第14卷 第3期 华侨大学学报自然科学版 Vol. 14 №. 3
JOURNAL OF HUAQIAO UNIVERSITY
1993年7月 (NATURAL SCIENCE) Jul. 1993

# 拓广 AUTOLISP 语言集方法及数组运算解释函数设计

侯 济 恭
〔计算机科学(电脑)系〕

**摘要** 利用 AUTOLISP 的潜在功能可拓广其函数集,不必涉及其存贮分配. 文中介绍 AUTOLISP 的潜在函数功能,讨论数组运算解释函数的设计方法.

**关键词** AUTOLISP,语言集,解释程序

## 0 引言

AUTOLISP 是开发 AUTOCAD(美国 AUTODESK 产品)的有力工具,但不具备一般程序设计语言应有的数组运算能力(如矩阵的运算),因而削弱其产生复杂曲面的能力. 幸好,AUTOLISP 潜在类似于 C 语言中的指针型变量,又具有独特字符串转变为变量名的能力. 利用这些特点以及其它基本函数,便可使 AUTOLISP 进行含有数组的运算. 根据开发 AUTOCAD 的经验,本文介绍 AUTOLIP 类指针型变量的设计、应用的基本方法,讨论数组说明,以及数组无素(下标变量)解释程序的设计方法. 最后,给出一个实用的数组变量解释程序. 为方便讨论,文中的大写字母加数字称下标变量名,如 A5,A235等,小写字母加数字称数组元素,如 $a_i$,$a_{i,j}$等.

## 1 基本函数

本节讨论构造数组解释程序的基本函数功能及其应用方法.

### 1.1 类指针型变量

利用 set 和 setq 函数,可实现如 C 语言中指针型的赋值运算

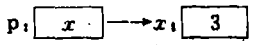
(setq p '$x$);将符号 $x$ 送入 p

(set p 3);将3存入 $x$

本文1992-05-07收到.

© 1994-2010 China Academic Journal Electronic Publishing House. All rights reserved. http://www.c

从 AUTOLISP 角度描述，p 指向表 $x$；从 C 语言角度看，p 是一指针，指向变量 $x$ 的存贮空间. 本程序段执行后，p 和 $x$ 的关系图是

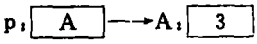



### 1.2 字符串转化为变量名

利用 read 函数，可将一字母字符串转化成内存变量名

```
(setq p (read "A"))
(set p 3)
```

前一语句将字符"A"转变为变量 $A$，再将 $A$ 存入 p，后一语句将3存入 p 所指之变量 $A$ 中. 执行后变量间的关系图是

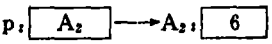



函数 itoa 可将一数值转化为字符串，函数 stract 可连接两个字符串(stract "A" (itoa 2))返回值为"A2"

将以上函数组合，可产生一下标变量名. 对该变量名赋值

```
(setq i 2)
(setq p (read (stract "A" (itoa i ))))
(set p 6)
```

执行后变量间的关系图是




### 1.3 求值函数

eval 可求出变量 $x$ 的数值，不论 $x$ 是否是指针型变量，利用此函数，可进行下标变量之间的相互赋值.

```
1: (setq i 2 j 3)
2: (setq x (read (stract "A" (itoa i))))
3: (setq y (read (stract "A" (itoa j))))
4: (set x (eval y))
```

本程序段相当于如下的 C 语言程序段

```
1: i=2; j=3; /* 说明 */
2: x=&a[i]; /* x is a pointer of integer */
3: y=&a[j]; /* y is the same as x */
4: *x=*y ; /* ai=aj */
```

本程序段相当于执行 $a_2=a_3$. 利用 eval 函数，还可产生数组维数表或下标表的赋值运算

```
(setq x' (10 20 30 ))
(setq table (read "x"))
(setq table (eval table))
```

执行第二条语句后，table 是指针变量，指向表 $x$，执行第三条语句后，table=(10 20 30).

### 1.4 表构造函数

利用表构造函数 cons，可产生数组的维数表或下标表

```
(setq table (cons '10 table))
(setq table (cons '20 table))
```

© 1994-2010 China Academic Journal Electronic Publishing House. All rights reserved. http://www.c

若 table 初值为 nil，则执行结果 table＝(20 10).

