关于横向程序变换的若干研究*

蔡经球

(厦门大学人工智能与计算机研究所)

RESEARTH ON PROGRAM TRANSFOR—MATION ON THE SAME LEVEL

Cai Jing-qiu

(*Xiamen University*)

ABSTRACT

In this paper, a series of work and results of us in the field of equivalent transformations of recursive programs are sorted and summaried. Some new equivalent transformations and techniques presented in our paper expand the knowledge on Program Transformation on the same level.

摘 要

本文对我们近几年来在递归程序等价变换方面之一系列研究工作[12-18]做了分类总结，所提出的一些新变换模式和技术进一步丰富了人们对横向程序变换技术的认识。

软件自动化是计算机科学的前沿课题之一[1]，其主要研究途径有：演绎综合途径、程序变换途径、实例推广途径和过程化途径，其中尤以程序变换途径得到研究工作者更多的关注[1]～[12]。程序变换大体上可分为横向变换和

* 1989 年 7 月 25 日收到，1989 年 10 月 15 日定稿，本研究课题得到国家自然科学基金资助。

© 中国科学院软件研究所 http://www.jos.org.cn

纵向变换两大类，所谓横向变换是指在相似的抽象级上将一个语言成分转化为另一个与之等价但效率更高的语言成分[2]。

横向程序变换的研究可追溯到1966年Cooper首次提出的Cooper变换[6]，而英国爱丁堡大学的Burstall和Darlington的研究以及他们所研制的ZAP系统是目前这方面最有代表性的工作[7]。

本文对横向程序变换作了较深入的探讨，其中有相当多一部分递归程序等价变换模式是笔者及其同事们近几年来陆续提出的[12]～[18]。

我们将采用函数式递归程序模型作为讨论的基础(参阅[12]之第七章)，并把递归程序变换模式分成若干类型加以阐述，限于篇幅，我们只简要叙述各类变换的输入模式、输出模式、可用性条件和应用示例，有关变换的正确性证明(采用结构归纳法)将不在此列出(感兴趣的读者可参阅有关文献)。

§1. 一阶递归程序的等价变换

以下讨论的递归函数只含有一个递归分支，称为一阶递归程序。

1.1 一阶Cooper变换(A型) [6][13][15]

这相当于1966年Cooper提出的Cooper变换之原型。

输入模式:

$$f(x) \equiv \text{if } b(x) \text{ then } h(x) \text{ else } F(f(k(x)), g(x))$$

输出模式:

$$f(x) \equiv G(x, e)$$
$$G(x, y) \equiv \text{if } b(x) \text{ then } F(h(x), y) \text{ else } G(k(x), F(g(x), y))$$

可用性条件:

①F满足结合律: $F(F(x, y), z) = F(x, F(y, z))$

②F有右单位元e，即 $F(x, e) = x$

③b, h, g, k 中均不含 f

应用示例:

[例1] 阶乘函数 FAC(N)=N! (N为自然数)

$$FAC(N) \equiv if\ N = 1\ then\ 1\ else\ FAC(N-1) * N$$
$$\Longrightarrow FAC(N) \equiv G(N, 1)$$
$$G(N, y) \equiv if\ N = 1\ then\ y\ else\ G(N-1, N * y)$$

© 中国科学院软件研究所 http://www.jos.org.cn

[例 2] 倒排表 L 的函数 $REV(L)$　　(L 为单表[1])

$$REV(L) \equiv if\ L = NIL\ then\ NIL\ else$$
$$APPEND(REV(CDR(L)), LIST(CAR(L)))$$
$$\Longrightarrow REV(L) \equiv G(L, NIL)$$
$$G(L, R) \equiv if\ L = NIL\ then\ R\ else$$
$$G(CDR(L), APPEND(LIST(CAR(L)), R))$$

Cooper 变换的基本思想是引入一个新变元，以便改变原递归算法的运算结构，从而把一类非尾递归型程序归结为等价的尾递归型程序，而后者直接对应于选代程序，具有较高的时空效率。

