JAERI-M
89-218

日本語入力を受け付ける知識ベースシステムの試作

１９８９年１２月

神林　　奨・上中　淳二*

日　本　原　子　力　研　究　所
Japan Atomic Energy Research Institute

JAERI-Mレポートは，日本原子力研究所が不定期に公刊している研究報告書です。
入手の問合わせは，日本原子力研究所技術情報部情報資料課（〒319－11茨城県那珂郡東海村）あて，お申しこしください。なお，このほかに財団法人原子力弘済会資料センター（〒319-11茨城県那珂郡東海村日本原子力研究所内）で複写による実費頒布をおこなっております。

JAERI-M reports are issued irregularly.
Inquiries about availability of the reports should be addressed to Information Division, Department of Technical Information, Japan Atomic Energy Research Institute, Tokai-mura, Naka-gun, Ibaraki-ken 319-11, Japan.

編集兼発行　日本原子力研究所
印　　刷　日立高速印刷株式会社

JAERI－M 89－218

日本語入力を受け付ける知識ベースシステムの試作

日本原子力研究所東海研究所計算センター
神林　奨・上中　淳二*

（1989年11月29日受理）

自然言語を用いて人間と対話ができ，しかも，さまざまな作業を行うことのできる知能ロボットは，あらゆる分野において切望されている。計算センターが行っているＨＡＳＰ（Human Acts Simulation Program）では，そのような自然言語インタフェースを持つ知能ロボットを想定し，知能ロボットが原子力プラント内で保守・点検作業する際の思考・動作過程のシミュレーションを試みている。現在は，シミュレーションを実現するために，高速でかつ安定した自然言語処理システム，自然言語処理と連動した作業計画システム，施設形状データベース，ロボット運動学のシミュレーション結果の表示システムなどの要素技術の開発を行っている。我々は，a)自然文で書かれた作業指示の理解，b)それに連動した自動的な作業計画の２点について，それに関わるシステムの実現を目指している。

自然言語処理に対しては，日本語解析プログラムＣＳ－ＰＡＲＳＥＲを知能ロボットとのインターフェースとして導入し，その実行環境を整えてきた。今回，ＣＳ－PARSERの効率的かつ高速な運用のために，新しく辞書システムを開発した。この辞書システムはＣＳ－ＰＡＲＳＥＲと切り放して辞書項目の修正を行うことが可能であり，日本語解析にとても有効な環境を与えることになった。

また，作業計画システムに関しては，その制御構造を調査するために，二つシステムを試作した。一つは，日本語生成に主眼をおいて物語生成プログラムMicro ＴＡＬＥ－ＳPINを改良したプログラム，もう一つは，日本文入力に主眼をおいてＣＳ－ＰＡＰＳＥＲと結合した作業指示システムである。これら二つの試作システムの評価結果から，今後の作業計画システムの構成を考察する。

東海研究所：〒319-11 茨城県那珂郡東海村白方字白根２－４
＊外来研究員，㈱ＣＳＫ

Task Planning Systems with Natural Language Interface

Shaw KAMBAYASHI and Junji UENAKA*

Computing Center
Tokai Research Establishment
Japan Atomic Energy Research Institute
Tokai-mura, Naka-gun, Ibaraki-ken

(Received November 29, 1989)

In this report, a natural language analyzer and two different task planning systems are described.

In 1988, we have introduced a Japanese language analyzer named CS-PARSER for the input interface of the task planning system in the Human Acts Simulation Program (HASP). For the purpose of a high speed analysis, we have modified a dictionary system of the CS-PARSER by using C language description. It is found that the new dictionary system is very useful for a high speed analysis and an efficient maintenance of the dictionary.

For the study of the task planning problem, we have modified a story generating system named Micro TALE-SPIN to generate a story written in Japanese sentences. We have also constructed a planning system with natural language interface by using the CS-PARSER. Task planning processes and related knowledge bases of these systems are explained.

A concept design for a new task planning system will be also discussed from evaluations of above mentioned systems.

Keywords : Artificial Intelligence, Natural Language Processing, Knowledge Base, Planning

---

* On leave from CSK Corp.

目　　次

Contents

1. はじめに

計算センターでは，昭和62年度より人間動作シミュレーション・プログラム（Human Acts Simulation Program, HASP）の研究を開始した[1]。HASPでは，自然言語による入力インターフェイスを持つ知能ロボットを想定し，知能ロボットが原子力プラント内で保守・点検作業する際の論理的思考過程および動作過程のシミュレーションを試みている。HASPプロジェクトの目標は，知能ロボットのシミュレーションのみならず，シミュレーション結果を用いて，知能ロボットや知能化・自動化されたプラントを設計するための基盤技術を開発することにある。現在は，知能ロボットシミュレーションのための要素技術を開発中である。それらの要素技術として，高速でかつ安定した自然言語処理システム，自然言語処理と連動した作業計画システム，施設形状データベース，ロボット運動学のシミュレーション，画像認識システム，被曝線量評価システム，シミュレーション結果の表示システムなどが必要と考えている（Fig.1.1）。本報告書では，上述の自然言語処理と作業計画に対応する，a)作業指示の理解，b)自動的な作業計画の生成の2点について議論する。

我々は，知能ロボットに対する命令として，日常的な日本語表現を用いる。そこで，知能ロボットが命令を理解して作業計画を行うということは，日本語表現の命令から，最終的にロボットに積まれた機器（アクチュエータ，TVカメラなど）へのコマンドを生成することと等価である。ここで，このような知能ロボットの命令理解過程を考える前に，我々人間がどのように自分のなした行動を表現しているのか考察しよう。簡単な例として，“ある人がタバコに火をつける”という行動を取り上げる。この行動を一言で表現すれば，

① タバコに火をつける

となる。しかし，“タバコに火をつける”ということは，さまざまな動作・作業の組合せからできている。それらの動作を表現すると，

② タバコを見つける
　タバコをつかむ
　タバコをつかんだ手を，口元に移動する
　タバコをくわえる
　ライターを見つける
　ライターをつかむ
　ライターを着火する
　ライターをタバコのそばに移動する
　タバコに火のついたことを確認する
　ライターの火を消す

という一連の動作系列となる。また，②に示した動作も，もとを正せば人間各部の筋肉の動きや視聴覚機構の働きに帰着されるために，上に示した動作系列はもっと細かく表現できる。

③ {筋肉の動き，視聴覚機構の働きの集合}

このように，ある行動を記述するための表現形式には，階層構造が存在する。すなわち，上に述べた①は概念的表現であり，②は動作表現であり，そして③は各機構の制御表現である。我々が“日常的な日本語表現”と考えているのは，①レベルの表現である。

それでは，知能ロボットが日本語で書かれた命令を受けて，その意味を理解して，作業を行うためには，どのようなことを考えるべきなのだろうか。我々の考えている知能ロボットは，あくまで自律的に作業を行う。現在まで，広範囲の作業に対して，自律的に自分自身の行動を制御することのできる実体は人間だけであったということを考慮すると，知能ロボットシミュレーションのために参考とすべき方法論は，我々自身を解析することから得られると考えられる。そこで上にみてきた行動表現の３つの階層は，そのまま知能ロボットが作業計画する過程に当てはめることができると思われる。つまり，概念的に指示された命令（①レベル）から，それを構成する動作例（②レベル）を知識から検索し，知能ロボットに搭載した各機器へコマンドや変数（③レベル）を転送するという３つの処理過程が必要であると考えられる。

以上の議論では，ある環境で既に起こってしまった行動の表現形態から知能ロボットシミュレーションに必要な処理を考えてきた。しかし，行動を取り巻く環境は，常に一定であるとは限らず，作業計画の中には環境の変化を取り込む余地がなければならない。例えば，“タバコに火をつける”の例で，ライターを用いて火をつけたが，常にライターを用いるわけではない（マッチを用いる場合も予想される）。さらに，知能ロボットに対する命令も，ロボットを取り巻く環境の変化と捉えることができるので，我々の目指すロボットシミュレーションの制御構造は，常に環境変化を感知し，それに関係する①～③レベルの作業計画（行動計画）を自動的に生成することのできるものでなければならない（Fig.1.2）。

以下では，これまで述べてきた命令理解と作業計画のための要素技術として，Fig.1.2に示した作業項目の概要を述べる。

日本語で記述された命令文から，命令の示す概念（①レベル）を導出するには，知能ロボットのおかれた環境での自然言語処理を行う必要がある。この際に用いる知識は，

１）構文・意味構造を導出するための辞書，文法規則

２）単語と環境内に存在する実体・動作概念との対応関係

の２つである。１）の知識を用いた命令文の構文・意味を導出するプログラムとして，日本語解析プログラムＣＳ－ＰＡＲＳＥＲ（㈱ＣＳＫ）を導入した。ＣＳ－ＰＡＲＳＥＲは，高木・伊東の提案した意味表現手法[2]に基づいて構成されている。高木・伊東の意味表現手法は，“風船がふくらむ”，“風船の体積が増加する”といった言い回し文（表面上は異なった構文だが，意味の同じ文）を同一の意味構造に表現可能という優れた特徴を持っている。ＣＳ－ＰＡＲＳＥＲをＨＡＳＰで運用するためには，命令文に合わせて辞書環境を整備しなければならない。このため，ＣＳ－ＰＡＲＳＥＲ用の辞書エディタを開発し，辞書項目の登録を行っている。ＣＳ－ＰＡＲＳＥＲの機能と辞書のエディタについては，２章で詳しく述べる。ＣＳ－ＰＡＲＳＥＲから出力される構文・意味構造は，２）の知識を用いて，環境に即した概念に変換される。

次に行う処理では，自然言語処理によって①レベルの概念に変換された命令文を，②レベル

の動作例に展開する。ここで用いる方法は，プランニングである。プランニングでは，

〔～する〕ためには〔～しなければならない〕

といったルール（知識）を用意しておき，命令概念に必要な動作列を導出する。この処理を検討するために，3章で物語生成プログラムMicro TALE－SPIN[3] のプランニング過程を解析する。Micro TALE－SPIN の用いているプランニング理論は，R.C.Schankらが提案したゴールとプランの理論である[4]。この理論では，ゴールの階層を定義し，それを解決するためのプランを用意して，事象の発生した原因や背景を表現する。Micro TALE－SPIN は，このゴールとプランの理論を用いて，登場人物が“喉の渇き”や“空腹”といった簡単なゴールを解決していく過程を，英語表現に変換して物語を生成する。さらに，この理論が日本語を対象としても利用できることを示すために，日本語表現の物語を生成するようにプログラムを書き換えた。これについては，3章に詳しく述べる。上に述べてきた知能ロボットシミュレーションのための処理をまとめた試作システムとして，日本語命令を受け，簡単な行動をとる知識ベースシステムを4章で述べる。このシステムでは，CS－PARSERを用いて命令文の日本語解析を行い，プランニングのための知識を用いて対象世界での行動に変換する。この結果，きわめて柔軟に日本語入力に対応するプランニングシステムが実現された。

ここまで述べてきた処理，すなわち①，②レベルの作業計画については，試作システムを構成するまでの方法論がはっきりしてきた。しかし，③レベルの処理は，まだ手をつけられない状態である。この処理では，ロボット運動の制御や視聴認識システムの制御などと協調した処理が必要であり，それら要素技術の展開を慎重に見極めて，我々の構築する作業計画システムを改良していかなければならない。また，知能化されたプラントを考えた場合，知能ロボットが参照する知識をどのようにデータベースにまとめたらよいのかという問題も残っている。次章以降，これら残された問題を検討する意味で，我々の試作してきたシステムを述べていきたい。

参考文献

1　浅井　他：原子力知能システム技術の研究，JAERI－M 89－023（1989）.

2　高木，伊東：自然言語の処理，丸善（1987）.

3　R.C.Schank and C.K.Riesbeck 編，石崎　監訳：自然言語処理入門，総研出版（1986）.

4　R.C.Schank and R.P.Abelson, “Scripts, Plans, Goals and Understanding”, Lawrence Erlbaum Associates, N.J.（1977）.

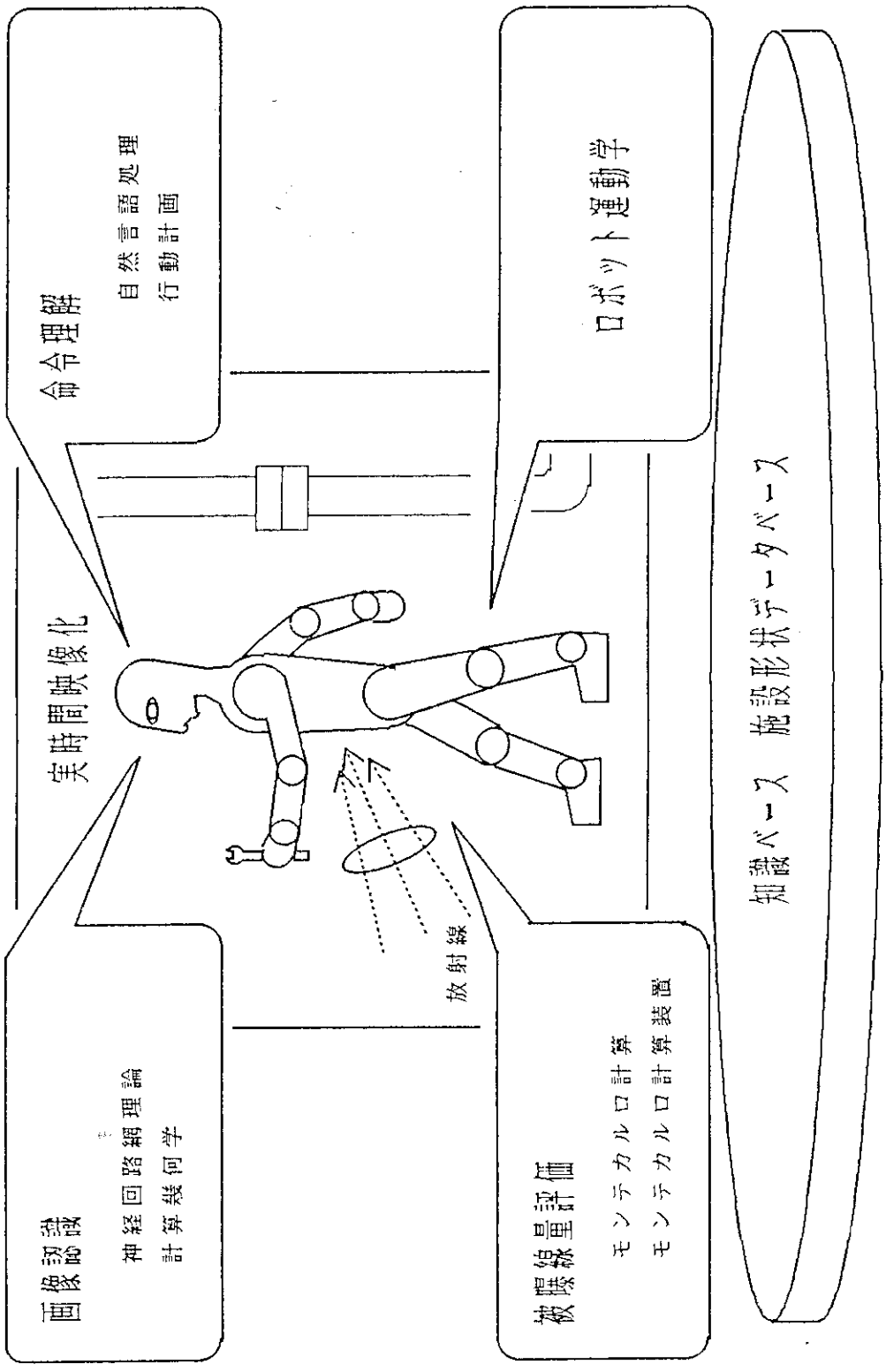


Fig. 1.1 A concept of Human Acts Simulation Program (HASP).

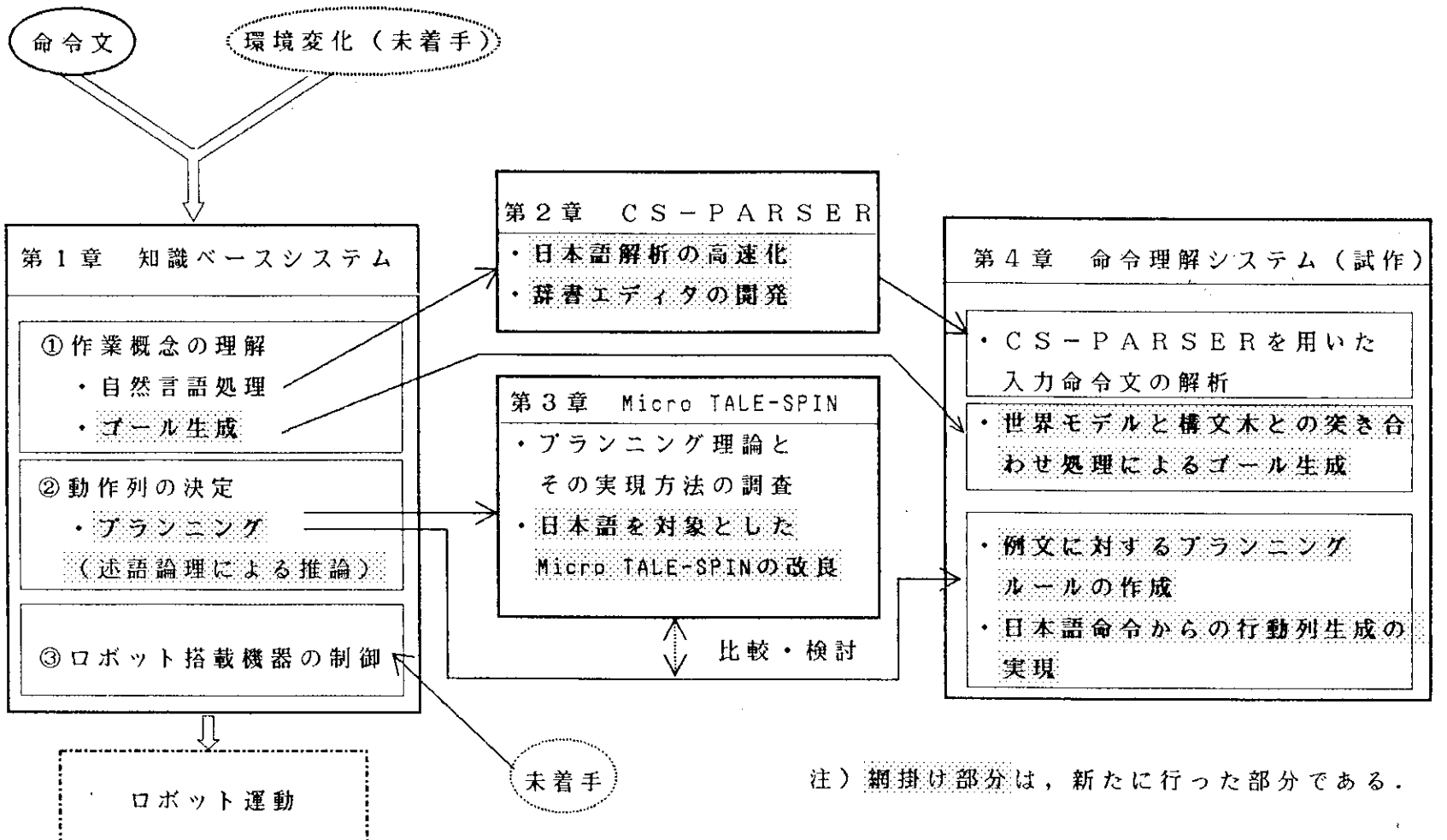


Fig. 1.2　A strategy of the action planning.