## 2　基本设计方法

本节讨论一维、二维数组元素的翻译方法，以及含有数组运算的程序设计基本方法.

### 2.1　一维数组下标翻译

在本解释程序中，下标变量名定义为：(2－1)　字母串‖数字串. 其中，字母串为一数组名. 例如，数组 $A$ 的元素 $a_1, a_2, \cdots a_{10}$ 在 AUTOLISP 中表示为 $A1, A2, \cdots A10$. 数组元素 $a_i$ 的下标变量名翻译可直接运用上节所介绍的方法，其解释函数如下

```
(defun arrayl(x i)
(read (strcat x (itoa i)))
)
```

其中，函数参数 $x$ 是字母字符串即数字型变量，$i$ 是整型变量. 本函数完成公式(2－1)的翻译工作，返回一个下标变量名. 其调用格式是

(arrayl　“数组名”　下标)

例1　$a_4$＝7的程序段如下

```
(setq i 4)
(setq y (array "A" i))
(set y 7)
```

例2　设 $A$＝(10,11,12,…,19,20)，求 $\sum_{i=1}^{10} a_i$，其程序编制如下.

```
1 ; * (defun c:sum( )
2 ;        ;the first part: set array 10 initial values.
3 ;        (setq x 1)
4 ;(while (<= x 10)
5 ;        (setq y (arrayl "a" x))
6 ;        (set y (+ x 10))
7 ;        (setq x (1+ x ))
8 ;)
9 ;;The scond part:get the sum.
10;(setq x 1 xum 0)
11;(while (<= x 10)
12;        (setq y (arrayl "a" x))
13;        (setq sum (+ sum (eval y)))
14;        (setq x (1+ x))
15;)
16;(foreach n '("\n The total is:" sum )(princ(eval n) ))
17;(print)
18;)
```

现作程序简要说明. 行3－8，为设置数组 $A$ 的初值. 行5取数组元素 $a_i$ 之下标变量名，行6执行

© 1994-2010 China Academic Journal Electronic Publishing House. All rights reserved.　http://www.c

$a_i=i+10$. 行10－15，求$\sum_{i=1}^{10}a_i$. 行12取数组元素 $a_i$ 之下标变量名，行13执行 sum＝sum＋$a_i$. 行16－17：打印结果.

## 2.2 二维数组下标翻译

二维数组元素 $a_{i,j}$ 的下标变量名翻译受其数组大小的影响. 下标的组织方式可按以行/列为序，即令数组 $A(d_1,d_2)$ 是 $d_1\times d_2$ 矩阵，以行为序计算，则 $a_{i,j}$ 的下标

(2－2) location＝$(i-1)d_2+j$.

其下标变量名

(2－3) "数组名"‖ location.

例如，令 $d_1=d_2=20$，则 $a_{12,15}$ 的下标是 location＝(12－1)×20]15＝235. 而 $a_{15,12}$ 的下标变量名为 A292.

二维数组元素的下标名解释程序如下

```
42: (defun array2( x i j / location)
43:       (setq location (+ ( * (1- i) dim ) j))
44:       (read (strcat x (itoa location) ))
45: )
```

本解释函数调用格式为：(array "数组名" 行下标 列下标).

调用本函数前，必须对数组的 dim 预先说明，即将 $d_2$ 值存入 dim 中. 本函数语句行43完成公式(2－2)的计算工作，行44完成公式(2－3)的转换工作，返回 $a_{i,j}$ 的下标变量名. 如 $a_{3,5}=7$ 的程序编制为

```
(setq i 3 j 5)
(setq y (array2 "a" i j ))
(set y 7)
```

又如，求矩阵 $A$ 的转置矩阵 $A'$. 转置部份程序编制如下

```
15:(setq i l)
16:(while (<= i dim)
17:       (setq j i )
18:(while (<= j dim)
19:       (setq y1 (array2 "a" i j))
20:       (setq y2 (array2 "a" j i ))
21:       (setq x (eval y1))
22:       (set y1 (eval y2))
23:       (set y2 x)
24:       (setq j (1+ j))
25:)
26:(setq i (1+ i))
27:)
```

下予程序说明. 行19，调解释函数 array2，取 $a_{i,j}$ 的下标变量名，并存入 y1. 行20，取 $a_{j,i}$ 的下标变量名，并存入 y2. 行21，取 $a_{i,j}$ 之值，存入临时变量 $x$。行22，取 $a_{j,i}$ 之值，存入 y1所指之变量 $a_{i,j}$. 行23，将 $x$ 值存入 y2所指之变量 $a_{j,i}$.

