大函数 ISFPRM 面积优化方法

瞿 婷,王伦耀,罗文强,夏银水
(宁波大学信息科学与工程学院,浙江宁波 315211)

摘 要: 针对以往在 ISFPRM 优化过程中只能处理小规模电路的不足,提出了一种新的乘积项十进制表示和处理方法来实现大电路 ISFPRM 面积优化. 具体包括:ISFPRM 多位变量的十进制数表示,基于二进制插值的极性转换方法,以及基于整数的位运算遗传算法实现 ISFPRM 面积优化. 提出的算法能有效地避免以往算法在处理输入较多的函数时效率低下甚至无法工作的情况,算法的性能用 MCNC 标准电路作为测试. 实验结果表明,提出的算法可以处理输入变量个数为 199 个的大电路,算法的速度对待处理电路的变量数不敏感特点,引入不确定项后,电路面积优化明显.

关键词: 不完全确定 RM 电路; 二进制插值; 位运算; GA 算法
中图分类号: TP391.72 **文献标识码:** A **文章编号:** 0372-2112 (2018)05-1101-06
电子学报 URL: http://www.ejournal.org.cn **DOI:** 10.3969/j.issn.0372-2112.2018.05.012

A Novel Method for Large ISPFRM Function Optimization

QU Ting, WANG Lun-yao, LUO Wen-qiang, XIA Yin-shui
(*School of Information Science and Engineering, Ningbo University, Ningbo, Zhejiang* 315211, *China*)

Abstract: In view of the problems of the published methods of the ISPFRM functions optimization which couldn't deal with large functions, a novel method for large ISPFRM function optimization was proposed which consists of the representation of product term in integer form, the polarity conversion method using the binary interpolation, and the circuit area optimization of ISFPRM using the bit-wise operation and the genetic algorithm. The proposed algorithm could deal with those functions with large inputs effectively, and has been implemented in C and tested under MCNC benchmarks. The experimental results show that it can deal with the large function with 199 inputs, and the speed of the algorithm is not sensitive to the functions' polarity. After the introduction of DC terms, the circuit area is further optimized.

Key words: ISFPRM(Incompletely Specified Fixed Polarity Reed-Muller); binary interpolation; bit operation; genetic algorithm

1 引言

逻辑函数的表示形式既可以采用基于"AND/OR/NOT"运算的传统 Boolean 形式来表示,也可以采用基于"XOR/AND"运算的 Reed-Muller(RM)逻辑来实现. 相应地,在逻辑综合方面,也可以大致分为基于的传统 Boolean 逻辑综合、基于 RM 的逻辑综合以及 Boolean 逻辑与 RM 逻辑相结合的双逻辑综合[1~3]. 相比于传统布尔逻辑,RM 逻辑在算术电路,奇偶校验电路,可逆逻辑综合,可测试性设计等方面的优势而吸引越来越多研究者的兴趣[4~7].

电路面积优化是逻辑综合与优化的一个重要内容,可以通过简化函数表达式来实现电路面积的优化. RM 函数的简化可以利用极性变换来实现[8,9],且通常以表达式中包含的乘积项的数量来衡量电路的面积[10,11]. RM 函数对应的电路面积优化中通常包含极性转换和极性搜索二个关键步骤. 通过极性转换获得某一极性下的 RM 函数表达式及包含的乘积项的数量;然后通过极性搜索得到实现 RM 表达式简化的最佳极性.

对于一个 n 变量的 RM 函数,一共有 2^n 种极性表示和 2^n 种繁简不一的表达式;若 RM 函数中有 α 个不确定项,则表达式的种类变成 $2^{n+\alpha}$[12]. 因此,当输入变量个数比较大时,极性转换过程中要处理的数据将变

收稿日期:2017-03-27;修回日期:2017-07-06;责任编辑:马兰英
基金项目:国家自然科学基金(No.61471211);国家自然科学基金重点项目(No.61131001)

得非常庞大,函数展开式的种类也变得非常庞大,尤其是含有不确定乘积项 RM 表达式种类将变得更加庞大,导致极性搜索变得更加困难,进而使得大 RM 函数对应的电路面积优化面临很大挑战.