JAERI-M 89-218

2. 日本語解析プログラムＣＳ－ＰＡＲＳＥＲの改良

ＨＡＳＰ実現における自然言語処理部分の核として,(株) ＣＳＫより日本語解析プログラム(parser, パーザ：商品名CS-PARSER[1])を導入した。また，パーザの自然言語インタフェースとしての性能を高めるために一部の高速化をはかり，操作性を向上させるために辞書編集ツールを作成した。本章では，ＣＳ－ＰＡＲＳＥＲ の基本機能の紹介と機能拡張の概要，実行例，そして結果の利用方法について述べる。

2.1 ＣＳ－ＰＡＲＳＥＲの機能概要

ＣＳ－ＰＡＲＳＥＲ の基本機能は形態素解析と構文解析に大きく分けられるが，形態素解析では辞書データの検索が行われるので，ＣＳ－ＰＡＲＳＥＲ は形態素解析部・構文解析部・辞書の３部分から構成されていると考えるべきである。ここではそれぞれの部分について説明する。

2.1.1 形態素解析部

形態素とは一般に「意味を有する最小の単位」とされるが，簡単のためにおおよそ単語と考えてさしつかえないものとする。形態素解析部では漢字・かな混じりの入力文字列が個々の形態素に分解され，後述する構文解析部が受け入れ可能な形式の情報にまで変換される。以下，ＣＳ－ＰＡＲＳＥＲ が行う処理を簡単に紹介する[2]。

① 形態素候補の切り出し

入力文字列から形態素候補を切り出すきっかけとしてＣＳ－ＰＡＲＳＥＲ では字種情報を利用している。漢字・平仮名・カタカナなどの字種の変わり目が形態素の区切りと重なることが多い（例えば「私の身長は２ｍです。」→「私／の／身長／は／２／ｍ／です／。」）という性質を利用し，字種情報解析によって効率よく形態素候補を切り出している。

② 語尾活用処理

①で得られた形態素候補と語尾活用テーブル（Table 2.1）を用いて辞書見出し，品詞，活用情報を推定する。

③ 辞書検索

②で推定された見出し，品詞，活用情報から辞書を検索する。辞書検索が成功した場合にはそれが求める形態素であると仮定し，その形態素に関する辞書情報を取り込む。辞書情報については2.1.3項で述べる。

④ 連接判定

③で取り込まれた辞書情報の中に活用連接情報が含まれている。③で定まった（と仮定された）品詞情報と活用連接情報を用い，形態素の前後間での接続の可否を判定する。判定には品詞連接テーブル（Table 2.2）と活用連接テーブル（Table 2.3）が参照され，品詞，活用の両面からチェックされる。

⑤　品詞推定

辞書に未定義の語が見つかった場合に連接情報等をもとに品詞を推定し，その品詞が標準的にもつ形態情報をその未定義語に付加して処理を進める。

以上の処理過程を経て入力文字列は形態素の情報の並びである一次元列として出力される。出力例をFig.2.1に示す。

2.1.2　構文解析部

機械翻訳システムなどにおける構文解析は句構造規則を用いていることが多い。しかし，我々が日常経験しているように，日本語解析には基本文型をテンプレートとして与えておくような方法は適していないように思われる。CS－PARSER では言語学的検討に基づいた中間表現（すなわち構文解析部の出力）[3)4)]を意識した解析方法を検討した結果，語と語の係り受け関係をチェックする方法を採用している。係り受け関係は品詞対品詞の形で構文解析規則としてCS－PARSER が解釈可能な専用言語によって記述されている。CS－PARSERは約1000の構文解析規則をもつ。構文解析規則の中では品詞レベルだけでなくその語の属する概念（意味素）をも考慮した場合わけがなされており，より高品質な日本語解析を行なわせることができる。Fig.2.2の例では「ヒューズの断線を点検する。」という例文について「ヒューズ→の→断線」という係り受け関係が正しいと評価されたのに対して，「ボルトの断線を点検する。」という例文では「ボルト→の」から「断線」への係り受け関係が棄却されたことを示している。

構文解析の結果，形態素解析により得られた語の一次元列は，係り受け関係を表わす１つの木構造にまとめあげられる。その出力例をFig.2.3に示す。

2.1.3　辞書

辞書情報は形態素解析時に検索される。辞書内のデータはCS－PARSER の出力する表現形式と密接に関わっており，見出し，品詞情報，活用情報，活用連接情報などの形態素解析部で述べた情報だけでなく，構文解析部で参照される意味素情報，係り受け情報などの情報も書き込まれている。CS－PARSER が枠組みとして用意している辞書のデータ項目をTable 2.4に示す。

CS－PARSERでいう品詞とは，我々が国語教育で学んだ品詞よりも詳細に分類されている。これは係り受け関係をもとに品詞を定義しているという背景から生じた性質であり，CS－PARSERでは69種類の品詞が定義されている。名詞類の品詞体系の例をTable 2.5に示す。

意味素は日本語解析の対象分野を考慮して定義するべきものと考えるが，我々は名詞概念の階層として約150の意味素を設定した[5)]。その一部をFig.2.4に示す。また，助詞についても約70の意味分類を行なったがここではそれらについては記さない。

格スロットと属性スロットは，品詞間の関係だけでは一意に定めることができないような係り受け関係をより詳細に規定するための情報を提供する[6)]。格スロットは，述語に対してどのような意味をもった体言が，どのような格として，どのような格助詞を介して係ってくるかを規定する（Table 2.6）。また属性スロットは，連体修飾や連用修飾時に係る語がどのような意味で，係られる語のどのような属性値に係ってくるかを規定する（Table 2.7）。

句の係り受け構造を表わす構造情報は，句を構成している語とそれらの係り受け条件より構成されるが，我々の研究では今のところこの情報を利用していない。

2.2 ＣＳ－ＰＡＲＳＥＲに対する機能拡張

2.1節で述べた基本機能をもつCS－PARSER に対してＨＡＳＰ研究への適用をはかるために機能拡張を施した。機能拡張のポイントは次のとおりである。

・出力データの表示に関するインタフェースの作成

・辞書データアクセス部の変更

これらの作業については，CS－PARSER の内部構造についてノウハウをもつ（株）ＣＳＫと協議の上で具体的に内容を決定した[7]。

2.2.1 出力データの表示に関するインタフェースの作成

CS－PARSER が解析結果として出力するデータはFig.2.5に示すように内部処理を経たままのデータである。CS－PARSER は日本語処理の出力からの標準的な出力インタフェースを装備しておらず，図で示した内部表現をもとにユーザの要求する出力形式へ変換するインタフェースのソフトウェアを新たに開発する必要があった。

昭和63年度作業における我々のCS－PARSER に対する出力要求は次の3点であった。一つには出力の木構造を人間が見て分かりやすい形の表示を行なうこと。そして意味素を考慮した表示がなされていること。最後に外部ファイルに出力を行なえることである。そしてこれら3点を基本仕様としてFig.2.5に示した解析結果の木構造を読み取り，係り受け関係を明確に表示するインタフェースを作成した。Fig.2.6中の各ノードに併記されている数字は品詞，意味素，係り方を表わすおのおの意味をもった記号である。なお現在，出力インタフェースはパーザ本体に組み込んだ形になっており，すでに2.1節で紹介したパーザの出力はインタフェースを通したものである。

2.2.2 辞書データアクセス部の変更

CS－PARSER は，基本機能の一環としてデータ構造の提供はしているが，標準辞書をもっていない。動作テストのために（株）ＣＳＫが我々に貸与した辞書はごく小規模であり，また，主記憶上に展開されており，我々がそのままの形でシステムを拡張しようとすれば，辞書データを主記憶上に常駐させなければならなかった。システムの能力向上に伴って辞書データの量（項目数と1項目に対する情報量）が増大するということから考えると，今後の拡張性という点から，どうしても辞書データを外部ファイルとしてもつ必要があった。

そこで昭和63年度作業として，CS－PARSER に外部辞書ファイルを設定し，辞書検索関数を変更した。CS－PARSER の当初の辞書データはＬＩＳＰ環境上にリスト形式で常駐しており，ＬＩＳＰ関数によって辞書検索が行われていた。変更後の辞書データはＵＮＩＸ環境上でＣ言語のＦＩＬＥ構造体として記述されている。そして辞書検索にはC言語が用いられ，その結果をＬＩＳＰのリスト形式に組み上げるようになっており，プログラムはＬＩＳＰとＣ言語が混在した形になっている（プログラミ

ング環境はKyoto Common Lisp)。実測データはないが（株）ＣＳＫの調査では辞書検索関数の変更によって検索速度がかなり向上したとのことである[7]。

また辞書データの変更により，辞書をメンテナンスするための専用エディタ（辞書編集ツール）が必要となった。Fig.2.7に辞書編集ツールの初期画面を，そしてFig.2.8に編集中の画面を示す。ＣＳ－ＰＡＲＳＥＲ の当初の辞書データは vi エディタで編集しなければならなかった上，そこで扱っていたデータがすでに内部データであったため，辞書の内部構造を熟知していないと辞書登録ができなかった。しかし，辞書編集ツールを作成したことにより，メニューにしたがって入力していけばよくなったので，辞書登録の作業効率がひじょうに向上した。その結果，熟練者はそれまでの10倍程度（もしくはそれ以上）のスピードで辞書登録を行えるようになり，修正に要する時間も大幅に短縮された。また初心者が辞書情報を学習する上でもひじょうに有効なツールとなった。

2.2.3　その他の変更

前の２つの項目で述べたほかに，ＣＳ－ＰＡＲＳＥＲ の自然言語処理インタフェースとしての能力を上げるため，解析速度の高速化に寄与する2か所の変更を施した。一つには品詞体系の縮小であり，もう一方は未定義語推定処理の簡略化である[7]。

2.1.3項で述べたようにＣＳ－ＰＡＲＳＥＲ では基本的に69種類の品詞が定義されている。まえにも述べたとおり，構文解析規則は品詞間の係り受け関係として記述されており，単純に品詞数を減らすことによって構文解析規則をも減らすことが可能である。昭和63年度作業では品詞体系を前の約半分の37種類に再構成し，高速化を実現することができた((株）ＣＳＫの調査では約200文に対する解析試験の結果品詞体系の縮小による弊害は認められなかったとのことである。）。また，この品詞体系の変更は，ＬＩＳＰ環境に表形式でその対応関係が記述されており，再度変更することが必要となった場合には，そこを変更するだけでよいのでメンテナンスにも都合がよいということがいえる。

未定義語推定処理は，当初のＣＳ－ＰＡＲＳＥＲ では処理にひじょうに時間がかかっていた上，推定後に解析失敗してしまうこともあった。また，未定義語の品詞というのは多くの場合名詞であるということが経験的に感じられた。そこでＣＳ－ＰＡＲＳＥＲ のインターフェースとしての位置づけを重要視して推定処理を大幅に簡略化し，未定義語が名詞であると限定した。未定義語推定処理の例をFig.2.9に示す。

2.3　ＣＳ－ＰＡＲＳＥＲの実行例と出力結果の利用方法

昭和63年度に導入したＣＳ－ＰＡＲＳＥＲ に対して辞書データを用意し，日本語解析を行なった結果について紹介する。また，平成元年度以降におけるＣＳ－ＰＡＲＳＥＲ の出力の利用方法について述べる。

2.3.1　ＣＳ－ＰＡＲＳＥＲ の実行例

ＣＳ－ＰＡＲＳＥＲ はSun Microsystems 社のSun３（日本語ＵＮＩＸ 4.2 bsd，主記憶容量

８ＭＢ）上のKyoto Common Lisp（ＫＣＬ）配下で実行環境が提供される。ただし，辞書編集ツールに関しては日本語ＵＮＩＸ環境上で利用できる。

昭和63年度作業では炉内作業に関する命令の例文として次の５文を正しく解析できる環境が整備された。

（例文１）計器盤の扉を開けろ。

（例文２）ボルトのゆるみを点検しろ。

（例文３）ゆるみの生じているボルトを増し締めしろ。

（例文４）リレーの設置値を確認しろ。

（例文５）ランプとヒューズの断線を点検しろ。

また例文１～５までを正しく解析するように辞書データを整備した結果，次のような文に対しては（正しく）解析を失敗することも確認された。

（例文６）ボルトの断線を点検しろ。

（例文７）ボルトのゆるみを交換しろ。

例文２・３の解析のようすをFig.2.10に，例文６・７の解析の様子をFig.2.11に示す。

これらの例文が解析できるように辞書に入力された語数は72個で，そのうちパーザの利用対象分野に存在しないと考えられる語（例えば主格の助詞「は」や句続点「。」「，」など）の数は56個にものぼった。また2.1.3項でも述べたとおり名詞概念に対して設定した意味素は約150，助詞に対しては約70である。

2.3.2　平成元年度以降におけるＣＳ－ＰＡＲＳＥＲの利用方法

我々がＣＳ－ＰＡＲＳＥＲに対して昭和63年度に要求した出力の仕様は，2.2.1項で示したように，あくまでもパーザが正しく作動するかどうかを確認するためのものだった。そして，ある程度のパーザの高速化を実現し，辞書データアクセス部を変更し，辞書編集ツールを用意したことによって，パーザの出力をどう加工するか（取り込むか）というところに全力を注ぐところまで準備が完了したものと考えている。

我々は平成元年度作業として，ＣＳ－ＰＡＲＳＥＲによる解析結果（係り受けの木構造）から動詞概念と名詞概念のフレーム[3]を読み取り，ロボットがもつ世界モデルと名詞概念とを対応づけ（referent)て命令文の言及している目標（goal）を作成する出力インタフェースの開発に着手している。出力インタフェースの処理概念をFig.2.12に示す。平成元年度におけるＣＳ－ＰＡＲＳＥＲに関する作業は，おおよそ次のような段階を経ることになるものと思われる。

①　パーザが処理する対象文の決定

②　辞書データの整備

③　対象文とフレーム化されたデータの整理

④　世界モデルの検討と整備

⑤　フレーム化処理アルゴリズムの検討

⑥　フレーム化処理のインプリメント

平成元年度中にパーザに処理させる対象文は，基本的な文を最小でも５文程度は用意する予定である。そしてそれらの対象文にあわせて辞書データ・世界モデル・ロボットが命令文から得

るべき目標を整理する。また，世界モデルはフレーム型の記述とし，命令文を解析したときのトップノード（図の例では「測定する」）にあたる動詞を中心としてまとめあげられた動詞フレームが目標となる。さらに平成元年度後半以降には，この目標を起点としてロボットが自己の行動を計画するシステムへと発展させていく予定である。

2.4 まとめ

昭和63年度に導入したCS－PARSERの機能の紹介と関連する作業の報告，今後の予定について述べた。昭和63年度作業によって，CS－PARSER をHASPのサブシステムとして利用できる環境は整ったと思われる。平成元年度より，「命令を読み取る」という立場から「命令を理解して行動を立案する」ということを目指して作業を進めている。

参考文献

1） CSK日本語解析システムCS－PARSER使用マニュアル，（株）CSK（1988）.
2） 高田潤，他：日英機械翻訳新方式システム，CSK技術通信，Vol.8－2，No.16，pp.9－14（1988）.
3） 高木朗，伊東幸宏：自然言語の処理，丸善（1987）.
4） 高木朗，吉田豊：次期機械翻訳システムにおける中間表現と構文解析の基本的な考え方，CSK技術通信，Vol.8－2，No.16，pp.3－8（1988）.
5） AI用構文解析プログラムの開発と作業指示用知識ベースの構築(II) 構文解析規則・意味素の修正一覧表，（株）CSK（1989）.
6） 野田庸男，他：新日英機械翻訳システムにおける意味と知識の取り扱い，CSK技術通信，Vol.8－2，No.16，pp.15－26（1988）.
7） AI用構文解析プログラムの開発と作業指示用知識ベースの構築(II) 作業報告書，（株）CSK（1989）.

Table 2.1 A conjugation of Japanese verb "akeru".

形態素候補	活用語尾テーブル	辞書見出し	活用情報
「開けろ」	「けろ」←「ける」 動詞 か行下一段 命令	「開ける」	動詞　か行下一段　命令

Table 2.2 Juncture conditions for the speech level.

前形態素	後形態素	連接可否	具体例
名詞	連体助詞	可	「計器盤の」
名詞	格助詞	可	「扉を」
名詞	接続助詞	不可	「扉て」
名詞	終助詞	不可	「扉ぞ」
格助詞	動詞	可	「を開ける」
格助詞	助動詞	不可	「をない」
動詞	助動詞	可	「開けない」
動詞	句点	可	「開ける。」

Table 2.3 Juncture conditions for the conjugation level.

前形態素	後形態素	連接可否	具体例
動詞　か行下一段　未然形	助動詞　ない活用	可	「開けない」
動詞　か行下一段　未然形	助動詞　れる活用	不可	「開けれる」
動詞　か行下一段　未然形	助動詞　られる活用	可	「開けられる」

Table 2.4 Data structure of the dictionary of the CS-PARSER.

形態情報	ＩＤ番号 見出し 活用 活用連接 品詞
意味情報	意味素 格スロット 属性スロット
構造情報	句の係り受け構造

Table 2.5 Categories of noun.

体系	品詞	分類
名詞類	名詞	関係節を作る体言
	文名詞	that節を作る体言
	準文名詞	that節も関係節も作れる体言
	数詞	数字や小数点など
	単位名詞	数詞を受けて体言を作る体言
	⋮	

Table 2.6 Case slots for verb "akeru" and "tenken-suru".

述語	格	連絡語（格助詞）	係る語の意味素
開ける	主格 対象格	は・が・の を	行動者 扉類
点検する	主格 対象格	は・が・の を	行動者 箱形装置，部品，部品状態保持，量

Table 2.7 Property slots for "dan-sen" and "settei-chi".

係られる語	属性	連絡語	係る語の意味素	棄却される例
断線	現象	の	電気回路部品	「ボルトの断線」
設定値	属性	の	計装品	「ボルトの設定値」

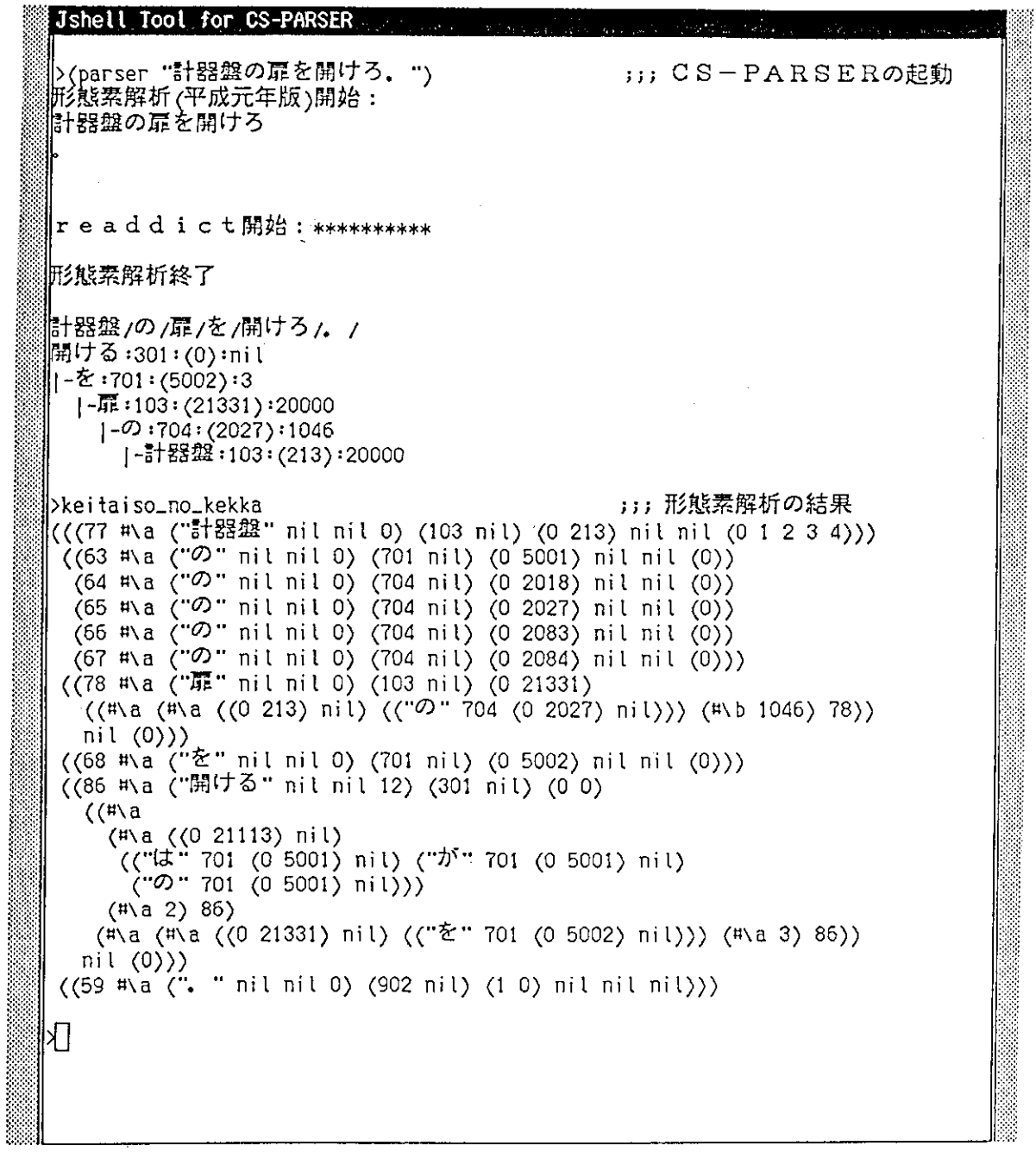


Fig. 2.1 A result of the morphomic analysis.