© 1994-2010 China Academic Journal Electronic Publishing House. All rights reserved. http://www.c

以上程序段(19—23行)执行 $a_{i,j}$和 $a_{j,i}$内容的互换操作，相应的 C 语言程序是

```
19: yl=&a[i][j] ;/* ai,j 地址送 yl */
20: y2=&a[j][i] ;/* aj,i地址送 y2 */
21: x=*yl ; /* ai,j值送 x */
22: *yl=*y2 ; /* aj,i值送 ai,j */
23: *y2=x; /*x 值送 aj,i */
```

运行本例程序，应预先将 dim 置为列最大值. 本程序段可作任何 $d_2\times d_2$的矩阵的转置运算.

## 3　任意维数组运算解释程序设计

对于多维数组下标变量名的翻译，必须考虑两个因素：(1)数组的内情向量表. (2)数组元素的翻译方法. 在一般编译原理[1]或数据结构中[2]对数组元素的下标计算方法都有详细描述，本设计对其一般公式加以化简，即设数组 $A(d_1,d_2,\cdots,d_n)$是 $n$ 维数组，$d_i$ 为各维尺寸说明，各维的下标均从1开始. 则数组元素淳 $a_{i1,i2,\cdots,in}$的下标计算公式

$$(i_1-1)d_2d_3\cdots d_n+(i_2-1)d_3\cdots d_n+\cdots+(i_n-1-1)d_n+i_n.$$

化简后得，

$$VAR=i_1d_2d_3\cdots d_n+i_2d_3\cdots d_n+\cdots+i_n-1d_n+i_n,\ constant=d_2d_3\cdots d_n+d_3\cdots d_n++d_n$$

其迭代公式

(3—1)　$VAR=(\cdots((i_1d_2+i_2)d_3+i_3)\cdots+i_{n-1})d_n+i_n$

(3—2)　$constant=(\cdots((d_2+1)d_3+1)d_4\cdots+1)d_n.$

下标变量名由数组名，VAR 与 constant 之差转为字符串后，连接成

(3—3)　“数组名”‖(VAR—constant).

现在考虑内情向量表的构成. 内情向量表只须考虑 constant 部份的保存(以减少计算量)以及维数表的保存，因此定义：数组变量名：寄存常数部份 onstant；第0下标变量名：寄存维数说明表. 由此，可得数组 $A$ 的逻辑内存分配表如图1. 例如，设 $M(10,20,30)$是10×20×30数组，计算 $m_{2,3,5}$的下标变量名，constant=(20+1)30=630，VAR=((2×20+3)30+5=1295. 所以，$m_{2,3,5}$的下标变量名是 M665. 相应的内存分配逻辑图见图2.

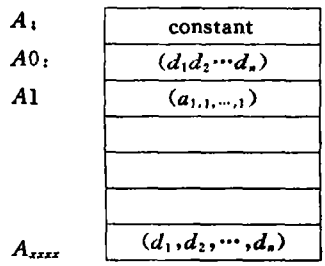



图1　A 的逻辑内存表

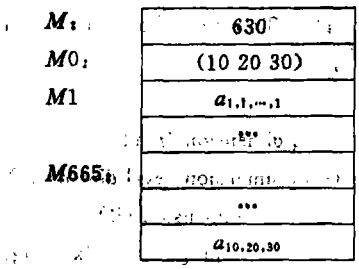



图2　M 的内存图

对数组的翻译，分为数组说明处理和数组元素翻译二部份：数组说明解释程序(即维数说明解释函数)示下. 本解释函数的调用格式：(dim　“数组名”　维数说明表). 设数组说明为 A

