布尔函数的代数因子分解*

曾献君　刘　俐　李思昆　　　　　　叶以正

（长沙工学院CAD中心　长沙　410073）（哈尔滨工业大学ICCAD室　哈尔滨　150001）

摘要　本文改进了布尔函数的代数因子分解算法ALG_DIV和QF，采用“分治”方式降低了分解过程的计算时间复杂性，提高了多级逻辑函数分解的效率，能获得近似优化的多级逻辑分解结果.

关键词　布尔函数分解，逻辑综合.

中图法分类号　TP391.72

布尔函数的多级逻辑分解是多级逻辑综合的关键技术.求取基于某一优化目标的最优多级逻辑分解形式已被证明为NP难题.在算法实现上，一般寻求近似优化的代数因子分解形式.[1~6]ALG_DIV和QF[1,2]是两个典型的布尔函数代数因子分解算法，它们对布尔函数$f=F(X)$的分解过程是：① 求布尔函数f的因子核集合$K(f)$及相关核集合$C(f)$；② 逐步从$K(f)$中选取收益最大的因子核，对f进行代数因子分解；③ 对f的代数因子分解形式中的各因子核进行替代变换，并对替代变换后的布尔函数作递归的代数因子分解；④ 对f的代数因子分解形式中出现的分解因子作递归的代数因子分解.在上述递归的代数因子分解过程中，由布尔表达式求因子核集合的计算时间与表达式中出现的布尔文字总数呈指数增长趋势，这从根本上制约了多级逻辑综合问题的规模.

本文在ALG_DIV和QF基础上提出了改进方法，通过“分治”策略降低因子核求取过程的计算时间复杂性，对满足一定条件的因子核对作因子乘积，提取优化的分解形式.

1　布尔函数的代数因子分解

布尔函数$f=F(X)$的二级逻辑极小化结果通常可表示为“积之和”的形式，如函数$F(X)$：

$$F(X)=x_2x_8+x_2x_9+x_3x_8+x_3x_9+x_4x_5\overline{x_7}+x_4\overline{x_5}x_7+x_4x_6x_7+x_1x_2x_5\overline{x_7}+x_1x_3x_5\overline{x_7}+x_1x_2\overline{x_5}x_7+x_1x_3\overline{x_5}x_7+x_1x_2x_6x_7+x_1x_3x_6x_7 \tag{1}$$

式(1)中的每个“积”项称为$F(X)$的一个立方$C(cube)$，如$x_4x_5\overline{x_7}$.立方C中布尔变量的出现形式称为C的布尔文字，C中出现的布尔文字构成C的支撑集合，记为$Sup(C)$，$Sup(C)$与C等价.立方$C=x_4x_5\overline{x_7}$的布尔文字为$x_4,x_5,\overline{x_7}$，即$Sup(C)=\{x_4,x_5,\overline{x_7}\}$；$F(X)$中的布尔文字集合记为$Sup(F)$，$F(X)$中出现的布尔文字总数记为$cost(F)$.对式(1)中的$F(X)$，$Sup(F)=\{x_2,x_3,x_4,x_5,\overline{x_5},x_6,x_7,\overline{x_7}\}$，$cost(F)=41$.

若布尔表达式$g=C_1+\ldots+C_k$满足$\bigcap_{i=1}^{k}Sup(C_i)=\varnothing$，称$g$为质布尔表达式，即$g$的各立方项$C_1,\ldots,C_k$没有公因子.

布尔函数$F(X)$的因子核：$F(X)$为布尔函数，$g=C_1+\ldots+C_k$为质布尔表达式，若存在布尔表达式h，使得$\forall C\in g\cdot h$，有$C\in F(X)$，则称g为$F(X)$的因子核，h称为g的相关核.$F(X)$的因子核集合记为$K(F)$，相关核集合记为$C(F)$.

