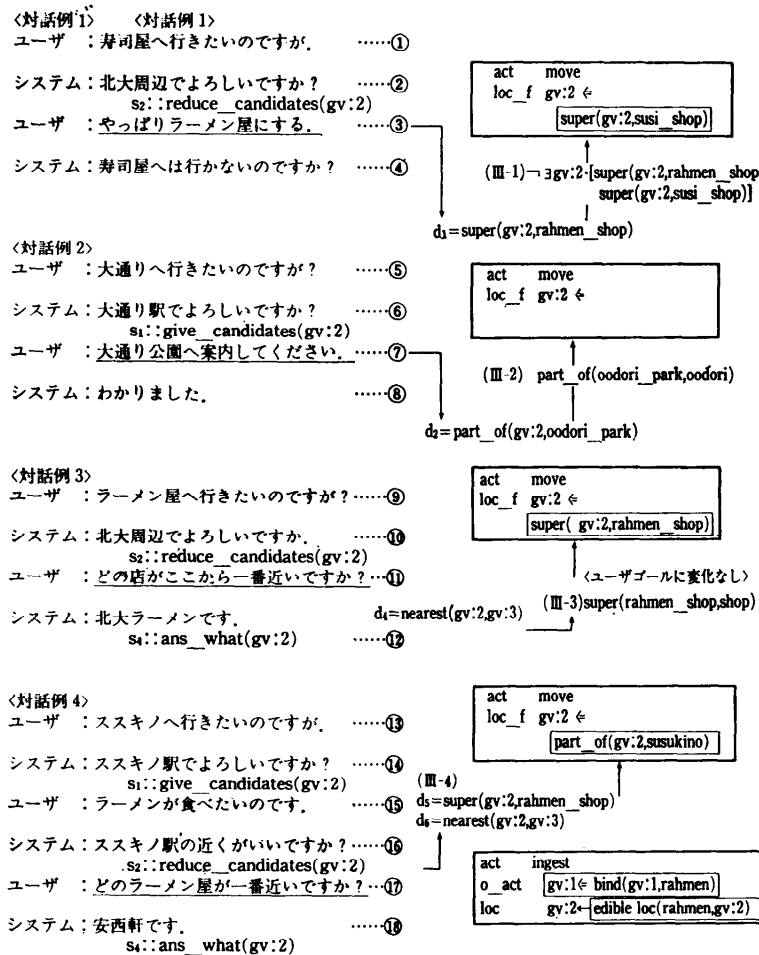


図-7 システムにおける対話例（ユーザゴールでは本文中の説明に必要な部分のみを示した。）

ただし、上の方法には、制約間の計算を自由に行うと計算が有限時間内で停止する保証がないという問題や、具体値を介さなければ矛盾や制約の計算ができないといった効率上の問題がある。しかし、この計算の結果として得られるゴール変数に対するd制約は、より柔軟な対話制御を行うための情報として有効だと考えられる。その理由は、システムとユーザとの対話では、対話が始まる前にユーザのゴールがすでに決定している場合とそうでない場合が考えられるが、ユーザゴールが確定しておらず対話の中で徐々に確定していく場合には、ユーザが発話するゴールに関するあいまいな情報を表現するための手段として、d制約が利用できるからである。しかも、ユーザゴールが確定している場合のゴール変数の束縛もゴール変数に対する特殊な制約であるともなせば、対話の中で抽出された

ユーザのゴールに関する情報は、すべてゴール変数に対するd制約として表現できるので、ユーザゴールにおける制約計算は単一の枠組みで表現できることになる。

一方、このようなd制約は、単に上に述べたようにあいまい性が表現できるというだけでなく、対話における文脈情報を保持していることにも注意すべきである。このような、d制約のもつ文脈情報がどのようにシステムのサービスに利用されているかについては、以下に述べることにする。