© 1994-2010 China Academic Journal Electronic Publishing House. All rights reserved.　http://www.c

$(d_1,d_2,\cdots d_n)$，调用本函数格式为(dim "A" '$(d_1d_2\cdots d_n)$)，执行过程解说如后.

```
47: (;
48: ;| —name: The name of a matrix.
49: ;| —table: The table of the matrix's dimension
50: ;| output value
51: ;| the matrix's name: The constant part of a element's location.
52: ;| the matrix's name0: The table of the matrix's dimension.
53: ;
54: defun dim (—name table / tab sum d)
55:     (setq tab (cdr —table))
56:     (setq sum (car tab))
57: (while (/= (setq tab (cdr tab)) nil)
58:             (setq d (car tab))
59:             (setq sum (* (1+ sum) d))
60: )
61: (setq tab (read (strcat —name "0")))
62: (set tab —table)
63: (setq tab (read —name))
64: (set tab (if (= sum nil) 0 sum)
)
```

行55，去掉维数说明表—table 之 $d_1$后存入维数子表 tab，即 tab=$(d_2d_3\cdots d_n)$. 行56，将 $d_2$值送 sum，为计算 constant 作准备. 行57－60，计算公式(3－2)，得 constant 于 sum. 行61－62，将维数表$(d_1d_2\cdots d_n)$存入 A0. 行63－64，将常数 constant 存入 A，若 A 为一维数组，则将变量 A 清 0.

下面为数组元素的下标变量名翻译程序.

```
4:
5: (defun array(—name —table / dimension row dx var)
6:     ;
7:     ;| —name: a matrix's name
8:     ;| —table: a table of the matrix's subscription.
9:     ;|   To return the location of the element.
10:  ;
11:
12:  (setq dimension (read (strcat —name "0")))
13:  (setq dimension (eval dimension))
14:  (if (=  dimension nil)
15:           (progn (setq dx (length —table))
16:           (while (/= dx 0)
17:           (setq dimension (cons '10 dimension))
18:           (setq dx (1-dx))
19:           )
```

© 1994-2010 China Academic Journal Electronic Publishing House. All rights reserved. http://www.c

```
20:             (dim—name dimension)
21:             )
22:     )
23:     (if (/= (length —table) (length dimension)) (mistake 1))
24:     (setq var (car —table))
25:     (if (/= (length dimension) 1)
26:             (progn
27:             (setq row var)
28:             (setq dx (car dimension))
29:             (while (/= (setq dimension (cdr dimension)) nil)
30:             (if (< dx row) (mistake 2)
31:                     (progn
32:                     (setq dx (car dimension))
33:                     (setq var ( * var dx))
34:                     (setq —table (cdr —table))
35:                     (setq row (car —table))
36:                     (setq var (+ var row))
37:                     )
38:                     )
39:                     )
40:                     )
41:                     )
42:                     (setq dx (read —name))
43:                     (setq var (— var (eval dx)))
44:                     (read (strcat —name (itoa var)))
45: )
46:
```

其调用格式是(array "数组名" 下标表). 例如，数组 $M$ 的元素 $m_{2,3,5}$的下标变量名，(setq y (array "M" '(2 3 5))，返回值为 $y=M665$.

设数组 $A$ 的元素 $a_{i1,i2,\cdots,in}$，语句为(array "A" (LIST $i1$ $i2$ $\cdots i_n$)). 程序3中，行12－13，取 $A0$值(即 $A$ 之维数表)存入局部量 dimension. 行14－22，若数组 $A$ 未预先说明(即未先调用 dim 函数)，则行15－19，求下标表长度(设为 $n$)，据此生成 $n$ 维表(10 10 …10)，行20，调用 dim 函数，代为生成内情向量表. 行23，判断维数表和下标表长度是否相同，即判断下标表的合法性. 行24，取 $i_1$送 VAR，为计算公式(3－1)作准备. 行25－41，计算公式(3－1)，结果存入 VAR. 其中，语句行30判断下标 $i_k$ 是否大于 $d_k$，即判断下标的合法性，语句行32－37计算选代公式 $i_{k-1} * d_k + i_k$. 行42－43，计算 VAR－constant. 行44，转换公式(3－3)，返回 $a_{i1,\cdots,in}$的下标变量名. 程序中的 mistake 函数是错误处理程序，显示用户错误类型后返回命令(command)状态. 程序十分简单.

