人工知能と対話技法 22－1
（1981. 9. 18）

# 定理証明システムSENRIの構成

山口高平† 西岡弘明†† 打浪清一† 手塚慶一†
（大阪大学 工学部† ・山口大学 理学部††）

## 1．はじめに

述語論理学における一般的な自動定理証明の手段としては，1965年にJ.A. Robinsonが発表した「導出原理」（resolution principle）があり，これは証明可能な定理は確実に導出法により証明できるという好ましい性質（完全性）を持つ。しかしながら，この方式は効率面で改善の余地があるため，導出の適用範囲を制限し探索空間を狭める研究－「制限付導出法」及び「戦略」の研究が，数多くなされてきた。また近年，述語論理をプログラミング言語とみなす「論理プログラミング」の研究が育ちつつあり，この場合パターン照合（unification）で動作する定理証明システムが，論理プログラミングシステムとして使用される。

過去における定理証明システムの研究は，C.L. ChangのTPU[2]・L.J. Henschen等[3]及び山崎等[4]のシステムがあるが，汎用記号処理言語LISPで構成されたものが数多い。この場合システムは，(1)入出力がユーザにとってわかりにくい。(2)データ構造に冗長がある。(3)廃品回収(GC)のコストが大きい。等の欠点を持ち，上記の様に定理証明及び論理プログラミングの研究が発展してくるに従って，定理証明を実行するのにより適したシステム（実用的なシステム）の必要性が大きくなっており，これは実用的な論理プログラミングシステムの必要性[5]にも深く関わっている。一方，効率的な定理証明アルゴリズム（新しい制限付導出法及び戦略）を発展させるためには，異なった制限付導出法及び戦略を比較実験するためのシステムが必要であるが[3]，この様な実験に適したシステムは，効率的な論理プログラミングの開発にも寄与すると考えられる。

以上述べた様に，実用的で且つ実験にも適した定理証明システムが少ない現状において，本稿では，

(1) 述語論理式（節）を表現するのに，より適したデータ構造を導入する。
(2) 入出力をより節の原形に近い形式にし，ユーザの取り扱いを容易にする。
(3) 各種制限付導出法（OL・SNL[6]・SPU[6]・TPU）を実行できるシステムにする。
(4) 頻繁に使用するルーチンのASSEMBLER VERSIONを作成する事により，単位導出時間を短縮する。
(5) 戦略をシステムに組み込む事により，探索空間を狭くして効率化を図る。
(6) ユーザが導出法と戦略を任意に選択し組み合わす事によって，異なった定理証明アルゴリズムを比較実験できる。

以上の項目を実現目標にし，実用的で且つ実験にも適した定理証明システムSENRIを構成したので報告する。

## 2．システム構成

### 2－1．システムの概観

SENRIの概観をFig.1に示す。まず，ユーザの選択した導出法と戦略及び入力節集合を入力モジュール（RESOLUTION & STRATEGY SELECTOR and INPUTER）によって読み込み，選択された制限付導出法実行モジュール（SNL EXECUTOR，SPU EXECUTOR，TPU EXECUTOR and OL EXECUTOR）に制御が移る。以下，戦略モジュー

ル（STRATEGY EXECUTOR）によって選択された戦略を適用しながら，推論モジュール（RESOLVENT GENERATOR and FACTOR GENERATOR）によって導出形及び簡約形を順次生成し，空節が導出されれば出力モジュール（OUTPUTER）により反駁木を出力して停止する。節リストは，LAVSと呼ばれるアレイ領域上で，節集合管理モジュール（SET OF CLAUSES ADMINISTRATER）によって管理されている。また，セルの操作・LAVSと文字データ領域の管理・リストコピー・代入操作とその応用操作・節の種類判定・GC等のモジュールは図示されていないが，システムの随所で数多く使用されている。以上の構成により，SENRIは他の同種のシステム[2]~[4]に比べて，以下の様な特徴を持つ。

(i) 異なったアルゴリズムで実験を行うためには，各種のリストをユーザが指定する必要があったが[3][4]，SENRIでは節（リスト）の種別をソフトウェアで行う事により，その手間を省きユーザの負担を軽減している。