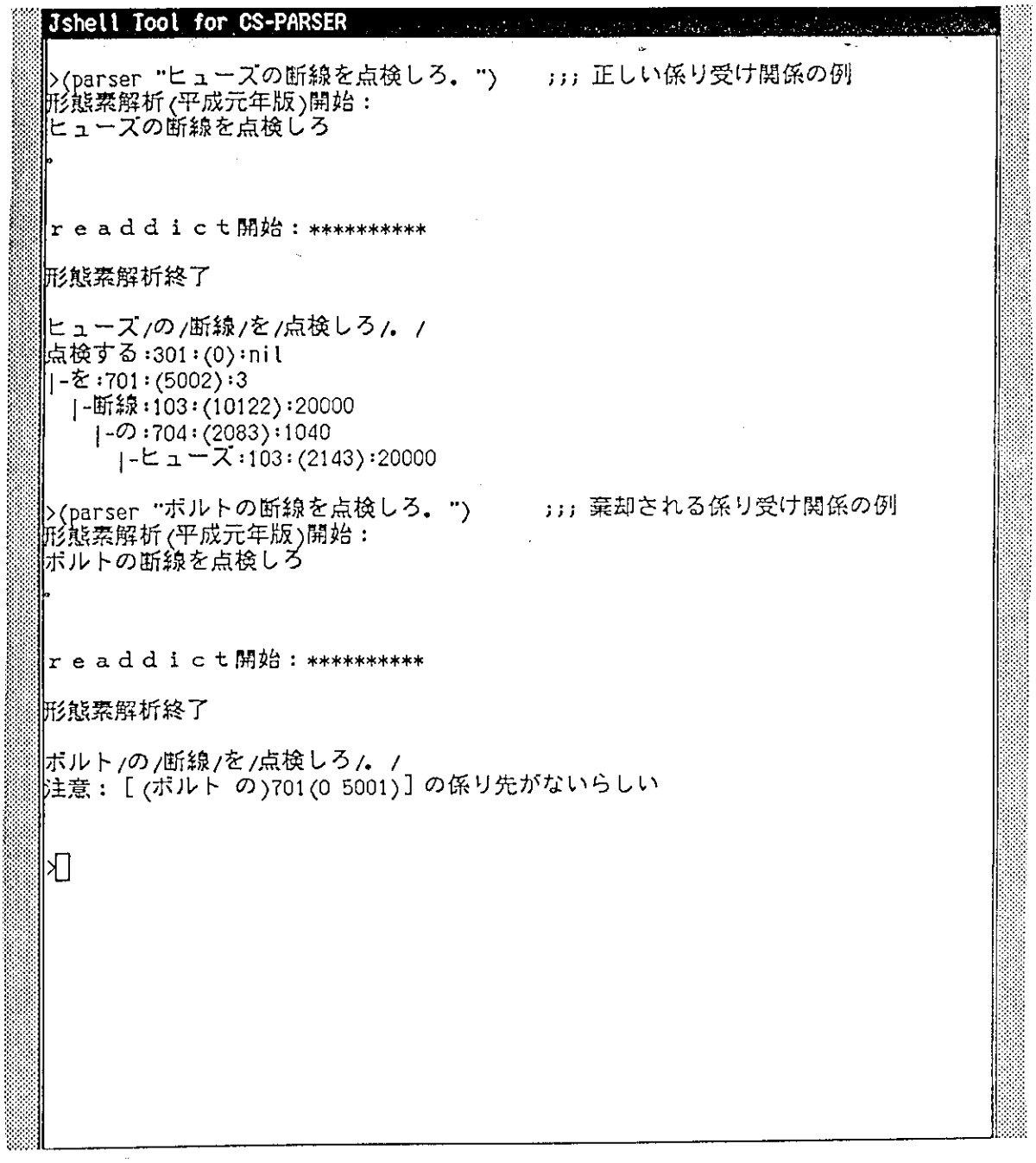
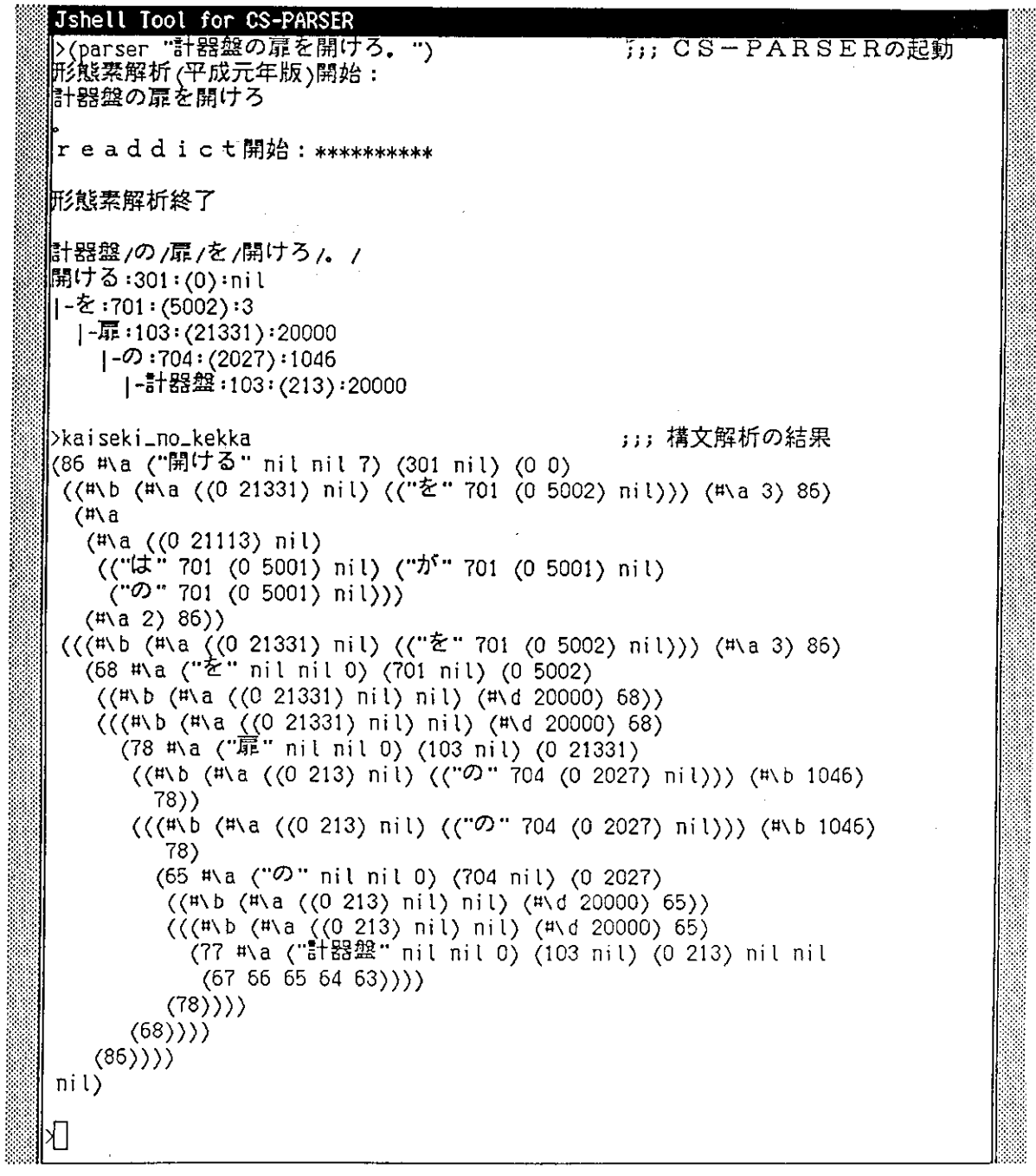


Fig. 2.2 A result of the syntactic analysis by considering semantic primitives.



```
Jshell Tool for CS-PARSER
>(parser "計器盤の扉を開けろ。")                    ;;; ＣＳ－ＰＡＲＳＥＲの起動
形態素解析(平成元年版)開始：
計器盤の扉を開けろ
。
ｒｅａｄｄｉｃｔ開始：**********

形態素解析終了

計器盤/の/扉/を/開けろ/。/
開ける:301:(0):nil
|-を:701:(5002):3
  |-扉:103:(21331):20000
    |-の:704:(2027):1046
      |-計器盤:103:(213):20000

>kaiseki_no_kekka                                  ;;; 構文解析の結果
(86 #\a ("開ける" nil nil 7) (301 nil) (0 0)
 ((#\b (#\a ((0 21331) nil) (("を" 701 (0 5002) nil))) (#\a 3) 86)
  (#\a
   (#\a ((0 21113) nil)
    (("は" 701 (0 5001) nil) ("が" 701 (0 5001) nil)
     ("の" 701 (0 5001) nil)))
   (#\a 2) 86))
 (((#\b (#\a ((0 21331) nil) (("を" 701 (0 5002) nil))) (#\a 3) 86)
   (68 #\a ("を" nil nil 0) (701 nil) (0 5002)
    ((#\b (#\a ((0 21331) nil) nil) (#\d 20000) 68))
    (((#\b (#\a ((0 21331) nil) nil) (#\d 20000) 68)
      (78 #\a ("扉" nil nil 0) (103 nil) (0 21331)
       ((#\b (#\a ((0 213) nil) (("の" 704 (0 2027) nil))) (#\b 1046)
         78))
       (((#\b (#\a ((0 213) nil) (("の" 704 (0 2027) nil))) (#\b 1046)
          78)
         (65 #\a ("の" nil nil 0) (704 nil) (0 2027)
          ((#\b (#\a ((0 213) nil) nil) (#\d 20000) 65))
          (((#\b (#\a ((0 213) nil) nil) (#\d 20000) 65)
            (77 #\a ("計器盤" nil nil 0) (103 nil) (0 213) nil nil
             (67 66 65 64 63))))
          (78))))
       (68))))
    (86))))
 nil)

>
```

Fig. 2.3 A result of the syntactic analysis.

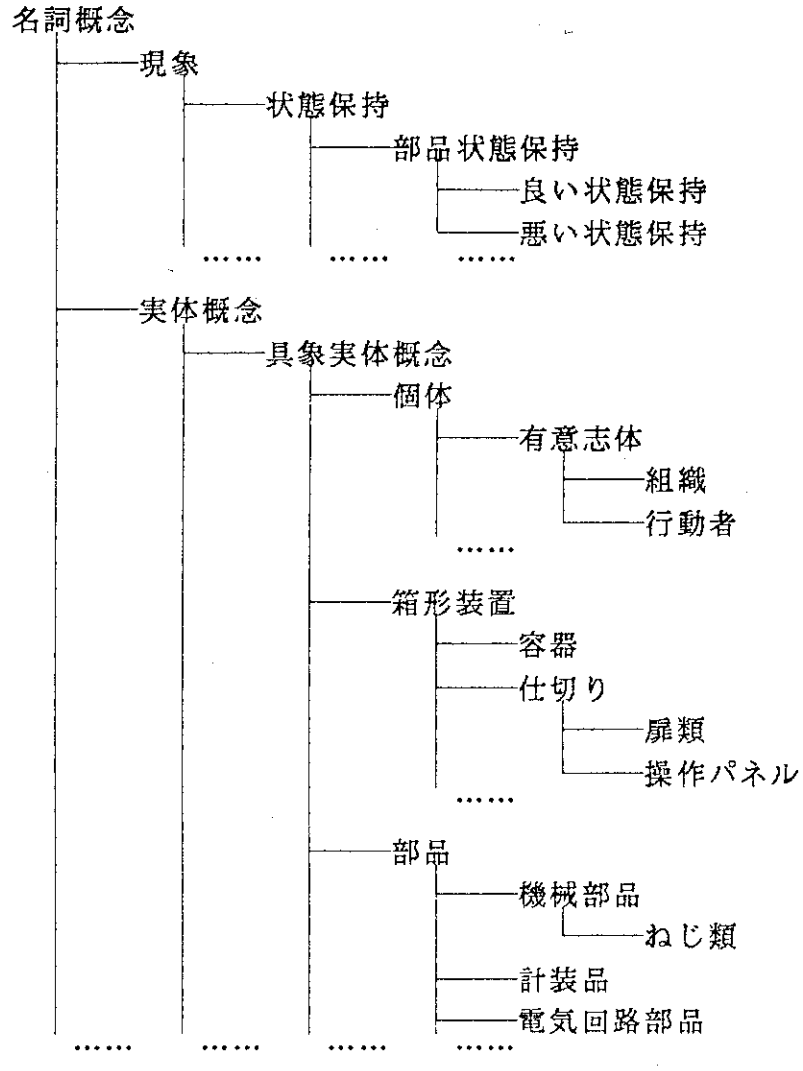


Fig. 2.4 The conceptual hierarchy of noun.

```
Jshell Tool for CS-PARSER



>(parser "計器盤の扉を開けろ。")          ;;; ＣＳ－ＰＡＲＳＥＲの起動
形態素解析(平成元年版)開始：
計器盤の扉を開けろ
。

ｒｅａｄｄｉｃｔ開始：**********

形態素解析終了


>kaiseki_no_kekka                        ;;; ＣＳ－ＰＡＲＳＥＲの出力を表示
(86 #\a ("開ける" nil nil 7) (301 nil) (0 0)
 ((#\b (#\a ((0 21331) nil) (("を" 701 (0 5002) nil))) (#\a 3) 86)
  (#\a
   (#\a ((0 21113) nil)
    (("は" 701 (0 5001) nil) ("が" 701 (0 5001) nil)
     ("の" 701 (0 5001) nil)))
   (#\a 2) 86))
 (((#\b (#\a ((0 21331) nil) (("を" 701 (0 5002) nil))) (#\a 3) 86)
   (68 #\a ("を" nil nil 0) (701 nil) (0 5002)
    ((#\b (#\a ((0 21331) nil) nil) (#\d 20000) 68))
    (((#\b (#\a ((0 21331) nil) nil) (#\d 20000) 68)
      (78 #\a ("扉" nil nil 0) (103 nil) (0 21331)
       ((#\b (#\a ((0 213) nil) (("の" 704 (0 2027) nil))) (#\b 1046)
         78))
       (((#\b (#\a ((0 213) nil) (("の" 704 (0 2027) nil))) (#\b 1046)
          78)
         (65 #\a ("の" nil nil 0) (704 nil) (0 2027)
          ((#\b (#\a ((0 213) nil) nil) (#\d 20000) 65))
          (((#\b (#\a ((0 213) nil) nil) (#\d 20000) 65)
            (77 #\a ("計器盤" nil nil 0) (103 nil) (0 213) nil nil
             (67 66 65 64 63))))
          (78))))
       (68))))
   (86))))
nil)

>
```

Fig. 2.5 An output data of the CS-PARSER.

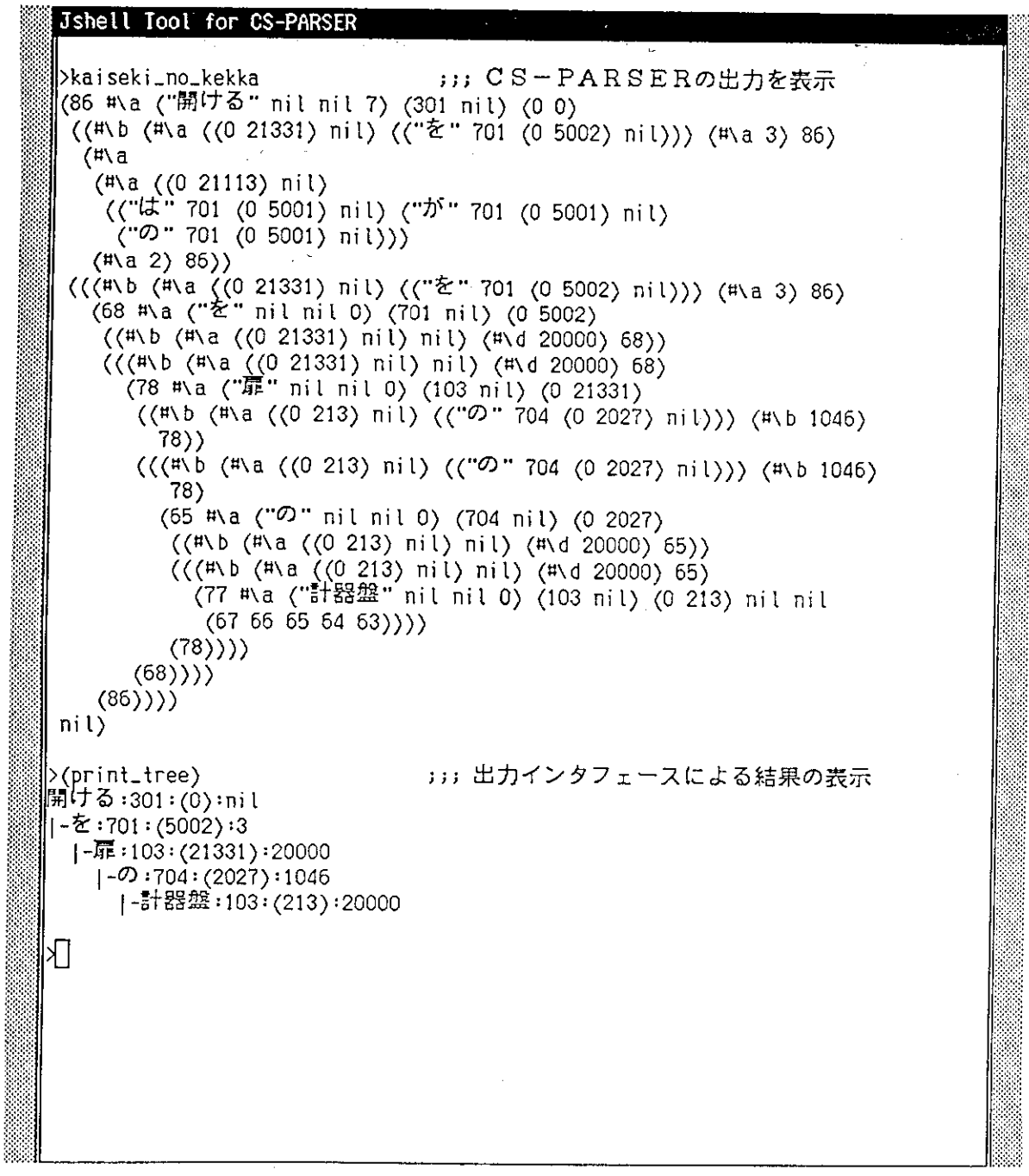


Fig. 2.6 A printing result of the interface function "print tree".

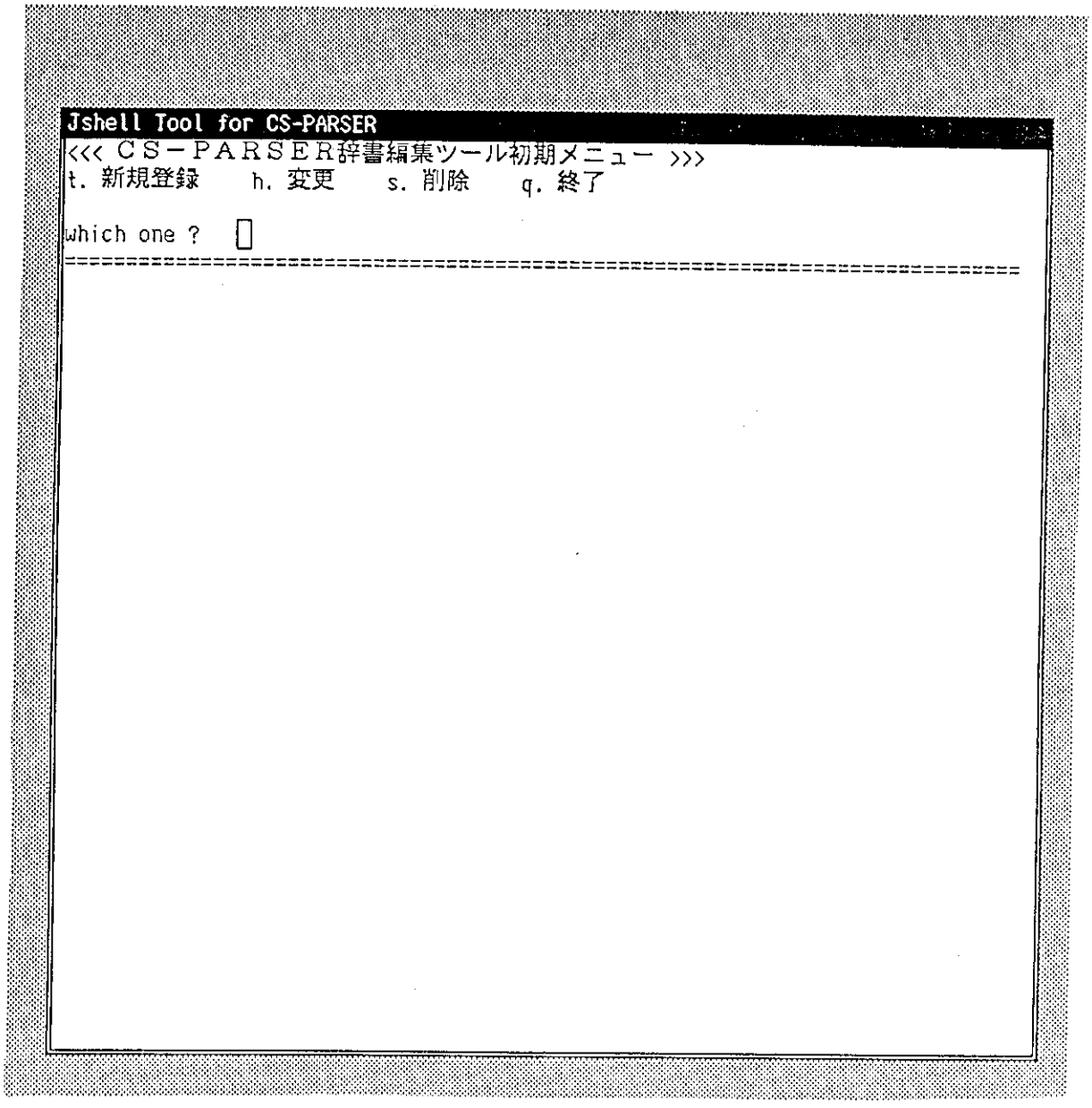


Fig. 2.7 An initial image of the dictionary editor.

