非決定的文解析エキスパートシステム

吉野　利明，　二村　高代，　泉田　義男，　牧之内　顕文
（富士通研究所）

１．はじめに

初心者向けのユーザインタフェース，コマンドを習得せずにそのシステムを使用したいユーザへのインタフェース，特定のシステムの使用頻度の低いユーザへのインタフェース等を考えると，自然言語（日本語）でユーザとシステムが対話するシステムが最も理想的である。すなわち，ユーザが日本語で命令や要求を入力すると，システムはその応答を日本語または日本語の解説付でグラフ，図形，音声等で返すというシステムである。

以上のことを目標として，我々は自然言語インタフェースを開発している。現在，この自然言語インタフェースを日本語によるデータベース検索システムＫＩＤ［２］に適用している。このような自然言語インタフェースシステムを構築しようとする際いくつかの問題が挙げられる。その中の一つにユーザの入力した入力文の解析の問題がある。本報では，ＫＩＤにおける自然言語解析の原理とそれに基づいた解析例について述べる。

ＫＩＤのパーサは以下のような特徴を持つ。

(1) システム全体をオブジェクト指向で構成し，構文ルールの実行はルール指向のメカニズムで行う。

(2) 日本語の文法カテゴリ文，節，単語を文法属性に基づいて分類し，階層構造を持つ言語モデルを定義する。このクラス内に構文ルールを記述することにより，ルールのモジュール化を促進し，変更を容易にする。

(3) 非決定的メカニズムを採用し，解析中の曖昧さをすべて保持できる。さらに入力文の多義の出力順は構文ルールと意味処理により制御可能にする。

文解析エキスパートシステムの構成を図１に示す。知識ベースとして，構文ルールを記述している言語モデルと応用分野の知識を記述している世界モデルを持つ。また，再実行機能とステップ機能を持つデバッガおよび説明機能を備えた推論制御部がある。さらに，ルール変更用のルールエディタを備える。

オブジェクト指向を利用して自然言語の構文・意味解析を行うシステムにはＯＤＤＳ［１］がある。ＯＤＤＳでは，構文処理用のメソッドと意味情報を同じ階層のオブジェクト内に記述し，意味主導型の構文解析を行っている。これに対し，本パーサは文法的カテゴリで構文ルールを分類，オブジェクト内に階層化している。分野の事物の意味関係は世界モデル［２］中に定義し，構文ルールからこの世界モデルを参照し，意味チェックを実行する方式を取っている。これは，構文ルールを分野のモデルから分離するためである。

２章ではパーサの開発方針，３章ではパーサが利用する言語モデル，４章ではメカニズム，５章ではデバッグツール，６章では解析例について述べる。

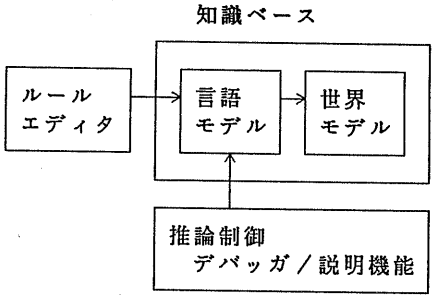


図１．文解析エキスパート

２．パーサの開発方針

パーサの開発に際して次の５個の方針を掲げた。

(1) 非決定的メカニズムの採用

我々は以前，構文解析ツールとしてＴＥＣ［３］［４］を用いていた。これはＭａｒｃｕｓが考案したパーサＰＡＲＳＩＦＡＬ［５］にボトムアップ解析機能を追加し，日本語解析用に変更したものである。ＴＥＣパーサの特徴は先読みを許し，意味解析を同時に行なうことにより，解析を一意に実行する決定的メカニズムを採用していることである。しかしながら日本語の入力文には，たとえ分野を特定しても曖昧な文が存在する。

（例１）　富士通屋は 500円で川崎工場が生産する商品を販売する。

例１において名詞節「 500円で」は「生産する」と「販売する」の２文節のどちらにも係る解釈が考えられる。このような場合，システムがこれらの解釈のうち一方を選択するよりもユーザに２つの解釈を提示して判断を仰いだほうが良い。すなわち解析途中で現れた入力文の曖昧さをシステムはすべて保持しながら解析を進め，文解析終了時でも曖昧さが残る時には，それらの解釈を順次ユーザに提示するような非決定的メカニズムを採用する。

