人 工 知 能 75−10
（1991. 3. 7）

コーパス解析に基づく事例ベースパーザ

島津秀雄 & 高島洋典
日本電気(株) C&C 情報研究所

コーパス・ベース解析システム (CBP) は、フレーズベースの自然言語解析システム構築の為の知識獲得機構である。パターン・概念対左辺側のパターンは、実際の会話 (コーパス) をパターン・概念対定義者がパターン汎化オペレータを使って変形・汎化して見本パターン (exemplar) として定義する。パターン・概念対右辺側の概念表現のためには、カテゴリー構造が用意されている。CBP は収集したコーパス内容に沿ってパターン・概念対をパターン・概念対定義者と共同して、漸増的に定義していく。

Acquiring Knowledge for Natural Language Interpretation Based-on Corpus-Analysis

Hideo Shimazu & Yosuke Takashima
C&C Information Technology Research Laboratories
NEC Corporation

Corpus-based Parsing system (CBP) is an incremental knowledge acquisition mechanism to develop phrase-based NL interpretation systems. The *pattern-concept* pairs are made of real corpus. Generalization operators are provided to generalize and modify raw language patterns. CBP's knowledge representation is exemplar-based. A category (the extensional equivalent of a concept) is represented as one or a set of generalized language patterns, called exemplar.

1 はじめに

自然言語解析手法の中で、フレーズベース (phrase-based) 解析手法は、狭い対象領域で自然な会話を理解・生成するシステムを構築する時には有効な手法である [1] [4] [6] [16]。フレーズベースシステムは、内部に多くのパターン・概念対を持ち、入力文にマッチするパターンを見つけるとそれに対応する概念を入力文の意味表現とみなすものである。フレーズベースシステムの性能は、それが保持するパターン・概念対知識ベースの質と量に依存する。ところが、パターン・概念対知識ベースは対象領域に非常に依存するので、ターゲットシステム毎に新しく構築しなくてはならない。これが、フレーズベースシステムの致命的な問題である。これを解決する1つの方法は、特定の対象領域のパターン・概念対知識ベースを簡単に構築する為の機構を提供することである。しかしながら、この観点からフレーズベースシステム用の知識獲得機構を開発するという研究はほとんど為されて来なかった。

本稿では、コーパス・ベース解析システム (Corpus-Based Parsing system, 以下 CBP と略す) を提案する。CBP は、フレーズベースシステムを構築する時に、自然にパターン・概念対を定義するための知識獲得機構である。CBP では実際の会話コーパスをパターン・概念対定義の素材として使う。パターン・概念対定義者は、会話コーパス中の文を変形・汎化してパターン・概念対の左辺とする。このとき、もし会話文をそのまま左辺に使ってしまうと、100 文を定義しても 100 種類の文しか受け付けられないので非常に効率が悪い。そこで、CBP では1つのパターン定義ができるだけ多くの種類の入力言語パターンとマッチするように変形・汎化させるための汎化オペレータを用意している。CBP の知識表現は exemplar 指向である [2] [14]。パターン・概念対の右辺側に相当する、意味を表わす概念は、カテゴリー構造として定義される。個々のカテゴリーには、1つ以上の一般化された言語パターン (これを見本パターン (exemplar) と呼ぶ) がくくり付けられている。見本パターンは、対応するカテゴリーを表わす単語、慣用的言回し、または文全体等である。見本パターンは、収集されたコーパス中の実際の事例から作られる。コーパス中の典型的な言い回し事例だけが見本パターンとしてくくり付けられる。ある事例は、その一部分が変形・汎化されて見本パターンになるし、別の事例はそのまま見本パターンとして定義されることもある。2つ以上の見本パターンが同一のカテゴリーにくくり付けられることも可能なので概念を表わす表現の多面性を実現することもできる。抽象カテゴリーは、それをより詳細化したカテゴリーの集合を保持しているカテゴリーである。ある抽象カテゴリーは別の抽象カテゴリーを保持する事が可能なので、カテゴリーの木構造の構築も可能である。また、1つのカテゴリーが複数の抽象カテゴリーから保持されることも可能なので、異なる視点による多重階層構造の実現も可能である。

