言語理解システム開発のための道具立て
── LISPを中心とするファシリティ群 ──

渕　一博　　田中穂積
（電子技術総合研究所）

はじめに　筆者たちの推論機構研究室では、数年前から「言語理解システム」の開発を研究テーマのひとつと考えてきた。これに取組むには、そのための基礎手段の充実がまず第一に必要だと考え、努力してきた。ここでは、そのいくつかを紹介したい。

このテーマは、短期決戦のものでなく、長期を要する大きなテーマである。これにアタックするには、周到な準備、十分な道具立てが必要である。一方、そのような用意をベースにすれば、この分野で前進することを可能とする諸条件はととのってきており、そうすべき時機が来ている、と考えられる。

このテーマを取上げる観点はいろいろあろうが、筆者たちの経歴からすれば、これは、ひとつには、計算機による情報処理の高水準化と二重写しに見えるものである。計算機技術の中核をなす部分 ──（人工）言語処理技術、システム制御（OS）技術、データベース技術等 ── は、このテーマに密接に関連している。これらの技術は、単に手段として利用できるだけでなく、言語情報、知識情報の処理のモデルと本質的にも関連し合っていると考えられる。

計算機技術の全体的な進歩によって十年前とは比較にならないパワーを利用できる時期に来ている。このパワーを十分に活用しなければならない。それとともに、このテーマは、逆に計算機技術そのものへも寄与することになろうと、筆者たちは予想している。それは、ソフトウェア的な面だけでなく、後で触れるように、計算機アーキテクチャへも影響することになろう。

ここに述べるような、計算機的な道具立てが、必要な準備の一部でしかないことはいうまでもない。言語学的、心理学的な成果の吸収は当然必要である。しかし、そのような成果の利用、さらには、その分野の人たちの協力にも、ここに述べるような道具立てが十分になされなければならないと考えられる。

ここで述べるのは、LISPを中心とするファシリティであるが、その前提となるものとして、会話型の共同利用計算機システムの重要性をとくに強調しておきたい。その効果は、体験上、観念的に予想するより、はるかに大きいものである。

LISP　種々のファシリティやシステムを組立てる基本的プログラミング言語として、我々はLISPを採用している。自然言語処理の観点から見たときも、LISPは多くの利点を持っているからである。

まず、言語情報、知識情報のような複雑な構造をもった記号データを処理するには、記号処理、リスト処理の機能をもったプログラミング言語とその処理システムが必要である。LISPはそのようなリスト処理言語として、歴史も古く、もっとも普及しており、標準言語的役割も果している。以下にLISPのいくつかの特徴を見よう。

(i) 再帰的呼出し：言語の統語構成には、再帰的な構造がよく現われる。それに対する処理プログラムも再帰的構造とすることによって、見通しが良くなる。

(ii) プログラムとデータの同一性：プ

ログラムもデータと同じ表現をもち，処理の対象とすることができる。規則や，語の意義，知識項目などについて「手続き的表現」をするときに有用である。

(iii) 記憶の自動管理：記憶スペースの割当てや「ごみ集め」(GC)が自動的に行なわれる。動的に可変な構造をもつデータの処理に必要である。自動的に管理されているため、ユーザ・プログラムで明示しなくてよく、見通しが良くなる。

(iv) 拡張性：さらに高水準のプログラミング言語(たとえば後述の LINGOL CONNIVER, PLANNER 等)を実現するのに、LISP を利用すると比較的容易である。

(v) 関数データ：変数束縛を含んだ関数を，まとめてデータとして扱える。この FUNARG 機能によって，多重処理的機能を持たせることができる。これは言語過程のモデル化に応用されよう。

(vi) 会話型利用：(ii)などの機能のため，インタープリータ(EVAL関数)が組込みになっている。このため会話型言語としても(結果的に)優れた特性を持つ。質問応答システムのような(高水準)会話システムのベースに適している。

(vii) 属性リスト：この機能によって，小規模ならば、辞書を納めることが容易であり，実験に適している。

(viii) 互換性：リスト処理言語としてはもっとも普及しており，プログラムの交換などに便利である。

このような特長から，自然言語研究のベースとして，LISP は現在もっとも適当な言語と考えられる。このことは，その改良や，自然言語処理にもっと適した(問題向，高水準)言語の開発と矛盾するものではない。