(ii) 制限付導出法実行モジュールと推論及び戦略モジュールを分離する事により，特別な導出や戦略の記述が容易となり，拡張性のあるシステムが実現されている。

(iii) (ii)と入力モジュールによって，ユーザがオプションの指定を変更するだけで，異なった定

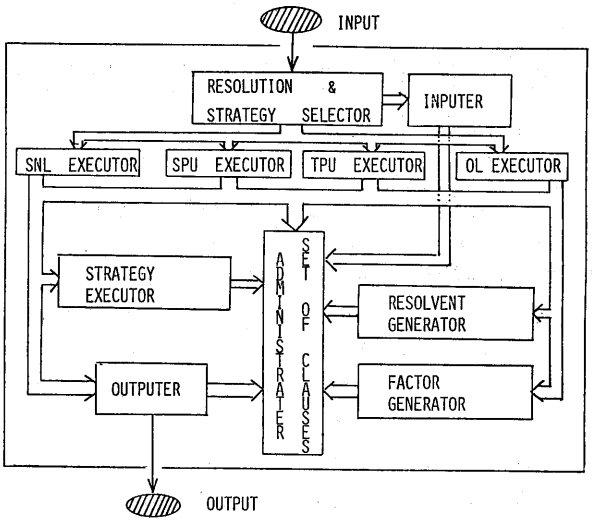
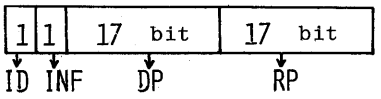



FIG.1 THEOREM PROVER 'SENRI' SYSTEM




ID部 ｛リストセルの時 → 0をセットする
　　｛データセルの時 → 1をセットする

INF部

| セルの型 | データの型 | 値 |
|---|---|---|
| データセル | 定数 | 0 |
| | 変数 | 1 |
| | 関数記号 | 0 |
| | 述語記号 | 0 |
| | 補の述語記号 | 1 |

(注) 同値のデータの型の区別は，セルの位置を情報として，ソフトウェアで行う。

DP部 → データセルの時は文字データを指すためのポインタをセットし，リストセルの時は下連結のためのポインタをセットする

RP部 → 右連結のためのポインタをセットする

FIG.2 セルの構造

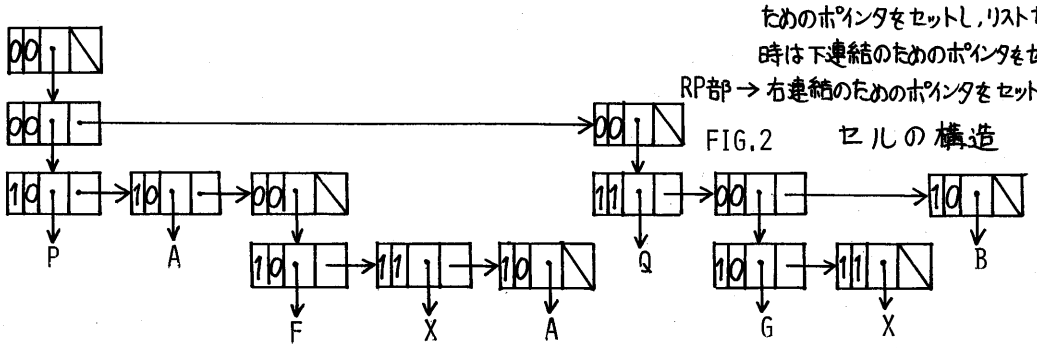



FIG.3 P(a,f(x,a))∨~Q(g(x),b) の内部データ構造

理証明アルゴリズムの比較実験が，容易に行える。

## 2．2　データ構造

SENRIのデータ構造の基本単位であるセルは，実際には計算機（大阪大学大型計算機センターのACOSシステム900）の1ワードで構成されており，Fig.2の様な構造を持つ。入力された節は，Fig.3の具体例で示す通り，セルを基本単位とするリスト構造によって表現される。

SENRIの基本データ構造は，LISPのもの（2つのポインタを持つセル）と比較して，INF部により述語論理式を無理なくコンパクトに表現しており，データ圧縮を行っていると言える。具体的には，変数の取り扱いに関して，LISPで構成された多くのシステムでは変数リストを独立に設ける事によって処理するのに対し，SENRIではINF部に1をセットしソフトウェアで管理する事が可能で，特別な変数リストは必要でない。また，補の述語記号の取り扱いに関しては，Fig.4に示す様に，LISPではNOTという余分なデータ領域を必要とするのに対し，SENRIではやはりINF部に1をセットしソフトウェアで管理する事ができる。