5. 発話意図に応じたサービス

5.1 発話意図に応じたサービスとゴール変数

発話の意図（以下発話意図と呼ぶ）を、ここでは、
発話者が対話者に対して何を期待するかの含意とみな

すことにする。このとき、ユーザに対してサービスを提供することを目的とした対話システムでは、ユーザの発話意図に応じたサービスを行うことが、もっとも大きな目的の一つになる。本システムでは、マネージャが、ユーザゴールとG表現を用いて、発話意図に応じたサービスの手続きを行う。手続きそのものは簡単で、G表現が示すゴールからプロトタイプゴールを下位のゴールに向かってたどり、あるゴールを満たすサービスが生成できるならば、そのサービスを発話意図に対応するサービスとみなす。ただし、サービスの生成には一般に適用条件があり、その条件も満たさなければならない（複数のサービスが適用可能な場合の選択については、ユーザに質問する）。

たとえば、図-7 ⑥の「ラーメン屋へ行きたいのですか」という文では、そのG表現によって示されるゴール g_3 から下位に向かってゴールをたどり、制約を満たすサービスが生成できるかどうか探索することになる（図-4 参照）。この例では、ユーザゴール g_6 に対応して $gv:2$ に関する情報を提供するサービス s_1 、ユーザゴール g_7 に対応して $gv:2$ の候補をしぼるサービス s_2 や具体的な候補を提示するサービス s_3 、それにユーザゴール g_8 に対応して $gv:2$ が具体化しているならばその値を答えるというサービス s_4 などがみつかると。このとき、 $gv:2$ には、 $super(gv:2, rahmen_shop)$ 以外の d 制約が存在しない（図-7 〈対話例3〉参照）ので、（実際に知識ベースを検索すると）制約を満たすラーメン屋（ $gv:2$ の候補）が 20 店以上存在する。

また、適用条件の中で用いられている述語などについては、(1)述語 $bind/2$ は、ゴール変数の値を束縛したとき、(2) $instance/1$ は束縛に限らず結果的に値が具体化したとき、(3) $cand_num(GV, [Min, Max])$ はゴール変数の候補の数 N が $Min \leq N \leq Max$ のとき、(4) $have_default(GV)$ はゴール変数 GV がデフォルト値をもつとき、それぞれ成り立つ。このようにして、「ラーメンが食べたい」という文は、本システムでは、 $super(gv:2, rahmen_shop)$ という d 制約をもった $gv:2$ の候補を限定するという内容のサービスを生成することによって、ユーザの発話意図に対応できると考えられる。

5.2 文脈情報を反映したサービスとゴール変数

システムによる発話文生成過程は、発話内容の意味表現を生成する WHAT 生成過程と、その意味表現から実際の文を生成する HOW 生成過程に分けられ

る。このとき、選択されたサービスを階層化されたサービス体系の中で具体化していくことは、システムの発話文生成過程における WHAT 生成過程に含まれる。したがって、サービス内容の表現に含まれたゴール変数を用いれば、システム側からの発話内容を、ユーザゴールがもつ文脈的制約を満たすように決めることができる。この点は、図-7 の対話例にもみられる。

たとえば、ユーザの発話文⑥のG表現 $[[act, ask_what], [aa, user], [target, (<, [super(gv:2, shop), nearest(gv:2, gv:3)]]]$ は、 $gv:2$ の値を問うものであるが、これに対するサービスは、図-4 からわかるようにユーザゴール g_8 に対応する s_4 である。このとき、この発話文⑥を文脈情報がないまま解釈すると、どのような店でも現在地（北海道大学文学部は出発地のデフォルト値である）からもっとも近い店が検索されるが、発話文⑥の存在によりその店はラーメン屋であると解釈すべきである。