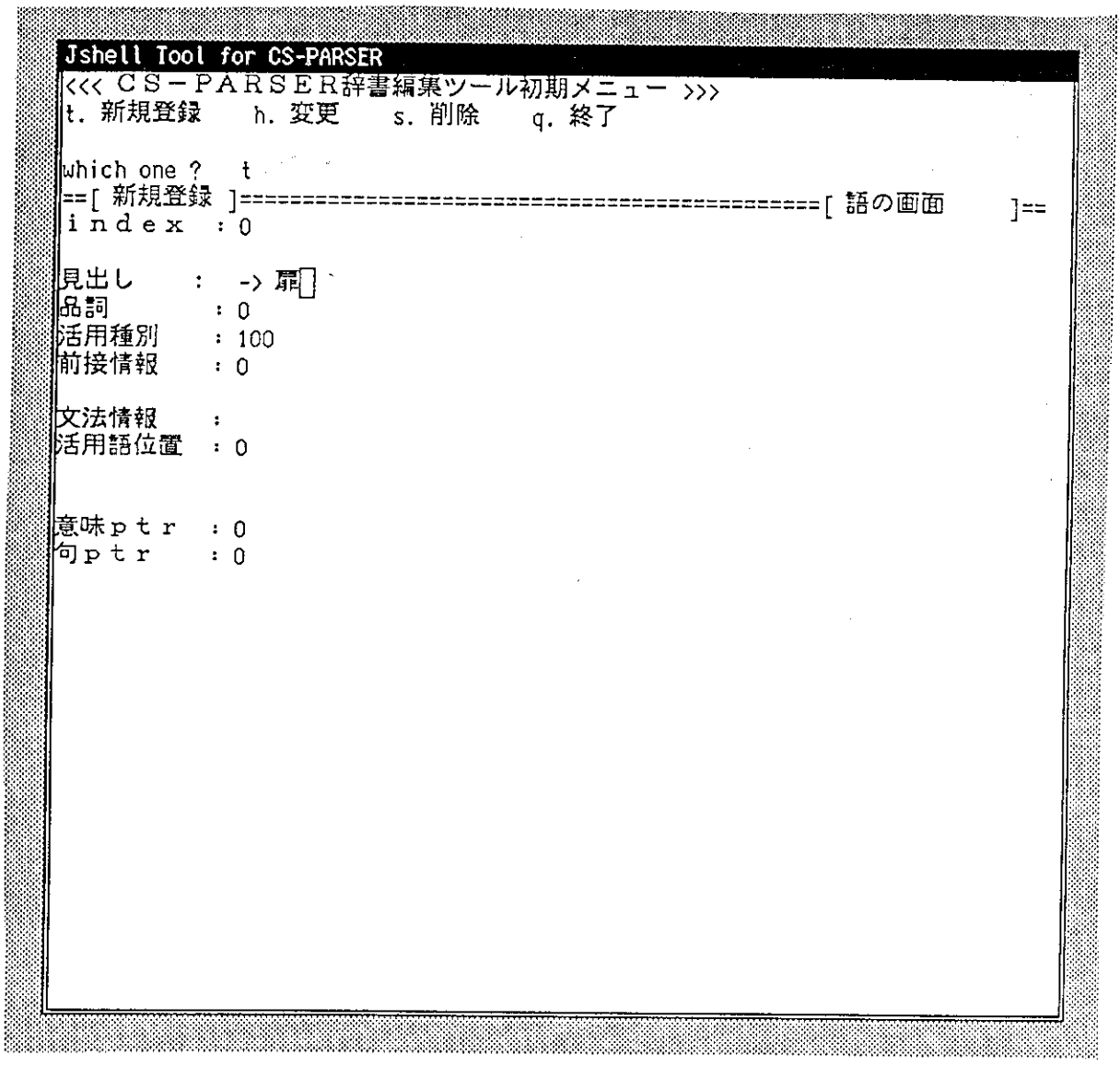


Fig. 2.8 An editing image of the dictionary editor.

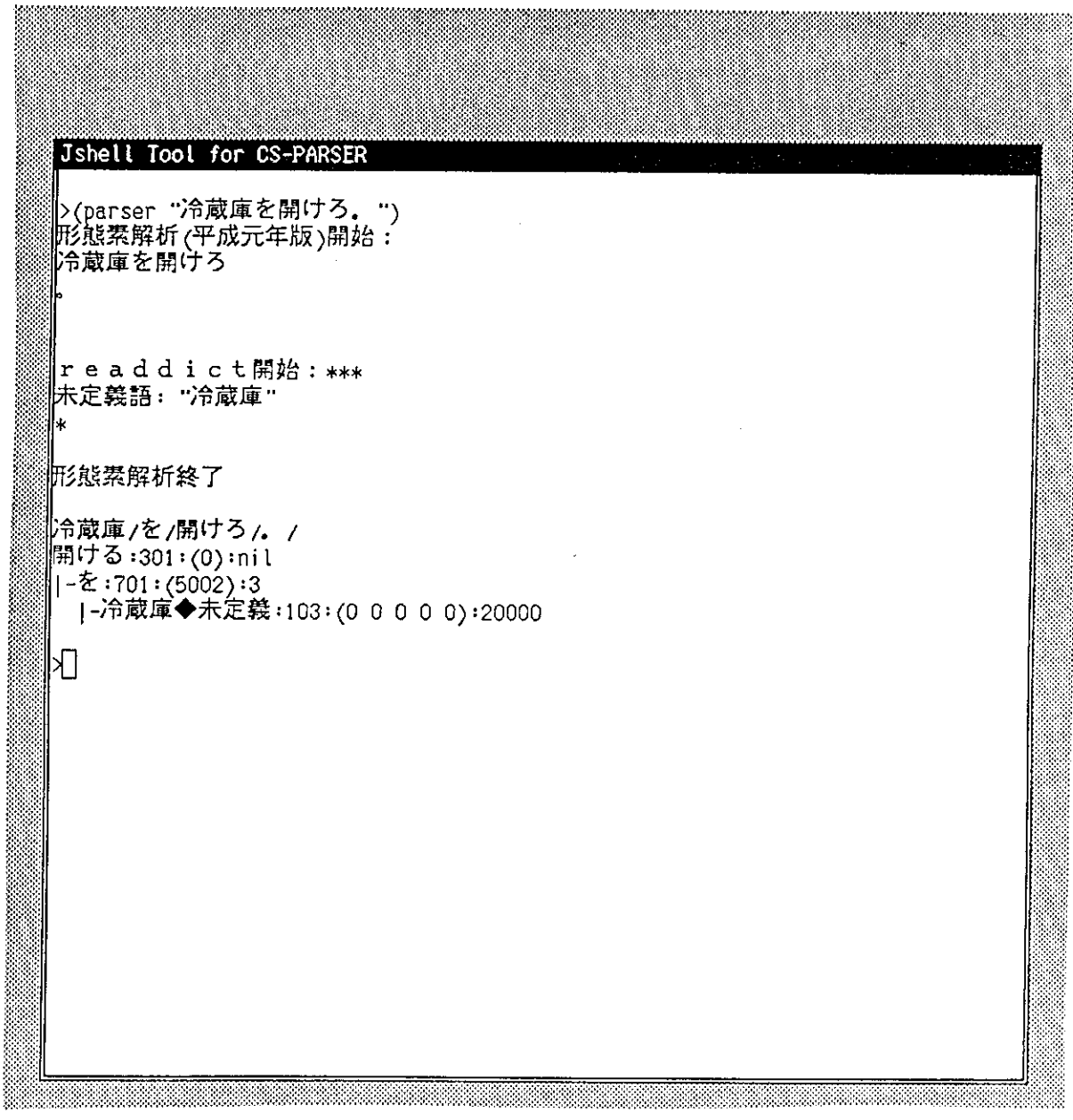


Fig. 2.9 A parsing result of a sentence with an undefined morpheme.

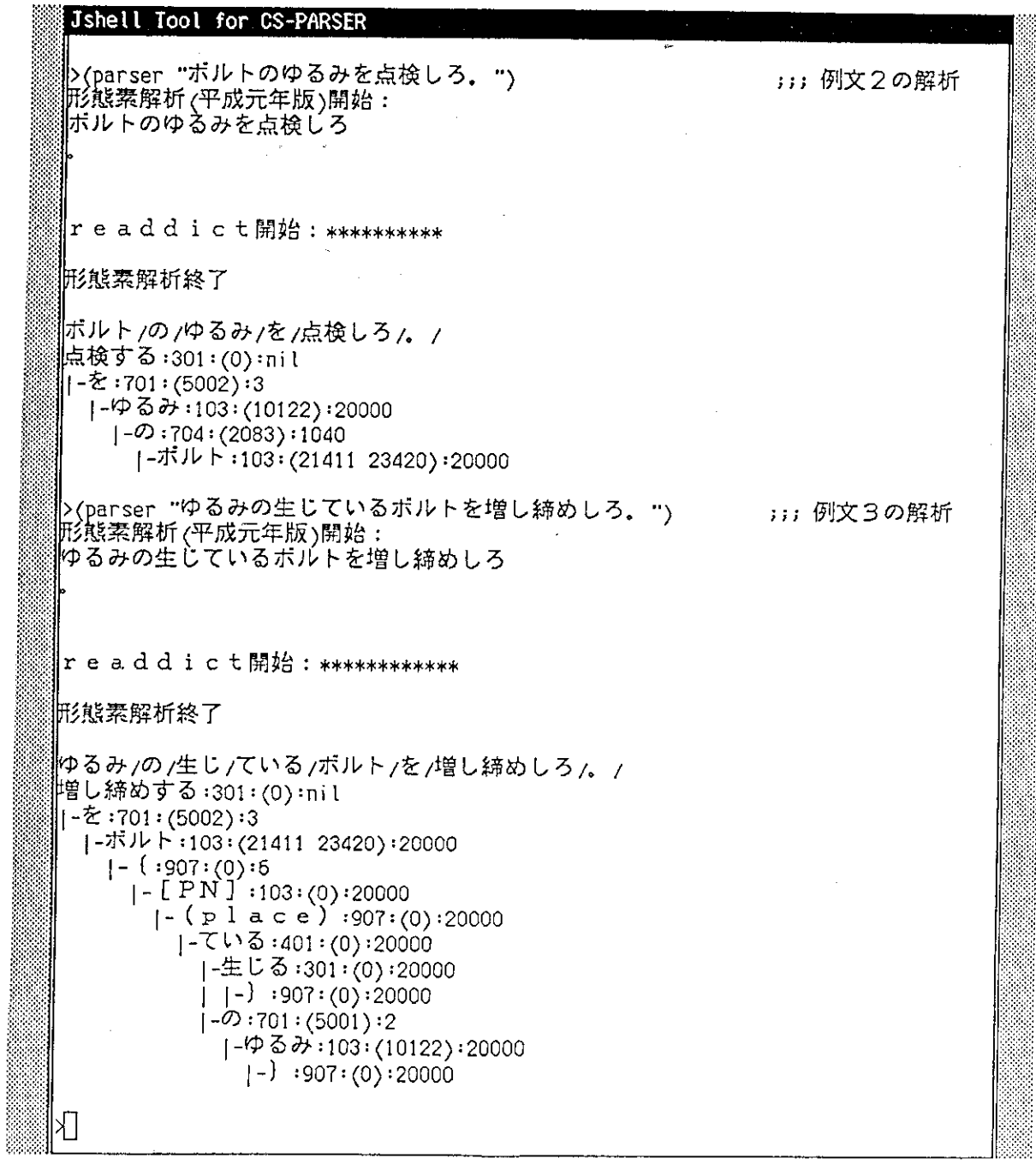


Fig. 2.10 Output data of the CS-PARSER for the examples No. 2 and No. 3.

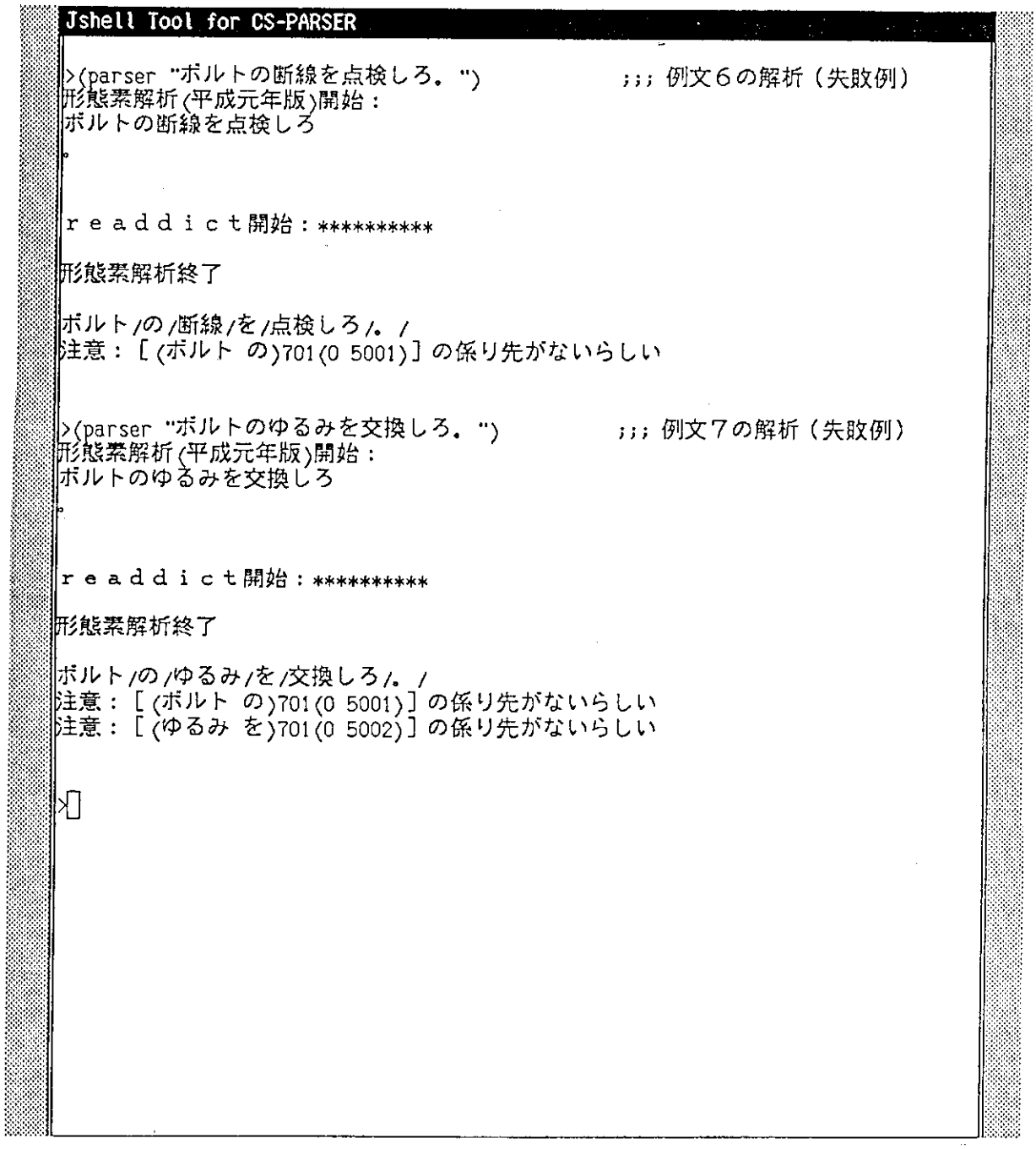


Fig. 2.11 Output data of the CS-PARSER for the examples No. 6 and No. 7.

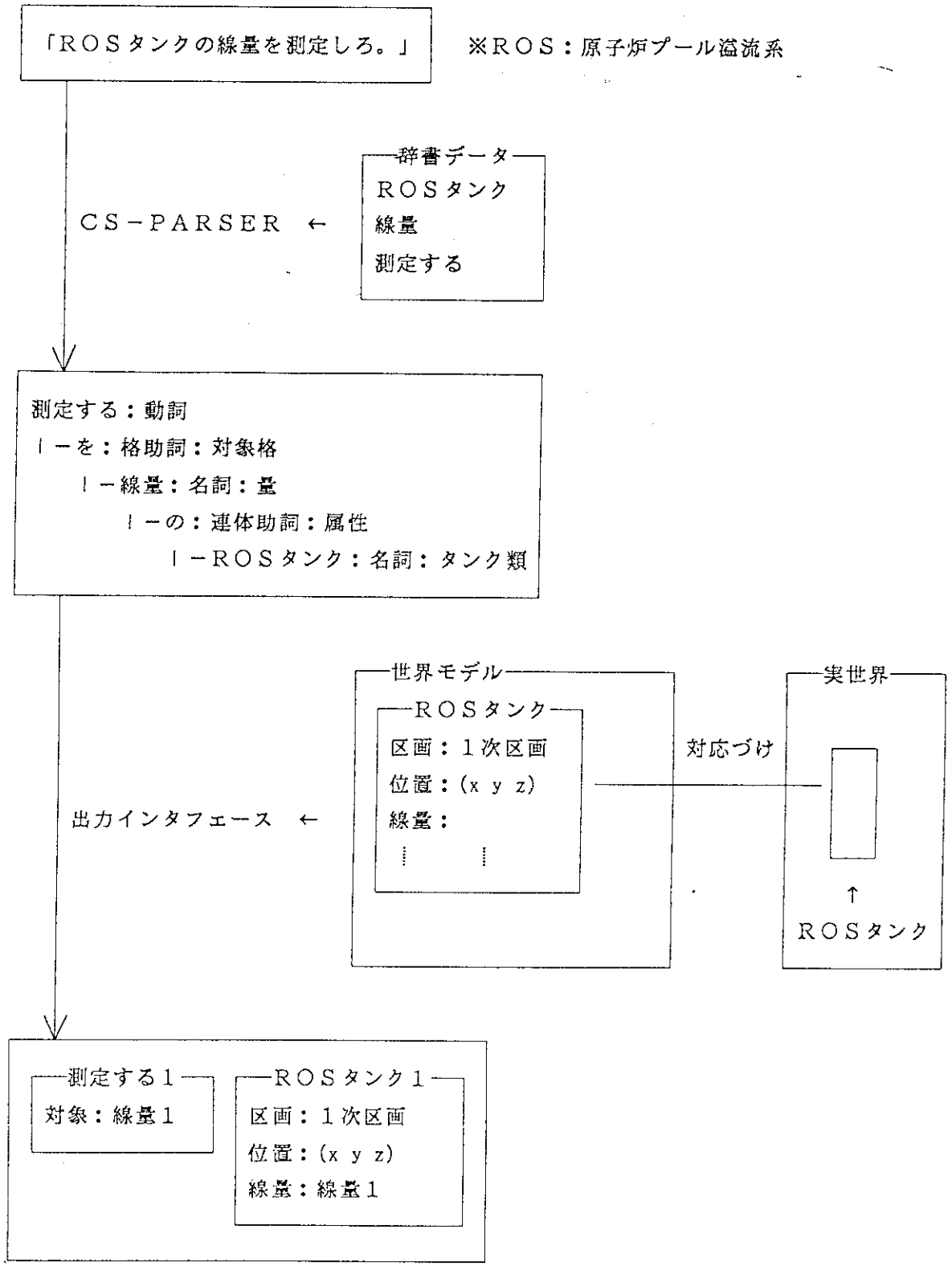


Fig. 2.12 The output interface between the CS-PARSER and the planning system.

3. プランニングを用いた物語生成プログラム

ＨＡＳＰ研究の中で，前章で述べた自然言語処理と密接な関係をもつ重要な研究項目が，行動計画システムの開発である。行動計画システムは，概念表現された命令（問題）を解決するための動作系列を生成し（問題解決），知能ロボットに搭載された機器へのコマンド・変数を転送する（Fig.3.1）。この作動計画の過程の中で，アクチュエータ等の制御部分を除いた問題解決部は，与えられた命令に関する論理的に構成された物語を生成することと等価なことである。本章では，R. C. Schank らのプランニング理論に基づき作成された物語生成プログラムMicro TALE−SPIN を解析し，我々の行動計画システムに必要な要素技術を分析する。

3.1 物語生成プログラムMicro TALE−SPIN

Micro TALE−SPIN は，プランニング理論に基づき論理的に構成された物語を生成する過程を学習するため，J. Meehanによって書かれたプログラムである[1]。我々が，Micro TALE−SPIN を取り上げた理由は２つあるが，その１つはソースプログラムが公開されていることである。Micro TALE−SPIN は，UCI Lisp[2] で記述されており，約1200行の規模である。我々のLispプログラミング環境は，SUN3／260（サンマイクロシステムズ社）上のKyoto Common Lisp（KCL)[3]であるので，プログラム全体をＫＣＬに書き換えた後，解析にあたった。Micro TALE−SPIN を取り上げた２つめの理由は，概念表現に概念依存性理論（Conceptual Dependency Theory：ＣＤ理論）を用いていることである[1]。ＣＤ理論は，R. C. Schank らによって提唱された意味表現手法で，２章で述べた CS−PARSER が用いている高木・伊東の意味表現手法と同様，意味的原素とそれらの結合規則を定義していることに特徴がある。このため，Micro TALE−SPIN を解析した結果を，効率よく我々の行動計画システムに応用することができると考えられる。以下，Micro TALE−SPIN が用いているＣＤ理論とプランニング理論について説明し，プログラムの動作例を用いて Micro TALE−SPIN の働きを解析する。

3.1.1 ＣＤ理論とプランニング理論

Micro TALE−SPIN で用いるＣＤ理論とプランニング理論がどのように物語と関係するのか，簡単な例をあげて説明する。物語の登場人物は，山に住んでいる熊のジョーである。

“この物語の主人公は，山の洞穴に住んでいる熊のジョーです。山にはきれいな水の流れる川があり，ジョーはよく川に水を飲みにいきます。

ある日，洞穴にいたジョーはとても喉が乾きました。そこで，喉の渇きをいやそうと考えました。喉の渇きには，川のきれいな水を飲むのが一番です。ジョーは川に行きました。そして，水をたくさん飲みました。ジョーは，喉が潤い，たいへん満足しました。お話はおわり。”

上の物語に記述してあることは，①状況設定，②主人公に起こった深刻な状況，③深刻な状況

を取り除くための問題設定，④問題を解決する手順，⑤手順を実行した結果，の５つである。①，②，⑤は事象を，③はゴールを，そして，④はプランを表している。これらの要素の記述において，事象の表現をＣＤ理論が受け持ち，ゴール，プランの表現をプランニング理論が行う。実際に上にあげた物語を，ＣＤ理論とプランニング理論を用いて書き下してみよう：

```
        ＣＤ表現                                    ゴール，プラン表現
-------------------------------------------------------------------------
①  (JOE (IS-A BEAR))
   (JOE (HOME CAVE))
   (LOC (ACTOR WATER))
        (VAL RIVER))
   (MLOC (VAL (CP (PART JOE)))
         (CON (LOC (ACTOR WATER)
                   (VAL RIVER))))
   (MLOC (VAL (CP (PART JOE)))
         (CON (LOC (ACTOR JOE)
                   (VAL CAVE))))
②  (MLOC (VAL (CP (PART JOE)))
         (CON (THIRSTY (ACTOR JOE)
                       (MODE POS)))) → ③ (S-THIRST JOE)
     ↑ IR                                    ↓
⑤  (PTRANS (ACTOR JOE)                  ← ④ (D-PROX JOE JOE
           (OBJECT JOE)                         WATER)
           (FROM CAVE)
           (TO RIVER))                      ↓
     ↑ r
   (MLOC (VAL (CP (PART JOE)))
         (CON (LOC (ACTOR JOE)
                   (VAL RIVER))))
     ↑ rE
   (INGEST (ACTOR JOE)                  ←    (DOIT (INGEST
           (OBJECT WATER))                       JOE WATER))
     ↑ r
   (MLOC (VAL (CP (PART JOE)))
         (CON (THIRSTY (ACTOR JOE)
                       (MODE NEG))))
```