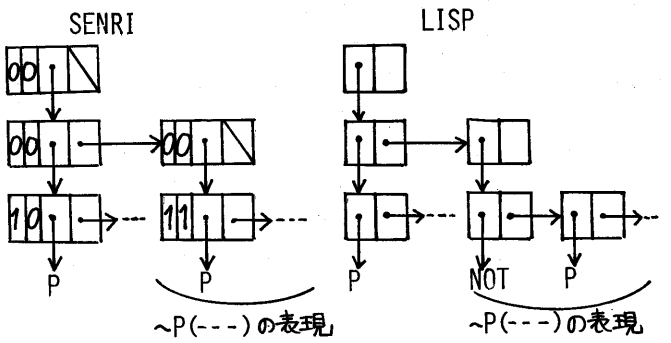



FIG.4　SENRIとLISPのデータ構造の比較

```
<INPUT> ::=< RESOLUTION> (< STRATEGY SEQUENCE >)< SET OF CLAUSES >.
<RESOLUTION> ::= SNL| SPU| TPU| OL
<STRATEGY SEQUENCE> ::= <STRATEGY> = <UNSIGNED INTEGER> |
                        <STRATEGY> = <UNSIGNED INTEGER> , <STRATEGY SEQUNCE>
<STRATEGY> ::= SUPPORT| TAUTO| EQRSLVNT| NOFACTOR| FDEPTH| RDEPTH|
               CLENGTH| TPUN2| TPUN3| TPUN4| TPUSUPPORT| OCLENGTH|
               ANSWER| NOTANS| PRINTRSLVNT| PRINTGC
<SET OF CLAUSES> ::= <NON-EMPTY CLAUSE>| <NON-EMPTY CLAUSE> ,
                     <SET OF CLAUSES>
<NON-EMPTY CLAUSE> ::= <LITERAL SEQUENCE>
<LITERAL SEQUENCE> ::= <LITERAL> |<LITERAL> / <LITERAL SEQUENCE>
<LITERAL> ::= <ATOMIC FORMULA> |- <ATOMIC FORMULA>
<ATOMIC FORMULA> ::= <PREDICATE SYMBOL> ( <TERM SEQUNCE> )|
                     <PREDICATE SYMBOL>
<TERM SEQUNCE> ::= <TERM>| <TERM> , <TERM SEQUENCE>
<TERM> ::= <CONSTANT>| <VARIABLE> |<FUNCTION>
<FUNCTION> ::= <FUNCTION SYMBOL> ( <TERM SEQUENCE> )
<VARIABLE> ::= $ <IDENTIFIER>
<CONSTANT> ::= <IDENTIFIER>
<PREDICATE SYMBOL> ::= <IDENTIFIER>
<FUNCTION SYMBOL> ::= <IDENTIFIER>
```

── 注意事項 ──

(1) 上文法において，/は論理和，－は否定記号を表す。
(2) 識別子の長さは任意であるが，9文字目以降は無視される。
(3) 入力は自由欄形式である。
(4) 注釈は，/* 文字列 */ の形式で，入力の任意の場所に挿入できる。
　但し，文字列には */を含まない。

FIG.5　SENRI　SYNTAX

```
SNL(NOFACTOR=1,FDEPTH=3,CDEPTH=4,EQ=1,CLENGTH=3,PRINT=1,GARBAGE=1)
/*     EXAMPLE OF GROUP THEORY (TPU 1)     */
P(G($X,$Y),$X,$Y) ; -P(K($X),$X,K($X)) ; P($X,H($X,$Y),$Y) ;
P($X,$V,$W)/-P($X,$Y,$U)/-P($Y,$Z,$V)/-P($U,$Z,$W) ;
P($U,$Z,$W)/-P($Y,$Z,$V)/-P($X,$Y,$U)/-P($X,$V,$W) .
```

FIG.6　INPUT　EXAMPLE

## 2－3．入力モジュール

SENRIにおける入力は，Fig.5の文法で示される様に，導出法(戦略,---)節集合の形式で行われるが，導出法と戦略はユーザのオプションとしてTABLE-1から指定されれば，RESOLUTION & STRATEGY SELECTORによって読み込まれる。戦略には，支持集合の指定(1)・インタラクティブに最適値を決定する必要がないもの(2-4)必要があるもの(5-7)・各導出法に専用のもの(8-12)が含まれ，その他ユーザの便宜のために(13-16)がある。また，節集合はINPUTERによって読み込まれるが，SNL導出法及びSPU導出法を実行する場合には，節集合はHorn 節集合に限定される。入力の具体例をFig.6に示すと共に，TABLE-2に入力モジュールの概要を示す。