以下、2節では、CBP の概要を例を使って説明する。3節では CBP の知識獲得アルゴリズムについて述べる。4節では、アルゴリズム中の3つの部分についてより詳細に説明する。5節では、我々の予備的な評価について説明する。6節では、CBP を他の研究と比較する。

2 例による CBP の説明

今、新しい電話機の使い方に関する自然言語質問応答システムを構築中だと仮定する。また、既にその電話機に関して初心者とエキスパートとの会話コーパスは収録済みだとする。ここで、初心者の発した新しい質問文 (1a) を取り上げ、パターン・概念対定義者がパターン・概念対を定義するとする。

1a) EXT [1] って、どこですか?

1b) general(EXT, X: object) [2] って、どこですか?

パターン・概念対定義者は、(1a) を見ると (1a) のまま見本パターンとして定義してもよいが、ここでは (1b) のように部分的に汎化した形に変形してから定義したとする。この汎化の意味は、「EXT」(すなわち「内線ボタ

[1] EXT は、対象電話機の内線用ボタン上につけられた記号である。

[2] general は、汎化オペレータの1つである。

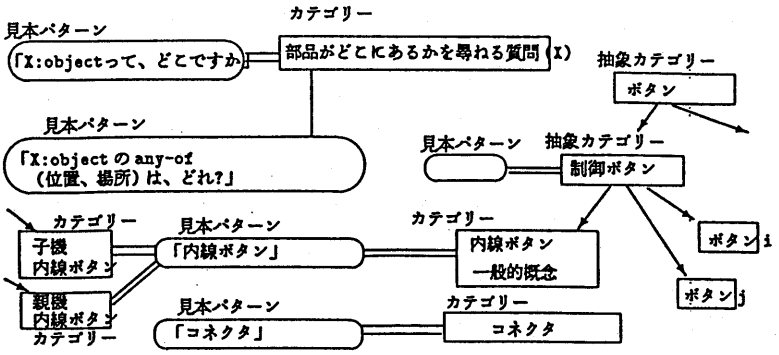

図1: (1a) (1b) (1c)(1d) から定義される内部表現

ン」) は、「object カテゴリーに属するものなら何でも」という表現に置換されることを意味する。次に、パターン・概念対定義者は (1b) の意味表現を定義する。この場合 (1b) の意味定義として、「部品がどこにあるかを尋ねる質問 (X:object)・カテゴリー」というカテゴリーが意味表現となる。また、「内線ボタン・カテゴリー」が「EXT」という言い回しに対応するカテゴリーとして定義される。もし、後から同じ意味を持つ形の違う言い回しがコーパス中に出現したらそれらはまた別の見本パターンとして保持されることになる。例えば、(1c) のような文が出現すると、パターン・概念対定義者はそれに汎化を施して (1d) のような形にする。

1c) コネクターの位置は、どれ?

1d) general(コネクター , X: object) の any-of(位置、場所) は、どれ? [3]

(1d) の意味は (1b) の意味と同じなので、両者の意味表現となるカテゴリーは同一であるべきだが、(1c) または (1d) の言い回しは、(1b) の言い回しとはマッチしない。そこで、パターン・概念対定義者は (1d) を「部品がどこにあるかを尋ねる質問 (X:object)・カテゴリー」の別の見本パターンとして保持することにする。図1は、(1a)、(1b)、(1c)、(1d) から生成される内部表現を表わしたものである。抽象カテゴリーである「制御ボタン・カテゴリー」は、電話機上の制御ボタン、例えば内線ボタンや割込み通話ボタン、留守録ボタン等を表わす一般的カテゴリーである。同一の見本パターンが複数の異なるカテゴリーに保持されることもある。例えば、見本パターン「内線ボタン」は、次の3つのカテゴリーに保持されている、「親機の物理的内線ボタン・カテゴリー」、「子機の物理的内線ボタン・カテゴリー」[4]、「抽象概念としての内線ボタン・カテゴリー」[5]。従って、入力文中やパターン・概念対定義者の記述中に「内線ボタン」というリテラルが出現した時には CBP はそれがどのカテゴリーを指示しているのか推測しなくてはならない。