1.2　一阶 Cooper 变换 (B 型) [15][16]

上述 *Cooper* 变换的原型并不适合于处理某些类型与表处理有关的算法，为此我们提出如下的一阶 *Cooper* 变换 (B 型)

输入模式:

$$f(x) \equiv if\ b(x)\ then\ h(x)\ else\ F(g(x), f(k(x)))$$

输出模式:

$$f(x) \equiv G(e, x)$$
$$G(y, x) \equiv if\ b(x)\ then\ F(y, h(x))\ else\ G(F(y, g(x)), k(x))$$

可用性条件:

(1) F 满足结合律

(2) F 具有左单位元 e, 即 $F(e, x) = x$

(3) b, h, g, k 中均不含 f

应用示例:

[例 3] 插入函数 $MERGE(X, L)$　　(X: 原子，L: 单表)

把原子 X 插入到已按序排列的单表 L 之适当位置，使新组成的单表仍按原序排列。

$$MERGB(X, L) \equiv if\ L = NIL\ then\ LIST(X)\ else$$
$$if\ x \geq CAR(L)\ then\ APPEND(LIST(X), L)\ else$$
$$APPEND(LIST(CAR(L)), MERGE(X, CDR(L)))$$
$$\Longrightarrow MERGE(X, L) \equiv G(NIL, X, L)$$
$$G(R, X, L) \equiv if\ L = NIL\ then\ APPEND(R, LIST(X))\ else$$
$$if x \geq CAR(L)\ then\ APPEND(R, APPEND(LIST(X), L)))$$
$$else\ G(APPEND(R, LIST(CAR(L))), X, CDR(L))$$

[1] 单表是指表元素均为原子的表，如 (A, B, C)。

© 中国科学院软件研究所 http://www.jos.org.cn

1.3 CZ变换[16]

有些情况下，*Cooper*变换的可用性条件①即F满足结合律不能成立，但F可具有另外的性质，为此我们提出如下的CZ变换。

输入模式：

$$f(x) \equiv \textit{if } b(x) \textit{ then } h(x) \textit{ else } F(g(x), f(k(x)))$$

输出模式：

$$f(x) \equiv G(x, h(x0)) \quad \text{其中}x0\text{满足}b(x0)\text{为}true$$

$$G(x,y) \equiv \textit{if } b(x) \textit{ then } y \textit{ else } G(k(x), F(G(x), y))$$

可用性条件：

①F满足如下性质：$F(x, F(y,z)) = F(y, F(x,z))$

②b, h, g, k中均不含f

[例4] 排序函数$SORT$ (L)　　(L: 单表)

$$SORT(L) \equiv \textit{if } L = NIL \textit{ then } NIL \textit{ else } MERGE(CAR(L), SORT(CDR(L)))$$

$$\Longrightarrow SORT(L) \equiv G(L, NIL)$$

$$G(L,R) \equiv \textit{if } L = NIL \textit{ then } R \textit{ else } G(CDR(L), MERGE(CAR(L), R))$$

1.4 函数反演变换[10] [12]

在*Cooper*变换的输入模式中，若函数$k(x)$具有反函数$k^{-}(x)$，则可使用下述函数反演变换。

输入模式：

$$f(x) \equiv \textit{if } b(x) \textit{ then } h(x) \textit{ else } F(f(k(x)), g(x))$$

输出模式：

$$f(x) \equiv G(x, x0, h(x0))$$

$$G(x,y,z) \equiv \textit{if } y = x \textit{ then } z \textit{ else } G(x, k^{-1}(y), F(z, g(k^{-1}(y))))$$