(注) モジュールの概要は右表の様に，モジュール構成のサブルーチンを用いて表わす。

| モジュール名 | |
|---|---|
| サブルーチン名 | サブルーチンの概要 |

TABLE-1　USER'S OPTION

| | | 指定形式 | 指定内容 |
|---|---|---|---|
| 導出法 | | SNL | SNL導出法の実行 |
| | | SPU | SPU導出法の実行 |
| | | TPU | TPU システムの実行 |
| | | OL | OL 導出法の実行 |
| 戦略 | 1 | SUPPORT=(N,..) | 支持集合の指定 |
| | 2 | TAUTOLOGY=1 | 恒真節の除去 |
| | 3 | EQRSLVNT=1 | 等位導出形の除去 |
| | 4 | NOFACTOR=1 | factoring 操作の除去 |
| | 5 | FDEPTH=N | 関数の深さをNに制限 |
| | 6 | RDEPTH=N | 導出の深さをNに制限 |
| | 7 | CLENGTH=N | 節の長さをNに制限 |
| | 8 | TPUN2=N | TPUシステムにおいてパラメータN2をNに指定 |
| | 9 | TPUN3=N | TPUシステムにおいてパラメータN3をNに指定 |
| | 10 | TPUN4=N | TPUシステムにおいてパラメータN4をNに指定 |
| | 11 | TPUSUPPORT=((N,..),..) | TPUシステムにおいて非単位節に対する支持集合の指定 |
| | 12 | OCLENGTH=N | OL導出法において mon-framed リテラルの数をNに制限 |
| | 13 | ANSWER=1 | unit answering clause での停止要求 |
| | 14 | NOTANS=1 | 述語名がNOTANSの負単位節での停止要求 |
| | 15 | PRINTRSLVNT=1 | 生成された導出形の出力要求 |
| | 16 | PRINTGC=1 | Garbage Collection 起動の際メッセージの出力要求 |

TABLE-2　入力モジュールの概要

| RESOLUTION & STRATEGY SELECTOR | |
|---|---|
| TPINPT(IR,IST,IS,ISUP,IPNT) | ユーザの指定した導出法と戦略を読み込む。IRとISTにそれらの値をセットし，ISには以下に示すSCLINの引数がセットされる。支持集合の指定がある場合はISUP(TPUを実行する時はIPNTに支持集合の管理情報)にその値をセットする |
| INPUTER | |
| CLIN(N) | 入力節を節リストに変換して，その番地をNにセットする。 |
| SCLIN(N) | CLINによって節を読み込み，節集合リストを生成し，その番地をNにセットする。 |

## 2－4．制限付導出法実行モジュール

このモジュールは，TABLE-1に示す制限付導出法を実行するためのものであるが，TABLE－1の戦略において SUPPORT=(N,---)・NOFACTOR=1・RDEPTH=N・TPUN2=N・TPUN3=N・TPUSUPPORT=((N,--),--)・ANSWER=1・NOTANS=1・PRINTRSLVNT=1・PRINTGC=1の処理は，この中に組み込まれている。TABLE-3に，制限付導出法実行モジュールの概要を示す。

## 2－5．推論モジュール

このモジュールは，導出形あるいは簡約形を生成するためのものである。生成される導出形のリストと簡約形のリストは，同じ節集合管理ルーチンLCSETで取り扱い易くするため，Fig.7 と Fig.8 に示す様に同様の構造を持つ。TABLE-4に

推論モジュールの概要を示す。

TABLE-3　制限付導出法実行モジュールの概要

| | |
|---|---|
| SNL EXECUTOR | |
| SNL(IST,IS,ISP) | 戦略IST(支持集合の指定はISP)の下で入力節集合ISからSNL導出法を実行する |
| SPU EXECUTOR | |
| SPU(IST,IS,ISP) | 戦略IST(支持集合の指定はISP)の下で入力節集合ISからSPU導出法を実行する |
| TPU EXECUTOR | |
| TPU(IST,IS,ISP,IPNT) | 戦略IST(支持集合の指定はISP,支持集合の管理情報はIPNT)の下で,入力節集合ISからTPUシステムを実行する |
| OL EXECUTOR | |
| OL(IST,IS,ISP) | 戦略IST(支持集合の指定はISP)の下で入力節集合ISからOL導出法を実行する |