3 CBP の知識獲得アルゴリズム

図2は、CBP システムの全体構成を表わしている。CBP は、パターンマッチャー (Pattern Matcher, PM と略す)、パターン・概念対知識ベース (Pattern-Concept Base, PCB と略す)、パターン・概念対定義者インタビュワー (Writer Interviewer, WI と略す) からなる。

図3は、CBP の処理アルゴリズムを表した図である。図3にそって処理の流れを説明していく。まず、対象領域の会話コーパスを収集する。パターン・概念対定義者は、その収集した会話コーパスをもとにして、パターン・概念対を定義していく。PM は新しい入力文を受け取ると、それとマッチする言語パターンを PCB 中から探す。

[3] any-of は、汎化オペレータの1つである。

[4] 対象となっている電話機は2つの受話器を持ち、それぞれ親機、子機と呼ばれている。

[5] これは次の様な状況での意味表現の為に使われる、「内線ボタン は、現在の通話を内線に転送する時に使われる。」

PMは、単純な文字列同士のマッチングより高い能力を持つ（より詳細は次節で説明する）。PCB中にマッチするパターンが見つかった時には、それに対応するカテゴリー構造がパターン・概念対定義者に提示される。パターン・概念対定義者がその結果に対して承認を与えると、CBPは会話コーパス中の次の文の処理を開始する。PMがPCB中にマッチするパターンを見つけられなかった時あるいは、パターン・概念対定義者がPMの出力結果に対して承認を与えなかった時には、WIはパターン・概念対定義者に対して、入力文に対する正しい意味表現を提示してくれるように頼む。パターン・概念対定義者は、もし望めば、意味表現を示す代わりに入力文のパラフレーズ（言い換え表現）を提示することもできる。パターン・概念対定義者がパラフレーズを提示した時には、そのパラフレーズは、元の入力文と意味表現を共有する別の言い回し事例とみなされる。このパラフレーズは、PMに渡される。もし、このパラフレーズパターンにマッチする見本パターンが存在すれば、その見本パターンに対応するカテゴリー構造が元の入力文とこのパラフレーズパターンの意味表現であるとみなされる。そして、元の入力文がこのカテゴリーの新しい見本パターンとして登録される。もし、パラフレーズパターンがPCB中のどれともマッチしない時には、パターン・概念対定義者はPCB中の正しいカテゴリーを提示しなくてはならない。もし、正しいカテゴリーがまだPCB中に存在しないのなら、パターン・概念対定義者は新しく意味カテゴリー構造を定義してやる。

パターン・概念対定義者はパターン汎化オペレータを使って、入力パターンを変形・汎化することができる（次節で説明する）。パターン・概念対定義者の記述に指示の曖昧性があることもある。WIは、その曖昧性を解消して解釈しなくてはならない（より詳細は次節で説明する）。変形・汎化された入力パターンは新しく定義されたカテゴリーの見本パターンとして保持される。

PMが入力パターンを正しくない見本パターンとマッチさせてしまった時には、その失敗は修正されなくてはならない。パターン・概念対定義者がWIに誤ってマッチしてしまった見本パターンと正しい見本パターンの間を区別する弁別表現[6]を示すと、その誤った見本パターンのところにテストとしてくくり付けられる。こうすることで、同じ間違いを再度起こすことが避けられる。CBPの修正機能は、CHEF [5]やDMAP [9]のそれのサブセットである。CBPの処理は本質的に、パターン・概念対定義者との共同作業を漸増的に行っていくというものなので、このようなパッチワーク的な失敗修正処理がどうしても必要になる。

コーパス中の全ての文が処理されると、最後に、質問型のカテゴリーの1つ1つについて、それぞれの質問への応答の仕方の手続きを記述していく。

4 CBPアルゴリズム中の詳細

前節のアルゴリズム中で詳しく述べなかった3つの点についてより詳細に説明する。

4.1 汎化オペレータ

生の言語パターンは、変形・汎化され、あるカテゴリーの見本パターンとして保持される。パターン・概念対定義者は、その変形・汎化の為に、汎化オペレータを使う。CBPでは、次のオペレータを用意している。

general(*original-pattern, generalized-expression*): は、*original-pattern* を *generalized-expression* によって示される概念に置き換えることを意味する。*Original-pattern* は、単語か単語列である。*Generalized-expression* は、見本パターンかカテゴリーのシステム中一意名のいずれかである。例えば、general(EXT, X: 制御ボタン) は、