電総研の LISP システム　処理の高速化，その他の理由から，スタンフォード大の LISP 1.6[2] をベースに出発した。これは、変数束縛等にスタックを利用することを前提にしている。これに、INTERLISP[3] を検討して，記号列のデータ型と基本関数を追加した。(これによって SNOBOL 的機能を実現することができる。)

その後改良を続けているが，主な点は次の通りである。

(i) 高速，大容量化：使用経験をベースに、主要関数のプログラムの改良等を行ない、高速化をはかった。その結果，高速 LISP のひとつである MAC-LISP と，速度の点でほぼ同等になった(表1)。

その後，スタックの利用方式，ごみ集め(GC)方式の改良を加え，さらに、総合的に2倍程度の性能向上が達成されつつある。

(ii) 処理機能の拡張：MACLISP[4] や INTERLISP とほぼ同等の機能をもつよう，機能の拡張を行っている。たとえば、CATCH, THROW 関数など。

(iii) デバッグ機能の拡張：会話型使用でとくに有効であり，かつ重要なデバッグ機能を強化し，実行中の状態の監視，回復等が便利になっている。

(iv) 入出力機能の強化：ファイル使用等の機能の強化が行なわれている。また、図形端末のためのグラフ基本関数が作製され、図形表示が LISP によって可能となっている。

使用経験から　電総研では，上記の LISP システムの上で，相当規模のプログラムを稼動させている。その経験から，LISP の使い勝手のうち二三を述べる。

(i) 速度：処理システムに改良を加えていった結果，かなりの高速性が得られている。このような応用分野で考えると，他のプログラミング言語(とそ

の処理システム）の使用と比較しても決して遅くないと思われる。これには後述のLINGOLの使用例を参考にして頂きたい。

(iii) 習得性：LISPの習得性については様々なことが言われている。しかし、実際に会話型の環境で使用していると，感じが大いに異なることを強調したい。予備知識のない人でも，一週間も会話的に使用すれば、基本的なことはほとんど習得できるし、評判の悪いカッコもさほど気にならなくなるようである。MILISYJなどの実例による訓練はとくに効果的である。

Sorting of 20 items	INTERLISP				MIT-MACLISP		LISP1.8	LISP1.9
	No spaghetti		With spaghetti					
Machine					PDP-10	MULTICS	T5600/170	
Interpreter (msec.)	3927		4036		1563	1318	609.	408.
Compiled object	No block	Block	No block	Block				
(msec.)	313.	217.	371.	219.	67.	56.	53.	51.
Date	Feb. 1975.				Nov. 1974		Apr 1975	Nov 1975

Table 1. Performance of LISP's.

MILISYJ　このシステムの原型はカーネギーメロン大の moran の作成したMILISYである。これは積木の世界についての簡単な記述を受入れ、記憶する。そして、その状景についての質問に答えることができる。

これは「小さな」質問応答システムであるが，パージング，データベースデータ検索など，質問応答システムに必要な諸要素が一揃い入っている所に特徴がある。自然言語処理の教育用に作られたものであるが，LISPで700行程度のプログラムであり，LISP自身の教材としても適当ということでスタンフォード大などでも使用されている。

MILISYJは，このMILISYを日本語用に改造したものである。位置関係についての推論規則など若干の拡張がなされている。[5)6)7)] 教育実習用に使用している。

語彙数は30語程度，文型は「ある」型の数種である。図1のように，入力文は統語規則と辞書を使用してパーズされる。これは簡単な変形規則によって変形され，「三つ組」の形でデータベースに登録される。質問に対する答は、このデータベースを検索することによって組立てられる。この際いくつかの推論規則が使用される。MILISYJとの会話例を図2に示す。