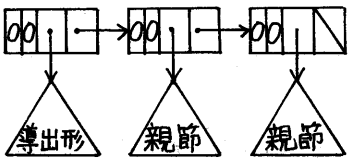



FIG.7
導出形の
リスト

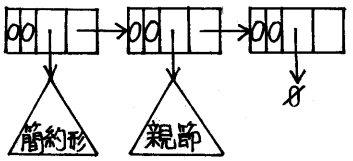



FIG.8
簡約形の
リスト

TABLE-4　推論モジュールの概要

| | |
|---|---|
| RESOLVENT GENERATOR | |
| RESOLV(LRESLV,IFLAG,ICC1,N1,ICC2,N2) | 節ICC1のN1番目のリテラルと節ICC2のN2番目のリテラルとから,一般の導出による導出形リストを生成する。非空節を生成した場合は,IFLAGに1をセットし,LRESLVにその番地をセットする。空節を生成した場合は,IFLAGに　をセットし,LRESLVにその番地をセットする。導出不可能の場合は,IFLAGに-1をセットし,対応するリテラルが存在しない場合は,IFLAGに-2をセットする。 |
| SNLRSV(LRESLV,IFLAG,ICC1,ICC2,N2) | 節ICC1の最右リテラルと節ICC2のN2番目のリテラルとから,SNL導出による導出形リストを生成する。LRESLVとIFLAGには,RESOLVと同様の値がセットされる |
| OLRSLV(LRESLV,IFLAG,ICC1,ICC2,N2,NLC) | 節ICC1の最右リテラルと節ICC2のN2番目のリテラルとから,OL導出による導出形リストを生成する。LRESLVとIFLAGには,RESOLVと同様の値がセットされ,NLCには簡約形を生成する時の制御情報がセットされる。 |
| FACTOR GENERATOR | |
| INFACT(IC) | 入力節ICの簡約形を生成し,それを入力節集合に加える。 |
| OINFCT(IC) | INFACT(IC)と同様であるが,OL導出を実行する時に使用される。 |
| RSFACT(IFACT,IC,NONNEW,NO) | 導出形ICから生成される簡約形の集合から,NO番目の簡約形を取り出し,I　CTにその番地をセットする。簡約形が生成されない場合は,IFACTに0がセットされる。NONNEWは,導出形のnon-new literalの数である |
| ORFACT(IFACT,IC,NLC,NO) | NLCが簡約形を生成する時の制御情報である以外は,RSFACT(IC)と同様である。 |

## 2－6．戦略モジュール

このモジュールは，制限付導出法実行モジュールの中で取り扱われていない戦略のためのものであり，TABLE-5に概要を示す。

TABLE-5　戦略モジュールの概要

| STRATEGY EXECUTOR | |
|---|---|
| ITAUTO(IC) | 節ICが恒真節ならば関数値を1とし，そうでなければ0とする。但し，OL導出法を実行する場合は，枠リテラルの取り扱いが必要であるため，IOTAUT(IC)を用いる。 |
| IEQ(IC) | 節IC1と節IC2が等しければ関数値を1とし，そうでなければ0とする。但し，OL導出法を実行する場合は，IOEQ(IC1,IC2)を使用する。 |
| LENGTH(IC) | 節ICの長さを関数値とする。OL導出法を実行する場合は，IOLENG(IC)を用いる。 |
| IDEPTH(IC) | 節ICの関数の深さを関数値とする。 |

## 2－7　出力モジュール

このモジュールは，証明結果を出力するためのものであり，TABLE-6にこの概要を示す。

TABLE-6　出力モジュールの概要

| OUTPUTER | |
|---|---|
| CLOUT(L) | 節Lを出力する |
| DTREE(L) | 節Lの演繹木を出力する。Lが空節ならば，証明結果を出力する事になる。 |

## 2－8　節集合管理モジュール

このモジュールは，節の取り出し及び節集合に導出形・簡約形を付加する事等のリスト処理を行うためのものであり，以下の4つの機能を持つ。

(1) 初期設定－入力節集合の最後の節にINPUTとLENDという2つの管理変数をセットし，INPUTは入力節の終りを示す変数として，LENDは節集合の終りを示す変数として以後取り扱われる。

(2) 導出形・簡約形の付加－節集合リストの最後に，導出形リスト及び簡約形リストを連結する。

(3) 節の探索－節集合中の節の順位から，節の番地を取り出す。

(4) 節の削除－節集合中の最後の節を削除する。

以下，節集合管理モジュールの概要をTABLE-7に示す。

TABLE-7　節集合管理モジュールの概要

| SET OF CLAUSES ADMINISTRATER | | | | | | |
|---|---|---|---|---|---|---|
| LCSET(C,L) | 節集合の管理を行い，引数Cによってその機能と関数値は以下の様になる | | | | | |
| | C | 機能 | 関数値 | C | 機能 | 関数値 |
| | 'S' | 初期設定 | LEND | 'G' | 節の探索 | L番目の節の番地 |
| | 'P' | 導出形の付加 | LEND | 'D' | 節の削除 | ―― |

