基于混合多值离散粒子群优化的混合极性 Reed-Muller 最小化算法

卜登立*①② 江建慧①
①(同济大学软件学院 上海 201804)
②(井冈山大学电子与信息工程学院 吉安 343009)

摘 要：针对布尔函数系统的混合极性 Reed-Muller(Mixed-Polarity Reed-Muller, MPRM)最小化问题，该文提出了一种混合多值离散粒子群优化算法。为解决多样性损失，改善优化结果，兼顾算法的效率和精度，算法采用多群协同优化方法，并提出了概率变异更新、没有重复的更新以及群间重复最优变异 3 种更新和变异策略。实验结果表明，和模拟退火遗传算法相比，所构造算法能够在获得基本相同优化结果的同时，提高 MPRM 最小化的时间效率。

关键词：数字电路；布尔函数系统；混合极性 Reed-Muller；多值离散粒子群优化；多群；更新和变异策略
中图分类号：TP331.2; TP391.72 **文献标识码**：A **文章编号**：1009-5896(2013)02-0361-07
DOI: 10.3724/SP.J.1146.2012.00790

Hybrid Multi-valued Discrete Particle Swarm Optimization Algorithm for Mixed-polarity Reed-Muller Minimization

Bu Deng-li①② Jiang Jian-hui①
①(*School of Software Engineering*, *Tongji University*, *Shanghai* 201804, *China*)
②(*School of Electronics and Information Engineering*, *Jinggangshan University*, *Ji'an* 343009, *China*)

Abstract: A novel hybrid multi-valued Discrete Particle Swarm Optimization (DPSO) algorithm for Mixed-Polarity Reed-Muller (MPRM) minimization of Boolean function system is proposed. To solve the problem of diversity loss, improve the optimized result and balance the efficiency and precision of DPSO, multi-swarm cooperative optimization is employed, and three update and mutation strategies of update with probabilistic mutation, update with no duplicates and mutation with best duplicates between swarms are proposed. The experimental results show that compared with Simulated Annealing Genetic Algorithm (SAGA), the proposed algorithm can obtain similar optimized results and improve the time efficiency of MPRM minimization.

Key words: Digital circuit; Boolean function system; Mixed-Polarity Reed-Muller (MPRM); Multi-valued Discrete Particle Swarm Optimization (DPSO); Multi-swarm; Update and mutation strategies

1 引言

相对于布尔函数的 SOP(Sum Of Product)表示，ESOP(Exclusive-OR Sum Of Product)表示能够得到更小更高效的实现，用 ESOP 表示实现的电路能够获得面积、功耗和可测性等方面的显著优势[1–3]。并且因为可以直接映射到目标电路，ESOP 表示已经在可逆电路和量子电路的综合方法中得到了应用[4]。然而，由于 ESOP 不是标准形表示，最小 ESOP 很难求解[5]，因此，其子类固定极性 Reed-Muller(Fixed-Polarity Reed-Muller, FPRM)和混合极性 Reed-Muller(Mixed-Polarity Reed-Muller, MPRM)的最小化成为人们比较关注的问题。相对于最小 FPRM，最小 MPRM 的乘积项数更接近于最小 ESOP 的乘积项数，因此 MPRM 最小化问题受到了广泛的关注。

布尔函数系统的 MPRM 最小化是 RM 电路逻辑综合过程中一个非常重要的阶段，求解乘积项数最少的 MPRM 表示可以为进一步的 ESOP 最小化提供一个初始覆盖[5]。布尔函数系统的 MPRM 能够通过其对应的混合极性值来确定，因此可以通过搜索布尔函数系统的混合极性空间来求解最小 MPRM。尽管采用穷举搜索策略遍历混合极性空间可以求得精确的最小 MPRM，然而由于极性空间与变量数之间呈指数关系，对于输入数大于 14 的电路而言，穷举搜索在时间上不可行[6]。因此，研究者更倾向于采用智能优化方法来求解最小 MPRM，如文献[7,8]采用遗传算法(Genetic Algorithm, GA)求解最小 MPRM。采用 GA 可以加快极性空间的搜索，能够在较短的时间内得到比较好的结果，并且优于爬山算法和模拟退火算法[7]。然而对于输入数较多的

2012-06-21 收到，2012-11-14 改回
国家自然科学基金(60903033)资助课题
*通信作者：卜登立 bodengli@163.com

电路，采用遗传算法由于过早收敛问题可能得不到全局最优解[6]，因此文献[6]采用模拟退火遗传算法(Simulated Annealing Genetic Algorithm, SAGA)求解最小 MPRM，采用 SAGA 能够获得比 GA 更好的结果和性能[6]。