布尔函数$F(X)$的代数因子分解，将$F(X)$表示成

$$F(X)=g_1\cdot h_1+\ldots+g_m\cdot h_m+r \tag{2}$$

形式的过程称为$F(X)$的因子分解，若式(2)满足

1) $Sup(h_i)\cap Sup(g_i)=\varnothing$，$1\leqslant i\leqslant m$；

---

* 本文研究得到国防科技大学青年基金资助.作者曾献君，1966年生，博士，讲师，主要研究领域为ASIC设计综合技术.刘俐，女，1963年生，工程师，主要研究领域为ASIC设计综合技术，软硬件混合仿真技术.李思昆，1941年生，教授，博士导师，主要研究领域为VLSI设计方法学，CAD，虚拟现实.叶以正，女，1937年生，教授，博士导师，主要研究领域为ASIC设计综合技术.

本文通讯联系人：曾献君，长沙410073，长沙工学院CAD中心

本文1996-08-09收到原稿，1997-05-04收到修改稿

2) $r\notin K(F)$，r 为质布尔表达式，
则称式(2)为 $F(X)$ 的代数因子分解.

$F(X)$ 的代数因子分解的优化目标是使式(2)中出现的布尔文字总数极小化，由于布尔函数的代数因子分解是多级逻辑综合过程中频繁调用的过程，这要求代数因子分解算法不仅需有良好的优化性能，还必须有较高的执行效率.

本文改进的布尔函数 $F(X)$ 的代数因子分解算法基于下述几点考虑：

1) $F(X)$ 的因子核集合 $K(F)$ 的求取过程决定了整个代数因子分解算法的计算时空复杂性，且与 $|Sup(F)|$ 呈指数增长趋势；

2) 采用"分治"策略，先求 $F(X)$ 相对于某个立方 $C\in F(X)$ 的因子核集合 $K_c(F)$，由于 $|Sup(C)|<|Sup(F)|$，故 $K_c(F)$ 的求取具有较短的计算时间，并且 $K(F)=\bigcup_{c\in F(X)}K_c(F)$，显然，采用"分治"策略求 $K_c(F)$ 可提高计算效率；

3) 针对 $K_c(F)$，对 $F(X)$ 作代数因子分解，抽取优化的分解形式，设 $F(X)$ 基于 $K_c(F)$ 的代数因子分解形式覆盖了其中的立方 $C,C_{i1},\dots,C_{im}$，可将这些立方置为 $F(X)$ 的无关项，并对 $F(X)$ 作进一步分解；

4) 对式(2)中的因子 $g_i,h_i(1\leqslant i\leqslant m)$ 作递归的代数因子分解，由于

$$|Sup(g_i)|<|Sup(F)|,\quad |Sup(h_i)|<|Sup(F)|,$$

故对 g_i,h_i 作递归代数因子分解有较低的计算时间复杂性.

2 代数因子分解算法

设 $F(X)=C_1+\dots+C_n$，$F(X)$ 中的立方按 $|Sup(C_i)|(1\leqslant i\leqslant n)$ 不减的顺序排列. 对 $\forall\, C_i\in F(X)$，$|Sup(C_i)|\geqslant 2$，采用因子核集合求取过程 $Kernel(F,Sup(C_i))$ 求 $K_{ci}(F)$，对 $F(X)$ 作基于 $K_{ci}(F)$ 的代数因子分解，获取 $F(X)$ 的待选因子分解形式；再从 $F(X)$ 的各种待选因子分解形式中选取优化的代数因子分解形式，并去除其中存在的逻辑冗余，获取式(2)的代数因子分解形式. 其中因子核求取过程 $Kernel(F,Sup(C_i))$ 在文献[1,2]中有详细阐述.