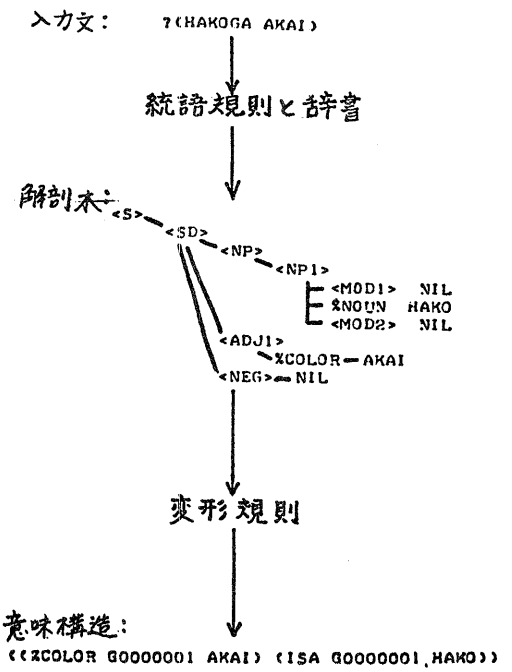

図1. 意味構造の抽出

```
#K1  (DOUZO YOROSHIKU).                                   どうぞよろしく.
     **
#N2  ?(KUROI OOKINA TSUKUENO UENI AKAI CHIISANA HAKOGA ARIMASU)   黒い大きな机の上に
                                                          赤い小さな箱があります.
#K2  (WAKATTAWA)                                          わかったわ.
     **
#N3  ?(AOI MARIGA SONO TSUKUENO UENI ARIMASU)             青いまりがその机の
                                                          上にあります.
#K3  (WAKATTAWA)                                          わかったわ.
     **
#N4  ?(SONO TSUKUEWA OOKII DESU KA)                       その机は大きいですか.
#K4  (HAI)                                                はい.
     **
#N5  ?(SONO TSUKUEWA AKAI DESU KA)                        その机は赤いですか.
#K5  (IIE)                                                いいえ.
     **
#N6  ?(SONO MARIWA CHIISAI DESU KA)                       そのまりは小さいですか.
#K6  (WAKARANAIWA)                                        わからないわ.
     **
#N7  ?(NANIGA SONO TSUKUENO  UENI ARIMASU KA)             何がその机の上に
                                                          ありますか.
#K7  (CHIISAI AKAI HAKO TO AOI MARI DESU)                 小さい赤い
                                                          箱と青いまりです.
     **
#N8  ?(NANIGA SONO MARINO UENI ARIMASU KA)                何がそのまりの上に
                                                          ありますか.
#K8  (NANIMO ARIMASEN)                                    なにもありません.
     **
#N9  ?(NANIGA SONO MARINO SHITANI ARIMASU KA)             何がそのまりの下に
                                                          ありますか.
#K9  (OOKII KUROI TSUKUE DESU)                            大きい黒い机です.
     **
#N10 ?(SONO TSUKUEWA SONO YUKANO UENI ARIMASU KA)         その机はその床の上に
                                                          ありますか.
#K10 (YUKA NANKA NAIWAYO)                                 床なんかないわよ.
     **
#N11 ?(OOKINA YUKAGA SONO TSUKUENO SHITANI ARIMASU)       大きな床がその机の下に
                                                          あります.
#K11 (WAKATTAWA)                                          わかったわ.
     **
#N12 ?(ANO HAKONO SHITANI NANIGA ARIMASU KA)              あの箱の下に何がありま
                                                          すか.
#K12 (OOKII YUKA TO OOKII KUROI TSUKUE DESU)              大きい床と
                                                          大きい黒い机です.
     **
#N13 ?S
#K13 ((UENI G0000002 G0000004) (%SIZE G0000004 OOKII) (ISA G0000004 YUKA) (UE
     NI G0000003 G0000002) (%COLOR G0000003 AOI) (ISA G0000003 MARI) (UENI GO
     000001 G0000002) (%SIZE G0000002 OOKII) (%COLOR G0000002 KUROI) (ISA GOO
     00002 TSUKUE) (%SIZE G0000001 CHIISAI) (%COLOR G0000001 AKAI) (ISA G0000
     001 HAKO))
```

図 2.　MILISYJとの会話例

現在，状景を図形端末に絵として表示するプログラムが作成されている。絵にするためには，常識としていくつかの前提を組込んでおく必要がある。図3にその一例を示す（作図もLISPによって直接行なわれている）。