上のＣＤ表現とゴール，プラン表現の意味は，Table 3.1 にまとめておいた[1]。

ＣＤ理論は，ＣＳ−ＰＡＲＳＥＲ が基礎としている高木・伊東の意味表現手法と同様に，動作概念，状態概念，そして実体概念の間の依存性（係り受け）を使って事象の表現を行う。また，事象と事象との間の関係（因果関係）を，↑ｒ（結果），↑Ｅ（可能），↑Ｉ（開始），↑Ｒ（理由）といったアークを用いて表現する。さらに，動作概念をTable 3.1 に示した少数の基本的行為（primitive actions）を用いて表わす。このため，表層の統語状態に縛られない事象表現を，さらに文脈の記述を可能にしている。

一方，プランニング理論（ゴール，プランの理論）は，＂登場人物が何故事象を起こしたのか＂という意図（知識レベル）の表現を行う。R. C. Schank らのゴールとプランに関する理論では，Table 3.1 にあげたゴールを含めて，次の７つのゴールを設定している[4]。

A−Goal	Achievement	長期にわたる達成目標
S−Goal	Satisfaction	空腹の解消，休養など
E−Goal	Enjoyment	オペラをみる，美術館に行くなど
P−Goal	Preservation	健康管理など
C−Goal	Crisis	家の修繕，救急車を呼ぶなど
I−Goal	Instrumental	準備をするなど
D−Goal	Delta	移動する，ものを所有する，何かを知るなど

これらのゴールは，登場人物のおかれた状態と対応している。例えば，＂ジョーは喉が乾いている＂に対応して，（Ｓ−ＴＨＩＲＳＴ　ＪＯＥ）なるゴールが発生する。登場人物がゴールを解決するには，ゴールのネットワークを逐次評価していくこと，他人にゴールを解決してくれるように頼むこと，というプランを実行する。Micro ＴＡＬＥ−ＳＰＩＮ に現れるゴールＳ−ＨＵＮＧＥＲ(空腹解消)，Ｓ−ＴＨＩＲＳＴ(喉の渇き解消）を例にとって，プランがどの様に構成されているか説明する。まず，主要なゴールはＳ−ＨＵＮＧＥＲ，Ｓ−ＴＨＩＲＳＴ の二つで，これらに対するサブゴールはＤ−ＰＲＯＸ（場所移動），Ｄ−ＣＯＮＴ（ものの所有），Ｄ−ＫＮＯＷ（情報の獲得）の三つである。これらのゴールを解決するために必要となるゴールのネットワークは以下のように表現される[1]。

Ｓ−ＨＵＮＧＥＲ → {Ｄ−ＣＯＮＴ（食べ物），ＤＯ（食べる）}
Ｓ−ＴＨＩＲＳＴ → {Ｄ−ＰＲＯＸ（飲物），ＤＯ（飲む）}
Ｄ−ＣＯＮＴ → {Ｄ−ＫＮＯＷ（where is ？），Ｄ−ＰＲＯＸ，ＤＯ（自分のものにする）} or {ＰＥＲＳＵＡＤＥ（持ち主）}
Ｄ−ＫＮＯＷ → {自分が知っているか調べる} or {ＰＥＲＳＵＡＤＥ（友達）}
Ｄ−ＰＲＯＸ → {Ｄ−ＫＮＯＷ（where is ？），ＤＯ（移動）} or {ＰＥＲＳＵＡＤＥ（移動対象）}

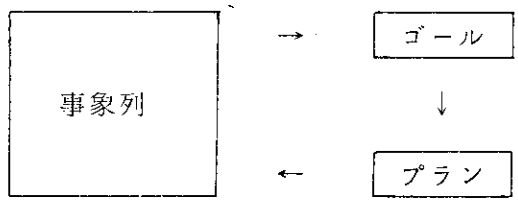
ＰＥＲＳＵＡＤＥ サブゴールは，登場人物が自分自身でゴールを達成できないときに，他人にそのゴールを達成するように依頼するものである。ＰＥＲＳＵＡＤＥ を達成するためのプランは，単純に依頼する（ＡＳＫ−ＰＬＡＮ），取り引きをする（ＢＡＲＧＡＩＮ−ＰＬＡＮ），脅す（ＴＨＲＥＡＴ−ＰＬＡＮ）といったものがある。ＰＥＲＳＵＡＤＥ サブゴールには，他人を説得するために

その人のいる場所に行って頼みたいことを伝達するというサブゴール（ＴＥＬＬ）が必要となる。

ＴＥＬＬ　→｛D－PROX（誰かのいる場所），DO（情報伝達）｝

上に述べたゴールのネットワークとプランには，DO（?）と行った記述が含まれている。この部分が動作を示す事象に対応している。

以上，Micro ＴＡＬＥ－ＳＰＩＮが用いているＣＤ理論とプランニング理論について説明した。我々が，これらの理論に学ぶべきことは，事象と知識の対応関係である。



登場人物のおかれた状況（事象）はゴールに対応し，ゴールを解決するためのプランが生成され，新たな状況が生まれる。これはまさに，｛環境認識→反応｝という人間の認知過程を示している。人間の認知過程は，常に活性化していることを考えると，我々の目指す命令理解システムにおいては，｛事象→ゴール→プラン→事象→・・・｝という巡回的な制御構造を実現することが重要である。

3.1.2　Micro ＴＡＬＥ－ＳＰＩＮの構成

Micro ＴＡＬＥ－ＳＰＩＮは，前節で述べたＣＤ理論とプランニング理論を，Lisp 言語を用いてプログラムの形で実現したものである。Micro ＴＡＬＥ－ＳＰＩＮは，次の３つの部分から構成されている（Fig.3.2）。

① 事象から因果関係を用いて新たな事象を生成する部分：
（事象主張関数，因果連鎖データベース）

② ゴールとそれに対するプランを評価する部分：
（ゴール評価関数，ゴール・プランデータベース）

③ ＣＤ表現された事象を英文に変換する部分：
（英文生成プログラム，辞書）

そして，これらの構成要素が参照・書換えするデータベースとして，世界モデル，登場人物の記憶がある。Micro ＴＡＬＥ－ＳＰＩＮを実行すると，まず，プログラマーが用意した世界モデルをロードし，次に物語の主人公とそれが達成すべきゴールの選択を要求する。この段階でゴールに対応する事象（たとえば＂ジョーは喉が渇いていると感じた＂というもの）が事象主張関数に引き渡される。これ以後は，事象主張関数とゴール評価関数が自動的に，物語に対応する事象列を生成し，英文生成関数によって物語が生成される。生成された事象は，事実データベースに追加され，もし登場人物が感知可能な事象であればその登場人物の記憶に追加する。

ここで実際にMicro ＴＡＬＥ－ＳＰＩＮが，3.1.1項で述べた簡単な物語を生成する過程を説

明する。（Fig.3.3）。主人公ジョーの初期データとして，

（LOC（ACTOR 水）（VAL 川））
（LOC（ACTOR ジョー）（VAL 洞穴））
（THIRSTY（ACTOR ジョー）（MODE POS））　①

を用意する。Micro TALE−SPIN 制御プログラムは，初期データが主人公の記憶に存在するという事象を次々に事象主張関数に入力する。①で示した事実が事象主張関数に入力された時点から，主人公ジョーの“のどの渇き解決”が始まる。

（MLOC（VAL（CP（PART ジョー）））
（CON（THIRSTY（ACTOR ジョー）（MODE POS））））

↓

［事象主張関数］

↓

THIRSTYという状態はゴールに対応するので，THIRSTYを解消するためのプログラムS−THIRSTY をゴール評価関数に引き渡す

↓

［ゴール評価関数］

この時点で，3.1.1項のゴールネットワークにしたがって事象列が生成される。（ここで，最終的ゴールは（THIRSTY（ACTOR ジョー）(MODE NEG))である。）

S−THIPSTY → {D−PROX（水），DO（飲む）}
D−PROX→D−KNOW（where is 水）；ジョーの記憶に存在するので真
DOIT（PTRANS）→（PTRANS（ACTOR ジョー）；事象主張関数を
（OBJECT ジョー）呼ぶ
（FROM 洞穴）
（TO 川））
DOIT（INGEST）→（INGEST（ACTOR ジョー）；事象主張関数を呼ぶ
（OBJECT 水））

そして，“水を飲んだ”という事象が生成された段階で，ジョーは喉が渇いていないということを事象主張関数が生成する。

（MLOC（VAL（CP（PART ジョー）））
（CON（THIRSTY（ACTOR ジョー）（VAL NEG））））

ここで，主人公ジョーのゴールが満足されたので，物語の生成が終了する。

上の例では，PERSUADE サブゴールを評価する部分は現れなかった。Micro TALE−SPIN は，登場人物間の友達関係（LIKE），強弱関係（DOMINATE），そして信頼関係（DECEIVE）を用いて，PRESUADE サブゴールの評価を実現している。PRESUADE サブゴールを実行するように初期設定を変更すると，登場人物間の駆引きを含んだかなり長い物語を生成することができる（Fig.3.4）。つまり，Micro TALE−SPIN の制御構造は普遍でも，登場人物のおかれた状況を変化させることで，さまざまなプランを生成することが可能である。そこで，我々の目指す行動計画システムは，知能ロボットが行う作業に関する

プランニング知識を十分用意しておき，Miro TALE−SPINと同様の制御構造を用いることで実現可能と考えられる。

3.2 Micro TALE−SPIN の改良

Micro TALE−SPIN は，もともと英語で記述された物語を生成するために作られている。しかし，3.1.2項で述べたとおり，Micro TALE−SPIN の構成部分の中で言語表現と関係しているのは，英文生成部のみである。それならば，英文生成部を日本文生成プログラムに置き換えるだけで，日本語で書かれた物語を作り出すことが可能と考えられる。これが実現できれば，Micro TALE−SPIN で用いているプランニング理論が，英語や日本語といった自然言語体系の違いに依存しないということを示すことにつながると考えられる。本節ではMicro TALE−SPIN のために我々が作成した日本文生成プログラムについて述べる。

Micro TALE−SPIN の英語生成プログラム（MUMBLE）では，CD表現に含まれている基本的行為それぞれに英文生成用のサブプログラムを定義している[1]。サブプログラムは，CD表現中に含まれる実体概念と基本的行為との依存性（ACTOR，OBJECT，etc.）を単語の並び方に置き換えることによって，英文を生成する。ここで，“ジョーは洞穴から川に行った”という事象を使って，英文が生成される過程を説明する。この事象のCD表現は，

```
(PTRANS (ACTOR JOE)
        (OBJECT JOE)
        (FROM CAVE)
        (TO RIVER))
```

である。まず，基本的行為PTRANSをキーワードとして，PTRANSで記述された事象を変換するサブプログラムが呼び出される[1]。

```
(COND ((EQUAL $(ACTOR) $(OBJECT))
       (SAY-SUBJ-VERB '(ACTOR) 'GO))
      (T (SAY-SUBJ-VERB '(ACTOR) 'MOVE)
         (SAY-FILLER '(OBJECT))))
(SAY-PREP 'TO '(TO))
```

このサブプログラムは，CD表現のACTOR，OBJECTスロットが同一のものかどうかによって，動詞GO，MOVEを使い分ける。そして関数SAY−SUBJ−VERB にACTORスロットを埋めている“JOE”と動詞“GO”を入力し，“JOE GOES”を印刷する。次に，どこへ行ったのかを示すTOスロットの値を関数SAY−PREP によって“TO RIVER”と印刷し，英文を生成する。

日本文を生成する場合，CD表現の依存性を表現する役割を担うのは，英文のような単語の順序ではなく，格助詞に相当する語，すなわち，「が」，「は」，「を」，「から」，「へ，に」である。そこで，基本的行為PTRANSから日本文を生成するためのサブプログラムは，以下のように記述できる。

```
(COND ((EQUAL $(ACTOR) $(OBJECT))
       (SAY-SUBJ '(ACTOR))
       (SAY-PROP 'に '(TO))
       (SAY-VERB 'GO))
      (T (SAY-SUBJ '(ACTOR))
         (SAY-PROP 'を '(OBJECT))
         (SAY-PROP 'に '(TO))
         (SAY-VERB 'MOVE)))
```

このサブプログラムを使って上に述べたＣＤ表現を変換すると，まず，ＳＡＹ−ＳＵＢＪによって「ジョーは」が印刷され，次にＳＡＹ−ＰＲＯＰによって「川に」が続き，さらに，ＳＡＹ−ＶＥＲＢが「行きます」を生成して，最終的に日本文「ジョーは川に行きます」を得る。他の基本的行為についても同様のサブプログラムを作ることができる。これらのサブプログラムを用いて，上に述べた簡単な文だけでなく，日本語としては少々不自然と感じられるが従属節を持つ文まで生成することができた（Fig.3.5）。

3.3　考察

本章では，論理的に構成された物語を生成するプログラムを説明してきた。Micro ＴＡＬＥ−ＳＰＩＮ は，本来，ゴールとプランの理論を学習するために作成されたものなので，生成できる物語（事象列）はきわめて簡単なものに限定される。さらにゴール・プランのネットワークを直接プログラムの形で記述してあるので，プランニングルールの変更の柔軟性に欠けている。また，一度制御プログラムを起動させてしまうと，ユーザが割り込みをかけることができないことも欠点である。しかし，ここで使われているゴールとプランの理論そしてＣＤ理論は，かなりの部分をＨＡＳＰで構築する動作計画プログラムに応用できると考えられる。そこで，今後動作計画部分を構築する際，どの様な修正を加えればよいのかを述べたい。

まず，ＣＤ理論はＣＳ−ＰＡＲＳＥＲ で用いている高木・伊東の意味表現手法に吸収できる。しかし，我々の行動計画システムでは最終的にロボット動作を生成することが目標となるので，ロボットの基礎的な動作を洗いだし，それを基本的行為として設定することが必要となる。その段階でロボットが扱うゴールの階層を作成し，それを解決するためのプランをルールベースの形で構築する。また，処理プログラムの制御構造は，いかなる場合でもユーザが割り込めるようにするために，日本語解析部，ゴール・プランネットワーク検索部，事象主張部をモジュールとしたループ構造をとる必要がある。

また，Micro ＴＡＬＥ−ＳＰＩＮ では場所の移動に際し，全く経路探索を行っていない。つまり，ある場所にあることを知っているという情報だけで，そこに移動するという事象を直ちに生成してしまう。ＨＡＳＰが対象とする世界では，移動に際してどの経路が最適なものであるかを評価しなければならない。そのため，経路を検索するという手段的ゴール（Ｉ−Goal）を設定して，それに対するプラン（実行関数）を作成する必要がある。また，ほとんどの作業で，使用する実体（装置）に依存した動作が必要と考えられるため，それに対応したＩ−Goalを作

らなければならないと考えられる[4]。

参考文献

1 R. C. Schank and C. K. Riesbeck, 石崎 監訳：自然言語処理入門, 総研出版(1986).

2 J. Meehan, " The new UCI LISP manual ", Lawrence Erlbaum Associates, N. J. (1979).

3 T. Yuasa and M. Hagiya, " Kyoto Common Lisp Report ", TEIKOKU INSATSU (1984).

4 R. C. Schankand R. P. Abelson, " Scripts, Plans, Goals and Understanding ", Lawrence Erlbaum Associates, N. J. (1977).

Table 3.1 A summary of (a) primitive actions, (b) states, and (c) goals for the Micro TALE-SPIN.

(a) 基本的行為の記述方法	意味
(PTRANS ACTOR OBJECT FROM TO)	ACTORがOBJECTをFROMからTOへ動かす
(ATRANS ACTOR OBJECT FROM TO)	ACTORがOBJECTの所有権をFROMからTOへ移す
(MTRANS ACTOR OBJECT FROM TO)	ACTORがOBJECTなる情報をFROMからTOへ伝える
(MBUILD ACTOR OBJECT)	ACTORがOBJECTなる情報を心に抱く
(GRASP ACTOR OBJECT)	ACTORがOBJECTをつかむ
(INGEST ACTOR OBJECT)	ACTORがOBJECTを体内に取り込む
(PROPEL ACTOR OBJECT TO)	ACTORがOBJECTを使ってTOに力を加える
(PLAN ACTOR OBJECT)	ACTORがOBJECTということを計画する
(CAUSE ANTE CONCEQ)	ANTEなる前提がCONCEQなる結果を導く
(WANTS ACTOR OBJECT)	ACTORがOBJECTなる状態を望む

(b) 状態の記述方法	意味
(CONT ACTOR VAL)	ACTORはVALに所有されている
(LOC ACTOR VAL)	ACTORはVALの近傍に存在する
(MLOC CON VAL)	CONなる情報がVALに記憶されている
(LIKE ACTOR TO MODE)	ACTORはTOをMODEで示す程度に好きである
(DOMINATE ACTOR TO MODE)	ACTORはTOよりMODEで示す程度，力が強い
(DECEIVE ACTOR TO MODE)	ACTORはTOをMODEで示す程度，だます
(HEALTH ACTOR MODE)	ACTORはMODEで示す程度，健康である
(SMART ACTOR MODE)	ACTORはMODEで示す程度，頭がよい
(HUNGRY ACTOR MODE)	ACTORはMODEで示す程度，空腹である
(THIRSTY ACTOR MODE)	ACTORはMODEで示す程度，喉が乾いている

(c) ゴール	意味
S-HUNGER	空腹を解消するゴール
S-THIRST	喉の渇きを解消するゴール
D-PROX	場所の移動を示すゴール
D-CONT	ものの所有を示すゴール
D-KNOW	情報の獲得を示すゴール

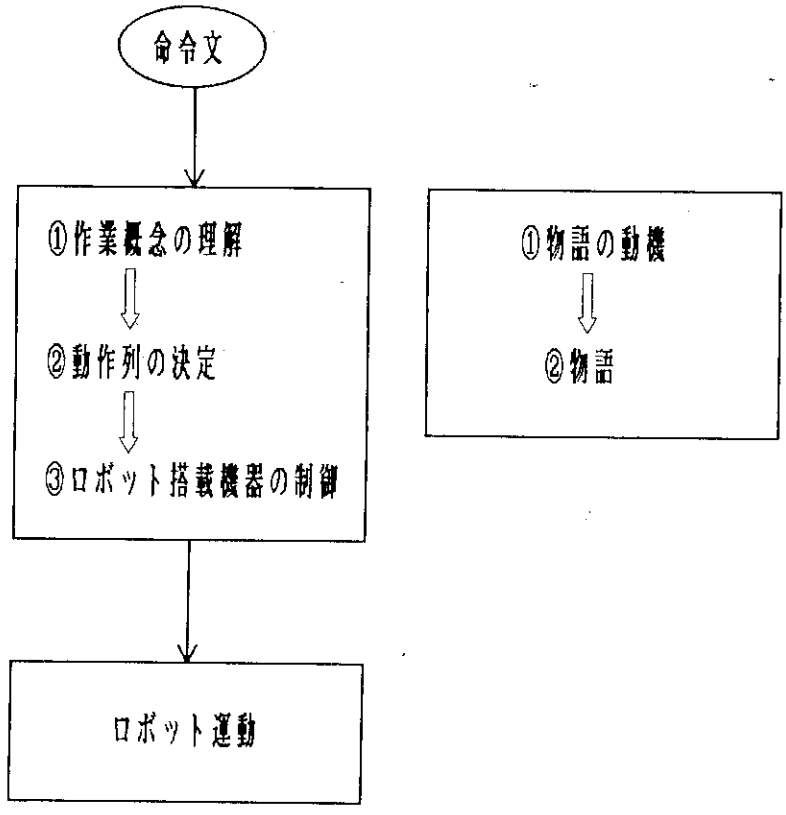


Fig. 3.1 Components of the action planning system.

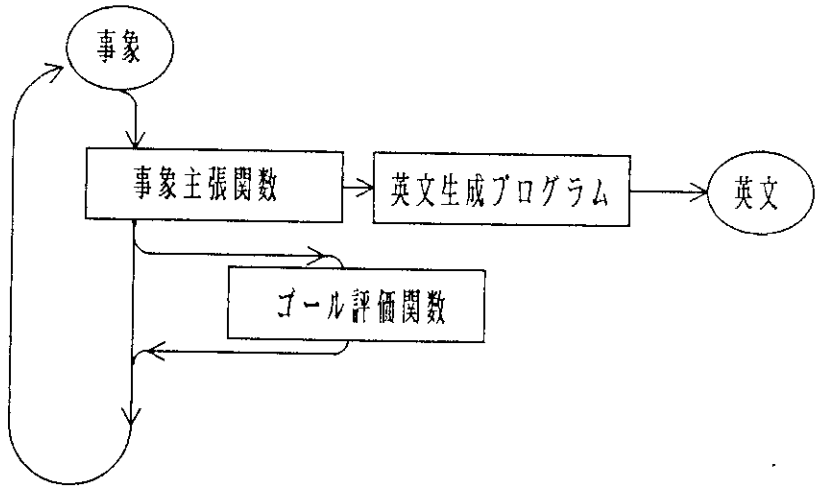


Fig. 3.2 System structure of the Micro TALE-SPIN.

