

L^6 についての覚え書き

統計数理研 渋谷 政昭

1)

L^6 はベル研究所の K. C. Knowlton が動画作製用の計算機言語として開発したもので、名前は laboratory's low level linked list language の 6 つの l によっている。この言語の最大の特徴は低水準といえる。 (リスト処理言語ではどのような特徴を高水準と呼ぶべきか必ずしもはっきりしないが。) Knowlton は IBM 7094, GE 635 で L^6 を開発した。日本では HIPAC-103 (統数研), TOSBAC-^{TOPS XII} 3400^{II} (日东科学技術研修所), FACOM 270, 20/30 (東大理, 後藤研) で implement されている。後者 2 つは近日公開されるであろう。

2)

L^6 の 4 つの基本実作とその特徴は次の通りである;

7 個の γ . 大きさが可変であり 2^n 論理語 ($n=0, 1, \dots, 7$) のものを使用できる。

区画. ブロックの、2語にわたる任意の位置に、任意の長さの区画を設け、これに数、文字、等のデータを記憶できる。区画の定義はすべてこのブロックに共通であり、1文字 ($A \sim Z$, $0 \sim 9$) の名前が付けられる。区画の定義は動的に (計算実行中に計算結果に依存して) 行なうこともできる。

指示子. 何らかの方法でブロックを指定するデータ (たとえば「ブロックの先頭の機械語の番地」) を指示子という。指示子は一定長以上の長さの任意の区画に記憶できる。システムより眺めて、指示子を他のデータと区別することはできる。

基地レジスター. 1論理語の大きさの記憶で指示子を格納するが作業番地としても用いられる。使用目的によ、2は出と呼ぶことがふさわしい。これも1字 ($A \sim Z$) の名前をもっている。

連結リストとは指示子により互に結びつけられたブロックと基地レジスターの集合のことである。あるブロックのある区画について演算する (区画の内容をとり出し、それについて演算し、とくに格納する) には、基地レジスターより出発して指示子の入っている区画の名前を順次に羅列することにより必要なブロックにたどりつき、最後に区画の名前を置くことにより指定できる。例. BPPD: 基地レジスター B にある指示子の指示するブロックの、区画 P にある指示子の指示

するブロックの・区分Pにある指示の指示するブロックの・区分D.

区分名を1字に限ることにより, 離れたブロックにある区分を短い語で指示でき, しかもある程度は記憶の便となる.

3)

L⁶の言語の単位は文 (FORTRAN 同様に原則として1行) であり次行形をしている.

a) 無条件文.

<名札> THEN <基本演算> . . . <名札>

<基本演算>の典型的な形は (<区分>, <演算コード>, <区分> または <リテラル>) であるが, 引数の数より多いものもある. 文末に <名札> があれば無条件飛越し, なければ次行の文に移る.

b) 条件文.

<名札> <ifs> <基本判別> . . . THEN <基本演算> . . . <名札>

<基本判別>は<基本演算>と同じ形をしており, 区分の内容を他の区分の内容またはリテラルと比較し, <真>または<偽>の値をとる. <ifs>は IFALL (またはIF), IFANY, IFNONE (またはNOT), IFNALL のいずれかで, これは x_i を基本判別の論理値とすると, それぞれ $\cap x_i$, $\cup x_i$, $\cap \bar{x}_i$, $\cup \bar{x}_i$ を意味する. この5通りの形式のためプロ

グラフは比較的見易い。

4)

自由ブロック・リストからのブロックの獲得 (get) の基本演算は ($\langle \text{区分} \rangle$, GT, $\langle \text{区分} \rangle$ または $\langle \text{整数} \rangle$) である。第3引数の指定する大きさのブロックを取って、それへの指示子を第1引数の区分に入れる。基本演算は ($\langle \text{区分} \rangle$, GT, $\langle \text{区分} \rangle$ または $\langle \text{整数} \rangle$, $\langle \text{区分} \rangle$) と第4引数をもつときには、第1引数の古い内容が第4引数の区分に入れられる。基地レジスタ R より出発して区分の指示子でつながられているブロックの単純リスト (冊) があるとき (R , GT, $\langle \cdot \rangle$, RL) は冊の押し下げとなる。

逆にブロックの自由化 (free) は ($\langle \text{区分} \rangle$, FR, 0) で行なう。第1引数の区分が指示するブロックを自由ブロックに戻し、区分に 0 を入れる。もしも ($\langle \text{区分} \rangle$, FR, $\langle \text{区分} \rangle$) ならば、^{自由化の後}第3引数の区分の内容が第1引数の区分に移される。上の冊の例で (R , FR, RL) は逆に跳ね上げとなる。ブロックの自由化は完全に使用者の責任であり、システムはごみ集めをしない (できずい)。

5)

サブルーティン呼出しは (DO , $\langle \text{名札} \rangle$) とし、基本演算で行なう。この基本演算の次にある基本演算への戻り着地が

システムの持つ棚にのせられ、〈名札〉の^文箱に命令の流れが移る。FORTRAN の RETURN に相当する DONE という特殊の〈名札〉があり、これが文末にあると、棚の一番上の戻り番地を跳ね上げて、そこに命令の流れを移す。カプルーティンは FORTRAN, ALGOL のようにはつくりとした系統ではなく、上のように DO, DONE で制御が行われるだけである。したがって帰納的を吟出しも可能である。

パラメータの授受は使用者の構成する棚によつて行われ、よいが、補助手段としてシステムが区分の内容 (field content) の棚をもっている。そこに載せ (save), 下す (restore) する基本演算は (S, FC, 〈区分〉), (R, FC, 〈区分〉) である。カプルーティンを一般化するためにもう一つ区別の定義の棚もある。

6)

L⁶ は再帰的であるために、諸現象・対象のもっている構造をそのまま記憶で表現できる。たとえば算術式を、節に演算記号をもった2進樹木として表わされる。(LISP などで) はすべてを葉先に移すために余分のブロックを要する。) しかもこの際に若干の考察により記憶を節約し、計算速度を早めるように区別の定義を行ふことができる。

対称リスト、環構造 (ring structure) などにも容易に構成

できるから、非数値アルゴリズムの教育用、実験研究用に、アセンブラー同様に異称に、しかも遙かに容易に使用できる。

システム（コンパイラースおよび実行時カプルーティン等）は比較的単純なため、小さな計算機にたいてい少るゝ労力でプログラミング可能である。

7)

TOSBAC 3400 の L^6 では、実数の使用可能、名札のブロッ
ク化、算術式の扱い、^{アセンブラー言語で書かれたカプルーティンが呼出し}、_、などの機能向上が行われている。

FACOM 270 の L^6 ではブロッックの大きさ、使用・未使用の表示のために1論理語当り1ビットの表で済ます 'tail lamp' 方式を採用している。

8)

これまでに L^6 で試験的に行なった課題としては 1. 迷路探索、2. ランダム文字の生成、3. 名札ブロッック処理、4. PURE LISP の EVALQUOTE、5. 凸多面体の射影、6. 多角形の包含関係判別、などがある。

9)

参考文献

Knowlton, K.C. (1966) : A programmer's description of L^6 , Comm. ACM, 9, 616-625.

高橋・原・徳石 (1969) : リスト処理言語 L^6 について,

第10回プログラミング・シンポジウム。

日科技術計算センター (1970) : T-L⁶。使用者のための
便覧。

—— (1970) : T-L⁶ によるリスト処理入門。