粒子群优化算法(Particle Swarm Optimization, PSO)是一种新的群智能优化算法，已被广泛地应用于各种复杂的优化领域[9]。与其他进化算法相比，PSO 具有容易实现和性能良好的优势，并且大量实验结果也表明了 PSO 是一种强大的优化工具[9,10]。本文提出了一种基于混合多值离散粒子群优化(multi-valued Discrete Particle Swarm Optimization, DPSO)的 MPRM 最小化算法。为改善优化结果，算法采用了多群协同优化方法，同时，受 GA 算法基本操作的启发，提出了概率变异更新策略、没有重复的更新策略以及群间重复最优变异策略。通过实验验证了所提出策略以及算法的有效性，并与基于 SAGA 的 MPRM 最小化算法进行了比较。

2 问题描述

2.1 MPRM 基本概念

一个 n -输入 m -输出的布尔函数系统 $\boldsymbol{F}:\{0,1\}^n \to \{0,1\}^m$，由 m 个布尔函数构成，每个布尔函数对应一个输出，其第 k ($0 \le k \le m-1$)个输出的 SOP 标准形表示如式(1)所示。

$$f_k\left(x_{n-1},x_{n-2},\cdots,x_0\right)=\sum_{i=0}^{2^n-1}a_{k,i}\omega_i \tag{1}$$

其中 $a_{k,i}\in\{0,1\}$ 为第个 k 输出表达式中乘积项的系数，ω_i 为第 i 个乘积项，也称为最小项。

将式(1)所示布尔函数系统的所有变量按照一定顺序通过正 Davio 分解、负 Davio 分解和 Shannon 分解[2]进行展开，根据变量在展开时所采用的分解类型，可以得到一个分解类型列表(Decomposition Type List, DTL)[2]，表示为 $\boldsymbol{D}=[d_{n-1},d_{n-2},\cdots,d_0]$，其中 $d_l\in\{0,1,2\}$ 表示变量 x_l 的分解类型分别为正 Davio 分解、负 Davio 分解和 Shannon 分解，又称之为变量的极性，DTL 对应的十进制值 $\sum_{l=0}^{n-1}d_l 3^l$ 为 MPRM 的极性值。由于每个变量可选的极性有 3 种，因此对于 n 个变量的布尔函数系统而言，MPRM 的极性空间大小为 3^n，共有 3^n 种 MPRM 标准形表示。

一个 n -输入 m -输出的布尔函数系统按照上述方式展开后，得到的极性值为 h 的 MPRM 标准形可以表示为如式(2)所示的异或和多项式形式。

$$f_k^h\left(x_{n-1},x_{n-2},\cdots,x_0\right)=\oplus\sum_{i=0}^{2^n-1}a_{k,i}\pi_i \tag{2}$$

其中 $\oplus\Sigma$ 为异或和操作，$a_{k,i}$ 为 MPRM 第 k 个输出的表达式系数，乘积项 $\pi_i=\prod_{l=n-1}^{0}x_l^{(d_l,i_l)}$，$i_l$ 表示索引 i 的第 l 个二进制位，并且 $x_l^{(d_l,i_l)}$ 的形式根据 DTL 以及系数索引按式(3)的方式确定。

$$x_l^{(d_l,i_l)}=\begin{cases}1, & (d_l,i_l)=(0,0)\text{或}(1,0)\\ x_l, & (d_l,i_l)=(0,1)\text{或}(2,1)\\ \overline{x}_l, & (d_l,i_l)=(1,1)\text{或}(2,0)\end{cases} \tag{3}$$

2.2 MPRM 最小化问题

MPRM 最小化的目的是通过对极性值进行选择以使得如式(2)所示的 MPRM 表达式中的乘积项数最少。对于一个如式(1)所示的布尔函数系统，其混合极性空间大小为 3^n，每个极性值使用一个 DTL 表示，并且一个 DTL 可以唯一地确定一个如式(2)所示的 MPRM。因此，MPRM 最小化问题可以转化为求解最优极性值的问题，并描述为：搜索混合极性空间，通过极性转换得到所搜索到的极性值对应的 MPRM，其中具有最少乘积项数的 MPRM 为最小 MPRM，相应的极性值即为最优极性值。