可用性条件：

①$b(x0)$为true

②$k(x)$存在反函数$k^{-1}(x)$

③b, h, k, g中均不含f

应用示例：

[例5] 阶乘函数 $FAC(N) = N!$　　(N: 自然数)

$$FAC(N) \equiv \textit{if } N = 1 \textit{ then } 1 \textit{ else } FAC(N-1) * N$$

$$\Longrightarrow FAC(N) \equiv G(N, 1, 1)$$

$$G(x,y,z) \equiv \textit{if } y = x \textit{ then } z \textit{ else } G(x, y+1, z*(y+1))$$

© 中国科学院软件研究所 http://www.jos.org.cn

1.5 Cooper 变换的改进型[13]

Cooper 变换(A 型)的可用性条件②即F具有在单位元e,这是为了保证在一开始情况时$b(x)$就为真时变换的正确性，因此只要将模式多加一个分支，将上述情况分离另作处理，则可用性条件②实际上可略去，于是我们提出*Cooper*变换(A 型)的改进型。

输出模式:

$$f(x) \equiv \mathit{if}\ b(x)\ \mathit{then}\ h(x)\ \mathit{else}\ G(k(x), g(x))$$
$$G(x,y) \equiv \mathit{if}\ b(x)\ \mathit{then}\ F(h(x), y)\ \mathit{else}\ G(k(x), F(g(x), y))$$

可用性条件:

①F 满足结合律

②b, h, k, g 中均不含f

[例7] 阶乘函数 $FAC(N) = N!$　　(N: 自然数)

$$\Longrightarrow FAC(N) \equiv \mathit{if}\ N = 1\ \mathit{then}\ 1\ \mathit{else}\ G(N-1, N)$$
$$G(N,y) \equiv \mathit{if}\ N = 1\ \mathit{then}\ y\ \mathit{else}\ G(N-1, N*Y)$$

§2. 二阶递归程序的等价变换

以下讨论的递归函数含有两个递归分支，称为二阶递归程序。由于两个递归分支的相互影响，其变换将出现较复杂的情况。

2.1 二阶Cooper 变换(A 型) [15]

输入模式:

$$f(x) \equiv \mathit{if}\ b(x)\ \mathit{then}\ h(x)\ \mathit{else}$$
$$\mathit{if}\ b1(x)\ \mathit{then}\ F1(f(k1(x)), g1(x))$$
$$\mathit{else}\ F2(f(k2(x)), g2(x))$$

输出模式:

$$f(x) \equiv G(x, e)$$
$$G(x,y) \equiv \mathit{if}\ b(x)\ \mathit{then}\ F1(h(x), y)\ \mathit{else}$$
$$\mathit{if}\ b1(x)\ \mathit{then}\ G(k1(x), F1(g1(x), y))$$
$$\mathit{else}\ G(k2(x), F2(g2(x), y))$$

可用性条件:

①F1 满足结合律，　　即 $F1(F1(x,y),z) = F1(x, F1(y,z))$

②F1 关于F2 满足结合律，即 $F1(F2(x,y),z) = F1(x, F2(y,z))$

③F1 具有右单位元e，　　即 $F1(x,e) = x$

④b, b1, h, k1, k2, g1, g2, 中均不含f.

应用示例:

© 中国科学院软件研究所 http://www.jos.org.cn

[例8] 函数 $E(x1,x2)=x1**x2$　($x1$: 自然数, $x2$ 非负整数)

$$E(x1,x2)\equiv if\ x2=0\ then\ 1\ else$$
$$if\ even\ (x2)\ then\ E(x1*x1,x2/2)$$
$$else\ E(x1,x2-1)*x1$$

可把第一个递归分支当作 $E(x1*x1,x2/2)*1$

$$\Longrightarrow E(x1,x2)\equiv G(x1,x2,1)$$
$$G(x1,x2,y)\equiv if\ x2=0\ then\ y\ else$$
$$if\ even\ (x2)\ then\ G(x1*x1,x2/2,y)$$
$$else\ G(x1,x2-1,x1*y)$$

2.2　二阶 Cooper 变换 (B 型) [15]