[6]現在は、Prologの述語の形で実現されている。

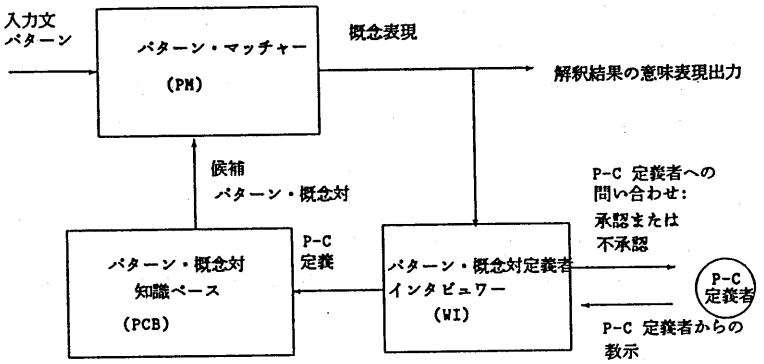

図 2: CBP の全体構成

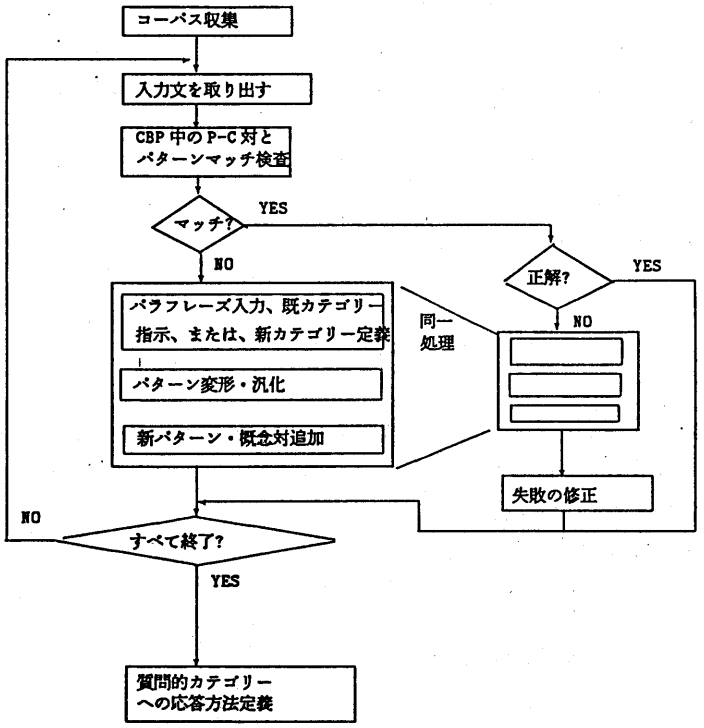

図 3: CBP の知識獲得アルゴリズム

文字列「EXT」が、「制御ボタン・カテゴリー」または「制御ボタン・カテゴリー」下のどれかのカテゴリーのものに置換されることを意味している。

sibling(*orignal-pattern, sibiling-expression*): は、*original-pattern* を *original-pattern* が参照しているカテゴリーと *sibling-expression* が参照しているカテゴリーとの間の両方に共通の抽象カテゴリーに置換する。例えば、**sibling**(EXT,「短縮キー」) は、文字列「EXT」を、「EXT」が参照するカテゴリーと「短縮キー」が参照するカテゴリーの両方の共通の抽象カテゴリーである「制御ボタン・カテゴリー」に置換する。

omittable(*orignal-pattern*): は、*original-pattern* が、元のパターン中から省略されても良いことを意味している。

any-of(*orignal-pattern, pattern-1, pattern-2, ...*): は、*original-pattern* が、*pattern-1, pattern-2, ...* のうちのどれかと入れ替えても意味が変わらないことを意味している。

interpolate(*new-pattern*): は、*new-pattern* が、元のパターンに挿入されても意味が変わらないことを意味している。

arbitrary-order(*pattern1, pattern2, ...*): は、*pattern1, pattern2, ...* の出現順序が任意であることを示している。