これに対して本システムでは、ユーザゴールに登録された $super(gv:2, rahmen_shop)$ と、⑥のG表現によってもたらされた d_3, d_4 の間の制約計算によって、 $super(gv:2, rahmen_shop) \wedge d_4$ が得られる。そして、⑥に対応するサービス s_4 、すなわちそのゴール変数 $gv:2$ をユーザに答えるというサービスが発話されることによって、⑥で、ラーメン屋で北大から一番近いところという d 制約を満たす「北大ラーメン」が応答される。また、同じくユーザの発話⑦の意図に対するサービスも s_4 となるが、応答すべき対象は「北大文学部からもっとも近いラーメン屋（北大ラーメン）」ではなく、すでに発話された③と⑤の制約を受けた $gv:2$ を介することにより、「ススキノで北大文学部から最も近いラーメン屋」、すなわち安西軒が応答の対象となる。このように、サービスの表現にゴール変数が含まれていることによって、対話文理解過程で得られる文脈情報を用いたサービスが可能になっている。

なお、動的に生成される制約を仮説とみなし、仮説生成推論によって文脈処理を行う方法もある。これについては、文献⁷⁾を参照されたい。

5.3 発話意図と対話状況

一般に、発話意図は対話の状況に依存すると考えられる。しかし、だからといって対話状況を記述すれば発話意図に応じた発話ができるというわけではない。問題は、対話状況のなにに注目し、注目した情報をど

のような形でとらえるかということである。

ふつう、サービス提供場面における対話では、ユーザ側が、自分のゴールを達成するのに役立つサービスを期待して発話を行うと考えられる。したがって、ユーザの発話意図は、基本的には、発話文から推測されるユーザゴールと、システム側のサービス提供可能な情報との間の計算によって決まると考えられる。たとえば、ユーザが「ラーメンが食べたい」と入力したとき、実際にシステムにラーメンをつくってもらうことを期待しているわけではないだろう。これは、システムにそのようなサービス能力がないことを、ユーザが知っているから期待しないですむわけである。

同様に、まったく同じ文に対しても、対話状況の違いによる発話意図の違いを、サービスとG表現（あるいはユーザゴール）との関係によって、統一的に説明することができる。たとえば、もしシステムにラーメンがつかれるなら、システムに対する「ラーメンが食べたい」という入力文の発話意図は、ラーメンをつくってもらうというサービスを要求することでありうるし、もし道案内ができるなら、同じ文の発話意図が、ラーメンが食べられる場所に関する情報を提供してほしいということでありうる。

このように考えていくと、ある発話文におけるユーザの発話意図は、`current_goal/1` の引数中で最も上位のゴールからその発話文のG表現に対応するゴールを経由してユーザが期待したサービスに対応するゴールまでのゴール列に対応すると考えられる。たとえば、`current_goal` (`[g3, g4, g5, g7]`) で、G表現に対応するゴールが `g3, g4, g7` の内のどれかであれば、その時点で推測される発話意図は `[g3, g4, g7]` となる。

ただし、上で発話意図に対応したサービスについて考えてきたが、実際のシステムの対話文理解過程では、発話意図そのものを利用しているわけではない。これは、対話文理解において発話意図を推論することが必須ではないという我々の仮説に基づいている。発話意図の抽出は認知的には興味深い問題であり、確かに発話者にはその発話を直接引き起こしたゴール（発話意図）が存在すると思ったほうがよい場合が多い。しかし、発話意図は対話文理解過程を説明するための概念としては必要であるが、実際の対話では、聞き手は話し手の発話意図を知る必要は必ずしもないとわれわれは考えている。

このような仮説をおく理由は、ユーザが、特定のサービスを期待して発話するほど詳しく、システムが

提供できるサービスについて知っているというわけではないこと、および、システムにとって最終的に必要なことは、ユーザの発話意図ではなく、発話意図に応じたサービスの提供だということにある。