本文主要讨论含有不确定项的固定极性 RM 表达式(Incompletely Specified Fixed Polarity Reed-Muller,ISFPRM)的简化,进而实现电路面积的优化. 在 ISFPRM 优化方面,文献[12]提出了一种基于 PSGA 算法的 ISFPRM 电路面积与功耗优化. 该算法在极性转换上采用了以最小项为基础的列表技术,在用列表技术进行极性转换时,算法的效率和能处理的电路规模直接受制于函数包含的最小项的数量;文献[13]提出了一种基于真值矢量(truth vector)和权重矢量(weight vector)的 ISFPRM 展开式最小化算法. 该算法提出的函数真值矢量表示实际上是最小项表示的另一种形式,真值矢量的长度等于 2^n,n 为变量数量,矢量中每个"1"代表一个最小项;文献[14]提出一种基于系数矩阵变换的 ISFPRM 展开式最小化算法. 该算法的极性转换过程是通过布尔矩阵和最小项的分解运算实现. 从实验结果来看,以上方法能处理的电路输入个数都没有超过 20. 其主要原因是由于以上方法以函数的最小项为基础实现极性转换,而逻辑函数最小项的数量与 2^n 成正相关. 因此,随着输入变量的增加,算法所要处理的数据量成指数级增加,导致算法效率减低,甚至无法工作. 因此要实现对大输入变量的 ISFPRM 函数的优化,应该避免以最小项为基础的极性转换.

本文将提出了一种以不相交乘积项为基础,结合提出的多位变量的十进制数表示方法,二进制插值极性转换方法,并利用遗传算法实现面积优化.

2 基于不相交项的不完全确定 RM 函数表示

含有 n 个输入变量逻辑函数可以用式(1)表示.

$$f(x_0,\cdots,x_{n-2},x_{n-1}) = \sum_{k=0}^{W-1}(p_k \mid 1) + \sum_{j=0}^{D-1}(q_j \mid y_j) \tag{1}$$

式(1)中,"$\sum$"是逻辑"或"运算,$(p_k \mid 1)$表示输入为 p_k 所示变量组合时,f 的取值为逻辑"1". 在本文中把$(p_k \mid 1)$略写为 p_k. q_j 为 f 的不确定项,y_j 是输入为 q_j 所示变量组合时 f 的取值,且 y_j 可以根据需要任取逻辑"0"或"1". W 和 D 分别为构成 f 的乘积项和不确定项的数量. 考虑到不相交乘积项之间的"或"与"异或"可以互换而不会改变逻辑函数的正确性. 因此,当构成式(1)中的乘积项为不相交乘积项时,式(1)就可以转化为式(2).

$$f(x_0,\cdots,x_{n-2},x_{n-1}) = \left(\oplus\sum_{i=0}^{Z-1}e_i\right)\oplus\left(\oplus\sum_{j=0}^{V-1}(u_j \mid d_j)\right) \tag{2}$$

式(2)中,$\oplus\sum_{i=0}^{Z-1}e_i$ 和 $\oplus\sum_{j=0}^{V-1}(u_j \mid d_j)$ 分别对应式(1)中 $\sum_{k=0}^{W-1}p_k$ 和 $\sum_{j=0}^{D-1}(q_j \mid y_j)$ 的不相交项的展开. 符号"$\oplus\Sigma$"表示多个乘积项的"异或". 由于没有对构成不相交乘积项 e_i 和 u_j 的变量的取值形式进行限定,即同一个变量可以以原变量和反变量的形式出现在不同的乘积项中,因此,式(2)实际上是不完全确定逻辑函数的混合极性的 RM 展开. 对于"异或"运算,由于存在 $1\oplus x=\bar{x}$,因此总可以利用 $1\oplus x_i=\bar{x}_i$ 或 $1\oplus\bar{x}_i=x_i$,使得式(2)中某一变量 x_i 的取值形式在所有乘积项中都一样,从而在不相交乘积项的基础上实现 f 的固定极性 RM(fixed-polarity Reed-Muller,FPRM)表示. 考虑到一个函数的不相交乘积项数量远小于它的最小项,且与输入变量的个数无关,因此基于不相交乘积项的极性转换算法往往可以处理更大的电路,同时算法的速度也更快[8].