4.2 見本パターンと入力文とのパターンマッチング

PM は、新しい入力文を受け取ると、PCB 中からマッチする見本パターンを見つけようとする。PM は、見本パターン中に汎化オペレータ表現が存在すればそれを解釈する。現在の実現方法では、汎化オペレータを含んだ見本パターン定義は、前もって DCG(Definite Clause Grammar) [12] 表現に展開されるので、入力文と見本パターン間のパターンマッチングは Prolog インタプリタをそのまま使っている。例えば、(2a) のような見本パターンは (2b) のように DCG 表現に展開される。

2a) Y: 受話器, 上, の, X: スイッチ , は, どれ, ?

2b) ' 見本パターン ' → ' 受話器 '(U), [上],[の], ' スイッチ '(V) , [は], [どれ], [?]

また、汎化オペレータは、それぞれ以下のように DCG 表現に展開される。

general(X, Y)⇒ Y

sibling(X, Y)⇒ Z (Z は、X のカテゴリーと Y のカテゴリーの両方を含む抽象カテゴリー)

omittable(X)⇒ ([X]; {true})

any-of(X, Y, Z)⇒ ([X] ; [Y] ; [Z])

interpolate(X)⇒ ([X] ; {true})

arbitrary-order(X, Y, Z)⇒ seq → [] ; s, seq. s → [X] ; [Y] ; [Z].

4.3 パターン・概念対定義者の記述の曖昧性解消

パターン・概念対定義者が汎化オペレータ **general**(*original-pattern, generalized-expression*) と **sibling**(*original-pattern, sibling-expression*) を使って書く記述には、あいまいな表現が含まれることがある。これらのオペレータ中の *original-pattern* は、複数のカテゴリーを参照していることがある。また、*generalized-expression* や *sibling-expression* も、同様に複数のカテゴリーを参照していることがある。WI は、パターン・概念対定義者の記述中に曖昧性が含まれている時には、以下の優先順序つけのための直感的ルールを使って、一番もっともらしい解釈候補を選択する。

1つめの直感的ルールは、候補となる抽象カテゴリー中で最も一般的でないものを選択すること、である。もし、パターン・概念対定義者の general または sibling 記述から複数の抽象カテゴリーが候補として出現した時には、元の詳細なカテゴリーからそれら抽象カテゴリーまでのカテゴリー階層木中の距離をそれぞれの候補抽象カテゴ

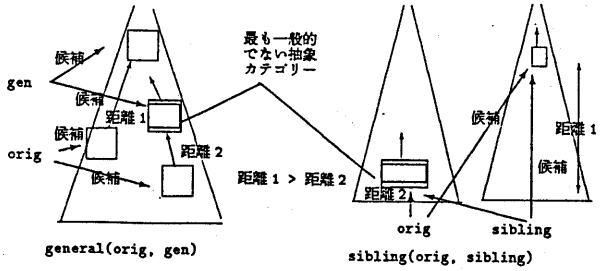

図 4: general(*orig, gen*) と sibling(*orig, sibling*) のもっともらしい解釈

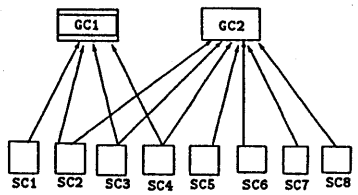

図 5: sibling 中の曖昧性解消に使われる「帰納的もっともらしさ」

リー毎に計算する。最も距離の小さい候補抽象カテゴリーが、最も一般的でない抽象カテゴリーとしてみなされ選択される (図 4 参照)。もう 1 つの直感的ルールは、帰納的にもっともらしいカテゴリーを選択すること、である。例えば、カテゴリー GC1 は、SC1, SC2, SC3, SC4 の抽象カテゴリーであるとし、カテゴリー GC2 は、S2, S3, S4, S5, S6, S7, S8 の抽象カテゴリーであるとする (図 5 参照)。ここで、パターン・概念対定義者が sibling(*some-pattern*, [SC2, SC3]) と記述すると、GC1, GC2 のいずれもこの汎化の候補解となる。この記述には、GC1 配下の詳細カテゴリー 4 つのうちの 2 つが含まれているので、2/4 が GC1 を支持しているとみなされる。一方、GC2 については 2/7 が支持している。そこで、GC1 の方が GC2 より「帰納的にもっともらしい」とみなされて、上の記述の評価として、GC1 が選択される。