## 3．SENRIシステムの効率改善

最初にも述べた様に，効率の良い定理証明システムは，定理証明の研究及び論理プログラミングの研究において非常に重要である。本章では，抽象代数等の数学理論の多くはHorn節集合のクラスの問題として表現できる事から，TPU EXAMPLE 1～9[1]（TPU EXAMPLE 9だけはrehaming[6]の操作によってもHorn節集合にならない。以後TPU-1～9と略記する。）を定理証明の標準的な問題と考え，すべて

の問題を比較的短時間（30秒を打ち切り時間とする）で証明できる事を目標とし，SENRIの効率改善を図る。

まず，Horn節集合に対して完全性を保持するＳＮＬ導出法及びＳＰＵ導出法をそのまま実行するSENRIシステムの効率をTABLE-8に示す。但し，各ルーチンはすべてFORTRANで作成されている。この表から，制限付導出法をそのまま適用するだけでは効率の良い実用的なシステムと言い難い事がわかる。制限付導出法の導入目的は，探索空間を狭める事による効率改善であったが，効率改善はFig.9に示す様に以下の２つの側面から捉える事ができる。

①各段の処理速度を速める事による，すなわち主に単位導出時間短縮による効率改善。

②探索空間を狭める事による効率改善。

TABLE-8 制限付導出法をそのまま実行するSENRI FORTRAN VERSIONの効率

| 制限付導出法 / 証明問題 | ＳＮＬ | ＳＰＵ |
|---|---|---|
| ＴＰＵ－１ | * | ８４５３ |
| ＴＰＵ－２ | * | * |
| ＴＰＵ－３ | * | * |
| ＴＰＵ－４ | * | * |
| ＴＰＵ－５ | ６２８ | * |
| ＴＰＵ－６ | ９７１ | * |
| ＴＰＵ－７ | * | ６６７６ |
| ＴＰＵ－８ | * | * |
| ＴＰＵ－９ | ９３６１ | REJECTION |

* : LAVS EMPTY or STACK OVER FLOW
unit time : msec
LAVS : 2500 cells (36bits/cell)
machine : ACOS series 77 NEAC SYSTEM 900

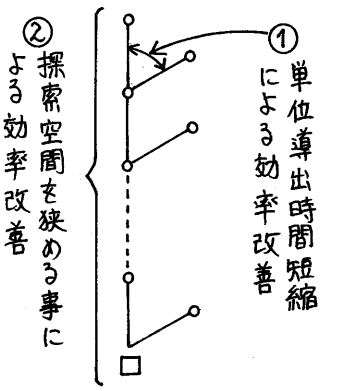



FIG.9 効率改善のための２つの側面

①の実現方法としては，定理証明を行うのにより適したシステム（言語）の導入が一般に考えられるが，本稿ではSENRI ASSEMBLER VERSIONを作成する事により実現した。一方，②の実現方法としては，制限付導出法・戦略・発見的手法及び補助制御言語の導入が一般に考えられるが，本稿ではＳＮＬ導出法及びＳＰＵ導出法に戦略を組み合わせる事により実現した。

①の実現結果
SENRI ASSEMBLER VERSIONを頻繁に使用するルーチン群（原始関数群）から順次作成していく事により効率改善を図った。結果をTABLE-9に示す。

①＋②の実現結果
TABLE-1に示されたス～７の戦略をＳＮＬ導出法及びス～４の戦略をＳＰＵ導出法に付加する事により効率改善を図った。結果をTABLE-10に示す。

TABLE-10より，ＳＮＬ導出法及びＳＰＵ導出法と戦略の組み合わせを実行するSENRI ASSEMBLER VERSIONは，システム全体として比較的短時間ですべての問題の証明を完了しており，効率の良い実用的な定理証明システムが実現できたと言える。

TABLE-9 制限付導出法をそのまま実行する SENRI ASSEMBLER VERSION の効率（①の実現結果）

| 証明問題＼制限付導出法 | SNL | SPU |
|---|---|---|
| TPU-1 | * | 1170 |
| TPU-2 | * | * |
| TPU-3 | * | * |
| TPU-4 | * | * |
| TPU-5 | 222 | * |
| TPU-6 | 265 | * |
| TPU-7 | * | 978 |
| TPU-8 | * | * |
| TPU-9 | 1569 | REJECTION |