なお，一つの質問に対して応答する速度は（計算機時間で）0.2～0.5秒程度である。

LINGOL　これはMITのPrattによって作成されたシステム（Linguistics Oriented Programming Language）で，中核に優れたパーザを組込んでいる。[8][9]

言語理解システムの分野では，現在Woodsの方式 Augmented Recursive Transition Network（ATN）をベースにするものが広く使われている。[10][11][12][13]

```
(AKAI HAKOGA TSUMIKINO HIDARINI ARU)
(WAKATTAWA)
(SONO HAKONO SHITANI AKAI TSUKUEGA ARU)
(WAKATTAWA)
(SONO HAKONO HIDARINI SANKAKUGA ARU)
(WAKATTAWA)
(SONO TSUMIKIWA OOKII)
(AH' SOONANO)
(G02WA G04NO UENI ARU)
(OSSYARUTOORINI DEKINAIWA, MOTTO YASASIKU ITTENE)
(G04WA OOKII)
(AH' SOONANO)
(G02WA SONO TSUKUENO UENI ARU)
(WAKATTAWA)
(G02NO HIDARINI AOI HAKOGA ARU)
(WAKATTAWA)
(G06WA G01NO UENI ARIMASU)
(WAKATTAWA)
```

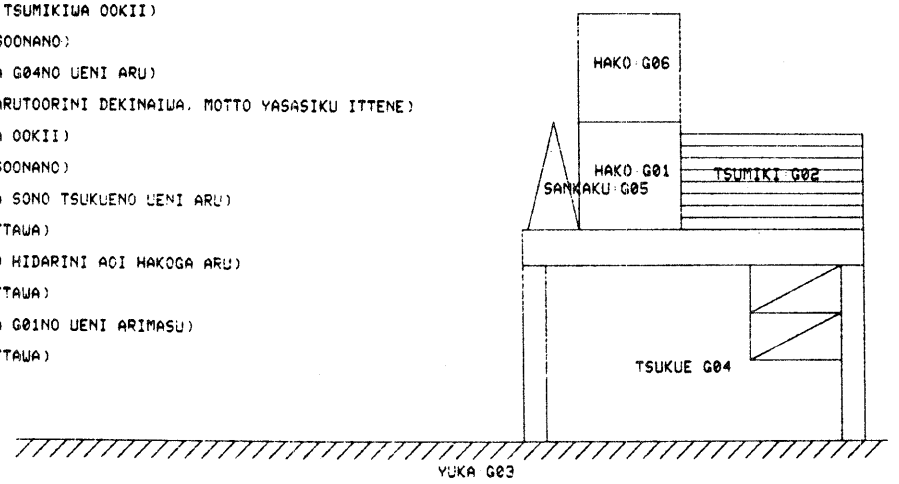

図3. 状景の作図

一方，Prattの方式はAugmented Context Free Grammarということができる。オートマトン理論的には，両者は等価であるが，パーザーの構成や，言語形式に相異がある。

LINGOLのパーザー　Context Free（文脈自由）文法に対する一般的なパージング・アルゴリズムとして，"bottom up"方式では，Cocke-Younger-Kasami方式があり，"top down"方式としてはEearly方式が現在代表的なものとされている。[14] LINGOLのパーザーは前者をベースにし，これに後者を組合わせている。一般的なCFパーザーとしてはもっとも高速な方式であろうと思われる。巧妙なプログラミング技法によって両者の結合が行なわれているために，速度，能力ともに両者の利点が活かされている。それと同時に，後述のような「助言」項が自然に導入されている。"bottom up"をベースにしているので，不要の「予測」は生成しない。一方，すでに生成されたものからの"top down"の予測を利用するので，それ以外の項の生成は抑止される。さらに，「あいまいさ」の処理も正しく行なわれるとともに，bottom upの過程で「もっともらしさ」を計算することができて（助言），全体として，もっともそれらしい木を最初にたどることができるようになっている。

LINGOLの木操作　パージングの結果生成される木は，単なるデータとしての木ではなく，「プログラム」の木となっている。すなわち，解剖木はそれ自身プログラムであり，パージングのあと，このプログラムを実行することによって，木操作が行なわれる。