これらの曖昧性解消の為の直感的ルールは、非常に直感的であり、論理的正当性はない。従って、WI は、これらの直観的ルールを使って仮に 1 つの候補を選択しても、その後にパターン・概念対定義者にそれがパターン・概念対定義者が意図したものであったかどうかを問い合わせる。ただ、これらの直感的ルールは、パターン・概念対定義者が、より自然にかつ手を抜いて定義でき、しかも大抵その解釈がうまくいくことが目的であるのでこれでよい。

5 実験と予備的評価

実験として我々は、電話機の利用方法に関する自然言語質問応答システムを CBP を使って試作中である。この節では、その実験から分かった予備的な評価について説明する。まず、我々はその電話機の初心者とエキスパートとの間の会話を収集した。

5.1 CBP は汎用自然言語インタフェースの機構として充分強力だろうか?

CBP が取っている手法は非現実的と考える人は多いに違いない。その人たちは、「我々が使う言語の言い回しは無限にあるので CBP の手法で自然言語インタフェースを実現しようとすると無限個のパターン・概念対を定義しなくてはならないはずだ」、という主張をすると思われる。もし、以下の 3 つの仮定の内の少なくとも 1 つが満足されれば、上の主張は正しい。その仮定とは、文中に無限個のボキャブラリがあること、文中に無限種類の文法があること、文の長さが無限長あること、である [7]。北野 [7] のスピーチコーパスの評価においては、これらの仮説は満足されない。例えば、スピーチは、人間にとって一種のリアルタイムのタスクなので、実際には人間は限定された数のボキャブラリと文法で発話をしている。また、人間のスピーチでは長さが短い文が大部分を占めてお

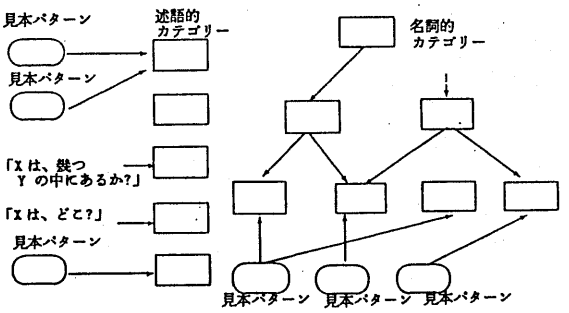

図 6: 構築されるカテゴリー世界の一般的構造

り、平均 10 語長程度でほとんどの発話は 20 語長未満であった。

我々の実験環境は、北野のそれとは少し条件が異なる。我々の場合、二人の会話はスピーチでなく、計算機端末間でのキーボードによる入出力の形で会話コーパスを収集した。しかしながら、それでも上記 3 つの条件を満足していない点では同様である。我々の実験環境でも、会話はリアルタイムタスクとみなせるので、最初の 2 つの仮定は満足していない。また、キーボード入力はスピーチより会話者にとって大変な仕事なので、我々の収集したコーパス中の文の長さは北野の評価結果より更に短く、平均語長は１０語未満であった。一般の自然言語インタフェースの環境を考えた時、その環境は我々の実験環境や北野のそれと似たものになると想定してもさしつかえない。それゆえ、CBP の手法は一般的な自然言語インタフェースを構築する為の機構として有効であると期待できる。