3 ISFPRM 表达式的极性转换及算法实现

设 n 变量 RM 函数 f 的初始极性为 P_0,待转化的极性,也称目标极性为 P_1,P_0 和 P_1 分别由 n 个"0"或"1"组成,分别表示对应的变量取原变量或反变量. 令极性为 P_0 的 RM 函数表达式中,逻辑值为"1"的乘积项集合为 V,不确定项的集合为 V_{dc}.

设 f 的某一乘积项 p_i 在极性为 P_0 时的表达式为 $p_i=\dot{x}_0\dot{x}_1\dot{x}_2\cdots\dot{x}_k\cdots\dot{x}_{n-1}$,符号 $\dot{x}_k$ 表示变量 x_k 取值为原变量,反变量或不出现三种情况之一. 假设从极性 P_0 向极性 P_1 转换时,p_i 中有 z 个变量的极性需要变换,$1\leqslant z\leqslant n$,它们分别为 $\dot{x}_{t0},\dot{x}_{v1},\cdots,\dot{x}_{v(z-1)}$. 考虑到 p_i 中的乘积项符合交换定律,为了表述方便,把该 z 个变量都放在相邻的位置上,并置于 p_i 的尾部,即 $p_i=\dot{x}_0\dot{x}_1\dot{x}_2\cdots\dot{x}_{n-1}\cdot(\dot{x}_{t0}\dot{x}_{v1}\cdots\dot{x}_{v(z-1)})$. 因此在极性为 P_1 时,p_i 的表达式转化为 $p_i=\dot{x}_0\dot{x}_1\dot{x}_2\cdots\dot{x}_{n-1}\cdot[(1\oplus\overline{\dot{x}_{t0}})(1\oplus\overline{\dot{x}_{v1}})\cdots(1\oplus\overline{\dot{x}_{v(z-1)}})]$. 用归纳法不难证明表达式$[(1\oplus\overline{\dot{x}_{t0}})(1\oplus\overline{\dot{x}_{v1}})\cdots(1\oplus\overline{\dot{x}_{v(z-1)}})]$的展开后有 2^z 个乘积项.

在本文中,用"0"表示变量在乘积项中不出现,用"1"表示变量按照极性 P_1 规定的形式出现,仍然可以归纳法证明表达式$[[(1\oplus\overline{\dot{x}_{t0}})(1\oplus\overline{\dot{x}_{v1}})\cdots(1\oplus\overline{\dot{x}_{v(z-1)}})]$的展开结果同 0 到$(2^z-1)$的二进制展开成对应关系. 例如,在 $z=2$,$(1\oplus\overline{\dot{x}_{t0}})(1\oplus\overline{\dot{x}_{v1}})$展开后的表达式为$\{1,\overline{\dot{x}_{v1}},\overline{\dot{x}_{t0}},\overline{\dot{x}_{t0}\dot{x}_{v1}}\}$,一共 4 项. 而 0 到$(2^z-1)$的二进制表示为$\{(00)_2,(01)_2,(10)_2,(11)_2\}$. 若规定$\overline{\dot{x}_{v1}}$为低位,$\overline{\dot{x}_{t0}}$为高位,"0"和"1"分别表示变量不出现和按照极性 P_1 规定的形式出现,则$\{1,\overline{\dot{x}_{v1}},\overline{\dot{x}_{t0}},\overline{\dot{x}_{t0}}\,\overline{\dot{x}_{v1}}\}$与$\{(00)_2,(01)_2,(10)_2,(11)_2\}$存在对应关系. 从上面例子不难发现,将乘积项从极性 P_0 向极性 P_1 转换中,可以通过比较 P_0 和

P_1 取值不同的位的数量和位置,然后按照二进制方式展开,就可以实现极性转换.