输入模式:

$$f(x)\equiv if\ b(x)\ then\ h(x)\ else$$
$$if\ b1(x)\ then\ F1(g1(x),f(k1(x)))$$
$$else\ F2(g2(x),f(k2(x)))$$

输出模式:

$$f(x)\equiv G(e,x)$$
$$G(y,x)\equiv if\ b(x)\ then\ F1(y,h(x))\ else$$
$$if\ b1(x)\ then\ G(F1(y,g1(x)),k1(x))$$
$$else\ G(F2(y,g2(x1),k2(x1))$$

可用性条件: (除③外同2.1)

③F1 具有左单位元e,　　即 $F1(e,x)=x$

应用示例: 当F1=F2=F 时有较多的应用

[例9] 交函数 $INT(L1,L2)$　　($L1,L2$: 单表)

$$INT(L1,L2)\equiv if\ L1=NIL\ then\ NIL\ else$$
$$if\ MEMBER(CAR(L1),L2)$$
$$then\ APPEND(LIST(CAR(L1),INT(CDR(L1),L2))$$
$$else\ INT(CDR(L1),L2)$$

把第一个递归分支当作 $APPEND(NIL,INT(CDR(L1),L2))$

$$\Longrightarrow INT(L1,L2)\equiv G(NIL,L1,L2)$$
$$G(R,L1,L2)\equiv if\ L1=NIL\ then\ R\ else$$
$$if\ MEMBER(CAD(L1),L2)$$
$$then\ G(APPEND(R,LIST(CAR(L1))),CDR(L1),L2)$$
$$else\ G(R,CDR(L1),L2)$$

© 中国科学院软件研究所 http://www.jos.org.cn

[例 10] 替代函数 $SUBST(X,L,A)$　　(X, A: 原子, L: 单表)

要求以 A 替代单表 L 中 X 的任何出现，而表 L 的其余元素保持原样，由此形成一新表。

$$SUBST(X,L,A) \equiv if\ L = NIL\ then\ NIL\ else$$
$$if\ CAR(L) = X$$
$$then\ APPEND(LIST(A), SUBST(X, CDR(L), A))$$
$$else\ APPEND(LIST(CAR(L)).SUBST(X, CDR(L), A))$$
$$\Longrightarrow SUBST(X,L,A) \equiv G(NIL, X, L, A)$$
$$G(R,X,L,A) \equiv if\ L = NIL\ then\ R\ else$$
$$if\ CAR(L) = X$$
$$then\ G(APPEND(R, LIST(A)), X, CDR(L), A)$$
$$else\ G(APPEND(R, LIST(CAR(L))), X, CDR(L), A)$$

2.3　C 变换 [14]

在二阶 *Cooper* 变换的输入模式中，若 $F1 \neq F2$ 时，有时可采用如下的 C 变换或下节的 T1 变换:

输入模式:

$$f(x) \equiv if\ b(x)\ then\ h(x)\ else$$
$$if\ b1(x)\ then\ F1(f(k1(x)), g1(x))$$
$$else\ F2(f(k2(x)), g2(x))$$

输出模式:

$$f(x) \equiv G(x, e)$$
$$G(x,y) \equiv if\ b(x)\ then\ F1(y, h(x))\ else$$
$$if\ b1(x)\ then\ G(k1(x), F1(y, g1(x)))$$
$$else\ G(k2(x), F2(y, g2(x)))$$

可用性条件:

① F1 满足:　　$F1(F1(x,y),z) = F1(x, F1(z,y))$

② F1, F2 满足:　　$F1(F2(x,y),z) = F1(x, F2(z,y))$

③ F1 具有右单位元 e

④ b, b1, h, k1, k2, g1, g2, 中均不含 f.