TABLE-10 制限付導出法＋戦略を実行する SENRI ASSEMBLER VERSIONの効率（①＋②の実現結果）

| 証明問題＼制限付導出法＋戦略 | SNL ＋ STRATEGIES | SPU ＋ STRATEGIES |
|---|---|---|
| TPU-1 | 805 | 2838 |
| TPU-2 | RETIRED | 23037 |
| TPU-3 | 16983 | 24803 |
| TPU-4 | 1478 | 4857 |
| TPU-5 | 37 | 941 |
| TPU-6 | 87 | 4278 |
| TPU-7 | 188 | 250 |
| TPU-8 | 5520 | 2572 |
| TPU-9 | 478 | REJECTION |

*：LAVS EMPTY or STACK OVERFLOW , RETIRED : MORE THAN 30 MSEC
unit time : msec , LAVS : 2500 cells (36 bits/cell)
machine : ACOS series 77 NEAC SYSTEM 900

## 4．システムの評価

多くの定理証明システムは，汎用記号処理言語LISPで構成されており，SENRIを評価する場合LISPとの比較は欠かせない。そこで，LISPで構成されたTPUシステムと同じ導出法と戦略をSENRIで構成し，比較検討を行った。

### 4－1 データ構造の比較

2－2でLISPのデータ構造とSENRIのものを比較した様に，SENRIでは第2フィールドのINF部によって述語論理式をコンパクトに表現している。LISPでは，節の本体リストの他に，変数リストや節の履歴情報を表すリストを設けているのに対し，SENRIでは変数や節の履歴情報をソフトウェアで管理している。この事は，リスト消費の軽減につながり記憶領域を節約するばかりでなく，GCの起動回数を削減し効率面にも寄与すると考えられる。

### 4－2．入出力の比較

TPU－1を例にとり，原データ，LISPとSENRIにおける入力形式及びLISPとSENRIにおける出力形式をFig. 10～14に示す。Fig. 11とFig. 12を比較する事により，SENRIの方が原データにより近い形式で入力ができ，入力が容易である事がわかる。また，Fig. 13とFig. 14を比較する事により，SENRIではどの節とどの節を親節にして導出を行ったかがよくわかる出力形式であるのに対して，LISPでは単に節の番号を出力する形式である。すなわち，SENRIの方が反駁の手順を容易に理解できる形式であると言える。

### 4－3．効率の比較

LISPコンテスト[8]において，TPU-1～9を各種LISPで実行した結果が報告されている。現在，SENRIで構成したTPUシステムでTPU-1～9の実行時間を測定し

(1) $P(g(x,y),x,y)$

(2) $P(x,h(x,y),y)$

(3) $\sim P(k(x),x,k(x))$

(4) $\sim P(x,y,u) \vee \sim P(y,z,v) \vee \sim P(x,v,w) \vee P(u,z,w)$

(5) $\sim P(x,y,u) \vee \sim P(y,z,v) \vee \sim P(u,z,w) \vee P(x,v,w).$

FIG.10 TPU EXAMPLE 1の原データ

```
(TPU
  @((1(X Y)((P(G X Y)X Y)))
    (2(X Y)((P X(H X Y)Y))))
  @((3(X)((NOT P(K X)X(K X)))))
  @((4(X Y Z U V W)((NOT P X Y U)(NOT P Y Z V)(NOT P X V W)(P U Z W)))
    (5(X Y Z U V W)((NOT P X Y U)(NOT P Y Z V)(NOT P U Z W)(P X V W))))
  @((3)NIL)
  @5
  @2
  @3
  @0)
```

FIG.11 LISPにおける入力形式

```
TPU(TPUSUPPORT=((3),()),MAXSELNU=2,MAXBFFDT=3,MAXFDUC=0)
/*     EXAMPLE OF GROUP THEORY (TPU 1)    */
P(G($X,$Y),$X,$Y) ; -P(K($X),$X,K($X)) ; P($X,H($X,$Y),$Y) ;
P($X,$V,$W)/-P($X,$Y,$U)/-P($Y,$Z,$V)/-P($U,$Z,$W) ;
P($U,$Z,$W)/-P($Y,$Z,$V)/-P($X,$Y,$U)/-P($X,$V,$W) .
```