5.2 パターン・概念対定義者の経験的な考察

我々は、収集したコーパスを CBP の機構で定義した。以下は、CBP によるパターン・概念対定義のしやすさに関する経験的な考察である。CBP のアプローチは、「人は同じまたは似ている言い回しを繰り返し使う」という希望的仮説に非常に強く依存している。実際のところ、似ている言語パターンは繰り返し使われる。とりわけ、同一の会話セッション中では、話者が交互に少し形を変えて似た言回しを使うことが多い。そのような時には、定義したパターン・概念対がすぐにも役に立つ。

多くの文は充分短いので、そのまま見本パターンになってしまう。数少ない複文構造のものは、パターン・概念対定義者によって分割されて定義される。「部品がどこにあるかを尋ねる質問 (X:object)・カテゴリー」のような述語的でそれが 1 つの文に対応するようなカテゴリーを表すための言い回しにはいろいろな表現方法がある。このような述語的カテゴリーは数え上げ的に定義されていく。また、多くの少しだけ形が異なる言い回しがそれぞれ別の見本パターンとして同一の述語的カテゴリーに保持されていく。つまり、述語的なカテゴリーに対する見本パターンは当初の我々の予測より少ししか一般化されなかった。その理由は、我々が言語パターンから直訳的でなく意訳的に変換するようなレベルにカテゴリー定義をしたためだと思われる。

一方、名詞的なカテゴリー、例えば「転送ボタン」のような物理的オブジェクトや「留守番電話」のような抽象的概念オブジェクトについては、概念によるカテゴリー構造の階層木を容易に構築できた。パターン・概念対定義者は、異なる視点ごとに、多くの小さいカテゴリー階層木を作っていく傾向がある。図 6 は、パターン・概念対定義によってつくられる典型的なカテゴリー世界の概念図である。

6 関連研究

この研究の最初の動機は、フレーズベースの自然言語解析機構におけるパターン・概念対定義の仕方を向上させることであった。CBP は、UC [16] の PHRAN [1] や ALANA [4] に比べるとパターン・概念対を定義しやすく

なっている。しかし、UCの研究者たちは、自然言語解析をプランニングや問題解決のような、より高次のレベルの推論機構と結合することに重点を置いていた。それゆえ、彼らはUCの意味表現に *Build and Construct* [9] パラダイムを採用した。我々は、その代わりに意味表現に分類パラダイム (heuristic classification) [3] を採用した。その理由は、その方がより静的で定義がしやすいからであるし、特定の対象領域をカバーすればよい自然言語インタフェースを構築するには分類パラダイムで充分と思われたからである。LIFER [6] は、CBPの祖先とも言える。CBPは一見するとLIFERと似ているように見えるが、CBPは自然言語パーザというより知識獲得機構というべきである。また、両者の知識表現は異なる。同様に、CBPは言語や単語の自動獲得を目指した研究とも一線を画する [8] [17]。CBPは機械学習でなく、あくまでも人が知識を入力していくのを支援する為の機構である。

CBPの知識表現は、exemplar-based な知識獲得機構であるPROTOS [2] に非常に強く影響を受けた。見本パターンやカテゴリー構造の考え方はPROTOSに負っている。CBPは、事例ベース推論とも親近性がある。事例ベース推論では、一般的な知識でなく、過去の特定の経験的知識を利用する。CBPのパターン・概念対記述は実際のコーパスを直接使ったものなので「事例ベース」の一種と考えることもできる。多くの事例ベースシステムの中で、CBPはDMAP [9] に近い。DMAPもまた、フレーズベースの自然言語理解システムである。DMAPが他のフレーズベースシステムと違う点は、パターン・概念対の概念側がパターン側を決定するのに使われる、という点である [11]。これは普通のフレーズベースシステムとちょうど逆である。DMAPは、新しい入力文を読むと自分自身の内部の知識ベースを変更していく。また、入力文を誤って理解した時の自動修正機構も有している。CBPの目的は、容易なパターン・概念対定義機構を構築することなので、DMAPのこのような野心的な特徴的機構に比べるとその特徴は保守的である。その代りに、パターン・概念対の定義の簡単さにおいてはMicro-DMAPに比べて勝っている [10]。CBPは、幾つかの事例ベース推論システムが誇る複雑で洗練された推論機構というよりは、コーパスデータそれ自身に依っているという点で、memory-based 推論機構 [15] の方により親近性があるとも言える。