木構造の表現に，プログラムの木（ネットワーク）構造をそのまま利用しているわけである。このため，文脈的情報を（局所変数等を使って）自然に行なうことができる。この「生成部」の巧みな構成もLINGOLのひとつの特長である。

```
$PICK UP A BLOCK

(ACHIEVE (PICK YOU (CHOOSE BLOCK) (UP)))
(362 )msec. ****(536 )msec by MACLISP.

$WHAT ARE YOU HOLDING?

(QUERY (GRASP (STATIVE (GRASP YOU ((WHAT))))))
(734 )msec. ****(928 )msec by MACLISP.

$WHAT COLOR IS IT?

(QUERY (BE IT (WHAT COLOR)))
(671 )msec. ****(853 )msec by MACLISP

$PUT IT BESIDE THE PYRAMID

(ACHIEVE (PUT YOU IT (BESIDE (DEF PYRAMID))))
(407 )msec. ****(560 )msec by MACLISP

$WHERE IS THE RED BLOCK?

(QUERY (BE (DEF RED BLOCK) ((WHAT LOC))))

(446 )msec. ****(594 )msec by MACLISP.

$WHAT IS IT ON?

(QUERY (BE IT (ON)))
(431 )msec. ****(542 )msec by MACLISP.

$WHAT SHAPE IS THE BIG RED THING?

(QUERY (BE (WHAT SHAPE) (DEF BIG RED)))
(906 )msec. ****(1176 )msec by MACLISP.

$IF YOU PUT ONE BLOCK ON ANOTHER, CAN YOU THEN PICK UP THE LATTER?

(ASSUMING (PUT YOU (CHOOSE BLOCK) (ON (CHOOSE OTHER)))
          (QUERY (POSSIBLE (PICK YOU (DEF LATTER) (UP) NIL))))
(1573. )msec. ****(1759. )msec by MACLISP.

$CAN YOU PUT A BLOCK ON A PYRAMID?

(QUERY (POSSIBLE (PUT YOU (CHOOSE BLOCK) (ON (CHOOSE PYRAMID)))))
(744. )msec. ****(976 )msec by MACLISP.

$HOW MANY BLOCKS CAN YOU PICK UP?

(QUERY (POSSIBLE (PICK YOU (CHOOSE (WHAT NIL) BLOCK) (UP))))
(925. )msec. ****(1114. )msec by MACLISP.

$WHY DID YOU DO THAT?

(QUERY (PAST (DO YOU (DEF) ((WHAT REASON)))))
(555. )msec. ****(703. )msec by MACLISP.

$WHICH IS TALLER, THE RED CUBE OR THE GREEN ONE?

(CHOICE (BE (CHOOSE (RANK NIL 1 )) (WHAT))
        (OR1 (DEF RED CUBE) (DEF GREEN)))
(1773. )msec ****(2244. )msec by MACLISP.
```

図4. SHRDLV

には、LISPのプログラムをそのまま書込むことができる。

LINGOLはLISPの機能を巧みに活している。それとともに、LISPの制御構造にCF規則によって起動されるACTOR[5]-likeな制御構造をつけ加えたものということができ、プログラムのモジュール化に役立っている。LINGOLはその意味でopen endedである。これは文法を書くための言語であって、文法自身ではないことを注意しておこう。

SHRDLV　LINGOLの使用例として、Prattが開発中の英語分析の例を図4に挙げておく。実行時間は電総研LISPで走らせた場合とMACLISPの場合を併記してある。この例は、WinogradのSHRDLUシステムとほぼ同程度のものをLINGOLで書くプロジェクトの一部である。

LINGOLのユティリティ

このシステムには、デバッグ等のために、各種のトレース機能が備えられている他、解剖木の図示ルーチンがある。これは仲々便利である。

LINGOLの形式　文法規則の形式は（<left> <right> <cog> <sem>）である。leftとrightはCF文法規則である。但し、rightは、現在、単項か2項に限られている。COGnition部は「助言」項で、あいまいさのあるときの選択基準を与える。いわゆる「意味素性」による選択操作がここで想定されている。cogを使わなければ単なるCFパーザーとなる。またCF部とcog部の振分けには、ユーザの任意性が入ることを注意しておこう。