极性转换后,表达式中可能会包含多个相同的乘积项可以合并. 假设在极性 P_1 下,某一个乘积项 e_i 与某一个不确定项$(u_j \mid d_j)$表达式一样,则可以将该二项合并成一项,即$\{e_i \mid (1 \oplus d_j)\}$. 更一般地,在极性为 P_1 情况下,式(2)可以表示为:

$$f(x_0,\cdots,x_{n-2},x_{n-1}) = \oplus \sum_{i=0}^{l-1} \left\{ e_i \left[\left(\oplus \sum_{j=0}^{n-m-1} d_j \right) \oplus \left(\oplus \sum_{l=0}^{m-1} 1 \right) \right] \right\} \tag{3}$$

式(3)表示在极性 P_1 下,经过相同乘积项合并后,f 一共有 l 个乘积项构成,对于某一乘积项 e_i,一共有 n 个相同乘积项,其中取值为"1"的有 m 个,属于不确定项的有$(n-m)$个,$n \geqslant 1$,$m \geqslant 0$. 考虑到不同极性下,式(3)中的 l,m 和 n 的值会发生变化,而偶数个"1"的"异或"结果为"0"可以删去,因此通过极性转换,可以实现 l 的减少;另外适当地选择不同的 d_j 的取值,可以使某些 e_i 的 $\left[\left(\oplus \sum_{j=0}^{n-m-1} d_j \right) \oplus \left(\oplus \sum_{l=0}^{m-1} 1 \right) \right]$ 取值为"0"而删除,从而实现式(3)的进一步简化.

RM 函数的极性转换伪代码如下,其中乘积项存储在链表 L_1 中,初始极性为 P_0,目标极性为 P_1,链表 L 和 L_new 为二个空的链表.

算法 1　polarity_convert(L,L_1 L_new,P_0,P_1)

```
{store product terms to List L1;
format_convert(L,L1);   //setp A1
P = P0 ∧ P1;            //step B1
pointer = head of L;
While (pointer ! = Null) {
p_dec = pointer->product;
p_need_cnt = P & p_dec;      //step C1
Z = counter_1(p_need_cnt);   //step D1
for(i = 0;i < (1≪z);i++) {
p_new = insect_binary(p_need_cnt,p_dec);//step E1
store p_new to the L_new;} }
}
```

算法 1 中 step A1 用来将乘积项转换成十进制形式. 当变量个数超过 30 个,又小于 60 个时,用 2 个十进制数存储,以此类推. 因此一个十进制数最多可以表示 30 位的变量. step B1 和 step C1 符号"∧"表示"位异或"运算,符号"&"表示位"与"运算. 由于 $P = P_0 \wedge P_1$,所以当 P 中位值取"1"时,对应的变量需要进行极性变换. 同时考虑到乘积项中没有出现的变量不需要极性转化,因此可以通过 $p\_need\_cnt = P\&p\_dec$ 运算,通过检查 p_need_cnt 中位值是否为"1"来确定需要极性变换变量,并用 step D1 中子程序 $counter\_1(p\_need\_cnt)$ 来统计需要极性变换的变量数量. step E1 通过二进制插值的方式实现对乘积项的极性转换,其方法如下:分别将数值 0 至(2^z-1)的二进制表示插入到 2^z 个 p_dec 中,插入的位置就是 p_need_cnt 中位值为"1"对应位. 用上述算法分别对 ISFPRM 表达式中确定的乘积项集合和不确定乘积项集合进行极性转换,并对结果进行合并,就实现了 ISFPRM 的极性转换,算法的伪代码如算法 2. 伪代码中,链表 L_1 存储确定的乘积项集合,L_{dc}存储不确定乘积项集合,初始极性为 P_0,目标极性为 P_1,链表 L_{isRM}为空的链表,用于寄存极性转换后的结果.

算法 2　ISFPRM_Polarity_convert(L_1,L_{dc},L_{isRM},P_0,P_1)

```
{creat 3 new Lists L_a,L_b and L_c;
polarity_convert(L_a,L1,L_b,P0,P1);
clear L_a;
polarity_convert(L_a,Ldc,L_c,P0,P1);
merge(LisRM,L_b,L_c);   //step A2
}
```

算法 2step A2 中,对于 L_b,L_c 中相同的乘积项用式(3)进行合并,并将结果寄存在链表 L_{isRM}中.