7 まとめ

コーパス解析なしにシステム設計を行うと、どのような知識表現形態・推論機構を採用しても結局実際の会話の処理から離れてしまう。実際の発話は、パターン・概念対定義者の想像力を大きく凌駕しているので、実際の発話データがパターン・概念対定義のブートストラップ (bootstrap) として使われるべきである。CBPでは対象領域の実際のコーパスデータを直接パターン・概念対定義に使っている。パターン・概念対定義の個々のパターンは、コーパスデータの変形・汎化形である。パターン・概念対定義者はCBPが提供する汎化オペレータを使って元のパターンを変形・汎化する。変形・汎化されたパターンは見本パターンとして保持されることになる。

CBPの知識表現はexemplar指向である。個々のカテゴリーは1つ以上の見本パターンの集合で表わされる。コーパス中の典型的な言い回しだけが見本パターンとして定義され保持される。CBPは対話的にパターン・概念対定義をしていく機構を提供している。収集したコーパスに沿って1つずつ文を定義していく。コーパス中に新しい文パターンを見つけると、CBPはパターン・概念対定義者と共同してその意味表現を定義していく。CBPはパターンマッチングの失敗から過去の定義の誤りを修正する機構も用意している。

現在、我々は特定の電話機に対する自然言語質問応答システムをCBPを使って構築中である。その経験によると、CBPのアプローチは一般的な自然言語インタフェース構築機構としても充分強力である。

参考文献

[1] Arens, Y., "CLUSTERS: An Approach to Contextual Language Understanding", Rep. UCB/CSD 86/293, Ph.D.

Thesis, 1986.

[2] Bareiss, E.R., "Exemplar-based Knowledge Acquisition: A Unified Approach to Concept Representation, Classification, and Learning.", Academic Press, 1989.

[3] Clancey, W.J., "Heuristic Classification", Artificial Intelligence, 27:289-350, 1985.

[4] Cox, C.A., "ALANA Augmentable LANguage Analyzer", Rep. UCB/CSD 86/283, 1986.

[5] Hammond, K., "CHEF: A model of case-based planning", Proc. of the National Conf. on Artificial Intelligence, 1986.

[6] Hendrix, G.G., Sacerdoti, E.D., Sagalowicz, D., and Slocum, J., "Developing a Natural Language Interface to Complex Data", In ACM Trans. on Database Systems, 1978.

[7] Kitano, H. and Higuchi, T., "Massively Parallel Memory-Based Parsing", Carnegie Mellon University, 1990.

[8] Jacobs, P. and Zernik, U., "Acquiring Lexical Knowledge from Text: A Case Study", Proc. of the National Conf. on Artificial Intelligence, 1988.

[9] Martin, C.E. and Riesbeck, C.K., "Uniform Parsing and Inferencing for Learning", Proc. of the National Conf. on Artificial Intelligence, 1986.

[10] Martin, C.E., "Case-based Parsing", In *Inside Case-based Reasoning* edited by R. Schank and C. Riesbeck, Lawrence Erlbaum Associates, Hillsdale, NJ, 1989.

[11] Martin, C.E., "Pragmatic Interpretation and Ambiguity", the 11th Annual Conference of the Cognitive Science Society, 1989.

[12] Pereira, F., and Warren, D.H.D., "Definite Clause Grammars for Natural Language Analysis", Artificial Intelligence, 13:231-278, 1980.

[13] Riesbeck, C.K., Schank, R.C., "Inside Case-based Reasoning", Lawrence Erlbaum Associates, Hillsdale, NJ, 1989.

[14] Smith, E., and Medin, D., "Categories and Concepts", Cambridge: Harvard University Press, 1981.

[15] Waltz, D.L., "Massively Parallel AI", Proc. of the National Conf. on Artificial Intelligence, 1990.

[16] Wilensky, R. et. al., "UC – A Progress Report", Rep. UCB/CSD 87/303, 1986.

[17] Zernik, U., and Dyer, M., "The self-extending phrase lexicon", Journal of Computational Linguistics, The special Issue on the Lexicon, 1988.