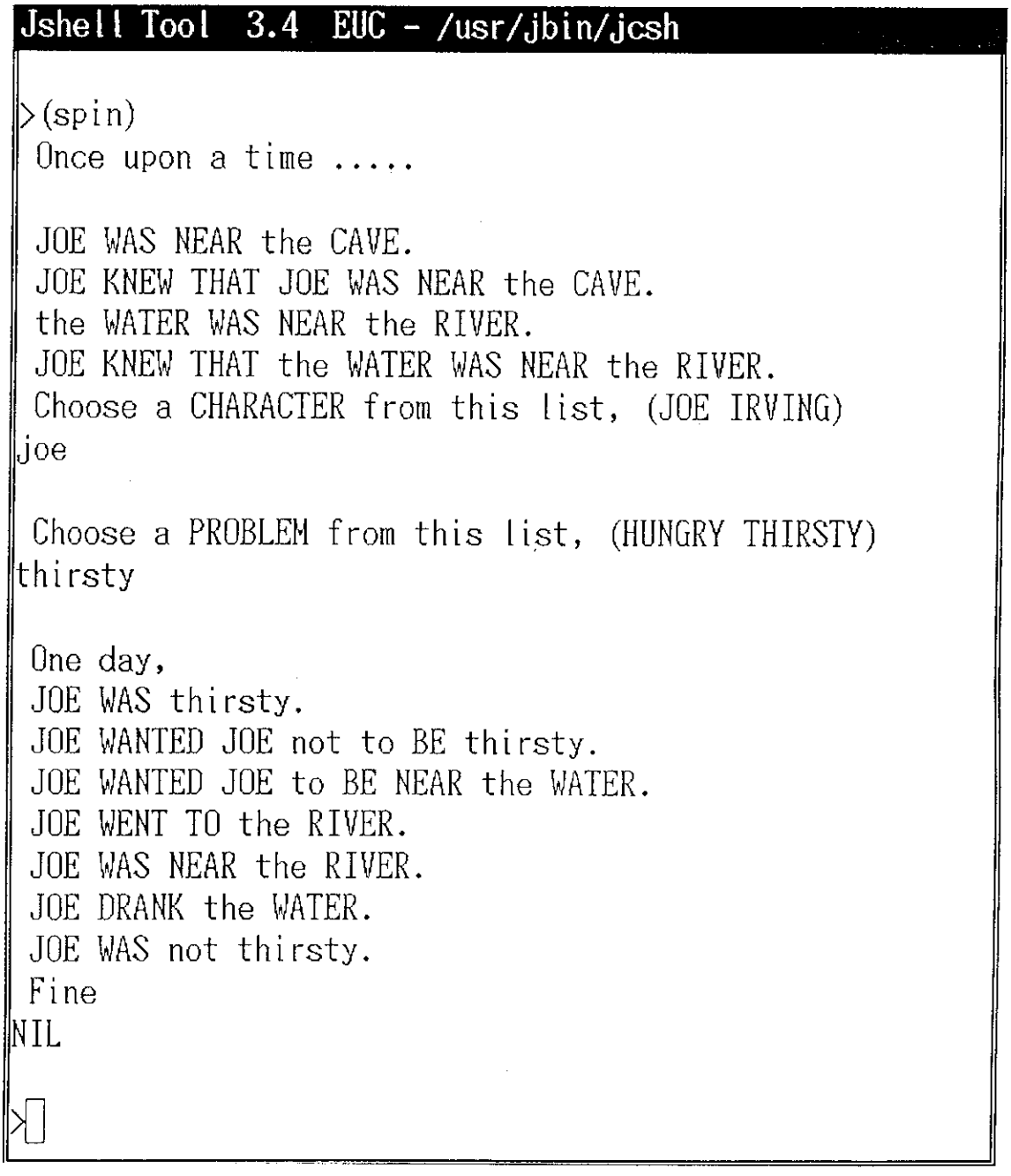


Fig. 3.3 An output of the Micro TALE-SPIN.

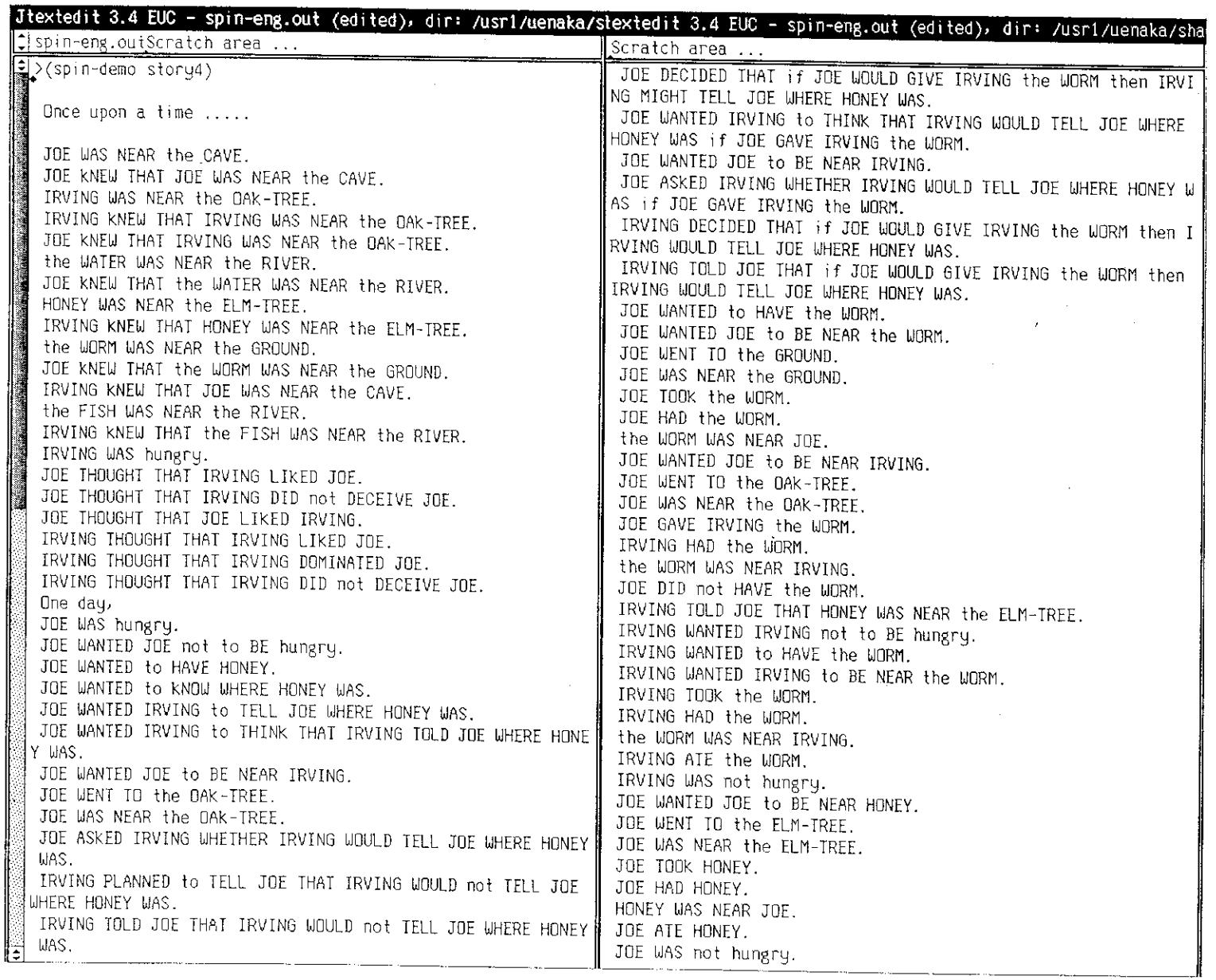


Fig. 3.4 A long story generated by the Micro TALE-SPIN.

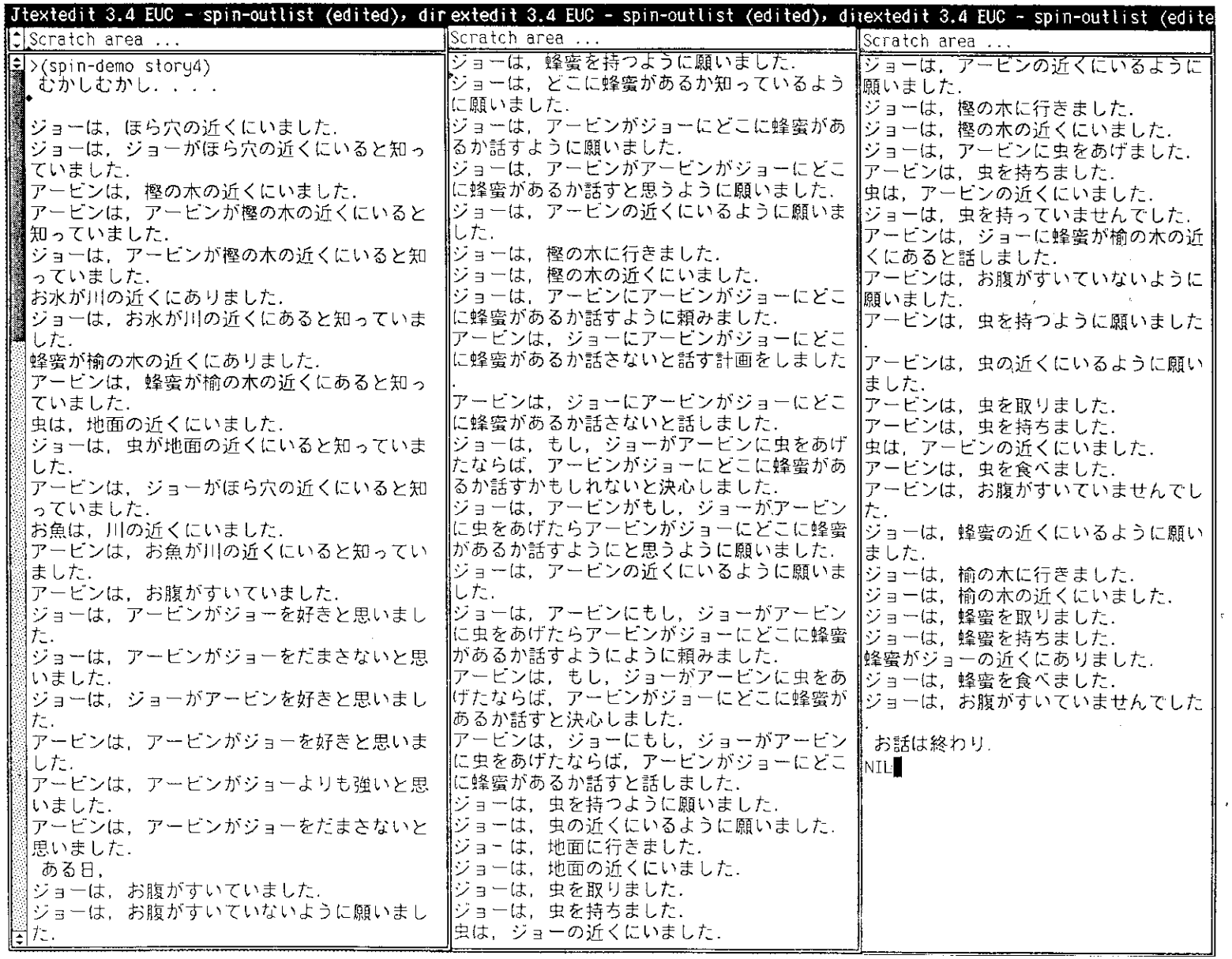


Fig. 3.5 A Japanese story generated by the Micro TALE-SPIN.

4. 命令理解知識ベースシステムの試作

本章では，２章で紹介した日本語解析プログラムＣＳ－ＰＡＲＳＥＲを用いて試作した知識ベースシステムについて述べる。システムが対象とする問題は，ソフトウェア上に仮想された知能ロボットが人間の命令を理解していく過程を実現することである。そして作業を進めるにあたっては，その理解過程で参照される知識のあり方を考えることを最重点項目とした。システムの試作には２段階の過程をとった。まず第一に，日常的になじみやすい簡単なロボットの世界における知識のあり方を考えた。そして次の段階として，ＨＡＳＰが対象とする世界の一部を基本モデルとしたテストケースに適用し，先に考えた知識のあり方を再検討した。以下ではロボットと言語の結びつき，試作システムの概要，システムの評価と今後の展望についてそれぞれ述べる。

4.1 ロボットと言語の結びつき

ロボットを制御するということは，結局のところロボットに対して関節角やトルクの変位を時間の関数として与えてやることである。しかし，人間とロボットが構成する系において，常に人間がロボットの関節角などを詳細に指示ができるかといえばそれには無理がある。そこで人間とロボットの中間に規則（文法）として位置づけられているのがロボット言語であり，人間がロボット言語の体系に従うことによってロボットを制御することができるわけである。ロボット言語は，その特徴によっていくつかのタイプに分類されるが，ここでは井上による５つの分類[1]を紹介する。

①作業目標レベル：作業目標だけを与える言語。処理系はその目標を達成するための詳細な動作手順等を自動生成する。

②対象物状態レベル：作業を対象物の間の状態変化として記述する言語。動作が起こる時点での対象物間の拘束条件などは必要であるが，手先の動きを陽に記述する必要がない。

③構造物動作レベル：ロボットの動作記述と対象物の動作記述を扱うことができるＰＡＳＣＡＬ風の言語。

④原始的動作レベル：ロボットの手先の動きに注目したＢＡＳＩＣ風の言語。

⑤コマンドレベル：ロボットを動かすためのもっとも原始的な命令で，アセンブラのイメージをもつ言語。

ロボット言語の具体例を上の分類に照らしあわせると，②ではＩＢＭワトソン研究所のＡＵＴＯＰＡＳＳ，③ではスタンフォード大学のＡＬ，④ではユニメーション社のＶＡＬ，そして⑤ではスタンフォード大学のＷＡＶＥなどがあげられるが，①にあたる言語はまだ開発されていない[2]。

産業用ロボットの世界では上で述べたようなロボット言語が存在し，人間とロボットのコミ

ュニケーションをとっている。しかし一方では人工知能の分野からのアプローチも存在する。その例としては，積木の世界を対象としたSHRDLU[3]（自然言語処理の応用システム）やSTRIPS[4]（プロダクションルールを用いた問題解決システム）などがある。特にSHRDLUは，自然言語形式の応答を行なって動作を起こすという点では上で述べたロボット言語の5つのレベルを超越したマン・ロボット・システム実現の可能性を提示しているといえるだろう。

ＨＡＳＰにおける命令理解研究では，自然言語形式の入力から，ロボット言語の分類でいう作業目標レベル（①）の記述を抽出することを目標としている。そして行動計画研究においては，命令理解研究の結果を引き継いで作業目標レベルの記述を構造物動作レベル（③）ないし原始的動作レベル（④）まで自動的に展開することを目標としている。それゆえ命令理解研究と行動計画研究のリンクによって，ロボットは自然言語形式の入力を理解して行動を起こすことができるわけである。これはSHRDLUにひじょうに近い立場のシステムとなるだろう。さらに付け加えるならば，ＨＡＳＰシステムは厳密な意味ではSHRDLUのように自然言語による応答はできない（日本語生成を柔軟に行なうような部分を用意していない）が，一方では積木の世界にとどまらずに炉内作業という世界を対象にしていることや，英語でも形式的に限定された日本語でもなく一般的な日本語表現の入力を許すといった特徴がある。このような点から，ＨＡＳＰの命令理解－行動計画システムはSHRDLUと違った意味で，先のロボット言語の5つのレベルを越えるマン・システムになりうるものと考える。

4.2　試作システムの概要

昭和63年度に作成した試作システムに関して，対象とした問題と処理の概要，システム化，そしてＨＡＳＰの題材を考慮した発展問題への適用についてそれぞれ述べる。

4.2.1　対象問題(1)

ＨＡＳＰが対象とする炉内作業の世界を考える前に，ごく日常的な対象モデルを設定して，システムに要求される処理機能と用いられるデータを考えることにした。Fig.4.1にロボットが対処すべき世界を示す。ロボットの世界においてロボットが取り扱う対象物はテレビ・ビデオデッキ・ビデオテープである。テレビ・ビデオデッキに対するロボットの操作は基本的にはボタンを押すだけに限り，その他にはテープを出し入れする操作を加えた。また，ロボットは片腕のみをもっており，人間でいう肩の部分がテレビ等を操作できる範囲の任意な位置に存在しているものとし，テレビ等の操作に関しては死角（操作範囲内であるにもかかわらず，手先をその位置に移動できない状態）が存在しないとした。ロボットの実行する素動作（動作の最小単位で，原始的動作レベルのロボット言語に匹敵する）としては，移動（move），掌握（grasp），解放（release），加圧（push），捜索（search）の5種類に限定した。実行動作の内容を示す。

① 移動…手先位置を移動する。

② 掌握…対象物をつかむ。

③　解放…掌握していた物をはなす。
④　加圧…対象物をわずかに押す。
⑤　捜索…状態データより位置座標を検索する。

本システムが想定しているロボットは，あくまでもソフトウェア上に構築されるのみで，ハードウェアの実現については全く考慮していない。それゆえ，②の「掌握」などは本来であれば「移動」や「位置ぎめ」などを組み合わせて表現するべきであるが，本システムでは「位置ぎめ」は自動的に行なわれるものとして①の「移動」と区別できる行為とした。また，⑤の「捜索」については，ロボットの視覚との協調ができればより良いのだが，本システムでは，状態（世界）データから値を検索することで動作を簡略化した。

一方，今回の試作システムでは，人間側が行う入力命令についても条件をつけた。入力命令は「テレビをつけろ。」，「ビデオを再生しろ。」など，文の中で必ず直接的な命令を下すようにし，状態を述べる文や疑問文などは当初含めないことにした。また，今回のシステム構築では動作列の自動生成部分の動作確認（すなわち理解されたかどうかということ）を主に作業を進めるため，まず入力文を「ビデオを再生しろ。」のみに限定し，データ生成にかかることにした。

4.2.2　処理概要

システムの処理概要を入力文「ビデオを再生しろ。」を例にとって説明する（Fig.4.2）。Fig.4.2の中では入力文が「ビデオを再生する。」という平叙文になっているが，ＣＳ－ＰＡＲＳＥＲ(構文解析）の出力としてはどちらも同じになるので本システムでは特に区別してしない（もちろん形態素解析の情報を利用して区別するように修正することも可能である。）。

まず，システムに入力文を与えると，システムは辞書データを基にして述語を根とした木構造の表現に変換する（Fig.4.2の(a)から(b)の部分）。木の枝部分にあたるところは連絡語（例えば助詞「を」）であり，枝がそのまま述語の格構造[5]を表現している。また，変換の際には必要に応じて，表層では述べられていない語の意味へ置き換える（例では「ビデオ」が「ビデオテープ」に置換されている）。さらに本システムにおいては，最終的に具体的行動を出力されなければ，システムが“理解している”と評価することができないので，名詞を具体物に特定化する処理も必要である。本システムでは格要素として抽出されたものが実際に世界（正確には世界と写像関係にある状態データ）に存在しているかどうかを判断し，「ビデオテープ」などの名詞概念が抽出された場合には，ロボットが既に知っているものの中から「ビデオテープ」という概念に属する具体物の中の最新情報を取り出して結びつける（Fig.4.3）。（ここまでがＨＡＳＰでいう命令理解部にあたる。これから先は行動計画の仕組みである。）

次に，前の処理によってトップレベルで得られた述語をキーワードとして述語－動作概念ＫＢを検索する。ＫＢの中には述語の格構造木を動作概念木に変換するためのルールが格納されている。システムは述語の格構造木のトップレベルにあたる述語をキーワードとして，知識としての述語の格構造木と，先に命令から導出された述語の格構造木とのマッチングを行なう。そして，そのときにマッチした値（すなわち命令から導出された格要素）を局所的な変数として束縛し，変換のための必要条件が検証される。必要条件の検証には状態データが参照され，

条件不成立の場合には，その条件を達成するようなサブゴールが新たに与えられる。Fig.4.2の中では示していないが，例えば「ビデオデッキの中にテープがセットされている。」という条件が成立していないような場合には，「ビデオデッキにテープをセットする。」というサブゴールが発生することになる（4.2.3項で示すシステムの処理例では，実際にサブゴールが発生している）。そして，条件が達成されると動作概念の木構造を生成する（Fig.4.2(b)から(c)の部分）。ただし，本システムにおける動作概念とは素動作よりもかなり大雑把な概念であり，対象モデルの設定上，前節で述べたように「押す」，「セットする」，「とり出す」など数種に限られる。以上のように得られた動作概念の木構造はさらに次の素動作生成へと利用される。

素動作生成は，動作概念ＫＢおよび名詞概念ＫＢを用いて行なわれる。システムは動作概念ＫＢから，動作概念・格の設定が先に得られた動作概念木と一致する知識を検索し，その知識に従って動作（素動作＋状態データの書き換え）を発生させるための手続きをとる。ここで動作概念は，動作を発生させるための方法は知っているが，直接的に動作を発生するわけではない。動作に関するデータは，動作主（ロボット）と対象物（ビデオデッキ）がそれぞれもっており，システムはスーパーバイザのように双者に対して動作の発生を促すメッセージを送信するだけでよい（Fig.4.4）。この方法は，例えば，素動作とそれに伴う動作主の状態データの変更手続きを動作主側に，そして素動作に伴なう（対象物側の）状態の変化を対象物側にもたせオブジェクトへのメッセージ送信で１つの事象を描くような場合に有効と思われる。

メッセージ送信の結果，ロボットのとる素動作列がシステムより出力され，同時に状態データの更新も行われる（Fig.4.2の(c)から(d)の部分）。

4.2.3　システム化

システム構成をFig.4.5に示す。このうち，述語の格構造木変換部はほとんど全てＣＳ－ＰＡＲＳＥＲ の機能であり，動作概念木変換部の要求仕様に基づいて構文解析の木構造を読み取る機能を付加した。各データベース（Data Base：ＤＢ）・知識ベース（ＫＢ）は最終的には外部記憶となることも考えているが，今回のシステムはごく小規模なプロトタイプモデルなので，全て主記憶上に展開させることにした。また，4.2.2項で示した概要では位置データを状態データに含めていたが，実際のインプリメントにおいては，メンテナンスの際の煩雑さ避けるため，ならびに検索の簡単化のために状態データから切り離した形にすることにした。システム化にあたって，ＣＳ－ＰＡＲＳＥＲ を除いた部分で新規に作成されたプログラムとデータの規模はTable 4.1のとおりである。