(2)　文法ルールをオブジェクト指向の考え方でモデル化する

パーサを含めシステム全体を知識プログラミングシステムＭＩＮＥＲＶＡで記述しモジュール間の制御を容易にする。特に構文ルールは，文，節，単語の単位を品詞，付属語等の文法属性で分類したクラスの中にプロダクションルールとして記述される。このクラスオブジェクトは階層構造を持ち，言語モデルと呼ばれる。この構文ルールの起動はルール指向のメカニズムで行われる。階層構造を持つ言語モデルの特徴を以下に示す。

・　文法事項チェック（例えば，「は名詞節」は「名詞節」の下位クラスである）に階層構造を使用することで，ルールの記述性を上げることができる。

・　指定クラスにルールセットがないときは，上位クラスのルールセットを遺伝することができる。そのために，記述するルールセット数を減らすことができる。さらに，下位クラスにそのクラス特有のルールセットの記述を省略した場合でも，上位クラスのルールセットの適用（粗い解析になる可能性があるが）をシステムは試みることができる。

・　意味処理関数は言語モデルのクラス内にメソッドとして記述することにより，構文ルールと統一的に管理できる。

(3)　構文・意味の融合型パーサとする

解析中に曖昧さを保持するメカニズムを取るが，構文解析と同時に意味チェックを行ない，曖昧さを減らす。構文解釈のみでは複数個の解析木が生成可能なときでも，意味チェックを同時に実行することにより，意味的に不整合な解析木の生成を止め，解釈の可能性の候補をへらすことができる。なお意味チェックには，上位・下位関係と属性関係の２つの関係およびクラスからなる世界モデルを使用する。

(4)　デバッグツールを整備する

非決定的メカニズムを採用すると，構文ルールや意味処理関数のバグのために思わぬ方向に解析が進んでしまいエラーとなったり，結果を何も出さずに解析が終了したりする場合が起りえる。このような非決定的な解析を採用したことにより生ずる解析が容易でない

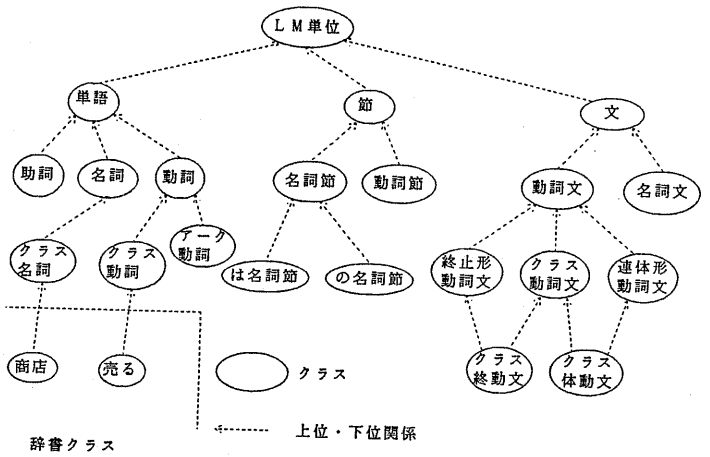


図２．言語モデルクラス

バグに対処するために，一文の解析終了後に，解析途中の任意の状態を調べる。さらに，ルールの修正後，解析途中の状態にもどり，再実行をする等のデバッグツールを整備し，デバッグを容易にする。

(5) ルールは頑強性を持つ

入力文には，助詞の省略，誤記等が含まれ，統語的にはいくぶん不自然な文がかなり含まれる。このような文であっても分野を特定した世界モデルを利用すると，解釈が決定してしまう例が見られる。不自然な入力文であっても，世界モデルを参照する範囲内で解釈可能ならば，それを実行できるような頑強性のある構文ルールを設計し，システムが受理できる入力文の範囲を広げる。

3．言語モデル

本章では，パーサが使用する言語モデルの構成の基本的考え方を挙げる。さらに言語モデルと意味処理で参照する世界モデル［2］の関係について概説する。

本パーサではクラスオブジェクトとして階層構造を持つ言語モデルクラスを定義し，構文ルールはこのモデルのクラス内に記述する。言語モデルを使用することにより構文ルールのモジュール化を実現し，拡張性，保守性を向上させている。