2.1 待选的因子分解形式的求取

对 $K_{ci}(F)$，选取 $K_{ci}(F)$ 中的因子核对 k_1,k_2. 设 k_1,k_2 对应的相关核为 ck_1,ck_2，使之满足

$$Sup(ck_1)\cap Sup(ck_2)=\varnothing \tag{3}$$

$$Sup(ck_1)\cup Sup(ck_2)=Sup(C_i) \tag{4}$$

易知 $K_{ci}(F)$ 中满足式(3)(4)的 k_1,k_2 必定存在，对 k_1,k_2 作因子乘积 $p=k_1\cdot k_2$，求 p 对函数 $F(X)$ 的立方覆盖矩阵 $A=(a_{ij})$，a_{ij} 的取值如下

$$a_{ij}=\begin{cases} m & k_{1i}k_{2j}=c_m\in F(X)\\ 0 & k_{1i}k_{2j}\notin F(X)\end{cases} \tag{5}$$

式(5)中 k_{1i} 为 k_1 的第 i 个立方项，k_{2j} 为 k_2 的第 j 个立方项，以 $|k_1|$，$|k_2|$ 表示 k_1,k_2 中的立方项数目.

定理 1. $F(X)$ 为布尔函数，C_i 为 $F(X)$ 的立方项，对 $K_{ci}(F)$ 中满足式(3)(4)的 k_1,k_2，因子积 $p=k_1\cdot k_2$ 对 $F(X)$ 的立方覆盖矩阵 A 中必存在行 r，列 l，使得：

1) $A(r,j)>0,\quad 1\leqslant j\leqslant |k_2|$

2) $A(j,l)>0,\quad 1\leqslant j\leqslant |k_1|$

3) $A(r,l)=i$.

证明，由式(3)(4)知，$C_i=ck_1\cdot ck_2\in ck_1\cdot k_1\subseteq F(X)$，$C_i=ck_1\cdot ck_2\in ck_2\cdot k_2\subseteq F(X)$，故有 $ck_2\in K_1$，$ck_1\in K_2$.

不妨设 ck_2 为 k_1 的第 r 个立方项，ck_1 为 k_2 的第 l 个立方项，则

$$k_{1r}\cdot k_{2j}=ck_2\cdot k_{2j}\in F(X),1\leqslant j\leqslant |k_2|;$$

由覆盖矩阵 A 的定义知：

$$A(r,j)>0,1\leqslant j\leqslant |k_2|;$$

同理有

$$A(j,l)>0,1\leqslant j\leqslant |k_1|.$$

$$k_{1r}\cdot k_{2j}=ck_1\cdot ck_2=C_i,故\ A(r,l)=i.$$ □

通常情况下，可由 $p=k_1\cdot k_2$ 的立方覆盖矩阵 A 中非零元素抽取 k_1,k_2 的子表达式 $sk_1\subseteq k_1$，$sk_2\subseteq k_2$，使 $C_i\in sk_1\cdot sk_2\subseteq F(X)$. 易知，$sk_1\cdot sk_2$ 为 $F(X)$ 的可选代数因子分解形式. 最坏情况下也可抽取出形如 $ck_1\cdot k_1$，$ck_2\cdot k_2$ 的可选代数因子分解形式.

由 $p=k_1\cdot k_2$ 中抽取可选代数因子分解式的算法如下：

1) 求 $p=k_1\cdot k_2$ 对 $F(X)$ 的立方覆盖矩阵 A；

2) 对 A 中元素进行行、列变换，使 A 中非零元素数目按行、列顺序逐步减少；

3) 分别按行、列次序抽取 A 中非零元素构成"长方形"。若"长方形"的长宽均不小于 2，则抽取相应的行、列对应的 k_1,k_2 的立方项构成 sk_1,sk_2，将 $sk_1\cdot sk_2$ 作为可选代数因子分解形式；

4) 继续 3)，抽取所有形如 $sk_1\cdot sk_2$ 的代数因子分解形式；

5) 若存在形如 $sk_1\cdot sk_2$ 的因子分解形式，转 6)；否则，抽取形如 $ck_1\cdot k_1,ck_2\cdot k_2$ 作可选的代数因子分解式；

6) 将抽取出的代数因子分解形式所覆盖的 $F(X)$ 的立方项置为无关项.

对布尔函数 $F(X)$ 的代数因子分解而言，可先求取全部的待选代数因子分解形式，再从中选取优化的代数因子分解形式.