本文的重点是对布尔函数系统混合极性空间的搜索策略进行研究，搜索过程之中的极性转换使用文献[1]中基于多输出 MPRM 表达式的列表转换技术完成。

3 基于混合 DPSO 的 MPRM 最小化

3.1 编码与适应度函数

PSO 算法在进行编码选择时需要遵守完备性、健全性和非冗余性 3 个基本原则[10]。由于 MPRM 中变量的极性为三进制表示，因此本文使用三进制进行编码，将 DTL 作为粒子的位置向量 $\boldsymbol{D}_j=[d_{j,n-1},\cdots,d_{j,0}]$，$d_{j,l}$ 表示粒子群中索引为 j 的粒子在 n 维搜索空间中第 l 维的位置，即 MPRM 第 l 个变量的极性。

由于 MPRM 表达式中所包含的乘积项数的多少表示了该 MPRM 的优劣，因此，将 MPRM 中的乘积项数作为粒子的适应度。在计算粒子的适应度时，首先对 MPRM 进行极性转换，然后再统计该 MPRM 所包含的乘积项数。

3.2 算法模型

PSO 将问题的一组可行解建模为搜索空间中移动的粒子群，在算法的每一次迭代过程中记忆粒子的个体最优位置以及邻域最优位置，并根据一定的相互作用规则来更新粒子的速度以及位置，使粒子向更好的区域移动，从而获得最优解。但传统的单群模型容易使算法过早收敛从而使搜索陷入局部极小[11]，因此，为了避免搜索陷入局部极小，本文采用多群方法[11]，利用多个粒子群同时搜索解空间中

的多个不同区域。多群方法非常适合于解决多峰优化问题，对于多峰问题采用多群要比采用单群能够获得更为优化的结果[11,12]。布尔函数系统的混合极性空间中可能有多个极性值具有相同的适应度，即多个 MPRM 具有相同的乘积项数，可见 MPRM 最小化问题具有多峰问题的特征，因此本文采用多群协同优化方法，并采用文献[12]中的主-从多群模型。

主-从多群模型使用 q ($q>1$)个群进行协同优化，每个群独立的运行 PSO 算法。其中一个群为主群，其他 $q-1$ 个群为从群，所有 q 个群的最优位置作为全局最优位置，即潜在的全局最优解。从群根据本群的群历史最优位置来更新群中粒子的速度，而主群则根据全局最优位置更新群中粒子的速度。这样既利用 $q-1$ 个从群的独立搜索来保证在搜索空间中的多个区域进行搜索，又通过主群和从群间的空间相互作用使主群在全局最优区域中进行搜索，从而保持搜索空间中的全局探索和局部最优周围的精搜索之间的平衡，避免搜索陷入到局部极小[12]。

假设第 s ($0\le s\le q-1$)个群中第 j 个粒子的速度向量使用 $\boldsymbol{V}_{s,j}=[v_{s,j,n-1},\cdots,v_{s,j,0}]$ 表示， $v_{s,j,l}$ 表示该粒子在 n 维搜索空间中第 l 维的速度，第 s 个群中第 j 个粒子的历史最优位置用 $\boldsymbol{P}_{s,j}=[p_{s,j,n-1},\cdots,p_{s,j,0}]$ 表示，第 s 个群的群历史最优位置用 $\boldsymbol{B}_s=[b_{s,n-1},\cdots,b_{s,0}]$ 表示，q 个群的全局最优位置用 $\boldsymbol{G}=[g_{n-1},\cdots,g_0]$ 表示，则 $q-1$ 个从群中粒子的速度更新公式如式(4)所示，主群中粒子的速度更新公式如式(5)所示。

$$\begin{aligned}v_{s,j,l}^{t+1}&=W\times v_{s,j,l}^{t}+C_1\times r_1\times(p_{s,j,l}^{t}-d_{s,j,l}^{t})\\&\quad+C_2\times r_2\times(b_{s,l}^{t}-d_{s,j,l}^{t})\end{aligned}\tag{4}$$

$$\begin{aligned}v_{s,j,l}^{t+1}&=W\times v_{s,j,l}^{t}+C_1\times r_1\times(p_{s,j,l}^{t}-d_{s,j,l}^{t})\\&\quad+C_2\times r_2\times(g_{l}^{t}-d_{s,j,l}^{t})\end{aligned}\tag{5}$$