应用示列:

[例 11] 函数 $E1(x1,x2) = x1 ** x2$　　(x1: 自然数, x2 非负整数)

$$E1(x1,x2) \equiv if\ x2 = 0\ then\ 1\ else$$
$$if\ x2 > 0\ then\ E1(x1, x2-1) * x1$$
$$else\ E1(x1, x2-1)/x1$$
$$\Longrightarrow E2(x1,x2) \equiv G(x1, x2, 1)$$
$$G(x1,x2,y) \equiv if\ x2 = 0\ then\ y\ else$$
$$if\ x2 > 0\ then\ G(x1, x2-1, y * x1)$$
$$else\ G(x1, x2+1, y/x1)$$

© 中国科学院软件研究所 http://www.jos.org.cn

2.4 T1 变换[13]

输入模式：

$$f(x) \equiv \mathit{if}\ b(x)\ \mathit{then}\ h(x)\ \mathit{else}\ \mathit{if}\ b1(x)\ \mathit{then}\ F1(f(k1(x)), g1(x))\ \mathit{else}\ F2(f(k2(x)), g2(x))$$

输出模式：

$$f(x) \equiv G(x, e1, e2)$$
$$G(x,y,z) \equiv \mathit{if}\ b(x)\ \mathit{then}\ F1(F2(h(x), z), y)\ \mathit{else}\ \mathit{if}\ b1(x)\ \mathit{then}\ G(k1(x), F1(F2(g1(x), z), y), z)\ \mathit{else}\ G(k2(x), y, F2(g2(x), z))$$

可用性条件：

①F1, F2 均满足结合律，并分别具有右单位元 $e1, e2$.

②F2 关于 F1 满足分配律，即：$F2(F1(x,y),z) = F1(F2(x,z), F2(y,z))$

③b, b1, h, k1, k2, g1, g2, 中均不含 f.

应用示列：

[例12] 计算函数：

$$f(x1,x2) \equiv \begin{cases} 1 & \text{当} x2 = 0 \\ f(x1, x2-1) + x2 & \text{当} x2 > x1 \\ f(x1-1, x2) * x1 & \text{当} x2 \le x1 \end{cases}$$

$$f(x1,x2) \equiv \mathit{if}\ x2 = 0\ \mathit{then}\ 1\ \mathit{else}\ \mathit{if}\ x2 > x1\ \mathit{then}\ f(x1, x2-1) + x2\ \mathit{else}\ f(x1-1, x2) * x1$$
$$\Longrightarrow f(x1,x2) \equiv G(x1, x2, 0, 1)$$
$$G(x1,x2,y,z) \equiv \mathit{if}\ x2 = 0\ \mathit{then}\ z + y\ \mathit{else}\ \mathit{if}\ x2 > x1\ \mathit{then}\ G(x1, x2-1, x2 * z + y, z)\ \mathit{else} G(x1-1, x2, y, x1 * z)$$

由于不同递归分支的交互作用，使得不同二阶变换的可用性条件之具体要求很不相同，但它们都是针对某些类型的递归程序而设计出来的.

§3.B 变换[13]

在 *Cooper* 变换的输入模式中，决定分支条件的布尔表达式均未出现递归调用，当布尔表达式中出现递归调用将使消除递归的问题变换更加复杂。这里仅讨论其中较为简单的情形。

考察以下的例子

© 中国科学院软件研究所 http://www.jos.org.cn

[例13 (A)] 求非空数表L中的最大元M(L)

$$M(L) \equiv if\ CDR(L) = NIL\ then\ CAR(L)\ else$$
$$if\ CAR(L) \geq M(CDR(L))\ then\ CAR(L)$$
$$else\ M(CDR(L))$$

在此例中，决定分支的布尔表达式$CAR(L) \geq M(CDR(L))$中就出现了递归调用$M(CDR(L))$。

借助于引入两个新变元，我们设计了如下的B变换(在[13]中称为T2变换):

输入模式:

$$f(x) \equiv if\ b(x)\ then\ h(x)\ else$$
$$if\ B(f(k(x)), d(x))\ then\ D(x)$$
$$else\ f(k(x))$$