对式(1)中的布尔函数 $F(X)$，$F(X)$ 基于立方项 $C_1=x_2x_8$ 的因子核集合 $K_{c1}(F)$ 及相关核集合 $C_{c1}(F)$ 为

$$K_{c1}(F)=\{x_8+x_9+x_1x_5\overline{x_7}+x_1\overline{x_5}x_7+x_1x_6x_7,\ x_2+x_3\},$$

$$C_{c1}(F)=\{x_2,x_8\}.$$

由于 $K_{c1}(F)$ 中的两个因子核 k_1,k_2 满足条件(3)(4)，作 $p=k_1\cdot k_2$ 并求 p 对 $F(X)$ 的立方项覆盖矩阵 A 如下：

$$A=\begin{array}{c|ccccc|} & x_8 & x_9 & x_1x_5\overline{x_7} & x_1\overline{x_5}x_7 & x_1x_6x_7 \\ x_2 & 1 & 2 & 8 & 10 & 12 \\ x_3 & 3 & 4 & 9 & 11 & 13 \end{array}$$

取 $sk_1=k_1,sk_2=k_2$，抽取 $sk_1\cdot sk_2$ 作为待选因子分解形式，并将 $sk_1\cdot sk_2$ 所覆盖的 $F(X)$ 的立方项 $C_1,C_2,C_3,C_4,C_8,C_9,C_{10},C_{11},C_{12},C_{13}$ 置为无关项，$F(X)$ 现仍有 C_5,C_6,C_7 未被 $sk_1\cdot sk_2$ 所覆盖. 选取 C_5，易求

$$K_{c5}(F)=\{x_5\overline{x_7}+\overline{x_5}x_7+x_6x_7,x_4+x_1x_2+x_1x_3\},$$

$$C_{c5}(F)=\{x_4,x_5\overline{x_7}\},$$

$K_{c5}(F)$ 中的两个因子满足条件(3)(4)，易求

$$A=\begin{array}{c|ccc|} & x_4 & x_1x_2 & x_1x_3 \\ x_5\overline{x_7} & 5 & 8 & 9 \\ \overline{x_5}x_7 & 6 & 10 & 11 \\ x_6x_7 & 7 & 12 & 13 \end{array}$$

取 $sk_1=x_4+x_1x_2+x_1x_3,sk_2=\overline{x_5}x_7+x_5\overline{x_7}+x_6x_7$，将 $sk_1\cdot sk_2$ 作为 $F(X)$ 的待选代数因子分解形式，并将 $sk_1\cdot sk_2$ 所覆盖的 $F(X)$ 的立方项置为无关项，至此，$F(X)$ 的所有立方项均被待选代数因子分解形式所覆盖.

2.2 待选代数因子分解形式的优化选取

设 $F(X)$ 的待选代数因子分解形式序列为 $p_1,p_2,\ldots,p_k,p_i$ 取 $sk_{i1}\cdot sk_{i2}(|sk_{i1}|\geqslant 2,|sk_{i2}|\geqslant 2)$ 或 $ck_i\cdot k_i$ 形式. 从中选取合适的分解形式序列 $p_{i1},p_{i2},\ldots,p_{im}$，使 $p_{i1}+p_{i2}+\ldots+p_{im}$ 构成式(2)中 $F(X)-r$ 的一个覆盖，$p_{ij}(1\leqslant j\leqslant m)$ 的选取方式如下：

(1) 构成 $p_1,p_2,\ldots,p_k$ 对 $F(X)$ 的立方覆盖矩阵 $C,C\in N^{k\times n}$，n 为 $F(X)$ 中的立方项数目，$C=(c_{ij})$，c_{ij} 取值如下：

$$c_{ij}=\begin{cases}0 & c_j\notin p_i\\ 1 & c_j\in p_i\end{cases} \tag{6}$$

(2) 若矩阵 C 中存在列 j，使 $\sum_{i=1}^{m}c_{ij}=1$，不妨设 $c_{ij}=1$，则称 p_i 为 $F(X)-r$ 覆盖的必选代数因子分解形式. 选取 $F(X)-r$ 的所有必选代数因子分解形式；

(3) 若某个立方(集合)尚未被已选取的代数因子分解形式所覆盖，但被其它多个待选的代数因子分解形式覆盖，则选择单位立方文字数最小的代数因子分解形式，即求

$$\min p_i\{Literal(p_i)/\sum_{j=1}^{n}c_{ij}\} \tag{7}$$

式(7)中，$Literal(p_i)$ 为 p_i 中的布尔文字总数，由式(7)选取具有最小值的 p_i；重复(3)，直至选取的代数因子分解形式序列 $p_{i1},p_{i2},\ldots,p_{im}$ 构成 $F(X)-r$ 的一个覆盖.