其中 W 称为惯性权重， C_1 和 C_2 为学习因子，一般情况下 $C_1=C_2$, r_1 和 r_2 是 $[0,1]$ 内均匀分布的随机数，这些参数用来控制历史速度以及历史最优位置对粒子速度的影响。上标 t 表示第 t 次迭代。另外，通过设置速度限制参数 $V_{\max}$ ，使速度控制在 $[-V_{\max},V_{\max}]$ 范围之内， $V_{\max}$ 的引入可以改善搜索的分辨率。

由于 DPSO 中粒子的速度表示粒子位置取值的概率，因此，受文献[13]的启发，本文采用的粒子位置更新公式如式(6)所示。

$$d_{s,j,l}^{t+1}=\left\lfloor 3\Big/\left(1+\exp(-v_{s,j,l}^{t+1})\right)\right\rfloor\tag{6}$$

其中 $\lfloor\cdot\rfloor$ 为下取整函数。由于式(6)中分母的取值范围为 $(1,+\infty)$ ，因此可以保证位置不会超出解元素取值范围 $\{0,1,2\}$ 。式(6)所示的更新公式实际上体现了启发式的更新规则，如果历史最优位置与粒子位置之间的差值($p_{s,j,l}^{t}-d_{s,j,l}^{t}$, $b_{s,l}^{t}-d_{s,j,l}^{t}$ 和 $g_{l}^{t}-d_{s,j,l}^{t}$)是正值，则此差值越大导致粒子的速度经逐代累积后也越大，因此潜在解的位置取值应该越大，相反，如果此差值是负值，则此差值越小导致粒子的速度经逐代累积后也越小，因此潜在解的位置取值应该越小。

3.3 更新和变异策略

对于 MPRM 最小化问题，标准 DPSO 存在着多样性损失导致的过早收敛问题，尽管采用主-从多群模型以及式(4)~式(6)所示的更新公式使得优化结果有所改善，但式(6)启发式的位置更新公式又导致算法的随机性和多样性受到一定损失，因此对一些电路测试多次所得结果的均值与标准差都比较大。为改善算法优化结果，本文提出了概率变异更新、没有重复的更新以及群间重复最优变异 3 种更新和变异策略。

3.3.1 概率变异更新策略　为了解决随机性损失和多样性损失，改善 DPSO 的优化结果，受 GA 基本操作的启发，本文提出了粒子概率变异更新策略。尽管文献[10,14]也采用了变异策略，但其中的变异策略是在完成粒子速度与位置的更新后对粒子的位置进行变异，而本文所提出的概率变异更新策略则是在对粒子的位置进行更新时以一定的概率对粒子的位置实施变异操作。概率变异更新公式如式(7)所示，其中 $r\in\{1,2\}$ 是一个均匀分布的随机数。

$$\left.\begin{aligned}v_{s,j,l}^{t+1}&=v_{s,j,l}^{t}\\d_{s,j,l}^{t+1}&=(d_{s,j,l}^{t}+r)\%3\end{aligned}\right\}\tag{7}$$

通过设置变异更新概率 p_{mu} ，以 $1-p_{mu}$ 的概率按式(4)~式(6)进行速度和位置的常规更新，以 p_{mu} 的概率按式(7)的方式进行变异更新，从而在增加算法随机性的同时增加了多样性。通过实验发现， p_{mu} 取一个小于等于 0.05 的值，可显著改善算法的优化结果。为减少算法参数的设置，本文通过在算法运行过程中随机产生一个在 $[0.01,0.05]$ 之内均匀分布的随机数作为变异更新概率 p_{mu} 。

3.3.2 没有重复的更新策略　没有重复的更新策略是受 GA 中没有重复的稳态替换策略[7,15]启发，其目的是维持群中粒子的多样性。假设已经根据式(4)~式(6)并结合概率变异更新策略对第 s 个群中的第 j 个粒子 $\mathrm{part}_{s,j}$ 的速度和位置进行更新得到了新粒子 $\mathrm{part}_{s,j}^{'}$ ，然后将 $\mathrm{part}_{s,j}^{'}$ 与第 s 个群中的所有粒子进行比较，如果该群中存在与 $\mathrm{part}_{s,j}^{'}$ 的适应度相同的粒子，并且 $\mathrm{part}_{s,j}^{'}$ 的历史最优适应度大于或等于该粒子的历史最优适应度，则保持 $\mathrm{part}_{s,j}$ 不变，否则使用 $\mathrm{part}_{s,j}^{'}$ 替换 $\mathrm{part}_{s,j}$ 。