4　大函数的 ISFPRM 的面积优化算法

本文用遗传算法实现有利于 ISFPRM 函数简化的最佳极性搜索. 遗传算法主要包括染色体编码,适应度函数以及遗传算子操作等几个方面.

在本文中将极性与不确定项的取值用二进制数进行染色体编码,并以 30 位编码为单位用一个整数表示. 染色体的前段表示不确定项取值编码,后段表示极性编码. 考虑到 ISFPRM 函数的乘积项的数量是衡量电路面积一个重要指标,因此用乘积项的数量作为适应度函数. 假设 G_i 表示第 i 种染色体编码,TermNum(G_i)表示极性取值和不确定项取值的组合为 G_i 时式(3)对应的乘积项数,则适应度函数如式(4)所示:

$$\mathrm{fitness}(i) = \frac{1}{\mathrm{TermNum}(G_i)} \tag{4}$$

显然 TermNum(G_i)越大,电路面积越大,适应度函数值越小,反之,电路面积越小,G_i 编码越好.

遗传算子操作主要包括染色体的选择、交叉和变异三个基本操作. 考虑到本文的染色体以多位染色体编码为对应一个整数的形式寄存的,因此遗传算法对染色体的上述操作实际上是对一个或多个整数某些或某个二进制位的复制、替换或取反运算.

(1)基于整数表示的染色体段复制和替换

整数的段复制和替换操作用来实现遗传算法中染色体的选择和交叉. 假设染色体的编码以二进制形式寄存

在整数 D 中,为了截取 D 中第 j 到 $(j+k-1)$ 连续 k 位二进制数,$j\geqslant 0$,$k\geqslant 1$,$(j+k-1)\leqslant 29$,可以先构造一个整数 F,使得 $F=(0_{29},0_{28},\cdots,1_{(j+k-1)},1_{(j+k-2)},\cdots,1_j,\cdots,0_0)_2$,其中 1_j 表示 F 的第 j 位取值为 1,同理 0_j 表示 F 的第 j 位取值为 0. 通过式(5)可以实现对的 D 连续 k 位二进制数复制到 D'.

$$D'=D\ \&\ F \tag{5}$$

而通过式(6)可以实现整数 D_0 的 j 到 $(j+k-1)$ 连续 k 位二进制数用整数 D_1 中相应位的二进制数替换.

$$D_0=[D_0\&(\sim F)]\mid[D_1\ \&\ F] \tag{6}$$

式(6)中符号"~"为逐位取反运算,"|"为位"或"运算.

跨多个整数的二进制段的复制和替换也可以用上述方法实现. 如待复制的二进制段长度为 L 位,则 L 总可以表示成 $L=m+k*30+r$. 其中 m 表示需要截取的段的头部覆盖了一个整数的低 m 位二进制数,r 表示段的尾部覆盖了另一个整数的高 r 位二进制数,k 表示二进制段覆盖了 k 个连续的整数. 对于这样的二进制段,它的高 m 位复制和替换时需要构造的整数为 $F=1\ll(m-1)$;它的低 n 位复制和替换时构造的另一个整数为 $F'=\sim[1\ll(r-1)]$;而中间 k 个整数可以通过完整的整数复制和替换实现.

(2)基于整数表示的染色体的位取反

整数的位取反用来实现遗传算法中的染色体变异操作. 假设要对表示整数 D 中第 n 位和第 m 位二进制数取反,$n\neq m$,可以构造一个整数 F,使得 $F=(1\ll n)\mid(1\ll m)$;然后利用 $(D\wedge F)$ 实现对 D 的第 n 位和第 m 位进行取反.

例如 $D=17$,要对 D 的第 3 和 4 位取反,则 $F=(1\ll 3)\mid(1\ll 4)=(11000)_2=24$,$(D\wedge F)=(10001)_2\wedge(11000)_2=(01001)_2=9$.