前面例子中 $F(X)$ 的待选因子分解形式为

$$p_1=(x_2+x_3)(x_8+x_9+x_1x_5\overline{x_7}+x_1\overline{x_5}x_7+x_1x_6x_7),$$

$$p_2=(x_4+x_1x_2+x_1x_3)(x_5\overline{x_7}+\overline{x_5}x_7+x_6x_7),$$

作 p_1,p_2 对 $F(X)$ 的立方项覆盖矩阵 C 如下：

$$C=\begin{array}{c|cccccccccccccc|} p_1 & 1&1&1&1&0&0&0&1&1&1&1&1&1 \\ p_2 & 0&0&0&0&1&1&1&1&1&1&1&1&1 \end{array}$$

由 C 易求出：p_1,p_2 均为 $F(X)$ 的质蕴含代数因子分解形式，$r=\varnothing$，$F(X)$ 可初始分解为 $F(X)=p_1+p_2$ 的形式.

2.3 逻辑冗余的消除

设优选出的 $F(X)$ 的代数因子分解形式为 $p_{i1},p_{i2},\ldots,p_{im},p_{i1}+\ldots+p_{im}$ 可能多次对 $F(X)$ 中的某些立方项进行了覆盖，并直接导致逻辑冗余和布尔文字数目增加；另一方面，在代数因子分解形式中，有些逻辑冗余的存在反而有利于获取良好的代数因子分解形式.[2,3]可采取一定的规则消除导致布尔文字数目增加的逻辑冗余，而保留对代数因子分解目标有益的逻辑冗余项.

冗余项去除法则：若 $F(X)$ 中的立方项 C 被代数因子分解式 $p_{i1},\ldots,p_{im}$ 多次覆盖，若从 $p_{i1},\ldots,p_{im}$ 中任意去除一个，产生 C 的乘积项，将使整个因子分解式中布尔文字数目增加或 $F(X)-r$ 中有些立方项不被覆盖，则该类逻辑冗余对最终的多级逻辑分解结果的优化有好处，应加以保留，否则，应尽量消除代数分解形式中出现的逻辑冗余.

代数因子分解式中逻辑冗余的去除方法如下：

1) 构造代数因子分解式 $p_{i1},\ldots,p_{im}$ 对 $F(X)$ 中立方的覆盖矩阵 C；

2) 对 $p_{ij}\in\{p_{i1},\ldots,p_{im}\}$，求 $S_{p_{ij}}=\{j|c_{ij}=1\wedge(\sum_{k=1}^{m}c_{kj})>1\}$，

由 p_{ij} 的立方覆盖矩阵 A 中消除 $S_{p_{ij}}$ 中元素后导致的布尔文字数目减少数目 $n_{p_{ij}}$，若消除 p_{ij} 中产生 $S_{p_{ij}}$ 立方项之后，$F(X)-r$ 中有多于一个立方不为 $p_{i1}+\ldots+p_{im}$ 所覆盖，则 $n_{p_{ij}}=0$. 寻找使 $n_{p_{ij}}$ 最大的因子分解形式 p_{ij}，消除 p_{ij} 中的逻辑冗余；

3) 重复 1) 和 2)，直至 $\forall\ p_{ij}\in\{p_{i1},\ldots,p_{im}\},n_{p_{ij}}\leqslant 0$ 或 $F(X)-r$ 中每个立方项只被 $p_{i1},\ldots,p_{im}$ 覆盖唯一的 1 次.