3．1言語モデルの構成

言語モデルの構成の概要を図2に示す。言語モデル中には構文ルールが記述してある。モデル内の各クラスは構文解析木のノードになりえるクラスである。クラスの分類の基準の概要は以下の通りである。

- 言語モデルのルールセットを含むクラスの最上位クラスは「単位クラス」であり，これは「単語クラス」，「節クラス」，「文クラス」に分けられる。
- 「名詞節クラス」，「名詞文（体言止めの文，述語節の中心が名詞の文）クラス」等の体言節，体言文は付属語（付属語に活用形があればその活用形も含めて）で分類する。日本語において名詞節の係りを決定するのは，名詞節中の助詞または助動詞とその活用形であるからこれを分類の基準とする。
- 「動詞節クラス」，「動詞文クラス」等の用言節，用言文は付属語（付属語がないときは用言そのもの）の活用形で分類する。用言節の係りを決定するのは，節全体の活用形であるからこれを分類の基準とする。
- 「単語クラス」は品詞で分類する。「単語クラス」の最下位クラスは辞書クラスであり，入力文中の単語列はこの辞書クラスのインスタンス列に変換される。
- 名詞（名詞節，名詞文），動詞（動詞節，動詞文）を文法基準以外に世界モデル対応の補助的基準でも分類している。例えば，「名詞クラス」は「クラス名詞クラス」と，それ以外に分類される。「クラス名詞クラス」とは世界モデルの中に概念的に対応するクラスを持っている名詞のことである。「動詞クラス」は「クラス動詞クラス」と「アーク動詞クラス」に分類される。前者は世界モデルの中に対応するクラスを持っている動詞のことである。後者は世界モデルの中に対応するアークを持っている動詞のことである。

階層関係はルールの記述性を高めるためにも有用である。図2において，「は名詞節」，「の名詞節」は「名詞節」の下位クラスであるため，「名詞節」の属性を継承しており，構文情報のチェック時にこの継承を利用できる。ルールセットや意味処理のメソッドもこの継承を利用する。

3．2 ルールセット

言語モデルクラスのルールセットは1～数個のルールより構成され，図3の形式を取る。

control 属性内の値について説明する。DO1 は最初に条件部を満たしたルールのみを実行する。DOALL は条件部を満たしたすべてのルールを実行する。WHILE1 はCONDスロットの値がNIL になるまでDO1 を繰り返す。WHILEALLはCONDスロットの値がNIL になるまでDOALL を繰り返す。

3．3 言語モデルと世界モデル

ここでは言語モデルと意味処理で参照する世界モデルとの対応について説明する。世界モデルはクラスとアークをもちいて，対象分野における事物と事物間の関係を表現したものである。

言語モデルクラスの下位クラスに辞書クラスがある。この辞書クラスは単語を表わす。入力文の単語列は辞書クラスのインスタンス列に変換される。言語モデル

クラスと辞書クラスと世界モデルクラスの対応の例を図4に示す。

世界モデルのクラスに対応する辞書クラスは名詞，動詞および他の自立語の一部である。対応は辞書クラスのオブジェクト中にWORLD-MODEL 属性の値として記述する。この辞書クラスに対応している世界モデルのクラスは入力文により参照されるとインスタンス化される。これらの世界モデルクラスのインスタンスは属性関係に沿ってネットワークを形成し，解析木のノードに付加される。そして，その解析木のノードが言及する対象分野での範囲を示す。

4．パーサのメカニズム

本章では，パーサの制御に関係するクラスオブジェクトとパーサのメカニズムについて説明する。パーサの中には，3つの制御に関係するオブジェクト　－　コントローラ，ステート，セグメンタ　－　が存在する。コントローラはパーサの全体の制御を行う。ステートは，解析途中の状態を保持する。セグメンタは入力文の単語分割処理を行うオブジェクトである。解析途中の状態はステートとして保持され，曖昧さが生じた時には複数個のステートが生成，保持される。さらに各々のステートにはプライオリティが定義でき，入力文が多義を持つ場合には，そのプライオリティに従ってユーザに順次解釈を提示できる。パーサ全体の制御はメッセージパッシングにより行なわれる。