输出模式:

$$f(x) \equiv if\ b(x)\ then\ h(x)\ else\ G(k(x), d(x), D(x))$$
$$G(x, y, z) \equiv if\ b(x)\ then\ [if B(h(x), y)\ then\ z\ else\ h(x)]$$
$$else\ [if B(D(x), y)\ then\ G(k(x), y, z)$$
$$else\ G(k(x), d(x), D(x))]$$

可用性条件:

① $t : B(t, d(x)) \wedge B(D(x), y) \Longrightarrow B(t, y) = true$
$\neg B(t, d(x)) \wedge \neg B(D(x), y) \Longrightarrow B(t, y) = false$

② b, h1, d, D, k 中均不含f

应用示例:

[例13 (B)] 对例13(A)施行B变换，并注意到这时d与D均为CAR (L)，故输出模式可省去一个变元，于是有

$$M(L) \equiv if\ CDR(L) = NIL\ then\ CAR(L)\ else\ G(CDR(L), CAR(L))$$
$$G(x, y) \equiv if\ CDR(x) = NIL$$
$$then\ [if\ y \geq CAR(x)\ then\ y\ else\ CAR(x)]$$
$$else\ [if\ y \geq CAR(x)\ then\ G(CDR(x), y)$$
$$else\ G(CDR(x), CAR(x))]$$

§4. RR 变换

为处理一类具有二重递归调用的程序，我们设计了如下的RR变换[14]。

输入模式:

$$f(x) \equiv if\ b(x)\ then\ h(x)\ else\ f(f(k(x)))$$

© 中国科学院软件研究所 http://www.jos.org.cn

输出模式:

$$f(x) \equiv G(x,0)$$
$$G(x,y) \equiv if\ b(x)\ then\ [if\ y=0\ then\ h(x)\ else\ G(h(x),y-1)]$$
$$else\ G(k(x),y+1)$$

可用性条件: [无]

应用示例:

[例14] *McCarthy* 91 函数

$$M(x) \equiv if\ x>100\ then\ x-10\ else\ M(M(x+11))$$
$$\Longrightarrow M(x) \equiv G(x,0)$$
$$G(x,y) \equiv if\ x>100\ then$$
$$[if\ y=0\ then\ x-10\ else\ G(x-10,y-1)]$$
$$else\ G(x+11,y+1)$$

注记: 本例中y起计算器的作用, 记录括号的层次。

§5. 多步递归程序的等价变换

对于诸如计算*Fibonacci*函数或*Legendre*函数这样的递归程序, 均出现相邻自变元的递归计算, 它们实际上是属于多步递归程序, 我们设计了如下的变换[17]:

输入模式:

$$f(x) \equiv if\ b1(x)\ then\ h1(x)\ else$$
$$if\ b2(x)\ then\ h2(x)\ else$$
$$F1(F2(f(k(x))),g1(x)),F2(f(k(k(x))),g2(x)))$$

输出模式:

$$f(x) \equiv if\ b1(x)\ then\ h1(x)\ else$$
$$if\ b2(x)\ then\ h2(x)\ else\ G(x,e2,e1)$$
$$G(x,y,z) \equiv if\ b2(x)\ then\ F1(F2(h2(x),y),F2(h1(k(x)),z))$$
$$else\ G(k(x),F1(F2(g1(x),y),z),F2(g2(x),y)))$$

可用性条件:

①F2 满足结合律, 即 $F2(F2(x,y),z)=F2(x,F2(y,z))$

②$F1(F1(x,y),z)=F1(F1(x,z),y)$

③F2 关于F1 满足分配律, 即 $F2(F1(x,y),z)=F1(F2(x,z),F2(y,z))$

④F1, F2 分别具有右单位元e1, e2

© 中国科学院软件研究所 http://www.jos.org.cn

⑤ $b2(x)$ <===> $b1(k(x))$

⑥ b1, b2, g1, g2, h1, h2 中均不含f

应用示例:

[例15] *FIBonacci* 函数 $FIB(x)$

$$FIB(x) \equiv if\ x = 0\ then\ 0\ else$$
$$if\ x = 1\ then\ 1\ else\ FIB(x-1) + FIB(x-2)$$
$$\Longrightarrow FIB(x) \equiv if\ x = 0\ then\ 0\ else$$
$$if\ x = 1\ then\ 1\ else\ G(x,1,0)$$
$$G(x,y,z) \equiv if\ x = 1\ then\ y\ else\ G(x-1, y+z, y)$$

§6. 并行递归的消除

在相当广泛一类有关表处理的算法涉及并行递归。本节讨论当递归程序中出现并行递归时，如何消除并行递归并将其转换为等价的尾递归形式。

为了讨论方便，我们可将表分为单表与复表。所谓单表是指表元素均为原子的表，如表(A B C);而复表则是指表元素可以是原子也可以是表的表，如(A (B C) (D))。

由于单表中的表元素均为原子，对其进行处理比较简单，所以在设计表处理算法时，可以先考虑单表的情形，然后再扩充为处理复表的一般情形，这时一般只需增加表头不是原子(即ATOM(CDR(L)) 为false) 时相应的递归分支。在处理复表的递归程序中往往出现并行递归调用，要直接消除这些并行递归调用是很困难的，我们提出如下的方法来间接消除这类并行递归调用。

①将复表改造成伪单表形式

例如可约定在复表内层用%代替左括号，&代替右括号。

于是复表(A (B (C)) (D)) 可改造成(A % B % C & & % D &) 的伪单表形式。

②设计相应的处理伪单表的递归程序，并选用适当变换。

它只需将处理单表的递归程序略加修改就可得到，且不会再出现并行递归调用。选用适当的变换模式可把这个程序转化为等价的尾递归形式。

③如有必要，可将处理后的结果表(伪单表形式) 再改造成复表形式。

方法应用示例:

[例16] 计算x在L中出现次数 $FRE(x,L)$. (x: 原子, L: 表)

①先考虑L为单表的情形，有

$$FRE(x,L) \equiv if\ L = NIL\ then\ 0\ else$$
$$if\ CAR(L) = x\ then\ FREN(x, CDR(L)) + 1$$
$$else\ FRE(x, CDR(L))$$

© 中国科学院软件研究所 http://www.jos.org.cn

注：当L为复表时，有

$$FREN(x,L) \equiv if\ L = NIL\ then\ 0\ else$$
$$if\ CAR(L) = x\ then\ FREN(x,CDR(L)) + 1\ else$$
$$if\ ATOM(CAR(L))\ then\ FREN(x,CDR(L))$$
$$else\ FREN(x,CDR(L)) + FREN(x,CAR(L))$$

最后一个递归调用分支中出现了并行递归$FREN(x,CDR(L))$和$FREN(x,CAR(L))$.

②在①之基础上考虑L为伪单表时的情形。在本例中，它恰好与L为单表的情形相同。

采用二阶*Cooper*变换(A型)很容易得到相应的尾递归型。

$$FRE(X,L) \equiv G(X,L,0)$$
$$G(X,L,Y) \equiv if\ L = NIL\ then\ Y\ else$$
$$if\ CAR(L) = 0\ then\ G(X,CDR(L),Y+1)$$
$$else\ G(X,CDR(L),Y)$$

在本例中，Y即为原子X在表L中出现的次数。它不涉及伪单表形式之结果复原成复表形式。

[例17] 将表L中全部原子倒排$REVA(L)$ (L: 表)

①先考虑L为单表的情形，这就是我们在例2中所得到的，即：

$$REV(L) \equiv if\ L = NIL\ then\ NIL\ else$$
$$APPEND(REV(CDR(L)), LIST(CAR(L)))$$

注：当L为复表时，将扩充成

$$REVN(L) \equiv if\ L = NIL\ then\ NIL\ else$$
$$if\ ATOM(CAR(L))$$
$$then\ APPEND(REVN(CDR(L)), LIST(CAR(L)))$$
$$else\ APPEND(REVN(CDR(L)), LIST(REVN(CAR(L))))$$