前面将 $F(X)$ 分解为 $F(X)=p_1+p_2$ 形式，其中 $F(X)$ 中的立方项 $C_8,C_9,C_{10},C_{11},C_{12},C_{13}$ 均被 p_1+p_2 覆盖 2 次，$F(X)=p_1+p_2$ 中存在逻辑冗余. 根据逻辑冗余去除过程的第 2) 步可计算出

$$S_{p1}=S_{p2}=\{8,9,10,11,12,13\},n_{p1}=9,n_{p2}=6,$$

即从 p_1 中消除产生逻辑冗余的项可使原始分解形式减少 9 个布尔文字，而对 p_2 作同样分析处理则只能减少 6 个布尔文字，故可从 p_1 的因子 $x_8+x_9+x_1x_5\bar{x}_7+x_1\bar{x}_5x_7+x_1x_6x_7$ 中去除产生逻辑冗余的项 $x_1x_5\bar{x}_7,x_1\bar{x}_5x_7,x_1x_6x_7$，从而使分解形式变为

$$F(X)=(x_2+x_3)(x_8+x_9)+(x_4+x_1x_2+x_1x_3)(x_5\bar{x}_7+\bar{x}_5x_7+x_6x_7)$$

2.4 因子的递归分解

设 p_i 为 $F(X)$ 的一个因子分解式，p_1 可为 $sk_{i1}\cdot sk_{i2}$ 或 $ck_i\cdot k_i$ 两种形式之一，由代数因子分解步骤 2.1～2.3 知，若因子中仅包含两个立方项时，则不可作进一步的代数因子分解，否则，应对相应的因子作递归代数因子分解，以进一步使代数因子分解形式中的布尔文字数目极小化. 函数 $F(X)$ 可递归分解为

$$F(X)=(x_2+x_3)(x_8+x_9)+(x_5\bar{x}_7+x_7(\bar{x}_5+x_6))(x_4+x_1(x_2+x_3))$$

2.5 分解过程 Near_Opt_Dc

给定布尔函数 $F(X)=c_1+\ldots+c_n$，$F(X)$ 的代数因子分解过程描述如下：

1) 将 $F(X)$ 中的立方项按其布尔文字数目的递增顺序排列；

2) 顺序从 $F(X)$ 中选取立方项 c，求 $F(X)$ 相对于 c 的因子核集合 $K_c(F)=Kernel(F,Sup(c))$，按 2.1 节描述的方法，求 $F(X)$ 基于 $K_c(F)$ 的所有待选的代数因子分解形式；

3) 按 2.2 节描述的方法，选取优化的代数因子分解形式，构成形如式(2)的 $F(X)$ 的代数因子分解形式；

4) 按 2.3 节描述的方法，消除 $F(X)$ 的代数因子分解形式中不必要的逻辑冗余；

5) 按 2.4 节描述的方法，对 $F(X)$ 的各分解因子进行递归的代数因子分解.

算法中，步骤 2) 和 5) 分别求取 $F(X)$ 基于 $K_c(F)$ 的所有待选的代数因子分解形式，对各因子分解形式递归代数因子分解，这是算法中"分"的方法；3) 和 4) 则为算法中"治"的具体步骤.

3 结果与讨论

改进的代数因子分解算法 Near_Opt_Dc 有比 ALG_DIV 和 QF 更优化的结果及更短的计算时间.