没有重复的更新策略有助于增加群内粒子间的排斥力，使群搜索可以在更大的区域进行，从而避免由于过早收敛导致搜索陷入到局部极小。通过实验发现，没有重复的更新策略可在一定程度上改善算法的优化结果。

3.3.3 群间重复最优变异策略 群间重复最优变异策略是为了增加群间的排斥力，以保证多个群能够搜索不同的区域，从而避免过早收敛，增强全局搜索能力。群间重复最优变异策略应用在 q 个群中的所有粒子已经完成速度和位置的更新，并且每个粒子的历史最优以及每个群的群历史最优均完成更新之后，如果存在群历史最优适应度相同的群，则对从群的群历史最优位置向量采用均匀变异方法实施变异操作。假设序号为 $q-1$ 的群为主群，第 s 个群的群历史最优对应的粒子为该群中的第 o 个粒子 $\text{part}_{s,o}$，下面给出群间重复最优变异策略的算法步骤如表 1 所示。

表 1 群间重复最优变异策略算法

(1)令 $s=0$；
(2)令 $u=s+1$；
(3)如果第 s 个群的群历史最优适应度等于第 u 个群的群历史最优适应度，则转步骤(5)；否则转步骤(4)；
(4)令 $u=u+1$，如果 $u<q$ 则转步骤(3)；否则转步骤(6)；
(5)对 $\text{part}_{s,o}$ 的历史最优位置向量 $\boldsymbol{P}_{s,o}$ 采用均匀变异方法进行变异，得到新位置向量 $\boldsymbol{P}'_{s,o}$，计算 $\boldsymbol{P}'_{s,o}$ 对应的适应度，并和 $\text{part}_{s,o}$ 的当前位置向量 $\boldsymbol{D}_{s,o}$ 对应的适应度进行比较，适应度较小的位置向量作为 $\text{part}_{s,o}$ 新的历史最优位置向量 $\boldsymbol{P}_{s,o}$，同时更新 $\text{part}_{s,o}$ 的历史最优适应度；
(6)令 $s=s+1$，如果 $s<q-1$ 则转步骤(2)，否则结束。

群间重复最优变异策略可以增加从群之间以及从群与主群之间的排斥力，使从群能够搜索更大区域的同时，主群保持在全局最优区域进行搜索。通过实验发现，群间重复最优变异策略可以起到反收敛的作用，能在一定程度上改善算法的优化结果。

3.4 初始化以及结束条件

初始化包括 q 个群中粒子位置以及速度的初始化，初始位置特别是初始速度会影响算法的结果。本文对 q 个群分别进行初始化，通过随机生成 $[0,2]$ 内均匀分布的整数对位置进行初始化，速度则初始化为 $[-V_{\max}/2, V_{\max}/2]$ 内均匀分布的随机数。

算法的结束条件既可以为达到预先指定的最大迭代次数，也可以根据特定的优化问题以及特定的实例来确定结束条件。本文采用后一种方法，在达到最大迭代次数之前，如果最优结果没有改善所累计的次数达到一个阈值则结束迭代过程。该阈值和布尔函数系统的规模有关，可以作为一个参数进行设置。为减少参数的设置，本文根据布尔函数系统的变量个数 n 设定该阈值为 $20\times\ln(n)$ [5]。

3.5 混合 DPSO 算法描述

下面给出采用主-从多群模型、式(4)~式(6)所示的粒子更新规则，以及本文提出的 3 种更新和变异策略的用于 MPRM 最小化的混合 DPSO 算法，其中序号为 $q-1$ 的群为主群。具体算法如表 2 所示。

表 2 混合 DPSO 算法(序号为 $q-1$ 的群为主群)

(1)设定参数：最大迭代次数，群数 q 以及群规模 q_{sz}，惯性权重 W，学习因子 C_1 和 C_2，速度限制参数 $V_{\max}$；
(2)读取网表文件；
(3)生成初始的 q 个群，计算群中粒子的适应度，更新粒子历史最优、群历史最优和全局最优信息；
(4)迭代次数赋 0 值；
(5)令 $s=0$；
(6)令 $j=0$；
(7)令 $l=0$；
(8)生成均匀分布的随机数 $p_{mu}\in[0.01,0.05]$ 作为变异更新概率，生成 0~1 之间均匀分布的随机数 rm，如果 $rm>=p_{mu}$ 则转步骤(9)，否则转步骤(10)；
(9)如果 $s<q-1$ 则按式(4)更新 $v_{s,j,l}$，否则按式(5)更新 $v_{s,j,l}$，同时确保速度不越界；并按式(6)更新 $d_{s,j,l}$，然后转步骤(11)；
(10)根据式(7)对 $d_{s,j,l}$ 实施变异更新操作，并转步骤(11)；
(11)令 $l=l+1$，如果 $l<n$，则转步骤(8)；
(12)令 $j=j+1$，如果 $j<q_{sz}$ 则转步骤(7)；
(13)令 $s=s+1$，如果 $s<q$ 则转步骤(6)；
(14)计算 q 个群中所有更新后的粒子的适应度，使用没有重复的更新策略替换相应群中的粒子，并更新 q 个群中所有粒子的历史最优以及每个群的群历史最优；应用群间重复最优变异策略，并更新全局最优；迭代次数+1，统计最优结果累计没有改善的次数，如果不满足结束条件则转步骤(5)；
(15)输出最优 MPRM 结果，算法结束。