4．1　コントローラクラス

コントローラクラスは，パーサの制御の中心となるオブジェクトである。一入力に対してコントローラクラスが一つだけインスタンス化（単にコントローラというときには，このインスタンスを指す）される。このインスタンスは，複数のステートを保持することができる。このインスタンスにパースのメッセージを送ると，もしこのインスタンスがステートを持っていればその中から実行プライオリティの最も高いステートを選び，そのステートにパースのメッセージを出し，解析の実行を促す。もしインスタンスがステートを一つも持っていなければセグメンタクラス（制御関係オブジェクトの一つ）のインスタンスに対し，単語分割の次候補を要求する。次候補（入力文の単語のインス

```
オブジェクト名
  (スロット)    (バリュー)
  control       :  制御情報 do1, doall, while1,
                   whileallの一つをもつ
  workspace     :  メッセージの引数を示す
  cond          :  ルールセットの繰り返し実行
                   (ｗｈｉｌｅ1，ｗｈｉｌｅａｌ
                   １ ) の条件を示す
  tempvar       :  ルールセット内のローカル変数
  rulevar       :  ルールセット内のローカル変数
                   条件部のパターンマッチ変数として
                   使用
  tracevar      :  トレースの対象変数
  tracevar-after-action
                :  実行部実行後のトレース対象変
                   数
  option        :  トレースの指定
  rules         :  (ルール記述部)
                   ルール名１
                     IF  条件部１
                     TNEN  実行部１
                     ・  ・  ・
                   ルール名Ｎ
                     IF  条件部Ｎ
                     THEN  実行部Ｎ
```

図3　ルールセットの記述形式

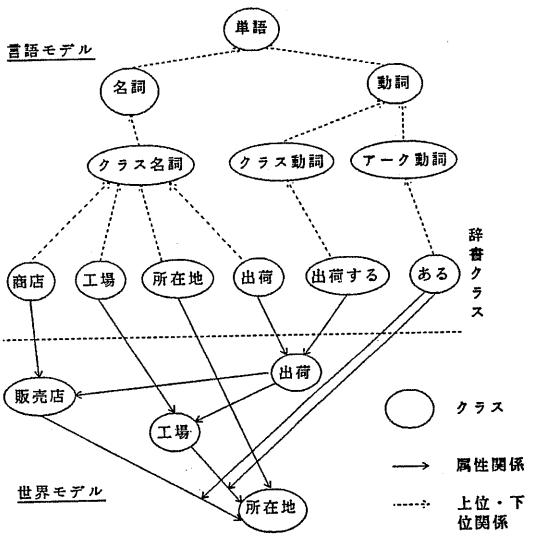


図4．言語モデルと世界モデルの関係

タンス列）が返ってくればステートを生成し，そのlook-aheadバッファ（図5参照）に単語のインスタンス列を入れる。そしてそのステートにパースのメッセージを送る。次候補がなければ解析木の生成が不可能であるからエラーである。パースのメッセージ動作が終了すると今回実行したステートをコントローラより消去する。

4．2　セグメンタ

セグメンタはパーサからのセグメント実行メッセージを受け取ることにより起動する。ユーザの入力文を単語単位に分割し，辞書クラスのインスタンス列を生成する。分割候補の優先度付けには文節数最小法［6］［7］を採用している。セグメンタは一回目の呼び出し時には，ユーザに入力文を要求し，一番目の候補，すなわち最小文節数の候補を返す。二回目以降の呼び出しでは次候補を一つずつ順に返す。分割の候補がなくなるまでこれを繰り返す。

4．3　ステートクラス

ステートクラスは制御関係オブジェクトの一つである。このクラスのインスタンスは2つのバッファを持つ。バッファ中には解析途中の木と木のノードに対応するユーザの言及範囲を世界モデルの部分ネットワークとして記述している。さらにそのインスタンスはステート（ステートと呼ぶときは，ステートクラスのインスタンスを指す）自身の優先度を示すステートプライオリティを持っている。