首先，整个代数因子分解过程的计算时间复杂性取决于因子核求取过程 $Kernel(F,Sup(c_i))$. 由于 $|Sup(c_i)|<$

$|Sup(F)|$，故 $Kernel(F,Sup(c_i))$ 较 $Kernel(F,Sup(F))$ 有低得多的计算时间复杂性，由文献[1,2]易知：$Kernel(F,Sup(c_i))$，$Kernel(F,Sup(F))$ 的计算时间复杂性分别为 $O(a^{|Sup(c_i)|})$，$O(a^{|Sup(F)|})$，$a>2$；由于 $K_{ci}(F)\subset K(F)$，故求基于 $K_{ci}(F)$ 的待选代数因子分解形式的计算时间低于逐步从 $K(F)$ 中选取优化的分解因子所需的时间；由 $\bigcup_{c\in F}K_c(F)=K(F)$，即使对 $\forall\ c\in F$ 求出所有基于 $K_c(F)$ 的待选代数因子分解形式，但从中选取优化的代数因子分解形式的计算时间复杂性应与逐步从 $K(F)$ 中选取优化的分解因子的计算时间复杂性相同. 故算法 Near_Opt_Dc 的计算时间复杂性应取决于 $Kernel(F,Sup(c_i))$ 的计算时间及 $F(X)$ 中的立方项数目 m. 在最坏情况下，其计算时间复杂性不超过 $O(a^{|Sup(F)|})$；其次，若 ALG_DIV 和 QF 可获取形如 $sk_{i1}\cdot sk_{i2}$ 的因子分解形式，设 $sk_{i1}\cdot sk_{i2}$ 覆盖 $F(X)$ 中的立方 $C_{i1},\ldots,C_{ik}$，由本文叙述可知，选取 $C_{i1},\ldots,C_{ik}$ 中的任意一个立方 C 并对 $F(X)$ 作基于立方 C 的分解，必可求解一个待选因子分解形式 $sk'_{i1}\cdot sk'_{i2}$，使得 $sk_{i1}\sqsubseteq sk'_{i1}$，$sk_{i2}\sqsubseteq sk'_{i2}$ 成立. 由于待选的因子分解形式的选取是以分解结果的布尔文字数目最小化作为标准，故 Near_Opt_Dc 至少可获取与 $sk_{i1}\cdot sk_{i2}$ 类似或更优化的分解因子形式；第三，将 $F(X)$ 作基于某个立方 C 的代数因子分解，至少可选取 $ck\cdot k$ 这种因子分解形式，但 Near_Opt_Dc 还可获得形如 $sk_1\cdot sk_2$ ($|sk_1|\geqslant 2$，$|sk_2|\geqslant 2$) 的待选因子分解形式，而 ALG_DIV 和 QF 则需经过一系列的因子提取、代换等过程才有可能获取类似的代数因子分解形式.

参考文献

1 Brayton R K. Factoring logic function. IBM Journal of Research Development, 1987, 31(3): 187～198

2 Brayton R K, Rudell R *et al.* Mis: a multilevel logic optimization system. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 1987, CAD-6(6): 1052～1081

3 Mathony H, Baitinger U G. CARLOS: an automated multilevel logic design system for CMOS semi-custom integrated circuits. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 1988, CAD-7(3): 346～355

4 莫云和，高文，刘明业. 多级函数分解. 计算机辅助设计与图形学学报，1993，5(2)：1～7
(Mo Yun-he, Gao Wen, Liu Ming-ye. Factoring multilevel logic functions. Journal of Computer-Aided Design and Computer Graphics, 1993, 5(2): 1～7)

5 Brayton R K, McMullen C. The decomposition and factoring of Boolean expressions. In: Proceedings of 1982 IEEE International Symposium on Circuits and Systems (ISCAS-82). Rome: IEEE Circuits and Systems Society Press, 1982. 48～54

6 Singh K J, Wang A R, Brayton R K *et al.* Timing optimization of combinational logic. In: Proceedings of IEEE International Conference on Computer-Aided Design (ICCAD-88). Santa Clara, California: IEEE Computer Society Press, Nov. 1988. 282～285

The Algebraic Decomposition and Factoring of Boolean Functions

ZENG Xian-jun　LIU Li　LI Si-kun

(*CAD Center　Changsha Institute of Technology　Changsha*　410073)

YE Yi-zheng

(*ICCAD Laboratory　Harbin Institute of Technology　Harbin*　150001)

Abstract　This paper is introduced improvement the algorithm ALG_DIV and QF that decompose and factor the Boolean functions. In this paper, the "divide and conquer" strategy is used to reduce the computing complexity and to improve the computing efficiency of the process to decompose and factor the Boolean functions. The improved algorithm can get a near optimal multilevel logic forms of the Boolean functions.

Key words　Boolean functions decomposition and factorization, logic synthesis.

Class number　TP391.72