## 4　应用实例

仍以求矩阵 $A$ 的转置阵为例，验证一下本解释的正确性，转置矩阵程序为

© 1994-2010 China Academic Journal Electronic Publishing House. All rights reserved.　http://www.c

```
1 : (defun c:trans()
2 :     (setq f (open "trans.dat" "w"))
3 :     (setq size (getint"\n Enter the dimension:"))
4 :     (printf "The array original values are: \n")
5 :     (dim "a" (lust size size))
6 :     (setq i 1 k 1)
7 :     (while (<= i size)
8 :             (setq j 1)
9 :             (while (<= j size)
10:                     (setq y (array "a" (list i j)))
11:                     (set y k )
12:                     (printf y)
13:*                    (setq j (1+ j) k (1+ k))
14:             )
15:             (printf "\n")
16:             (setq i (1+ i))
17:)
18:(setq i 1)
19:(while       (<= i size)
20:             (setq j i)
21:             (while (<= j size)
22:                     (setq y1 (array "a" (list i j)))
23:                     (setq y2 (array "a" (list j i)))
30:;ouput the resrut\
31: (printf "The matrix of transfer \n")
32:
33:(setq i 1)
34:(while (<= i size)
35:             (setq j 1)
36:             (while (<=j size)
37:                     (setq y (array "a" (list i j)))
38:                     (printf y)
39:                     (setq j (1+ j))
40:             )
41:             (printf "\n")
42:             (setq i (1+ i))
43:)
44:               (close f)
45: )
46:
47: (defun printf(y)
48:             (foreach n '((eval y) " ")(princ (eval n) f))
```

© 1994-2010 China Academic Journal Electronic Publishing House. All rights reserved. http://www.c

```
49;)
```

下面作程序简要说明. 行2-5,初始化准备,读入方阵 $A$ 的维数说明 sine,调用 dim 函数生成内情向量表. 6号行6－17,生成矩阵 $A$ 的初值,以行为序,将1－sine²存入 $a_{i,j}$. 行18－29,转置该矩阵,即令 $a_{i,j}$ 与 $a_{i,j}$ 内容互换. 行33－34,打印结果,设输入 *sine* 值为3,则运行结果为

```
The array original values are
  1  2  3
  4  5  6
  7  8  9
Thematix of transfer
  1  4  7
  2  5  8
  3  6  9
```

## 5　结束语

本文所有程序均在 IBM PC/AT 机上调试成功,运行结果满意. 若将 dim 和 array 函数加入 ACAD. LSP 中,可使用户程序能方便调用,且 AUTOLISP 便具有数 组运算能力. 所供方法,可自行设计数组间各种运算解释程序,如数组的整体赋值,求逆等,还可进行指针型运算和结构数组运算. 本解释程序不仅拓广了 AUTOLISP 的语言集,而且为进一步拓广其语言集打下了基础,具有广泛的实用价值.

## 参　考　文　献

[1] 陈火旺等编,程序设计语言编译原理,国防工业出版社,(1984),14－16.
[2] 王广芳等编,数据结构,湖南科学技术出版社,(1984),13－15.
[3] 周克绳等编,AUTOCAD 计算机绘图软件,国防工业出版社,(1989),338－391.
[4] 李卫华等编,宏 LISP 语言人工智能程序设计,湖南科学技术出版社,(1986),96－98.
[5] 黄昌宁等译,LISP 程序设计,清华大学出版社,(1983).

# Method for Expanding AUTOLISP Language Set and Design of Analytic Function for Array Operation

Hou Jigong
(*Department of Computer Science*)

**Abstract**　The AUTOLISP language set can be expanded without touch upon its memory allocation by making use of the latent function of AUTOLISP. In this paper, the author briefs its latent function; and discusses the design of analytic fucnction for array operation.

**Key words**　AUTOLISP, amgisge set, interpreter

© 1994-2010 China Academic Journal Electronic Publishing House. All rights reserved.　http://www.c