FIG.12 SENRIにおける入力形式

```
((6 3 4 4)(11 2 6 2)(15 1 11 1)(CONTRADICTION 1 15)).
```

FIG.13 LISPにおける出力形式

```
+++ DEDUCTION +++

*** INPUT CLAUSE( 1) ***
P(G($X,$Y),$X,$Y)

*** INPUT CLAUSE( 4) ***
-P($X,$Y,$U) / -P($Y,$Z,$V) / -P($X,$V,$W) / P($U,$Z,$W)

*** INPUT CLAUSE( 3) ***
-P(K($X),$X,K($X))

*** RESOLVENT( 6) OF ( 4) AND ( 3) ***
-P($X1,$X2,K($X3)) / -P($X2,$X3,$X4) / -P($X1,$X4,K($X3))

*** INPUT CLAUSE( 2) ***
P($X,H($X,$Y),$Y)

*** RESOLVENT( 11) OF ( 6) AND ( 2) ***
-P($X1,$X2,K(H($X2,$X3))) / -P($X1,$X3,K(H($X2,$X3)))

*** RESOLVENT( 15) OF ( 11) AND ( 1) ***
-P(G($X1,K(H($X1,$X2))),$X2,K(H($X1,$X2)))

*** RESOLVENT( 26) OF ( 1) AND ( 15) ***
EMPTY
```

FIG.14 SENRIにおける出力形式

```
+++ TPU SYSTEM END +++
```

ているが，今までの所 LISP コンパイラ とほぼ同程度である。しかしながら，この報告ではＧＣの時間が含まれていない値が多い。従って，ＧＣの起動回数がLISPと比べて少ないと考えられるSENRIとLISPを比較する場合，実行時間とＧＣの稼動時間を合わせて比較する必要がある。

## 4－4　ＧＣの特徴

SENRIにおいて再使用の可能性がリストは，節集合リストだけであり，代入リストや節のコピーリスト等は再使用の可能性が無い。従って，再使用の可能性があるリストをマークするためにスタックを用いる必要がなく，節集合の開始番地から１回たどる事によりマーク付けを完了する事ができる。この様に，簡易的な手法によってＧＣを取り扱う事ができ，ＧＣの稼動時間はLISPに比べて短縮できると考えられる。

## 5．むすび

本稿で述べた定理証明システムSENRIは，

(1) リストの指定変更等は不必要であり，オプションの指定を変更するだけで，異なった定理証明アルゴリズムを比較する事ができる。
(2) システムはモジュール別に構成されており，拡張性がある。
(3) リスト構造のデータ圧縮を行っている。
(4) 入力が容易で，出力が見易い。
(5) 記憶領域を節約している割には，処理効率はあまり悪くない。
(6) 簡易的な手法で，ＧＣを取り扱う事ができる。

以上の特色を持つ実験に適して拡張性があると共に，ユーザに則した実用的なシステムであり，定理証明及び論理プログラミングの研究において重要な位置を占めると考えられる。

－謝辞－ 御討論いただいた手塚研究室オートマトングループの諸兄，特に勝一善氏に感謝します。

## －参考文献－

(1) C.L.Chang and R.C.Lee :"Symbolic Logic and Mechanical Theorem Proving"(1973)
(2) C.L.Chang:"The unit proof and the input proof in theorem proving"JACM vol.17 no.6 (1974)
(3) L.Henschn,R.Overbeek and L.Wos:"A Theorem-Proving Language for Experimentation " CACM vol.17 no.6 (1974)
(4) 山崎,山本:"定理の自動証明のためのプログラミングシステムの構成" 情報処理 vol.17 no.5 pp.410～416 (1976)
(5) K.A.Bowen:"PROLOG" ACM 0-89791-008-7/79/1000/0014 (1979)
(6) D.Kuehner:"Some Special Purpose Resolution Systems" Machine Intelligece 7 pp.117 - 128 (1973)
(7) L.Henshen and L.Wos :"Unit Refutation and Horn Sets" JACM vol.21 no.4 (1974)
(8) 竹内："第２回 LISPコンテスト" 情報処理，vol.20，No.3 (1979)
(9) 山口・西岡・打浪・手塚："拡張された定理証明システムSENRIについて"信学会全国大会予稿 No.41(1979)
(10) 上記同："定理証明システムSENRIのSNL導出への拡張について"信学会全国大会予稿 No.1200(1980)
(11) 上記同："定理証明システムSENRIとその応用"信学技報，EC 80-26 (1980)