ステートは図5に示すようにlook-back とlook-aheadの2つのバッファを持つ。look-back バッファには，解析を延期した解析木のノードがはいる。右から（文頭に向って）L1（left1），L2，L3と呼ぶ。look-aheadバッファには，まだ解析されていないノードがはいっている。左から（文末に向って）C（current），R1（right 1），R2，R3と続く。

ステートがパースのメッセージを受け取ると，バッファ中のC（current）にルールRUNのメッセージを送る。それにより，C中のルートノードのルールセット（ルールノード自身はインスタンスなので，そのクラス中のルールセットまたはそのクラスにルールセットがないときは上位クラス中のルールセット）が起動される。

ルールセット内のルールの条件部と実行部では主に次のことを行う。

(a)　条件部

・　バッファの構文事項チェック

・　Cノードと他のバッファ中のノードとの意味チェック

(b)　実行部

・　ステートクラスインスタンスの生成

・　必要があれば新ノード（言語モデルクラスのインスタンス）を生成する。

・　ノード同士のアタッチ（木の生成）

・　新ステートへステートプライオリティをセットする。

・　新ステートをコントローラに追加する。

「の名詞節」ルールセットの概略を図6に示す。

4.4　プライオリティ

コントローラ中には複数個のステートが存在している。このとき，次にどのステートを実行すべきかはステートの持つプライオリティとステートの状態によって決る。以下に本パーサのプライオリティ決定のための要因について概説する。

(1)　構文的要因

ルールセット中のルール実行部に決定要因を記述する。これはルールセットの中で特に構文的要因を考慮して構文ルール内に記述するプライオリティ決定要因である。例えば，連体形に並立と解釈可能な名詞が続く時，並立の処理を先に実行したいなら，連体形の処理を延期して並立処理を行うようにしたステートのプライオリティをあげる。ルール内に記述するプライオリティ決定要因の代表的な値を下にあげる。

プライオリティ

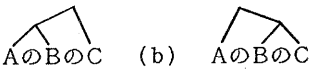
0.8　：　早く実行したいステートを生成する時に使用

(例)　　AのBとC

「BとC」の並立を早目に見るため，「Aの」の係り処理を一時延期し，「Bと」をCURRENTバッファにいれたステートを生成する時に使用する。

1　：　基準値

1.2　：　実行を延期したいステートを生成するときに使用する。



「AのBのC」とうい表現では例に示す2つの解析木を作ることが可能である。(a)を生成する処理を(b)よりもはやく実行したい時は，「Aの」の係り受け処理を行う時，(a)をつくるためのステートを生成するルール内の構文的プライオリティ要因を1とし，(b)をつくるためのそれを1.2 とする。

(2) 意味関数要因

意味関数がプライオリティを決定する要因を返す。意味関数の結果，特に，入力文中の格と辞書中の格情報が不一致のときに優先順位を下げる。格が不一致のときでも，そのステートのプライオリティを下げてコントローラ中に残す。そして他の可能性が失敗したとき，再実行される。これにより頑強性が保障されている。

プライオリティ

1　：　基準値

2　：　動詞の格処理で格が不一致

(3) ノード数要因

残っている節（ルートノード数）の少ない方を優先する。これはdepth first 的な探索を導入するためである。

本パーサでは，実行プライオリティとステートプライオリティの2種のプライオリティを定義している。

ステートプライオリティは，各々のステートが持つプライオリティで前述の(1)，(2)と前ステートのプライオリティの積で定義される。

ステートプライオリティ

＝前ステートプライオリティ

×ルールセット内要因

×意味関数要因

このステートプライオリティを基にして実行プライオリティを定義する。この実行プライオリティの最も高い（数値としては最も小さい）ステートをコントローラが選び実行する。

実行プライオリティ

＝ステートプライオリティ

×1.1＊＊残っている節数

5．デバッグツール

非決定的メカニズムを採用すると，構文ルールや意味処理関数のバグのために思わぬ方向に解析が進んでしまいエラーとなったり，結果を何も出さずに解析が終了したりする場合が起りえる。このような非決定的な解析を採用したことにより生ずる解析が容易でないバグに対処するために以下のようなデバッグ機能を備えている。