对由多个整数构成的长染色体中的某些位取反可以转化为对某一个整数的某一个位的取反来实现. 假设要对染色体上第 m 和 n 位取反,则 m 和 n 总可表示成为:

$m=m'+k_m*30,m'<30,k_m\geqslant 0$,

$n=n'+k_n*30,n'<30,k_n\geqslant 0$.

其中 m',k_m 表示第 k_m 个整数中的第 m'位,同理 n表示第 k_n 个整数中的第 n位. 对第 m'位和第 n位取反过程与对一个整数中的某一位取反的操作一样.

结合提出的 ISFPRM 不同极性间转化算法,以及基于整数段复制、替换和位取反运算的染色体编码操作,利用遗传算法可以实现 ISFPRM 电路优化的最佳极性搜索算法. 搜索算法的伪代码如算法 3.

搜索算法中,种群个数由 step A3 完成,n 为待处理函数的输入变量数,当$\lfloor 2^{n-1}\rfloor$小于50时,种群个数取$\lfloor 2^{n-1}\rfloor$,否则种群个数取值为 50. 在 step B3 中用基于不相交乘积项列表技术将"与/或"形式的逻辑表达式转化为零极性下的 RM 表达式. 在 step C3 中,先用搜索算法得到不同染色体下的函数表达式,并得到相应的乘积项的个数,然后用式(4)计算每个染色体的适应度. 在 step D3 到 step E3 中,利用提出的整数段复制和替换技术实现不同染色体交叉操作产生新的染色体;step F3 利用整数的位取反操作,实现染色体变异. 变量 Bestproduct 为最佳染色体下的 ISFPRM 的乘积项数,并用来衡量电路的面积.

算法 3 GA _ ISFPRM _ Opt(Benchmark,Bestproduct)

```
{let  NG=50;
popsize=pop(n);  //step A3
SOPset=read _ pla(Benchmark);
ISFPRMset=SOP _ to _ ISFPRM(Polarity _ 0,SOPset);  //step B3
Oldpop=initialize(popsize,DC _ number,Polarity);
fitness(Oldpop,ISFPRMset);//step C3
find _ best _ individual(Oldpop,best _ chrom,ISFPRMset);
while(NG ! =0){
if(i=0;i<popsize;i++){
    mate1=select(oldpop,popsize);  //step D3
    mate2=select(oldpop,popsize)
    crossover _ Operation(mate1,mate2,newpop);//step E3
    mutted1=mutation(mate1,mate2,newpop);    //step F3
    mutted2=mutation(mate1,mate2,newpop);  }
    fitness(newpop,ISFPRMset);
    find _ best _ individual(newpop,best _ chrom,ISFPRMset);
    Update _ Population(oldpop,newpop);  }
Bestproduct=counter _ products(best _ chrom);
}
```

5 实验及结果分析

本文算法用 C 语言实现,并对部分 MCNC(Microelectronics Center of North Carolina)电路进行了测试. 所用的计算机配置为 Windows XP 操作系统 4GB 内存和的 3.30GHz 处理器. 遗传算法种群最大个体为 50,最大的迭代数为 50,染色体交叉率为 50%,变异率为 6%. 由于原始的待测电路并没有包含不确定项,因此不确定项都是额外随机添加的. 在本文中. 额外添加的不确定项通过全集与待测电路乘积项集合之间通过集合之间不相交锐积结果中随机取一部分作为不确定项. 考虑到逻辑电路的面积和逻辑函数的乘积项数密切相关,在本文中用逻辑函数的乘积项数来衡量电路的面积. 具体实验结果如表 1 所示.

表 1　实验结果

电路名称	*i/o/p*	No _ dcterms		With _ dcterms			reduced(%)
		products	Time (s)	dcterms	products	Time (s)	
wim	4/7/22	12	0.289	9	2	0.511	83.3
rd53	5/3/67	20	0.299	13	11	0.531	45.0
pm1	16/13/37	28	1.747	20	21	2.124	25.0
tcon	17/16/32	25	0.437	22	11	0.952	56.0
pcle	19/19/45	24	5.396	35	19	7.070	20.8
cc	21/20/52	42	2.356	33	35	3.072	16.7
cm150a	21/1/17	83	4.061	16	5	4.269	93.9
pcler8	27/17/61	52	32.909	50	36	33.201	30.8
unreg	36/16/48	84	4.294	32	73	4.814	13.1
cht	47/36/120	85	7.986	11	83	8.914	2.4
example2	85/66/369	685	14.279	102	532	18.423	22.3
x4	94/71/607	1036	76.212	175	776	99.134	24.2
i6	138/67/239	211	24.907	201	112	62.603	46.9
i7	199/67/302	330	36.738	322	202	88.685	38.8