4 实验设置以及结果分析

由于文献[6]得出了基于 SAGA 的 MPRM 最小化算法(简称为 SAGA)要优于基于 GA 的 MPRM 最小化算法，因此本文仅与文献[6]中的 SAGA 算法进行比较。本文的基于混合 DPSO 的 MPRM 最小化算法(简称为 HDPSO)与文献[6]中的 SAGA 均使用 C++实现，程序运行的硬件环境为 Intel® Core™2 Duo CPU E8400@3GHz CPU 2GB 内存，软件环境为 Windows XP Professional 操作系统。

4.1 实验设置

HDPSO 中的参数通过初步实验确定，惯性系

数 $W = 0.9$，学习因子 $C_1 = C_2 = 3$，速度限制参数 $V_{\max} = 9$，群规模 q_{sz} 设置为 5，群数 q 设为 6，最大迭代次数设为 180。为使比较相对公平，SAGA 中的种群规模设定为 30。

为了验证本文提出的算法并和 SAGA 进行比较，选择了 21 个不同规模的输入数大于 14 的 MCNC 基准电路进行测试。由于 HDPSO 和 SAGA 均具有一定的随机特性，因此实验中对每个基准电路都测试 20 次，并统计算法结果的最小值、均值和标准差，以及算法迭代次数和所花费 CPU 时间的平均值。

4.2 结果分析

表 1 给出了 HDPSO 和 SAGA 的测试结果，从左至右依次给出了基准电路名称、电路的输入数/输出数、20 次测试所得到的最优 MPRM 中乘积项数的最小值、均值和标准差。从表 3 可以看出，两种算法优化结果的最小值完全相同，均值也基本相同，并且对于大多数电路而言，两种算法均能稳定地获得最小 MPRM 结果。尽管对于部分电路而言，两种算法不能总是得到最小 MPRM，但除电路 in2 外，均值都等于或者非常接近最小值，并且标准差比较小。

另外，由表 3 可以看出，对于某些电路 HDPSO 总能得到最小 MPRM，而 SAGA 却不能，如 tcon。同样，对于某些电路 SAGA 总能得到最小 MPRM，而 HDPSO 却不能，如 spla。这种情况下，由于电路不同，不能直接根据均值或者标准差来判断算法的优劣。因此，为了比较两种算法，图 1 给出了 HDPSO 和 SAGA 结果的变异系数(不包含两种算法结果的变异系数均为 0 的电路)，变异系数等于标准差除以均值，用于衡量不同量纲或者不同均值时数据的离散程度或者稳定性。由图 1 可以看出，从结果的离散程度或者稳定性角度看，两种算法基本相同。可见，HDPSO 能够获得和 SAGA 基本相同的优化结果。

表 3 MCNC 电路测试结果

基准电路	输入/输出	最小值	均值(标准差)	
			HDPSO	SAGA
b12	15/9	64	64(0.40)	64(0.22)
m181	15/9	65	65(0.36)	65(0.22)
b10	15/11	222	222(0.68)	222(0.87)
gary	15/11	242	242(0.22)	242(0.36)
ryy6	16/1	48	48(0)	48(0)
t481	16/1	13	13(0)	13(0)
b9	16/5	105	105(0)	105(0)
b2	16/17	333	333(0)	333(0)
spla	16/46	628	629(2.62)	628(0)
table5	17/15	559	559(0)	559(0)
t2	17/16	81	81(0)	81(0)
tcon	17/16	24	24(0)	25(0.5)
vda	17/39	93	93(0)	93(0)
pcle	19/9	24	24(0)	24(0)
in2	19/10	262	267(7.30)	268(7.35)
shift	19/16	100	100(0)	100(0.43)
cm150a	21/1	32	32(0)	32(0)
mux	21/1	16	16(0)	16(0)
cc	21/20	41	41(0.43)	41(0)
duke2	22/29	209	209(0)	209(0)
cordic	23/2	1980	1980(0)	1980(0)