最后的递归分支中出现了并行递归调用。

②把复表改造成伪单表形式，将上述$REV(L)$修改成如下形式：

$$REVA(L) \equiv if\ L = NIL\ then\ NIL\ else$$
$$if\ CAR(L) = \%\ then\ APPEND(REVA(CDR(L)), LIST(\&))\ else$$
$$if\ CAR(L) = \&\ then\ APPEND(REVA(CDR(L)), LIST(\%))$$
$$else\ APPEND(REVA(CDR(L)), LIST(CAR(L)))$$

© 中国科学院软件研究所 http://www.jos.org.cn

用 *Cooper* 变换(A 型)可将 $REVA(L)$ 转换成等价的尾递归形式

$$REVA(L) \equiv G(L, NIL)$$

$$G(L,R) \equiv \textit{if } L = NIL \textit{ then } R \textit{ else}$$
$$\textit{if } CAR(L) = \% \textit{ then } G(CDR(L), APPEND(LIST(\&), R)) \textit{ else}$$
$$\textit{if } CAR(L) = \& \textit{ then } G(CDR(L), APPEND(LIST(\%), R))$$
$$\textit{else } G(CDR(L), APPEND(LIST(CAR(L)), R)$$

③把上述 $REVA(L)$ 的结果R(伪单表形式)再还原为复表形式。

在[18]中我们还提出了消除并行递归的另一种方法，此处不再赘述。

§7. 结束语

本文对我们近几年来在递归程序等价变换方法之若干研究工作做了分类总结，所提出的一些新变换模式和变换技术进一步丰富了人们对横向程序变换技术的认识。

我们最近研制了一个横向程序变换的实验系统XDPTS，该系统已将本文所提及的大部分变换模式纳入其模式库中(并提供进一步扩充的能力)，关于XDPTS系统的功能和实现技术，我们将另文阐述。

参考文献

[1] 徐家福，软件自动化，计算机研究与发展，1988.11。

[2] 吕建，算法设计自动化研究综述，计算机科学，1988.3。

[3] Broy, M., 等 程序变换：程序设计的一种形式化方法，计算机科学，1988.2。

[4] Manna, Z., Mathematical Theory of Computation, McGraw-Hill, New York, 1974.

[5] Manna, Z. & Waldinger, R., The Logic of Computer Programming, IEEE Trans on Software Engineering, SE-4, 1978, 3.

[6] Cooper, D. C., The Equivalence of Certain Computations, Comp. J., 1966. 9.

[7] Burstall, R. M., Darlington, J., A Transformation System for Developing Recursive Programs, JACM, 1977.1.

[8] 黄皆民，程序的形式说明、推导和变换，计算机科学，1982. 2。

[9] 黄皆民，程序变换概论，电子学报，1983. 1。

[10] 仲萃豪，冯玉琳，陈友君，程序设计方法学，北京科学技术出版社，1985。

[11] 陈火旺，罗朝晖，马庆鸣，程序设计方法学基础，湖南科学技术出版社，1987。

[12] 胡正国，蔡经球，程序设计方法学，西北工业大学出版社，1987。

[13] 蔡经球，周松，递归算法的若干等价变换，厦门大学学报，1983. 2。

[14] 蔡经球，递归算法等价变换的若干新模式，人工智能学报，1983. 3。

[15] 蔡经球，关于Cooper变换的研究，厦门大学学报，1987. 1。

[16] 蔡经球，张克均，递归程序变换的一种新模式—CZ变换，厦门大学学报，1988. 4。

[17] 张克均，蔡经球，关于多步递归算法的等价变换模式，福建电脑，1988, 2。

[18] 蔡经球，关于并行递归程序变换的若干研究，小型微型计算机系统，1989, 11。

© 中国科学院软件研究所 http://www.jos.org.cn