SEMantics部は木操作の部分で、いわゆる意味処理の部分をここに埋込むことが想定されている。cog, sem

LINGOLの利用　筆者たちのグループでは、文法記述のシステムとして、このLINGOLを当分の間使用することに決めている。パージング・アルゴリズムが理論的に素性の知れたものであるとともに高速であること、全体で10頁程度のコンパクトなシステムであるとともに機能が豊富で、拡張も容易そうであることによる。

使用経験を通して、日本語用には形態素分析を強化する必要が感じられた。そこで、拡張の一つとして、分ち書き分析にCF的規則を導入する試みを行なった。この拡張はLINGOLに自然に埋込まれ、ほとんど自動的に、

```
HAHANOMURA.
                                              SENTENCE
(421     0 )MS PARSETIME                          !
                                                CNOUN
                                                  !
                         NPOS----------------------------------------CNOUN
                           !                                           !
         MNOUN---------------------------NO                          MNOUN
           !                              !                            !
         NOUN                            NO                          NOUN
           !                              !                            !
         HAHA                                                        MURA
           !                                                           !

STRUCT-NP
     ((HAHA    S0000002))
          FEATURE
               KASAN  NOURYOKU
               YUUSEI
               NINGEN
               ONNA   -KOYUU
          BE
               MOD=  JOUTAI
               IS=  ONNA
          MOD-FOR
               NAME=
                    KOYUU  ONNA
               HAS=
                    KASAN
                    ((MURA    S0000001))
          MOD-BY
               LOCATION=  BASHO
               CHILD=  NINGEN

STRUCT-NP
     ((MURA    S0000001))
          FEATURE
               KASAN  -NOURYOKU
               -YUUSEI
               BASHO
               -KOYUU
          BE
               MOD=  JOUTAI
               IS=  -YUUSEI
          MOD-FOR
               HAS=  NIL
          MOD-BY
               LOCATION=  BASHO
               MEMBER=
                    NINGEN
                    ((HAHA    S0000002))

(890     0 )MILLISECONDS
```

図5　意味構造の抽出例

統語情報を利用した分ち書き分析方式になった。[16]

*sem*部はほとんど*open*なので、ここをどう利用するかを考えなければならない。一例として、語の意味をフレーム構造で表わした分析の試みを行なっている。図5はその一部である。分ち書きを含めて統語分析がなされ、さらに、語の意味関係によって「の」の関係が確定される。

この部分には、言語過程のモデルが必要であるが、そのひとつのベースとして、CONNIVER, PLANNERから適当な機能を移すことを考慮している。

LINGOLの拡張計画のひとつに、文法規則の入力形式の拡張がある。一二項規則だけでなく、多項やオプションを許すなど、コンパクトな見やすい入力形式を組込むことを検討している。

CONNIVER, PLANNER

電総研のLISPの上で、CONNIVER[17] MICRO-PLANNER[18]の言語システムが稼動している。これらの言語には、

(i) パターン合せの機能
(ii) パターンによるデータ検索の機能
(iii) パターンによるサブルーチン呼出しの機能
(iv) データ・ベースのコンテクトと構造（CONNIVER）
(v) やり直し機能（MICRO-PLANNER）
(vi) コルーチン機能（CONNIVER）

などがある。これらは現在別々のシステムとして働いているが、機能を整理して、LINGOLなどと統合し、意味処理の問題に役立てることを検討している。

ALISP　LISPには属性リストがあり小規模なデータベースになっている。将来の研究のためにはもっと大容量なデータベースが必要となることが予想される。機能的にはCONNIVERのそれのようなものが、大容量、高速になることが望まれる。一方，より現実的な観点から研究されているRelational Data Base[19][20][21]などの成果も取入れる必要があろう。これらをLISP（あるいはその拡張システム）と有機的に結合する必要がある。

この方向の手始めとして，電総研では、「連想三つ組」[22]をベースとするシステム（通称ALISP）を試作した。これはLISPから呼出されるディスク上のデータベースシステムであるが、高速化のため（ソフトウェア）ページング方式等を採用している。