システムの開発環境はハードウェアがＳＵＮ３／260（Sun Microsystems社製）で，主記憶８Ｍバイトである。また，ソフトウェア環境としてはＯＳが日本語ＵＮＩＸ 4.2ＢＳＤで，使用言語はKyoto Common Lisp（通称ＫＣＬ）である。ＣＳ－ＰＡＲＳＥＲ はＫＣＬ環境下から呼び出しが可能である。

使用言語としてＬＩＳＰを採用していることと関連して，本システムのデータベースも全てリスト表現をとっている。Fig.4.6には動作概念ＫＢの例を，そしてFig.4.7には名詞概念ＫＢの例を示す。また，処理例をFig.4.8に示す。

4.2.4　対象問題(2)への適用

ＨＡＳＰが対象とする炉内作業の世界を考慮して，新たにロボットの世界を設定した。その世界をFig.4.9に示す。新しいロボットの世界では，ロボットは部屋の中を移動する設定になっている。ロボットが取り扱う物体はとりあえず計器盤のみとし，「計器盤の扉を開けろ。」もしくは「運転表示盤の扉を開けろ。」という命令に対して動作させることを目標とした。世界が変わったことにともない，対象問題(1)で考えた素動作の中の「移動」について３つの種類に分けることにした。それぞれの動作とその内容を示す。

①　体移動 …… ロボット全体で移動する。ただし経路探索は考慮せず，目標地を与えれば不自由なくその地点まで移動できるものとする。

②　空手移動… 手に何も保持しない状態で手先を移動する。

③　荷重移動… 手に何かを保持した状態で手先を移動する。

②と③については大雑把に考えると手先の移動ということで同じ動作になるわけであるが，ここでは何かを保持した状態で手先を移動するためにはより複雑な制御が要求されるであろうと考えて特に区別した（例えばここで取り上げた問題では，把手をつかみにいくときの手の移動と扉の動きに従いながら把手を引くときの手の移動を区別した。）

システム化については新規に作成した処理プログラムはなく，対象問題の違いによるデータの変更だけで作業を終了できた。新規に作成されたプログラムとデータの規模をTable 4.2に示す。また実際の処理例をFig.4.10に示す。

4.3　システムの評価と今後の問題

現段階におけるシステムの評価としては，まだまだおもちゃの域を脱していないというのが正直なところである。ただしシステムの開発期間が正味１週間程度（実際にかかった期間としては約１か月）という短期間であったということ，新規作成のプログラム量（処理部分）がひじょうにコンパクトであったという点では評価できるものと思う。また，対象問題(1)から(2)への拡張について，主プログラムを変更することなく対処ができたという点で予想以上にうまくいったのではないかと考えている。今後このままの形でシステムを成長させていくとすれば，さらにいくつかの例について処理プログラムがうまく動作するかを検討した上で，ＤＢとＫＢを追加・修正するための専用エディタを用意しなければならない。また，そのエディタには知識の矛盾を排除するような配慮が必要であり，知識の構造化をサポートするような機能も要求されるものと思われる。

一方，今回試作したシステムでは，入力情報を得てから動作を終了するまでが一連の手続きとなっており，途中での動作の中断や変更はできないという弱点がある。ＨＡＳＰが目指す自律型の知能ロボットが対処すべき問題として，「命令の随時割り込み・中断・再開」という能力をもつべきであると考えられる[6]が，本システムの方法ではまだ対処ができていない。そこでシステム全体にFig.4.11で示すような巡回的な制御構造（メインループ）を用いて，処理を細かくステップ刻みに進めていく方法が考えられる[6]。図の処理内容を説明すると，まず，外部からの入力があれば外部入力感知部（①）で感知され，入力情報は入力解析部（②）で解析

されてシステム内部向けの情報として保存される。本システムにおけるCS－PARSER は入力解析部の一部となり，その他にセンサーから入ったデータを処理するルーチンなどもCS－PARSERと同レベルのサブプログラムとして考えられる。処理準備部（③）ではそのループでシステムが行なうべき処理の知識（メタ知識）が取り込まれる。例えば，本システムで試作したような動作系列の生成を行うための知識や，生成された動作系列をそのときの外界の状況を考慮しながら実行するための知識などのうちから１つのメタ知識を選択する。そしてメイン処理部（④）では③で得られたメタ知識に基づいて１ステップ処理をすすめられ，１回のメインループの処理が終了する。将来的に我々の目指すシステムはこのような制御構造をもつことが望ましいと考えられ，今後はメタ知識の種類をどのくらい用意すればよいものなのかを検討していく必要があると思われる。

4.4 まとめ

本章では，昭和63年度に試作した，日本語解析プログラム CS－PARSER を用いた知識ベースシステムについて述べた。システム（ソフトウェア上に仮想されたロボット）は日本語の命令文を読み取り，自分のなすべき行動を考えて実行する。対象モデルとして日常的になじみやすい単純な世界と，ＨＡＳＰの目指す炉内作業を考慮したごく簡単な世界の２種類を考えたところ，同一の処理プログラムに対して知識データを入れ換えることで目標としていたシステムの動作を実現することができた。ただし今回の試作システムでは，１つの命令を受けると，そこから得られる目標を達成する動作をすべて終了するまで実行を中断することができなかった。そのような処理を実現するためには，一連の手続きを細かいステップに刻んで巡回的な処理手順で随時外界の状況を取り込みながら１ステップずつ処理を進めていくような方法が有効と思われる。今後はそのような制御構造のもとに知識のあり方を再検討していく予定である。

参考文献

1） 井上博允：ロボット言語の研究課題，日本ロボット学会誌，２－２，pp. 3－6（1984）

2） 荒牧重登：ロボットプログラミング，井上書院（1988）.

3） Winograd, T ：" Understanding Natural Languages "，Academic Press（1972）（訳）淵一博，田村浩一郎，白井良明：言語理解の構造，産業図書（1976）.

4） Nilsson, N. J. ：" Principles of Artificial Intelligence "，Tioga Publishing（1980）（訳）白井良明，辻井潤一，佐藤泰介：人工知能の原理，日本コンピュータ協会（1983）.

5） 長尾真：言語工学，昭晃堂（1983）.

6） 阿部康昭，中嶋要：環境状態に合わせた問題解決を行う自律ロボットの制御構造，CSK技術通信，Vol. 9－1，No.17，pp. 38－49（1989）.

Table 4.1 System size for the application (1).

部分	ＬＩＳＰ行数	使用ファイル容量
処理プログラム	３４４	約１２ＫＢ
辞書以外ＤＢ・ＫＢ	８３	約　２ＫＢ
辞書ＤＢ	(※)　９	－－－
システム周辺プログラム	２０	約５７０Ｂ

(※) 新規登録語数

Table 4.2 System size for the application (2).

部分	ＬＩＳＰ行数（追加分）	使用ファイル容量
処理プログラム	３４４　(　０)	約１２ＫＢ
辞書以外ＤＢ・ＫＢ	１９１　(１０８)	約　５ＫＢ
辞書ＤＢ	(※) １４　(　５)	－－－
システム周辺プログラム	３０　(　１０)	約８４０Ｂ

(※) 新規登録語数

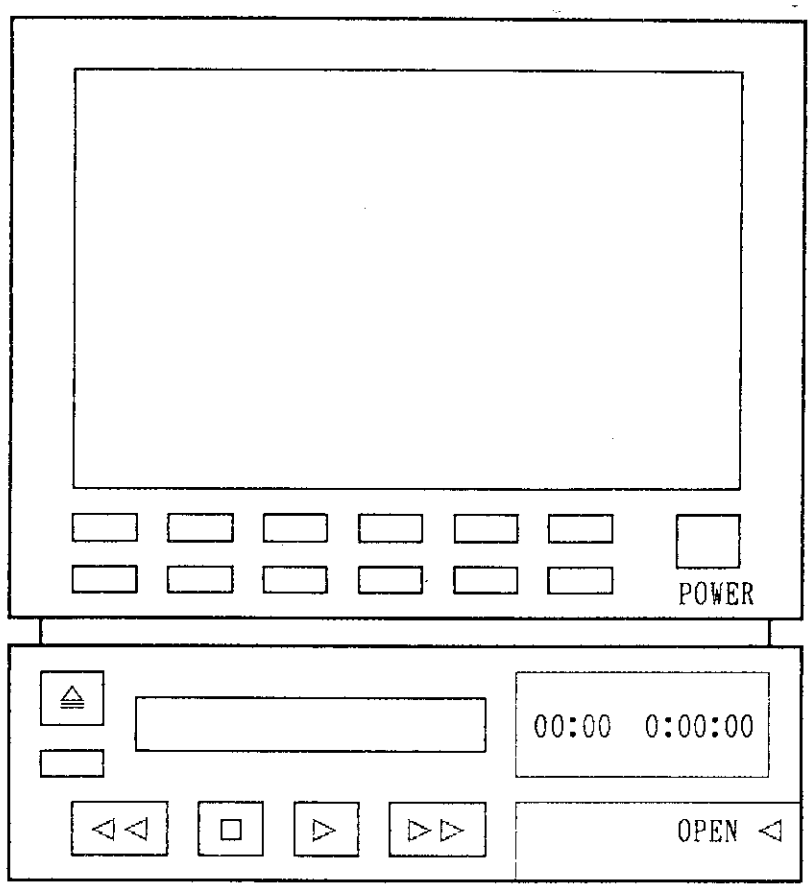


Fig. 4.1 The world model for the application (1).

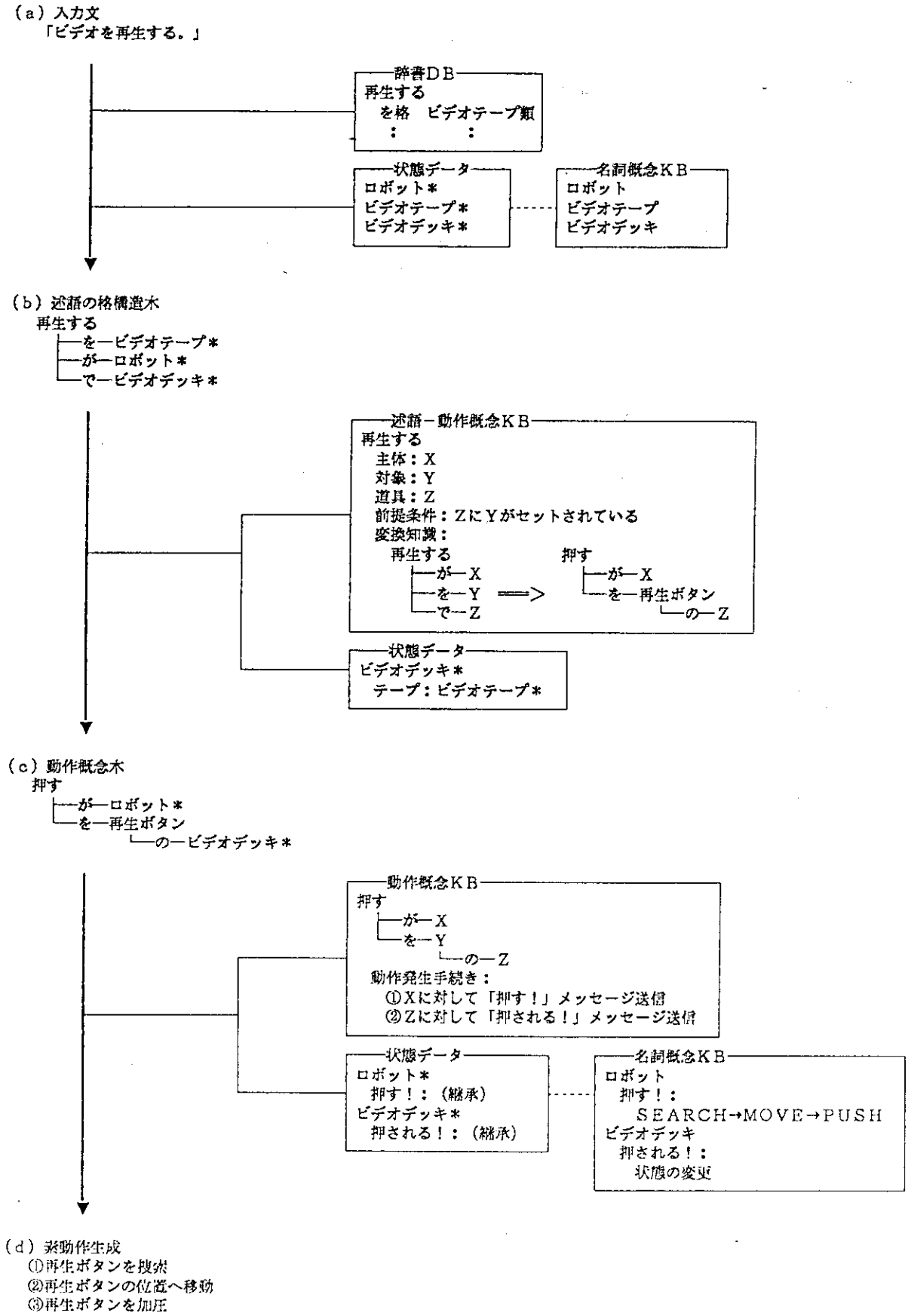
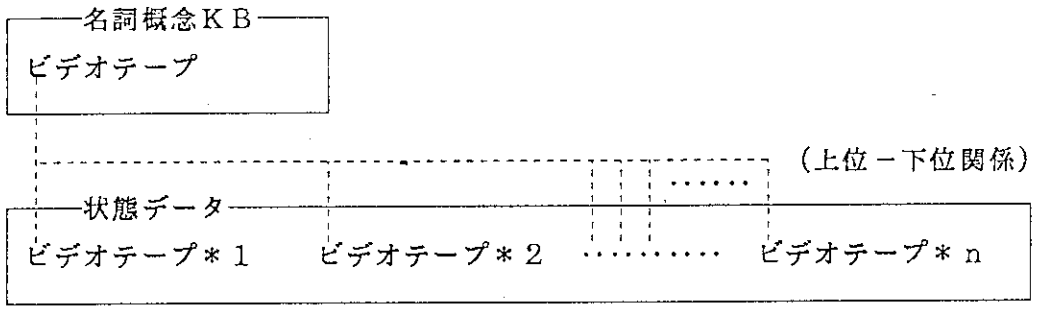
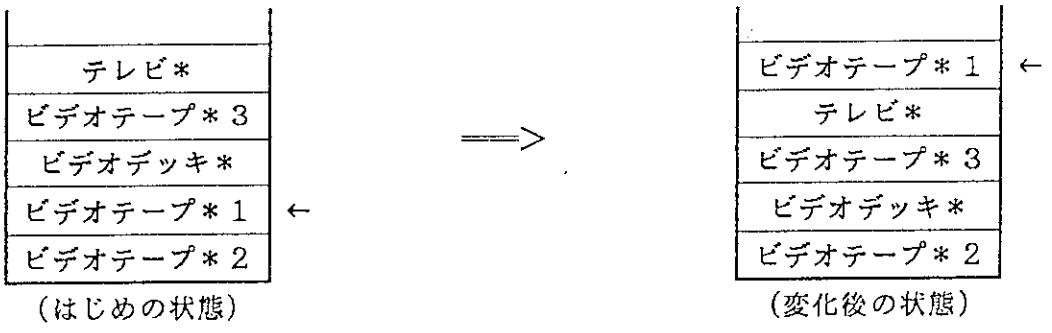


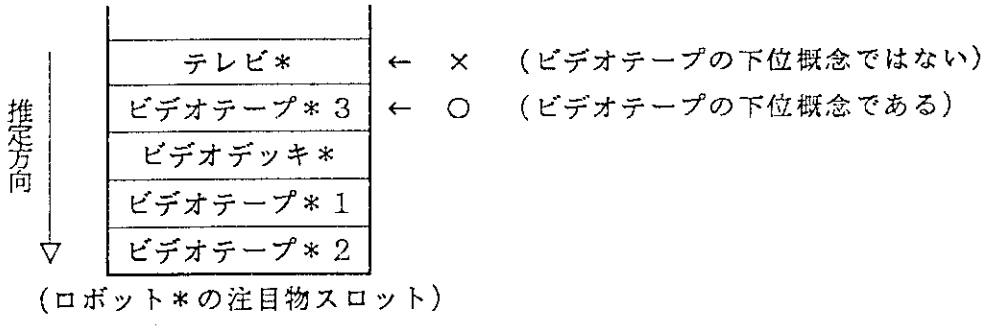
Fig. 4.2 An overview of the planning system.



（ａ）名詞概念と状態データの間の上位－下位関係



（ｂ）「ビデオテープ＊１」と言及された時のロボット＊の注目物スロットの変化



（ｃ）「ビデオテープ」と言及された時の具体物の推定

Fig. 4.3 Referent relations in the data base.

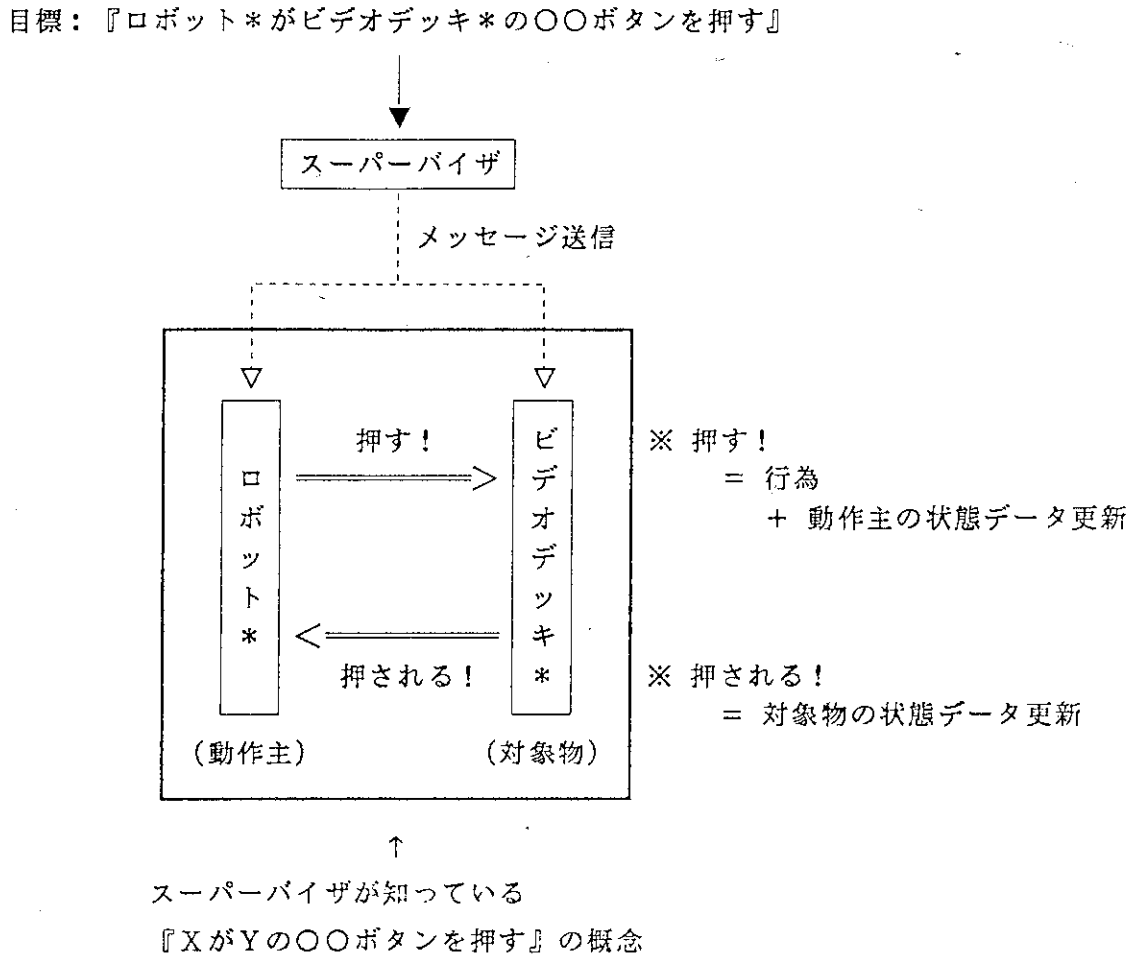


Fig. 4.4 A motion representation by using message passing.