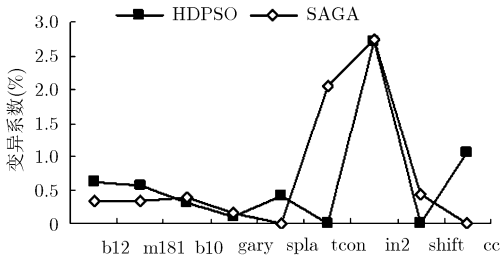

图 1 HDPSO 和 SAGA 结果的变异系数

表 4 给出了每个电路对两种算法分别进行 20 次测试求解最优 MPRM 平均所需的迭代次数和所花费的 CPU 时间，时间的单位为 s。SAGA 在 GA 中加入模拟退火过程的目的是加快算法的全局收敛速度，由表 4 可以看出，从迭代次数角度看，除电路 tcon, pcle 和 shift 外，SAGA 和 HDPSO 相比，MPRM 最小化所需的迭代次数大为减少。从总体角度看，SAGA 所需的迭代次数比 HDPSO 平均减少了 30%。但从算法所花费时间角度看，由于 SAGA 在迭代过程中加入了模拟退火过程，需要计算个体的适应度以使用 Metropolis 准则以一定概率接受从邻域产生的个体，因此 SAGA 每一次迭代过程所花费的时间要比 HDPSO 长。另外，SAGA 使用平均增量方法确定初始温度的过程需要计算从个体邻域产生新个体的适应度，也使得所花费的时间有所增加。因此，除电路 spla 外，SAGA 所花费的时间均比 HDPSO 长。对于电路 spla, SAGA 所需迭代次数比 HDPSO 减少的幅度最大，达到了 60%，这使得 SAGA 由于迭代次数减少导致时间的降低超过了由于模拟退火过程导致时间的增加，因此 SAGA 所花费的时间低于 HDPSO 所花费的时间，但 CPU 时间仅减少了 0.9%。从总体角度看，HDPSO 所花费的 CPU 时间比 SAGA 平均减少了 36%。可见，HDPSO 相对于 SAGA，能够提高 MPRM 最小化的时间效率。

表 4 算法迭代次数以及时间

基准电路	迭代次数		CPU 时间(s)	
	HDPSO	SAGA	HDPSO	SAGA
b12	98	57	1.0970	1.3556
m181	103	57	1.1687	1.2959
b10	85	51	0.8965	1.2860
gary	88	50	0.9309	1.2561
ryy6	71	46	1.6045	2.1586
t481	98	55	2.3067	2.5951
b9	66	45	1.4899	2.1839
b2	94	49	1.8669	2.3445
spla	107	43	3.1526	3.1239
table5	103	45	4.3243	5.2874
t2	77	65	3.3735	7.1346
tcon	82	86	4.9717	13.8960
vda	78	72	4.6923	13.6800
pcle	78	83	17.2680	38.4850
in2	112	61	20.6630	29.6790
shift	76	99	19.1030	63.7470
cm150a	85	77	83.9780	151.7200
mux	80	78	72.1020	149.3600
cc	101	93	106.9300	180.7500
duke2	125	76	220.7000	349.4700
cordic	106	56	425.9900	548.8200

5 结束语

由于布尔函数的 ESOP 表示能够获得更小更高效的实现，使得 ESOP 表示得到了广泛的应用。FPRM 和 MPRM 均是 ESOP 的标准形子类，和 FPRM 相比，MPRM 有可能获得更为简洁的表示，因此得到了更为广泛的关注。由于 MPRM 的极性空间为 3^n，因此对于输入数较多的电路，为提高 MPRM 最小化的时间效率，需要采用智能优化方法。本文提出了一种用于 MPRM 最小化的混合 DPSO 算法，为了能够在更大的区域进行搜索，避免陷入到局部极小，算法采用了多群协同优化方法，并提出了概率变异更新、没有重复的更新以及群间重复最优变异策略，实验结果验证了所提出算法的有效性。和 SAGA 对比的结果表明，本文提出的算法能够在获得与 SAGA 基本相同优化结果的前提下提高 MPRM 最小化的时间效率。