語彙情報や，より広く知識情報にはより効果的な表現形式が研究されなければならない。それとともにハードウエア・サポートを含めて，大容量で高速なベースが必要である。

推論機械　これまで紹介したのは，電総研で整備してきたソフトウェア的な道具立てである。言語理解システムあるいはひろく，言語情報，知識情報のためには，将来は，ハードウェア的なサポートも考慮する必要がある。通常の計算機の進歩によって，自動的に高速，大容量，経済的になる面はあるけれども，一方，計算機アーキテクチャを改良することによって性能向上をはかることも考えなければならない。

このための要素技術は，これまでの進歩によってほとんど用意されている。

(i) LSIの進歩
(ii) マイクロプログラム方式の普及
(iii) バーチャルメモリー方式の普及
(iv) CADの進歩，低廉化

等である。一方，これまで開発してきたソフトウェア道具立ての経験から，必要な基本操作が抽出される。これにはMACLISP, INTERLISP, MUDDLE,[23] CONNIVER, PLANNER(PLASMA)[24] 等がベースになろう。

推論機械といっているのは拡張LISPマシンの積りである。これのすべてをハードウェア化する必要のないのは当然のことで，マイクロプログラミング等を適宜採用すべきである。しかし，有効な基本操作はハードウェア化していくことを考えなければならない。原始操作としては，

(i) スタック操作
(ii) タグ(フィールド)操作
(iii) ハッシング操作
(iv) バーチャル化操作

等がある。

最近，このような考えを実現しようという動きが出てきていること[25]は注目されてよい。

謝辞　ここに述べた道具立ての開発には，電総研の多くの人の協力によるものである。とくに，推論機構研究室横井俊夫君はLISP開発，ALISP開発を担当した。CONNIVER, MICRO-PLANNERの稼動は，計算機方式研究室島田俊夫君らによる。また，LISPの改良には，東芝総研黒川利明氏が非常な情熱をもってあたられた。これらの方々，その他の方々に感謝します。

参考文献

1) Norman, Rumelhart: Explorations in Cognition, Freeman. (1975)
2) Quam: Stanford LISP 1.6 Manual, Stanford U., 1970
3) Teitelman: INTERLISP Reference Mannal, XEROX, 1973
4) Moon: MACLISP Reference Manual, MAC-MIT, 1974
5) 田中，他：実験用日本語質問応答システム，情報処理学会大会，1975.11
6) 田中，他：日本語質問応答小規模実験システムについて，信学会オートマトンと言語研究会，1975.11
7) 小さな日本語質問応答システム MILISYJ プログラム説明書，推論機構研究室資料，1975
8) Pratt: a linguistics oriented programming language, 3rd IJCAI, 1973
9) Pratt: LINGOL - progress report, 4th IJCAI, 1975
10) Woods: Augmented Transition Networks for Natural Language Analysis, Harvard U., 1969
11) Bobrow, Fraser: An augmented state transition network procedure, 3rd IJCAI, 1969
12) Kaplan: Augmented transition networks as psycological models for sentence comprehension, AI, 1972
13) 長尾，辻井：自然言語処理のためのプログラミング言語 PLATON，「情報処理」，1974
14) Aho, Ullman: The Theory of Parsing, Translation and Compiling, Prentice-Hall (1972)

15) Hewitt et al : a universal modular ACTOR formalism for artificial intelligence, 3rd IJCAI, 1969
16) 田中：構文情報を用いた自動わかち書きプログラム，情報処理学会大会，1975.11
17) Mc Dermott, Sussman, The CONNIVER Reference Manual, AIL-MIT, 1972
18) Sussman et al : MICRO-PLANNER Reference Manual, AIL-MIT, 1972
19) Codd : Relational model of data for large shared data bases, CACM, 1970
20) Codd : Relational completeness of data base sublanguges, in Data Base Systems, Prentice-Hall (1972)
21) Date : An Introduction to Data Base Systems, Addison-Wesley (1975)
22) Feldman, Rovner : An algol-based associative language, CACM, 1969
23) Pfister : a MUDDLE Primer, MIT, 1972
24) Hewitt, Smith, Towards a programming apprentice, IEEE TRSE, 1975
25) Greenblatt : The LISP Machine, AIL-MIT, 1974