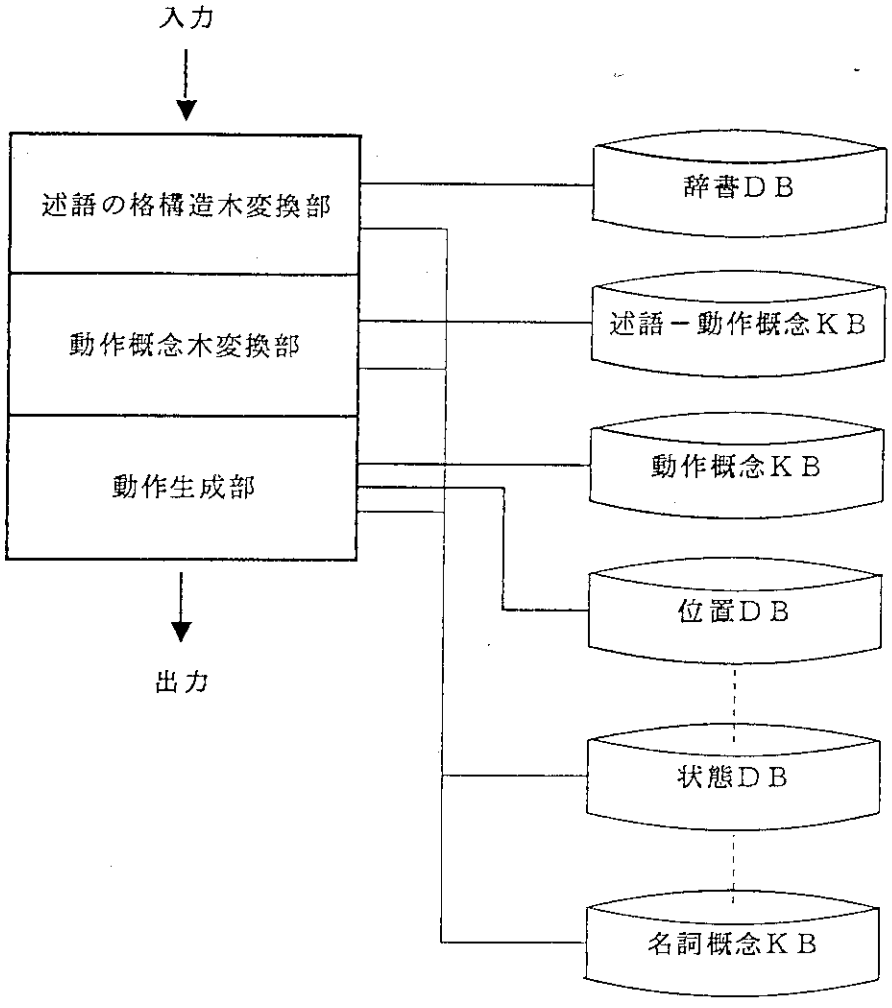


Fig. 4.5 An overview of the software.

Fig. 4.6 An example of the knowledge base for motions.

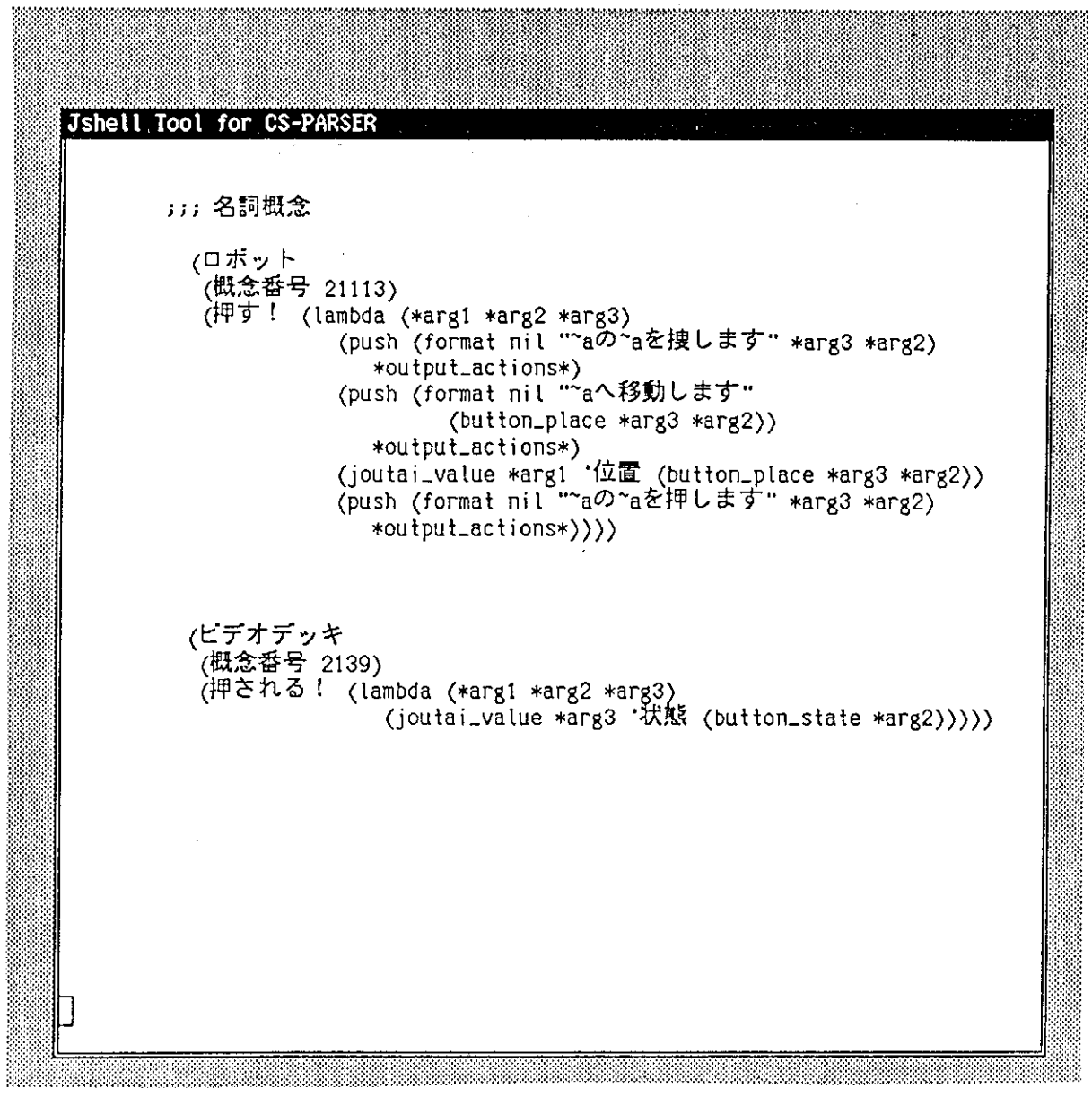


Fig. 4.7 Examples of the knowledge base for objects.

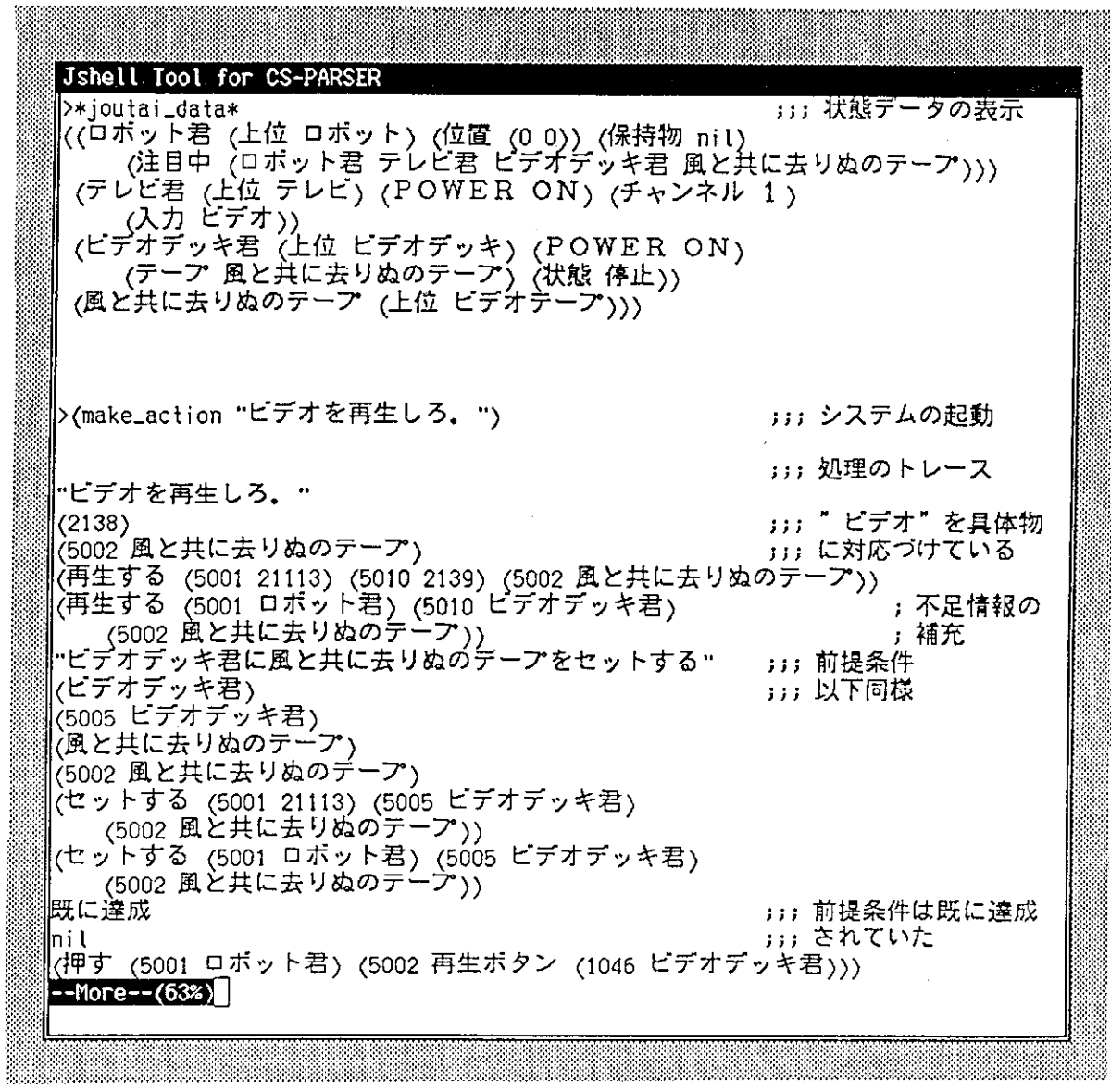
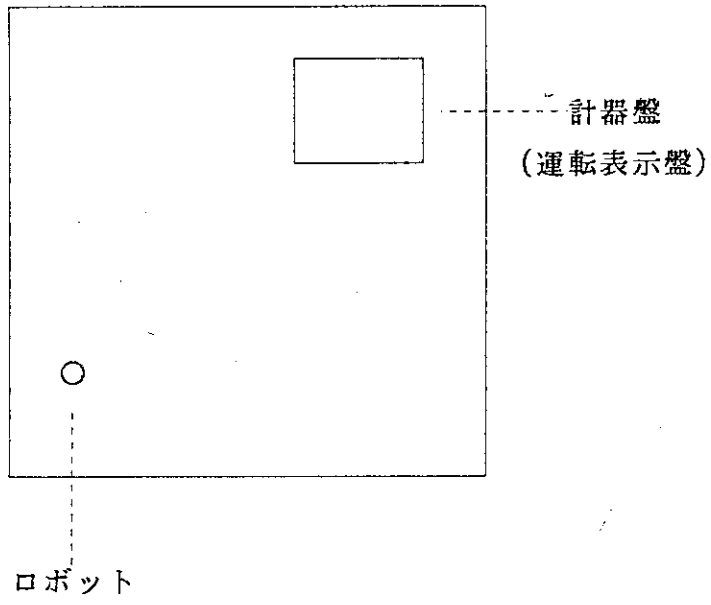
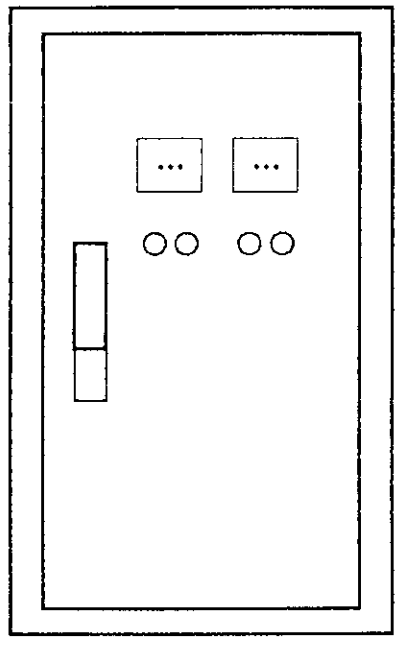


Fig. 4.8 Generated actions for the application (1).

Fig. 4.8 (Continued)



(a) ロボットの世界 (部屋を上から見たところ)



(b) 運転表示盤 (扉が開いていない状態)

Fig. 4.9 The world model for the application (2).

```
Jshell Tool for CS-PARSER
>*joutai_data*                                          ;;; 状態データの表示
((ロボット君 (上位 ロボット) (位置 (0 0)) (手位置 (0 0 1)) (保持物 なし)
     (注目中 (ロボット君 運転表示盤 扉ちゃん 把手ちゃん)))
 (運転表示盤 (上位 計器盤))
 (扉ちゃん (上位 扉) (全体 運転表示盤) (状態 閉じている))
 (把手ちゃん (上位 把手) (全体 扉ちゃん) (状態 出ている)))


>(make_action "計器盤の扉を開けろ。")                    ;;; システムの起動

        ************    処理は終わり    ************

        ************    動作を始めます  ************

"運転表示盤を捜します"
"運転表示盤の前に体移動します"
"扉ちゃんを捜します"
"把手ちゃんを捜します"
"把手ちゃんの位置に空手移動します"
"把手ちゃんをつかみます"
"扉ちゃんに従って荷重移動します"
"把手ちゃんをはなします"

        ************    動作終わり      ************


>*joutai_data*                                          ;;; 動作後の状態データ
((ロボット君 (上位 ロボット) (位置 (5 4)) (手位置 (6 4 2)) (保持物 なし)
     (注目中 (ロボット君 運転表示盤 扉ちゃん 把手ちゃん)))
 (運転表示盤 (上位 計器盤))
 (扉ちゃん (上位 扉) (全体 運転表示盤) (状態 開いている))
 (把手ちゃん (上位 把手) (全体 扉ちゃん) (状態 出ている)))

>(make_action "運転表示盤の扉を開けろ。")                ;;; 再度システムを起動


--More--(59%)
```

Fig. 4.10 Generated actions for the application (2).

```
Jshell Tool for CS-PARSER
"扉ちゃんに従って荷重移動します"
"把手ちゃんをはなします"

          ************    動作終わり      ************


>*joutai_data*                                          ;;; 動作後の状態データ
((ロボット君 (上位 ロボット) (位置 (5 4)) (手位置 (6 4 2)) (保持物 なし)
     (注目中 (ロボット君 運転表示盤 扉ちゃん 把手ちゃん)))
 (運転表示盤 (上位 計器盤))
 (扉ちゃん (上位 扉) (全体 運転表示盤) (状態 開いている))
 (把手ちゃん (上位 把手) (全体 扉ちゃん) (状態 出ている)))

>(make_action "運転表示盤の扉を開けろ。")               ;;; 再度システムを起動


          ************    処理は終わり    ************

          ************    動作を始めます  ************

          ************    動作終わり      ************


>*joutai_data*                                          ;;; 状態データの確認
((ロボット君 (上位 ロボット) (位置 (5 4)) (手位置 (6 4 2))
     (保持物 なし)
     (注目中 (ロボット君 テレビ君 ビデオデッキ君 風と共に去りぬのテープ
              運転表示盤 扉ちゃん 把手ちゃん)))
 (テレビ君 (上位 テレビ) (POWER ON) (チャンネル 1)
     (入力 ビデオ))
 (ビデオデッキ君 (上位 ビデオデッキ) (POWER ON)
     (テープ 風と共に去りぬのテープ) (状態 停止))
 (風と共に去りぬのテープ (上位 ビデオテープ))
 (運転表示盤 (上位 計器盤))
 (扉ちゃん (上位 扉) (全体 運転表示盤) (状態 開いている))
 (把手ちゃん (上位 把手) (全体 扉ちゃん) (状態 出ている)))

>
```

Fig. 4.10 (Continued)

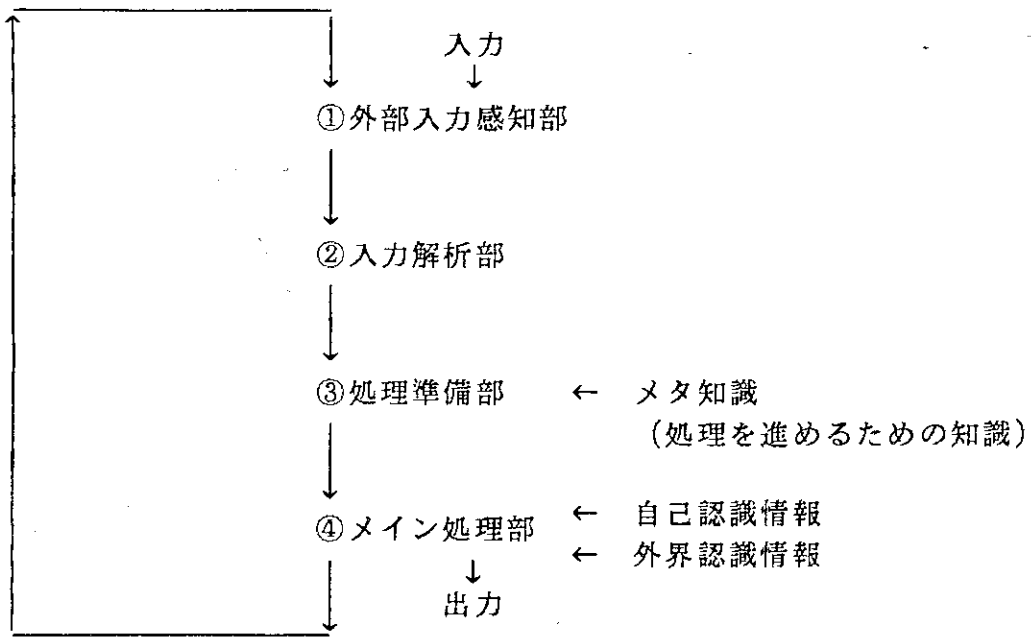


Fig. 4.11 A control structure of the interruption.

5. おわりに

本報告では，ＨＡＳＰにおける自然言語解析・行動計画とそれにかかわる知識ベースについて述べてきた。日本語解析については，ＣＳ－ＰＡＲＳＥＲをひな型として辞書の拡張作業を残すのみとなった。行動計画については，対象領域がかなり広範囲にわたるため，簡単なモデルを設定して，その中でプロトタイプとなるべきシステムを模索している段階である。平成元年度作業では，日本語入力に柔軟に対処できる行動計画プログラムを作成する予定である。ここではそのシステムの構成を概念的に説明し，本報告で述べた手法との対比を述べたい。

本報告で述べた知識ベースシステム（動作計画プログラム）は，あるきっかけをシステムに与えてしまうと二度とユーザがプログラムに働きかけることができない。このようなシステム制御は，人間が常に発話される事柄に注意を傾けているという事実から考えて，満足できない。そこで，常にシステムが入力待ちの（日本語を入力できる）状態をとることが必要となる。これを達成するためには，①日本語解析部，②動作計画部，③動作実行部がモジュールとして存在し，①→②→③の処理フローを基本要素とする巡回的な処理構造を考えなければならない。さらに，ある動作列を実行中にその動作をやめさせることが要求されるため，動作計画部は実行部が参照するフローチャートを作成し，実行部はフローチャートを１回の巡回処理で１ステップだけ処理するということが必要である。

上に述べた制御構造を採用した場合，次に述べる利点が考えられる。ある日本語入力に対してそれと同等なゴールとそのゴールを解決するためのフローチャートが生成されるため，それらの結果と環境とを保存しておけば，次に似たような命令が与えられた場合の参考として利用可能と考えられる（ある種の学習）。これらの情報は，人間が与えた知識ベースとは異なり，ロボット自身が作り出した記憶である。そこで，記憶を構成する方法論が与えられた場合，我々の考えている制御構造は自動的な知識の構造化の可能性を持ち合わせている。

謝　　辞

ＣＳ－ＰＡＲＳＥＲの使用の際お世話になった，（株）ＣＳＫ・ＡＩ科学技術事業本部の阿部康昭氏，中嶋要氏に深く感謝します。

5. おわりに

本報告では，ＨＡＳＰにおける自然言語解析・行動計画とそれにかかわる知識ベースについて述べてきた。日本語解析については，ＣＳ－ＰＡＲＳＥＲをひな型として辞書の拡張作業を残すのみとなった。行動計画については，対象領域がかなり広範囲にわたるため，簡単なモデルを設定して，その中でプロトタイプとなるべきシステムを模索している段階である。平成元年度作業では，日本語入力に柔軟に対処できる行動計画プログラムを作成する予定である。ここではそのシステムの構成を概念的に説明し，本報告で述べた手法との対比を述べたい。

本報告で述べた知識ベースシステム（動作計画プログラム）は，あるきっかけをシステムに与えてしまうと二度とユーザがプログラムに働きかけることができない。このようなシステム制御は，人間が常に発話される事柄に注意を傾けているという事実から考えて，満足できない。そこで，常にシステムが入力待ちの（日本語を入力できる）状態をとることが必要となる。これを達成するためには，①日本語解析部，②動作計画部，③動作実行部がモジュールとして存在し，①→②→③の処理フローを基本要素とする巡回的な処理構造を考えなければならない。さらに，ある動作列を実行中にその動作をやめさせることが要求されるため，動作計画部は実行部が参照するフローチャートを作成し，実行部はフローチャートを１回の巡回処理で１ステップだけ処理するということが必要である。

上に述べた制御構造を採用した場合，次に述べる利点が考えられる。ある日本語入力に対してそれと同等なゴールとそのゴールを解決するためのフローチャートが生成されるため，それらの結果と環境とを保存しておけば，次に似たような命令が与えられた場合の参考として利用可能と考えられる（ある種の学習）。これらの情報は，人間が与えた知識ベースとは異なり，ロボット自身が作り出した記憶である。そこで，記憶を構成する方法論が与えられた場合，我々の考えている制御構造は自動的な知識の構造化の可能性を持ち合わせている。

謝　　辞

ＣＳ－ＰＡＲＳＥＲの使用の際お世話になった，（株）ＣＳＫ・ＡＩ科学技術事業本部の阿部康昭氏，中嶋要氏に深く感謝します。