- 主記憶上のオブジェクトのルールセットクラスのルールを変更，修正するルール編集機能
- ルール変更，修正後，解析途中のユーザ指定のステートにもどり，再実行する機能
- ルールセットの起動を指定ステップずつ行うステッパ機能
- 指定したステートが，どのステートから，どのルールを使用して生成されたかをユーザに日本語で提示する説明機能

デバック機能の実例は次章で述べる。

6．解析例

入力文：川崎市の工場が出荷する商店は？

は，単語分割がなされ，文節統合処理により，図5に示す状態になる。この状態では，「の名詞節」ノードがCURRENT の位置にあるから「の名詞節」のルールセット（図6）が適用される。ルール『の名詞節ルール』の条件部がチェックされる。ここでは，2つの構文条件（R1が連体接続形でない）（L1が名詞節）を満足する。次の「の」意味チェックは「の名詞節」自身に意味チェックメソッドの起動のメッセージを送る。この例では，世界モデル中の「所在地クラス」と「工場クラス」に属性関係があるかをチェックする。条件部を満足するとルール実行部にはいる。ルール実行部では新ノード「が名詞節クラス」のインスタンスを生成し，木の生成およびプライオリティを設定してSTATE2

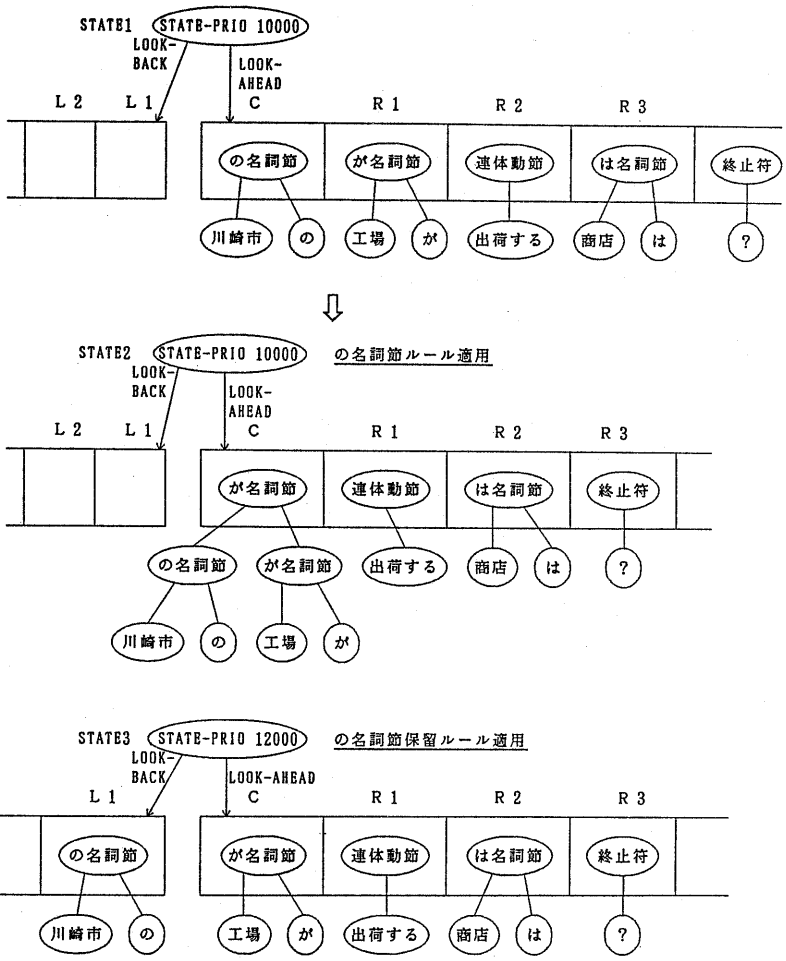


図５．解析状態例（「の名詞節」ルールセット適用例）

```
(の名詞節
   (メタクラス        (SEA:RULESET ))
   (スーパークラス    (名詞節)(連体接続形))
   (制御              (ＤＯＡＬＬ))
   (ワーク変数        ・・・          )
   (トレース変数      ・・・          )
   (一時変数          ・・・          )
   (ルール            (の名詞節ルール
                        トレース情報
                        (ＩＦ  (ＡＮＤ  (左１バッファ  が  連体接続形  でない)
                                        (右１バッファ  が  名詞節)
                                        (「の」意味チェック)))
                        (ＴＨＥＮ  新ステートの生成
                                   解析木の変形
                                   優先度の設定))
                      (の名詞節保留ルール
                        トレース情報
                        (ＩＦ  右２以降のバッファに名詞節あり)
                        (ＴＨＥＮ  新ステートの生成
                                   バッファの左シフト
                                   優先度の設定))

                        ・  ・  ・

                                                  ))
```