表 1 中,*i/o/p* 分别表示以“与/或”形式表示的测试电路的输入/输出/乘积项数,No _ dcterms 表示不包含不确定项情况下的电路优化结果,With _ dcterms 表示包含不确定项的电路优化结果. dcterms 表示待优化逻辑函数中添加的不确定项数量,products 表示经极性优化后 RM 函数包含的乘积项数,Time 表示算法运行时间,reduced 表示测试电路包含不确定项后的电路面积优化情况.

$$\text{reduced}\% = \frac{\text{products(No\_dcterms)} - \text{products(With\_dcterms)}}{\text{products(No\_dcterms)}} \times 100\%$$

从表 1 的实验结果中可以看出,相比于不包含不确定项的面积优化,引入不确定项后,测试电路的面积优化均有不同程度的提升. 由于 ISFPRM 优化程度与添加的不确定项有关,从已经发表的论文来看,不确定项都随机加入. 因此进行 ISFPRM 优化算法性能比较时,除了电路面积优化程度外,更重要的一点是算法所能处理的电路规模和运算速度. 文献[12,13]的测试电路均随机生成,最大输入均小于 15 个;文献[14]只说明提出的算法可以处理最大变量数大于 16 的 ISFPRM 函数,而并没有给出具体电路的实验结果. 而从表 1 的实验结果中可以看出,本文提出的算法能处理的输入变量为 199 个的大电路,其主要原因:(1)不同于文献[12~14]算法,本文优化算法以不相交项为基础,由于逻辑函数的不相交乘积项的数量远小于最小项的数量,并且不相交乘积项的数量与逻辑函数输入的变量数没有必然的联系,使得本文算法适合处理大电路,且运算速度具有对电路的输入变量个数不敏感的特点. 表 1 的数据也反映出算法运算的时间与输入变量数没有明显的关联;(2)本文采用一个十进制数表示多位变量的方法. 用提出的二进制插值进行极性转换后,虽然会增加比较多的乘积项,但存储空间增加不大,因此提出的算法在处理乘积项很多的函数时,在内存需要上有优势;(3)在 ISFPRM 优化过程中,式(3)中的相同乘积项的查找与合并是一个耗时的过程,尤其是乘积项数非常庞大时,查找速度对算法速度影响更大. 而采用一个十进制数表示多位变量的方法可以实现二个十进制数比较就可实现多位变量的比较,从而加快查找速度. 上述特点使得提出的方法能有效的克服了文献[12~14]提出的方法无法处理大电路的缺陷,并且具有较快的运算速度.

6　结论

本文在不相交乘积项基础上,通过极性搜索实现 ISFPRM 电路的面积优化. 提出的方法主要包括下面三方面内容:(1)ISFPRM 多位变量的十进制数表示方法;(2)利用二进制插值的实现 ISFPRM 极性转换方法;(3)基于整数位操作实现遗传算法中染色体的复制、交叉和变异. 其中多位变量的十进制数表示方法,二进制插值方法在有利于运算内存的减少,同时提高运算速度的提高. 相比于已发表的基于最小项的 ISFPRM 优化的方法,本文实现了大函数 ISFPRM 电路面积优化,并且具有算法运算速度对待处理函数的输入变量个数不敏感的特点.

参考文献

[1] Khan M H A. An ASIC architecture for generating optimum mixed polarity reed-muller expression[J]. Engineering Letters,2006,13(3):236 - 243.

[2] Jankovic D,Stankovic R S,Moraga C. Optimization of polynomial expressions by using the extended dualpolarity

[J]. IEEE Transactions on Computers, 2009, 58(12): 1710 – 1725.