关于 MPRM 的一个重要研究方向是多目标优化，如兼顾面积、功耗、可测试性等技术指标，而 PSO 常用来解决多目标优化问题，下一步的研究工作则是使用 DPSO 解决 MPRM 的多目标优化问题。

参 考 文 献

[1] 李辉, 汪鹏君, 王振海. 混合极性列表技术及其在 MPRM 电路面积优化中的应用[J]. 计算机辅助设计与图形学学报, 2011, 23(3): 527–533.
Li Hui, Wang Peng-jun, and Wang Zhen-hai. Tabular techniques for mixed-polarity and its application in area optimization of MPRM circuits[J]. *Journal of Computer-Aided Design & Computer Graphics*, 2011, 23(3): 527–533.

[2] 汪鹏君, 李辉. 基于 OKFDDs 的 Reed-Muller 逻辑混合极性转换算法[J]. 电子与信息学报, 2011, 33(4): 932–937.
Wang Peng-jun and Li Hui. An algorithm of Reed-Muller logic mixed-polarity conversions based on OKFDDs[J]. *Journal of Electronics & Information Technology*, 2011, 33(4): 932–937.

[3] Rahamana H, Dasb D K, and Bhattacharya B B. Testable design of AND-EXOR logic networks with universal test sets[J]. *Computers and Electrical Engineering*, 2009, 35(5): 644–658.

[4] Drechsler R, Finder A, and Wille R. Improving ESOP-based synthesis of reversible logic using evolutionary algorithms[J]. *Lecture Notes in Computer Science*, 2011, 6625: 151–161.

[5] Finder A and Drechsler R. An evolutionary algorithm for optimization of pseudo Kronecker expressions[C]. International Symposium on Multiple-Valued Logic, Barcelona, 2010: 150–155.

[6] Wang Peng-jun, Li Hui, and Wang Zhen-hai. MPRM expressions minimization based on simulated annealing genetic algorithm[C]. International Conference on Intelligent Systems and Knowledge Engineering, Hangzhou, 2010: 261–265.

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

[8] 杨萌, 徐红英, Almaini A E A. 针对混合极性的并行表格技术的遗传算法[J]. 计算机辅助设计与图形学学报, 2011, 23(11): 1938–1943.
Yang Meng, Xu Hong-ying, and Almaini A E A. Optimization of mixed polarity functions using genetic algorithm with parallel tabular technique[J]. *Journal of Computer-Aided Design & Computer Graphics*, 2011, 23(11): 1938–1943.

[9] Banks A, Vincent J, and Anyakoha C. A review of particle swarm optimization. Part I: background and development[J]. *Natural Computing*, 2007, 6(4): 467–484.

[10] 郭文忠, 陈国龙, Xiong Nai-xue, 等. 求解 VLSI 电路划分问题的混合粒子群优化算法[J]. 软件学报, 2011, 22(5): 833–842.
Guo Wen-zhong, Chen Guo-long, Xiong Nai-xue, *et al*.. Hybrid particle swarm optimization algorithm for VLSI circuit partitioning[J]. *Journal of Software*, 2011, 22(5): 833–842.

[11] Blackwell T and Branke J. Multi-swarm optimization in dynamic environments[J]. *Lecture Notes in Computer Science*, 2004, 3005: 489–500.

[12] Niu Ben, Zhu Yun-long, He Xiao-xian, *et al*.. MCPSO: a multi-swarm cooperative particle swarm optimizer[J]. *Applied Mathematics and Computation*, 2007, 185(2): 1050–1062.

[13] Veeramachaneni K, Osadciw L, and Kamath G. Probabilistically driven particle swarms for optimization of multi valued discrete problems: design and analysis[C]. IEEE Swarm Intelligence Symposium, Honolulu, Hawaii, 2007: 141–149.

[14] Reis C and Machado J A T. Circuit synthesis through combination of evolutionary and swarm algorithms[C]. WSEAS Internatiosnal Conference on Computers, Rodos Island, 2009: 115–120.

[15] Burke E K and Kendall G. Search Methodologies: Introductory Tutorials in Optimization and Decision Support Techniques[M]. New York: Springer, 2005, Chapter 4.

卜登立：　男，1975 年生，博士生，副教授，研究方向为 VLSI 设计和可靠性评估.

江建慧：　男，1964 年生，博士，教授，博士生导师，研究方向为可信系统与网络、软件可靠性工程、VLSI/SoC 测试与容错.