図６．「の名詞節」ルールセットの一部

を作る。同様に「の名詞節保留ルール」が適用されSTATE3を作る。この2つのステートがコントローラに格納される。ここでコントローラはデーモンを使用して実行プライオリティを計算し，これの高いSTATE2にルールRUN のメッセージを送る。それにより「が名詞節」のルールセットが適用される。

図7に例文の解析結果を示す。この例では2つの解釈が得られている。解析結果を出た状態あるいはエラーに陥った状態で，ユーザは，生成されたステートのトレースをとることができる。また任意ステートに生成された説明をもとめることができる。さらに，ルールを確認するため，またはルールを修正後に，任意のステートに対して再実行を要求できる。ステップ機能をもちいて，一または数回ずつにルールセットを適用することも可能である。このデバッグツールによりルールのバグの発見，再現が容易になると共に，バグ修正後，解析の途中ステートからの実行が可能となり，ルール修正の確認が容易になっている。

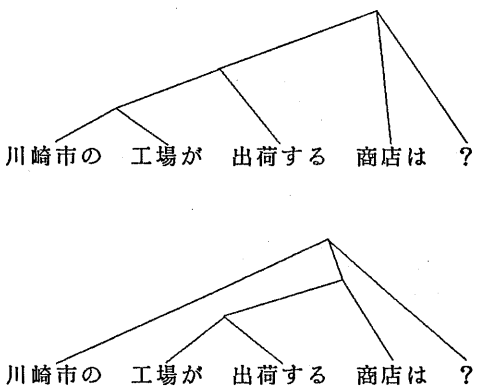


図7．解析結果

7．評価とまとめ

本パーサを使用して，販売分野の入力文の解析を試みた。この実験は，入力対象として並立記述，品詞としては形容詞，副詞を除いて行った。この実験での応答時間はＴＥＣパーサを使用した時の1／3という結果を得た。これは本パーサの方がルールの適用回数が少ないためである。

本稿では以下の特徴を持つ自然言語パーサについて述べた。

(1) 言語モデルクラス単位で文法ルールのモジュール化を実現し，拡張性に富む。

(2) 解析途中の任意のステートから再実行を行う等の機能を持つデバッグツールを持つ。

(3) 非決定的メカニズムの採用で，入力文の多義を順次出力する。

(4) ルール自身は頑強性を持つ等の特徴を実現している。

今後の課題としては，以下のようなものが残されている。

(1) プライオリティの計算関数と要因の評価，改良

(2) 先読みやプライオリティの低いステートの強制的切り捨てにより，ステート数の爆発を防ぐルール，メカニズムの作成

(3) 意味解析関数の整備

参考文献

(1)大澤・米澤；オブジェクト指向方式による対話理解システム，自然言語処理研究会 44-7，1984
(2)泉田他；対象世界のモデルを利用したデータベース検索システム，データベース・システム研究会 43-2, 1984
(3)杉山他；自然言語解析向き解析木作成ツールＴＥＣ，情報処理学会第２５回全国大会, pp. 1033-1034, 1982
(4)吉野他；ＫＩＤにおける日本語の構文・意味解析，情報処理学会第３０回全国大会, pp. 1425-1426, 1985
(5)M. P. Marcus；A Theory of Syntactic Recognition for Natural Language, 1980, The M. I. T. Press
(6)吉村他；文節数最小法を用いたべた書き日本語文の形態素解析，情報処理学会論文誌，Vol. 24, No. 1, pp. 40-46, 1983
(7)星合他；ＫＩＤにおける日本語文の形態素解析，情報処理学会第３０回全国大会, pp. 1423-1425, 1985