[3] 王伦耀,夏银水,陈偕雄. 逻辑函数的双逻辑综合与优化[J]. 计算机辅助设计与图形学学报, 2012, 24(7): 961 – 967.
Wang Lunyao, Xia Yinshui, Chen Xiexiong. Logic synthesis and optimization based on dual logic[J]. Journal of Computer- Aided Design & Computer Graphics, 2012, 24(7): 961 – 967. (in Chinese)

[4] M Amy, D Maslov, M Mosca, M Roetteler. A meet-in-the-middle algorithm for fast synthesis of depth-optimal quantum circuits[J]. IEEE Transactions on Computer-Aided Design ofIntegrated Circuits and Systems, 2012, 32(6): 818 – 830.

[5] A Rafiev, A Mokhov, FP Burns, JP Murphy, A Koelmans. Mixed radix reed-muller expansions[J]. IEEE Transactions on Computers, 2012, 61(8): 1189 – 1202.

[6] Saeed S M, Sinanoglu O, Almukhaizim S. Predictive techniques for projecting test data volume compression[J]. IEEE Transactions on very Large scale Integration Systems, 2013, 21(9): 1762 – 1766.

[7] L Amarú, PE Gaillardon, GD Micheli. A circuit synthesis flow for controllable-polarity transistors[J]. IEEE Transactions on Nanotechnology, 2014, 13(6): 1074 – 1083.

[8] 王玉花,王伦耀,夏银水. 基于不相交项并行列表技术的FPRM 实现[J]. 电子与信息学报, 2014, 36(9): 2258 – 2264.
Wang Yuhua, Wang Lunyao, Xia Yinshui. FPRM conversion using parallel tabular technique with disjointed products[J]. Journal of Electronics & Information Technology, 2014, 36(9): 2258 – 2264. (in Chinese)

[9] Al Jassani B A, Urquhart N, Almaini A E A. Manipulation and optimisation techniques for Boolean logic[J]. IET Computers & Digital Techniques, 2010, 4(3): 227 – 239.

[10] 卜登立,江建慧. 基于对偶逻辑的混合极性 RM 电路极性转换和优化方法[J]. 电子学报, 2015, 43(1): 79 – 85.
Bu Dengli1, Jiang Jianhui. Dual logic based polarity conversion and optimization of mixed polarity RM circuits[J]. Acta Electronica Sinica, 2015, 43(1): 79 – 85. (in Chinese)

[11] L Xiao, Z He, L Ruan, R Zhang. Optimization of best polarity searching for mixed polarity reed-muller logic circuit[A]. Publishing House of System-on-chip Conference[C]. Beijing: 2015. 275 – 280.

[12] 汪鹏君,汪迪生,蒋志迪,张会红. 基于 PSGA 算法的ISFPRM 电路面积与功耗优化[J]. 电子学报, 2013, 41(8): 1542 – 1548.
Wang Pengjun, Wang Disheng, Jiang Zhidi, Zhang Huihong. Area and power optimization of ISFPRM circuits based on PSGA algorithm[J]. Acta Electronica Sinica, 2013, 41(8): 1542 – 1548. (in Chinese)

[13] D Debnath, T Sasao. Exact minimization of FPRM for incompletely specified functions by using MTBDDs[J]. IEICE Transactions on Fundamentals of Electronics, Communications and Computer Science, 2005, 88(12): 3332 – 3341.

[14] M K Habib. A new approach to generate fixed-polarity Reed Muller expansions for completely and incompletely specified functions[J]. International Journal of Electronics, 2002, 89(11): 845 – 876.

作者简介

瞿 婷 女,1992 年生于贵州铜仁,现为宁波大学硕士研究生. 主要的研究方向为 RM 逻辑的混合极性综合.
E-mail: quting _ qt@ 163. com

王伦耀 男,1972 年生于浙江宁波,博士,教授. 主要的研究方向为 Reed-Muller 逻辑以及基于 Reed-Muller 逻辑与传统布尔逻辑的双逻辑综合与优化.
E-mail: wanglunyao@ nbu. edu. cn