もちろん、`current_goal/1` に含まれるゴールをサービスに結びつける操作が発話意図の解釈に沿ったものである場合、あるいは、発話意図に対応すると考えられるサービスが複数存在して確定できない場合には、サービスを限定する過程が発話意図を確かめることに対応する。しかし、この場合も、発話意図そのものを抽出しているわけではない。たとえば、ほかにd制約がない状態で「時計台へ行きたいんだけど」という文がシステムに入力された場合 (`bind(gv:2, tokeidai)`)、時計台に関する情報を与えるというサービス `s1` と時計台への道案内をするというサービス `s5` が考えられる。本システムでは、ゴールをさかのぼって `s1` と `s5` の分岐点になる `g3` からどちらへ意図が伸びているか、すなわち `g4` か `g5` かあるいは `g6` か `g9` かを、ユーザに問うようになっている。

6. おわりに

本論文では、はじめに自然言語の二つの機能について述べ、次に、そのうちの外的機能、すなわちコミュニケーション機能の計算モデルの一例を、われわれが構築した日本語対話処理システムを例として説明した。

われわれは、コミュニケーションの中での言語理解が、言語の理解の重要な側面であると考えている。とくに、本論文で述べたような、言語理解における文脈や意図の計算モデル化は、対話モデルをとおして考えるのが有効な方法であると考えている。

しかしながら、本論文での説明からも示唆されるように、文脈や意図の情報処理を含む対話の計算モデルは、現状ではある程度複雑なものにならざるを得ない。本論文で述べたようなアプローチをさらに進めるには、対話のモデル自体の研究はもちろん、その要素となる意味理解、文生成その他の諸モデルを確定していくことが必要である。これらのモデルについては、本特集号の他の論文を参照していただきたい。また、われわれとしては、今後とくに、言語生成および対話管理の計算モデル研究を重点的に進める必要があると考えている。

このような研究方向に関連して、計算機を用いた対話処理研究において重要なことの一つは、全体として

作動するシステムを構築し、その中で一つ一つの要素システムの役割を確定していくことである。文脈に依存した言語理解においては、各要素システムの動作が、他の要素システムの動作を決めるためのいわば「文脈」としてはたらいっており、したがって、「全体が作られて初めて部分が理解できる」という面がある。たとえば、対話文理解過程でいえば、対話文を解釈した結果として何が必要で、それが対話処理システム全体にとってどのように利用されるのかを明確にしなければならない。本論文で説明したモデルを含め、これまでの対話処理研究においては、この点がややあいまいなままであったように思われる。そして、その理由として考えられることの一つは、対話状況を固定していた点にあると考えられる。つまり、システムが存在する状況が、実際にはかなり固定的なものであったために、文脈に依存した動的な情報処理ということの本質として、計算モデルのレベルでは、まだみえてきていないものがあるように思われる。

自然言語理解の構造を考えるにあたっては、コミュニケーションという場における、文脈に依存した言語情報処理を考えないわけにはいかない。そのとき、文脈に依存した理解という、定義の困難な言葉の裏にな

にがあるか、その一部分でも計算モデルによって理解すること、それは、人間の言語理解だけでなく、将来の自然言語理解技術の発展に対しても、必ず役に立つであろう。

参 考 文 献

- 1) ヴィゴツキー (柴田義松訳): 思考と言語上下, 明治図書 (1962).
- 2) ピアジェ (谷村覚・浜田寿美男訳): 知能の誕生, ミネルヴァ書房 (1978).
- 3) 安西祐一郎: 認知科学と人工知能, 共立出版, pp. 71-83 (1987).
- 4) 開 一夫・安西祐一郎: 文脈に依存した定性的空間表現とその自然言語対話処理システムへのインプリメンテーション, コンピュータソフトウェア, 6巻1号, pp. 56-64 (1989).
- 5) J. Barwise and J. Perry: Situations and Attitudes, MIT Press (1983).
- 6) 中島 誠: 意識の発達, 塚原伸晃編: 脳と意識, 平凡社, pp. 35-53 (1985).
- 7) 神岡太郎: 計算機による対話文理解システム, 心理学評論, 30巻3号, pp. 234-258 (1987).

(平成元年7月13日受付)